You must be at least this tall to ride: How standards help improve your organization's software quality

John R. Marien
Chief Software Engineer
Software Assurance/Cybersecurity Technical Evangelist
The MITRE Corporation
jmarien@mitre.org

YOU MUST BE AT LEAST THIS TALL TO RIDE

19 June 2019

Version 1.0

MITRE

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

MITRE is a not-for-profit organization that operates research and development centers sponsored by the federal government.

We operate FFRDCs—federally funded research and development centers—which are unique organizations that assist the United States government with:

- Scientific research and analysis
- Development and acquisition
- Systems engineering and integration

We also have an independent research program that explores new and expanded uses of technologies to solve our sponsors' problems.

MITRE's Sole Focus Is to Operate FFRDCs

MITRE is chartered to work in the public interest. We have no commercial interests. We have no owners or shareholders, and we can't compete for anything except the right to operate FFRDCs. This lack of commercial conflicts of interest forms the basis for our objectivity. We also have the ability to acquire sensitive and proprietary information from the government and industry to inform our work. These organizations are able and willing to share data because they know we won't use it for a competitive advantage.

Moreover, because we operate multiple FFRDCs, we foster a culture of knowledge sharing. We apply what we learn from addressing one sponsor's challenges to similar issues faced by other federal agencies. This means when sponsors engage with us, they have access to all the minds of MITRE.

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

- I work directly with the OSD Software Assurance Community of Practice and the DHS Software and Supply Chain Assurance Community of Practice

- I run MITRE's Software Quality Assurance Evaluation Teams
  - We've evaluated over 230 Million Lines of Code
  - Unclassified through TS SCI
  - Covered 120+ Languages
  - Are a Subject Matter Expert (SME) in-the-loop evaluation team for both the Source Code and the Documentation
  - We use multiple COTS static, and dynamic, code analyzers as reference input to the SMEs
  - We highlight the Good in the Code/Documentation equally with highlighting areas that require improvement – All non-trivial software can be improved

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

We hold these truths

to be self-evident*

Where is your data on that?

* Declaration of Independence

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

What do you say after 100+ evaluations?
After 84 MSLOC?
In over a dozen languages?

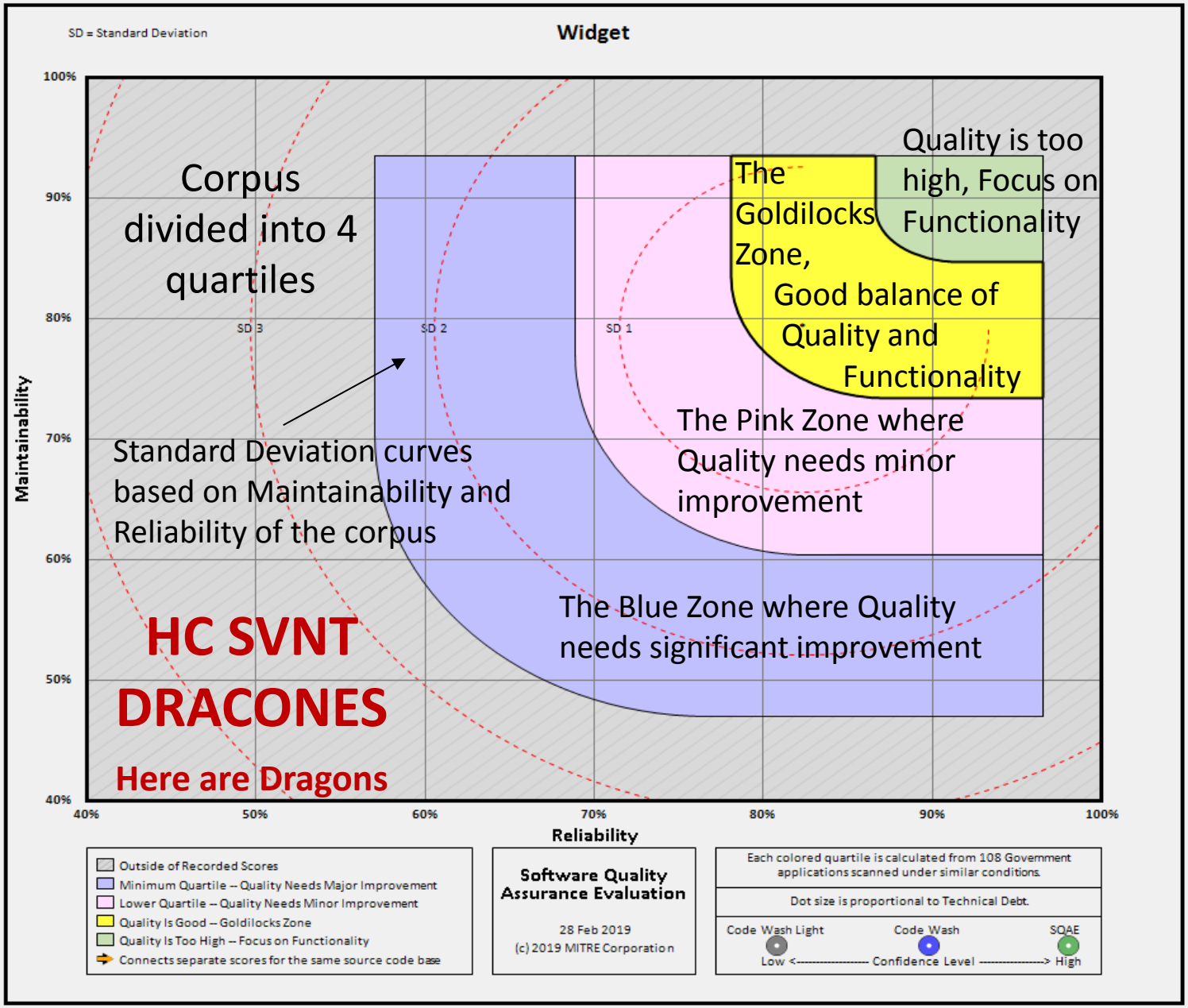
Whew!

More Data Please

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

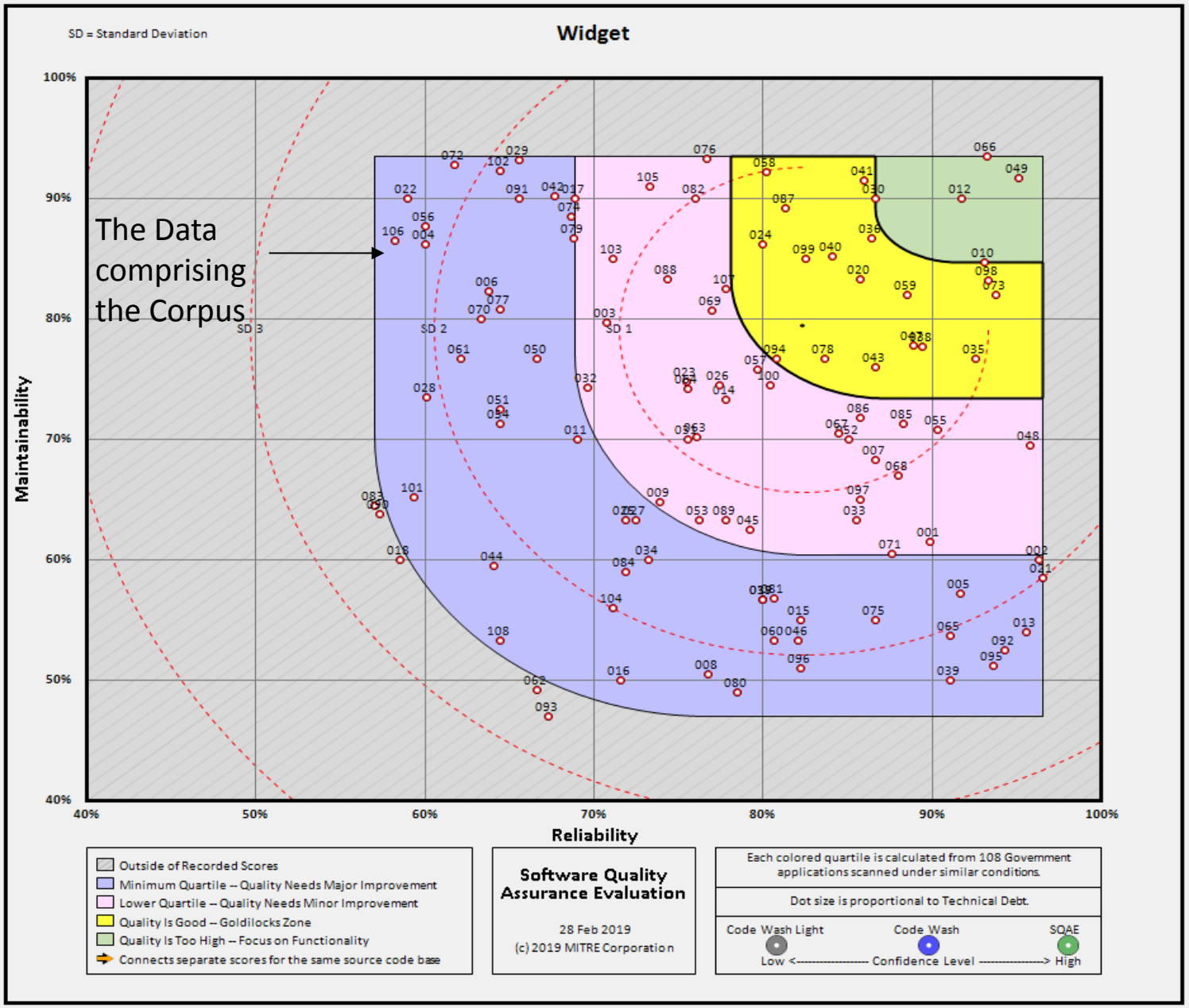
What Does 100+ evaluations tell us?

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

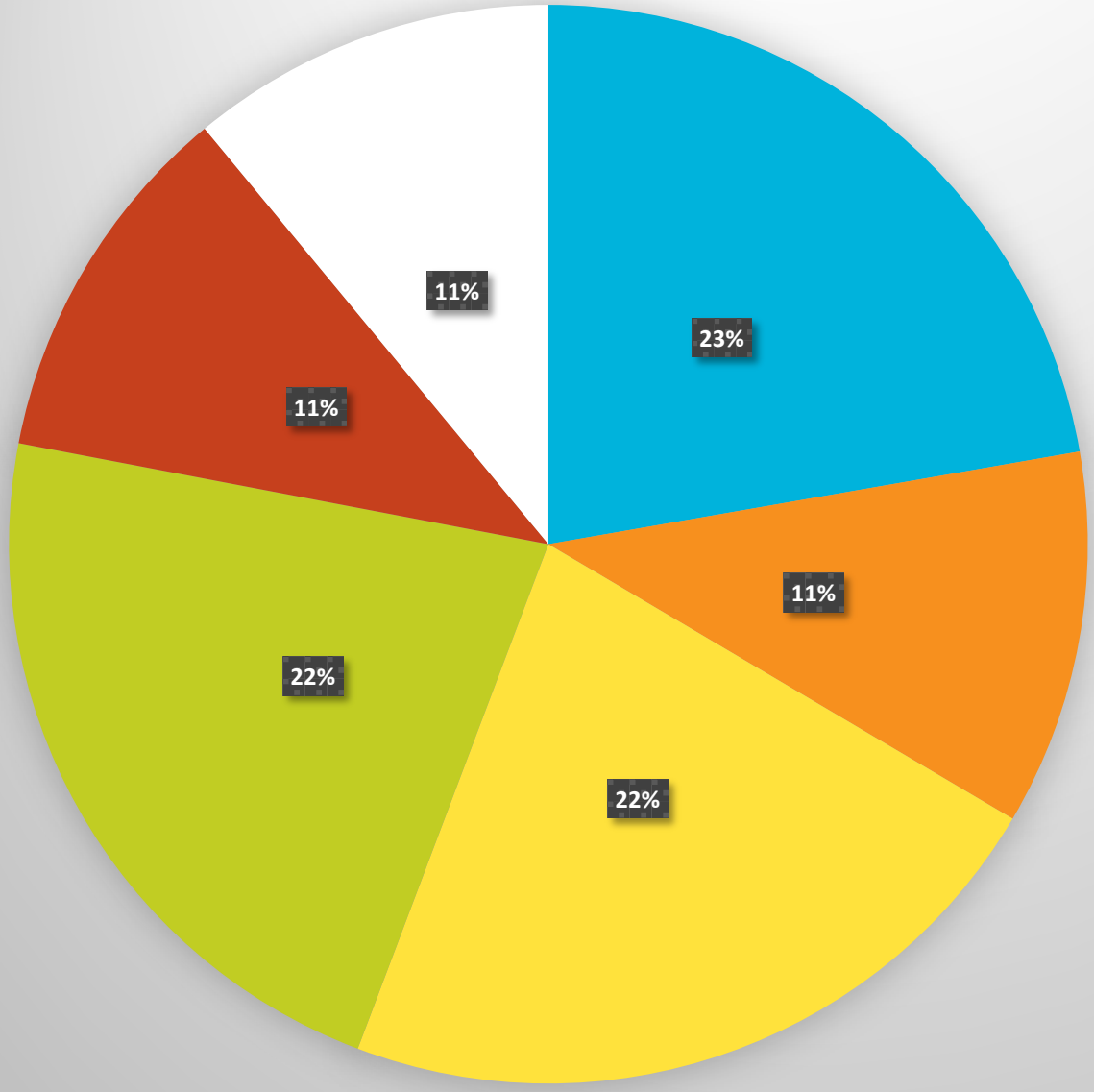
MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

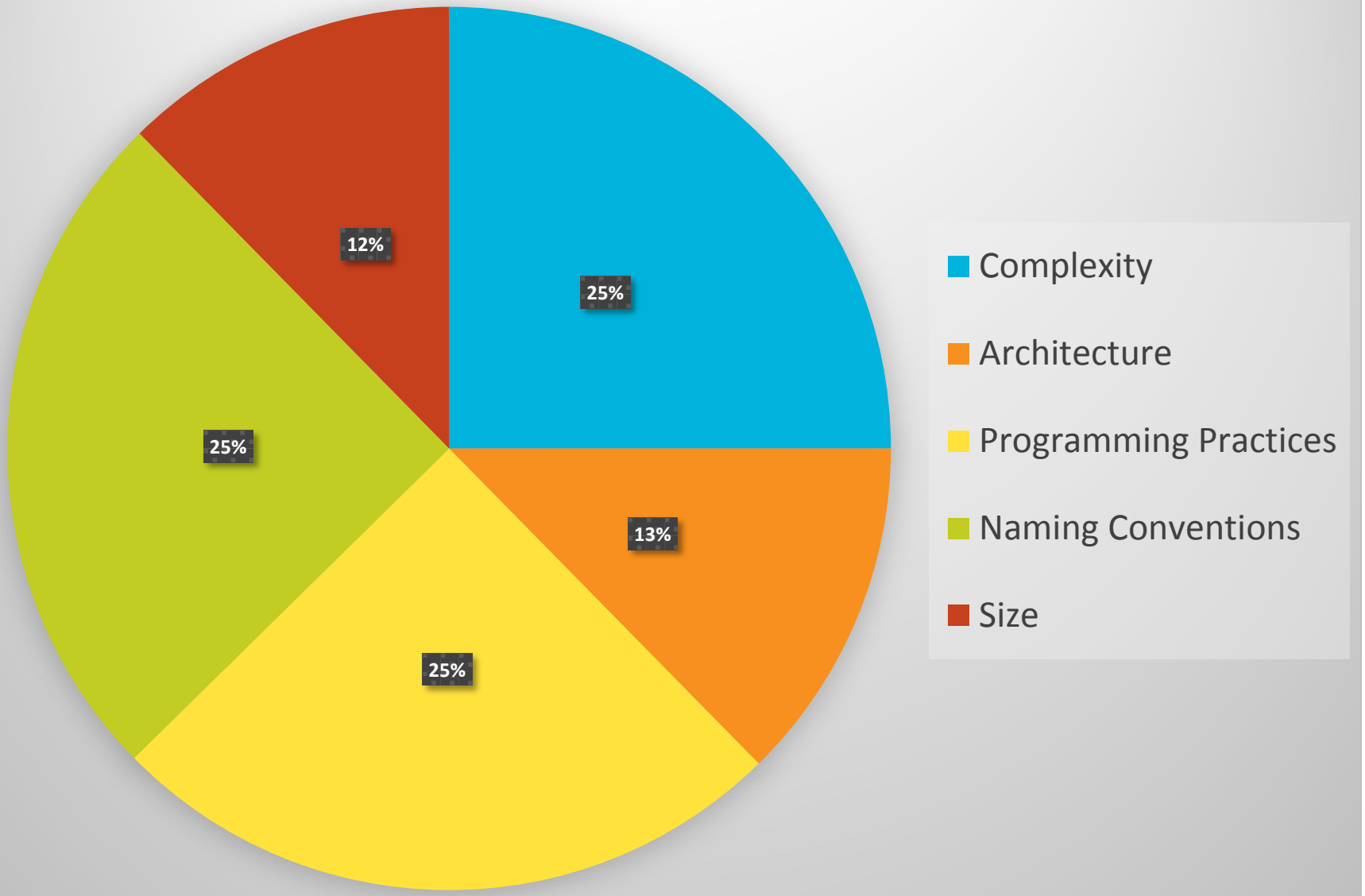
MITRE

Maintainability



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Reliability

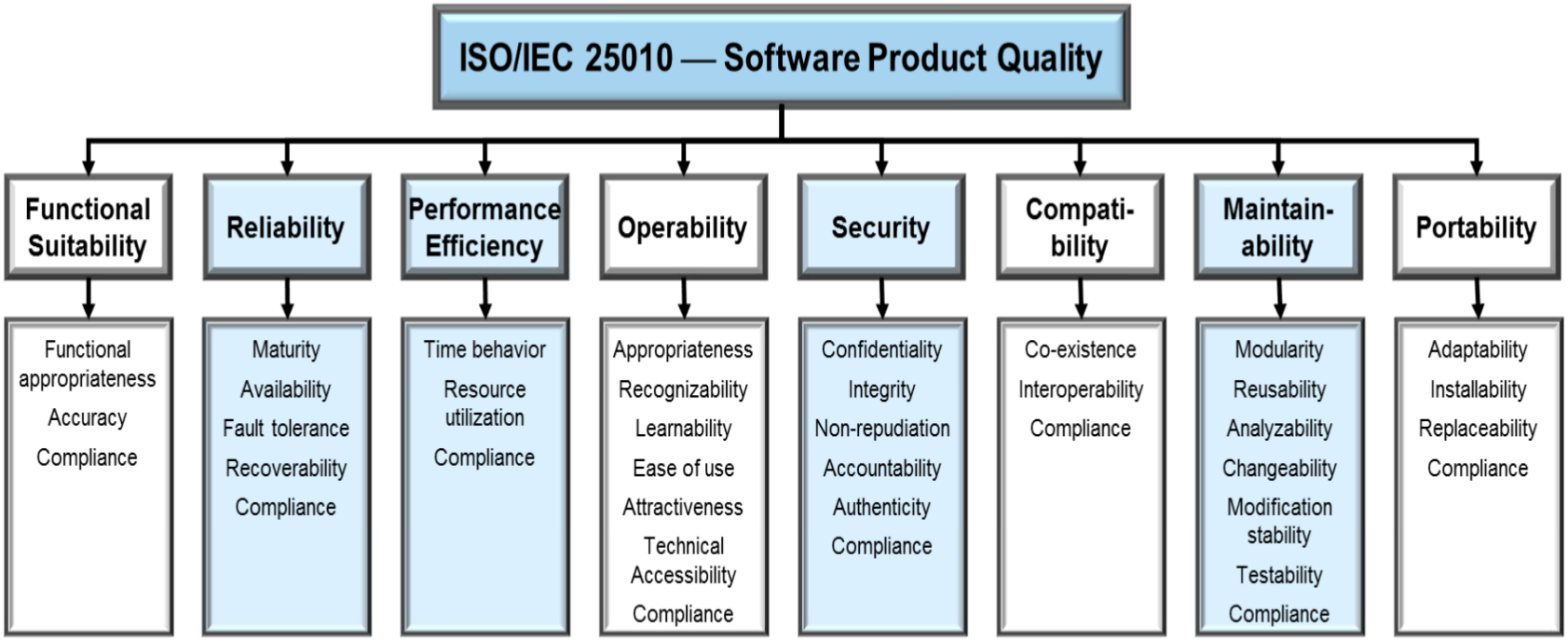


© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

CISQ Conforms/Supplements ISO 25000 standards

- ISO/IEC 25010 defines a software product quality model of 8 quality characteristics
- CISQ conforms to ISO/IEC 25010 quality characteristic definitions
- ISO/IEC 25023 defines measures, but not automatable or at the source code level
- CISQ supplements ISO/IEC 25023 with automatable source code level measures



CISQ automated structural quality measures are highlighted in blue

© 2019 Consortium for IT Software Quality (CISQ) www.it-cisq.org

Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

True or False?

- 79% of the software vulnerabilities that exist in the wild start off as quality issues in the code during the development phase of the Software Development Life Cycle.
- It costs $1 to fix a bug in the Development Phase, $100 to fix it in the Test Phase, and $1,000 to fix it in the field.
- The dependent libraries I use can be vulnerable even if I do not use the vulnerable functions.

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

True or False?

- As Code Size (Software Lines of Code : SLOC) increases the software becomes harder to maintain
- As Maintainability increases (becomes easier to maintain) Reliability increases (becomes more reliable)
- As Maintainability and Reliability increase so does Scalability
- Scanning source code does not improve the quality of the source code
- 80% of the cost of a project over its entire lifecycle is expended in the Maintenance Phase.

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

True or False?

- Agile Development Methodology is better than Waterfall
- Java is the best language to use for DoD programs.
- Technical Debt is just bad

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

The Envelope Please?

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

79% of the software vulnerabilities . . .

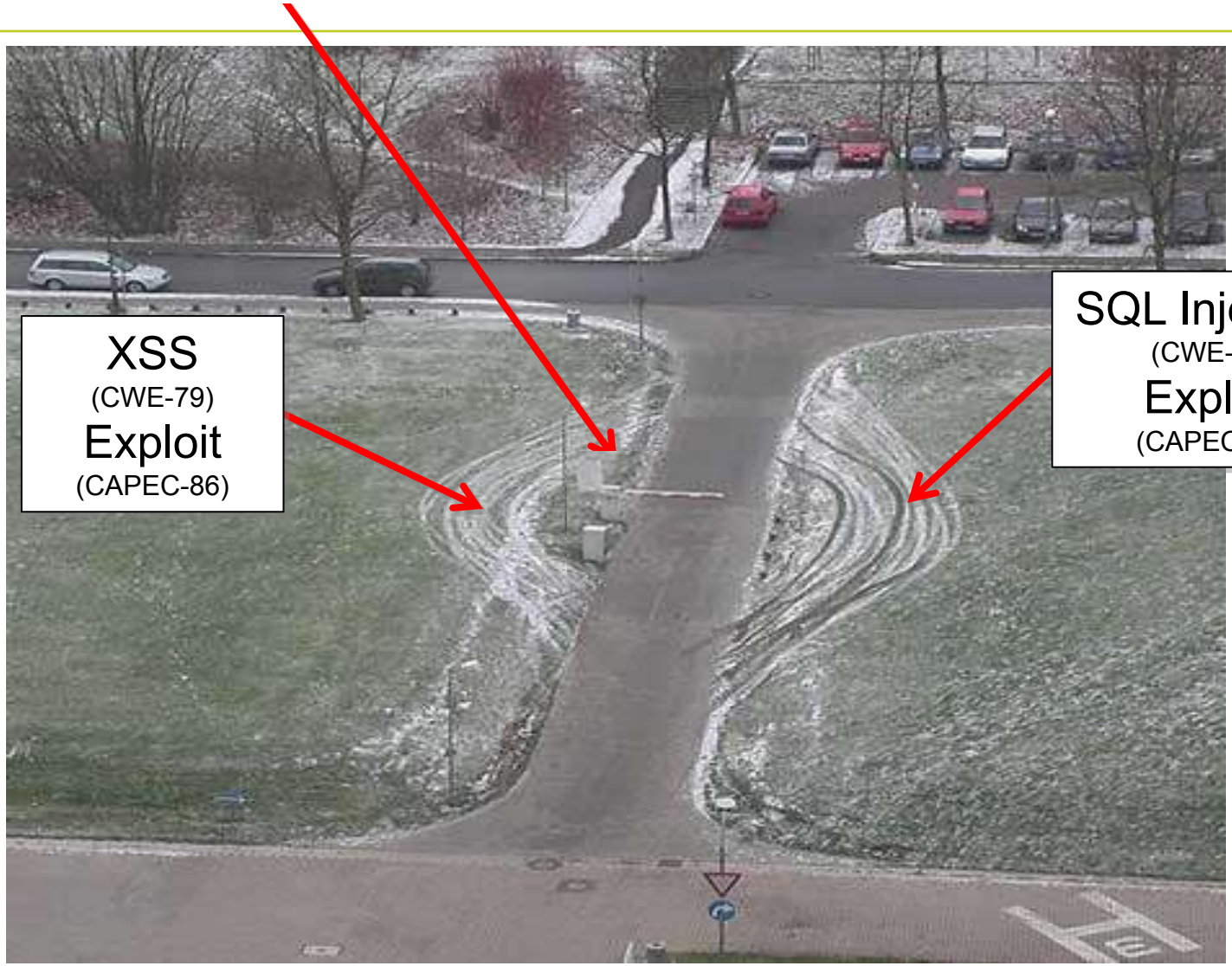
79% of the software vulnerabilities that exist in the wild start off as quality issues in the code during the development phase of the Software Development Life Cycle.

True, so far…

- **A review of 350 CWEs identified the root cause of 276 CWEs was a quality issue in the code that should have been contained in the Development Phase of the SDLC**
- **There are over 1000 CWEs**
- **Our data has not been fully analyzed, yet (we are at 35%)**
- **DAU claims 84% . . . I have not seen where they get this number from**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Security Feature



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

It costs $1 to fix a bug . . .

It costs $1 to fix a bug in the Development Phase, $100 to fix it in the Test Phase, and $1,000 to fix it in the field.

Actually, according to Caper's Jones*

- **Typical data for cost per defect varies from study to study but resembles the following pattern circa 2013:**
  - Defects found during requirements = $250
  - Defects found during design = $500
  - Defects found during coding and testing = $1,250
  - Defects found after release = $5,000
- **While such claims are often true mathematically, there are three hidden problems with cost per defect that are usually not discussed in the software literature:**
  1. Cost per defect penalizes quality and is always cheapest where the greatest numbers of bugs are found.
  2. Because more bugs are found at the beginning of development than at the end, the increase in cost per defect is artificial. Actual time and motion studies of defect repairs show little variance from end to end.
  3. Even if calculated correctly, cost per defect does not measure the true economic value of improved software quality. Over and above the costs of finding and fixing bugs, high quality leads to shorter development schedules and overall reductions in development costs. These savings are not included in cost per defect calculations, so the metric understates the true value of quality by several hundred percent.
- **The cost per defect metric has such serious shortcomings for economic studies of software quality that a case might be made for considering this metric to be a form of professional malpractice for economic analysis of software quality**

* http://www.ifpug.org/Documents/Jones-CostPerDefectMetricVersion4.pdf

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

The dependent libraries I use can . . .

The dependent libraries I use can be vulnerable even if I do not use the vulnerable functions.

True, BUT . . . Do you really want to accept that risk?

- **If there is executable code in your deliverable, it *can be executed* even if you do not call that function**

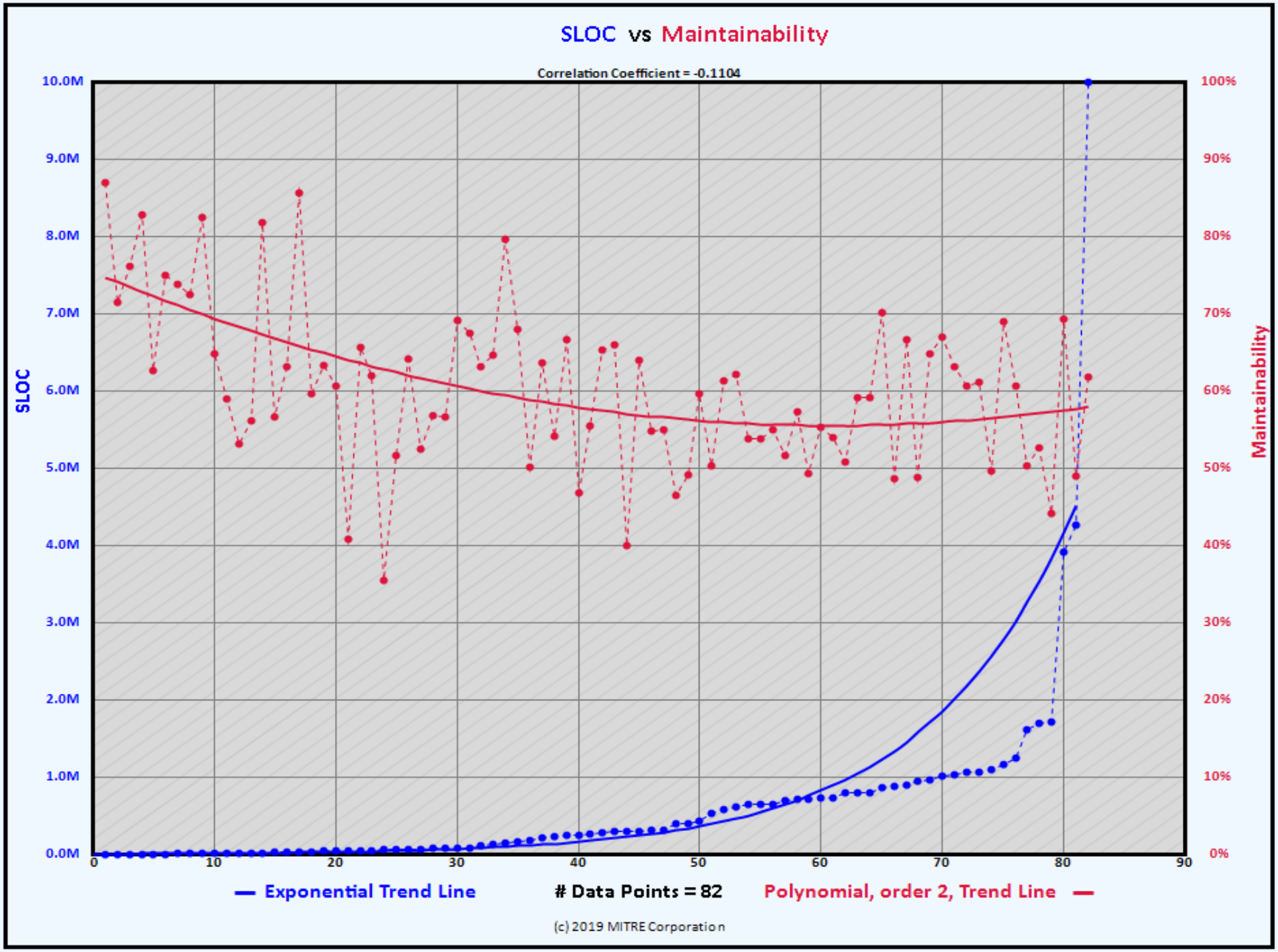
© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

SLOC vs Maintainability

As Code Size (Software Lines of Code : SLOC) increases the software becomes harder to maintain

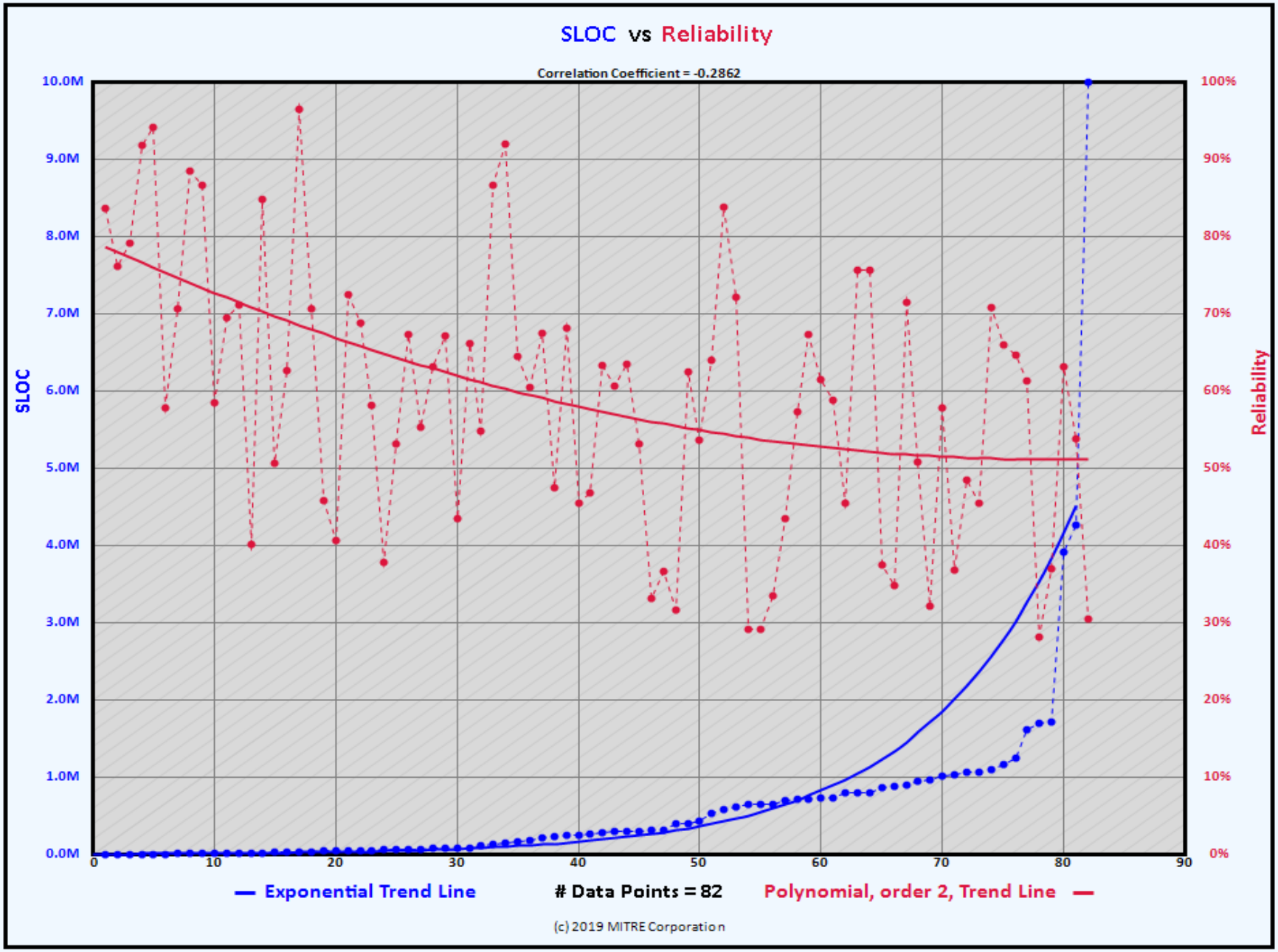
- **True**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

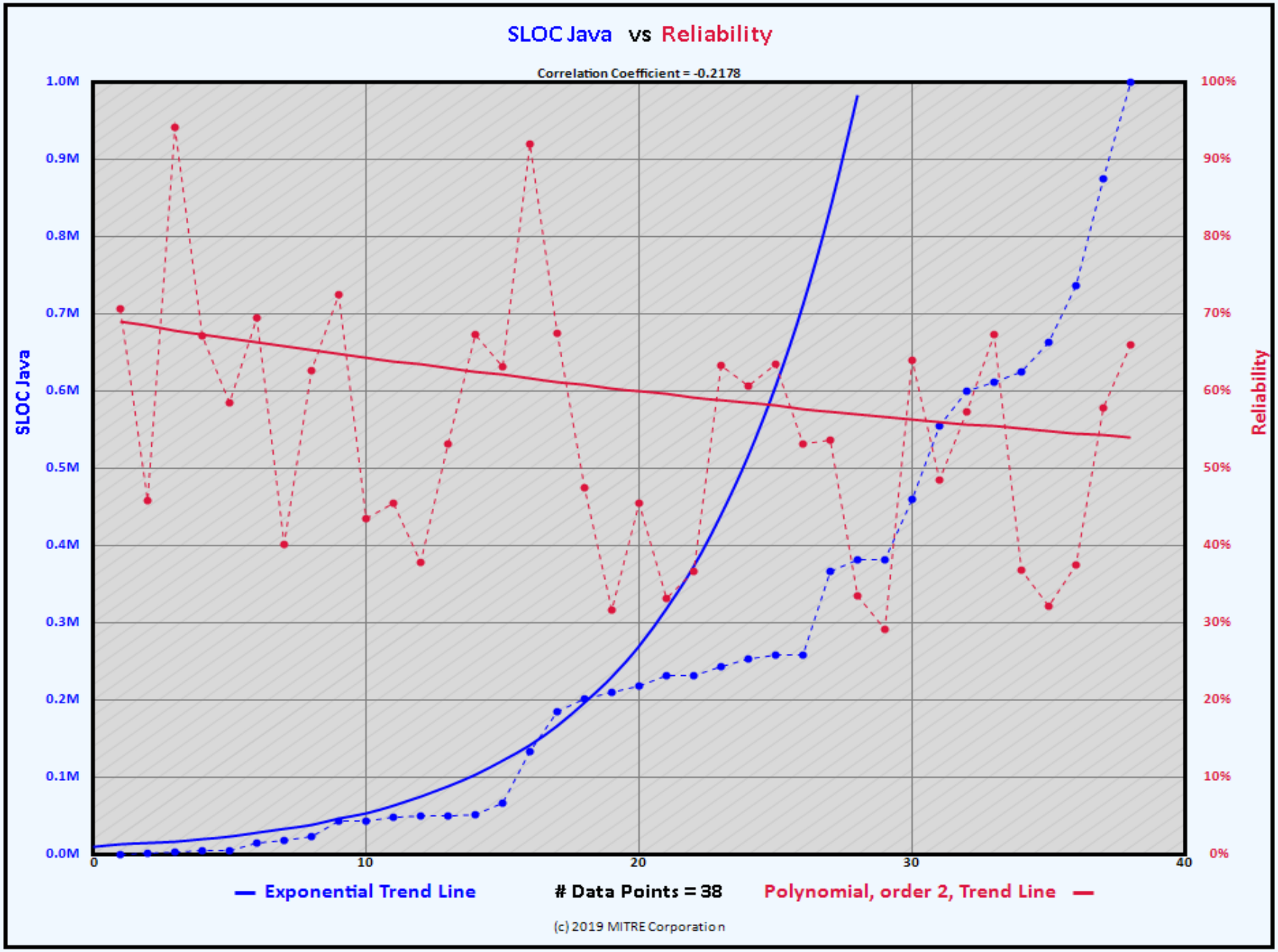
MITRE

Java SLOC vs Reliability

Java is the best language to use for DoD programs

- **False**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

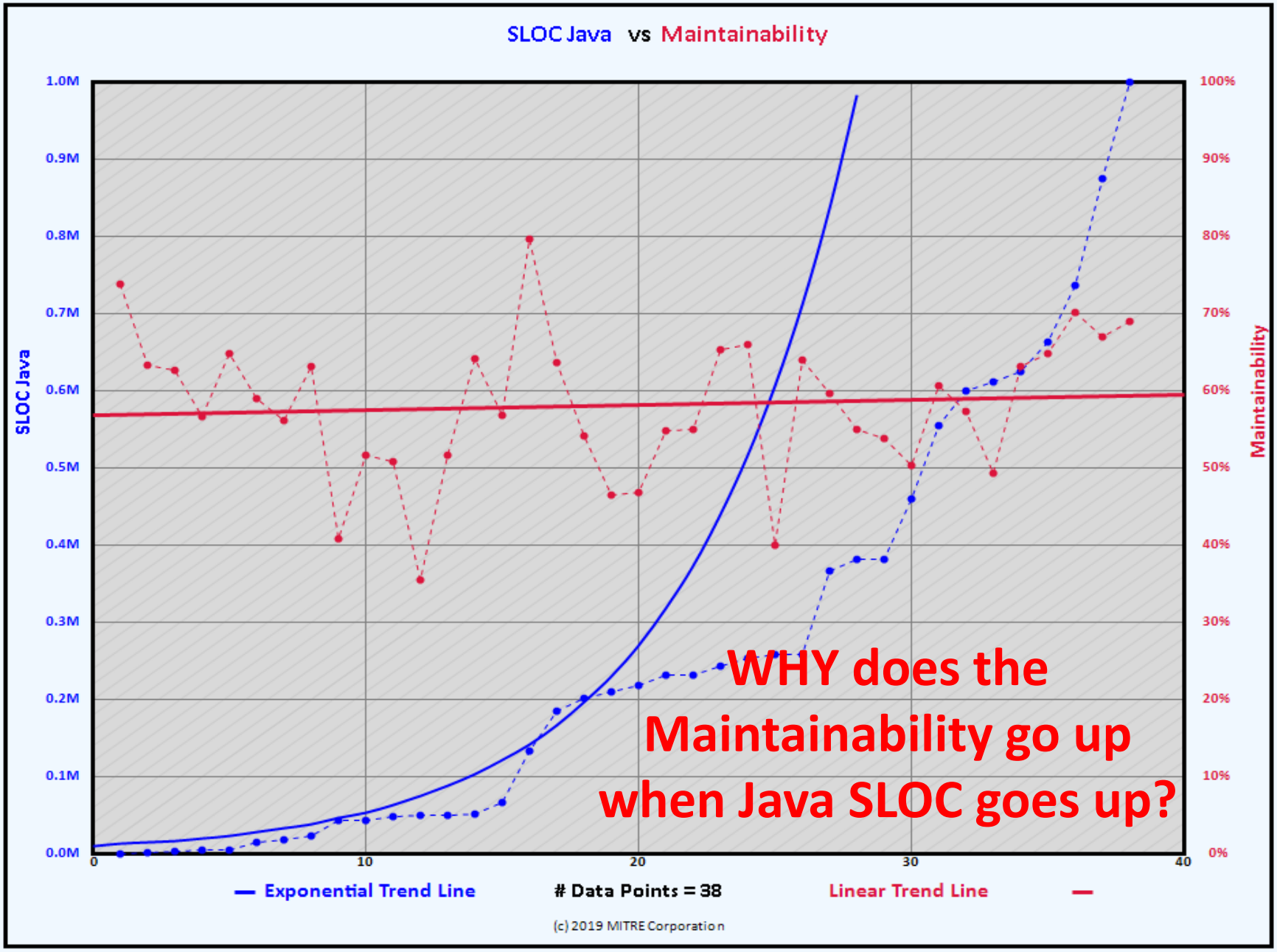
MITRE

Java SLOC vs Maintainability

Java is the best language to use for DoD programs

- **False**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Why Maintainability Increases as Java SLOC increases

1. There are only 38 samples in the corpus. But that should be statistically significant
2. You must look at the Java code to better understand this abnormal trend
3. Looking at the Java code, within the DoD, we see that most of the Java code is used to tie together various libraries and frameworks
4. We have gotten very good at writing Glue-Code in Java
5. Consider that Java Libraries are chock full of knowns vulnerabilities, well….

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

We Need More Data!

- I have asked CAST Software if they will share with me their database of Java code scanned within the DoD to add additional data points to this graph to see if it settles into an expected indicator that as SLOC increases that Maintainability decreases.

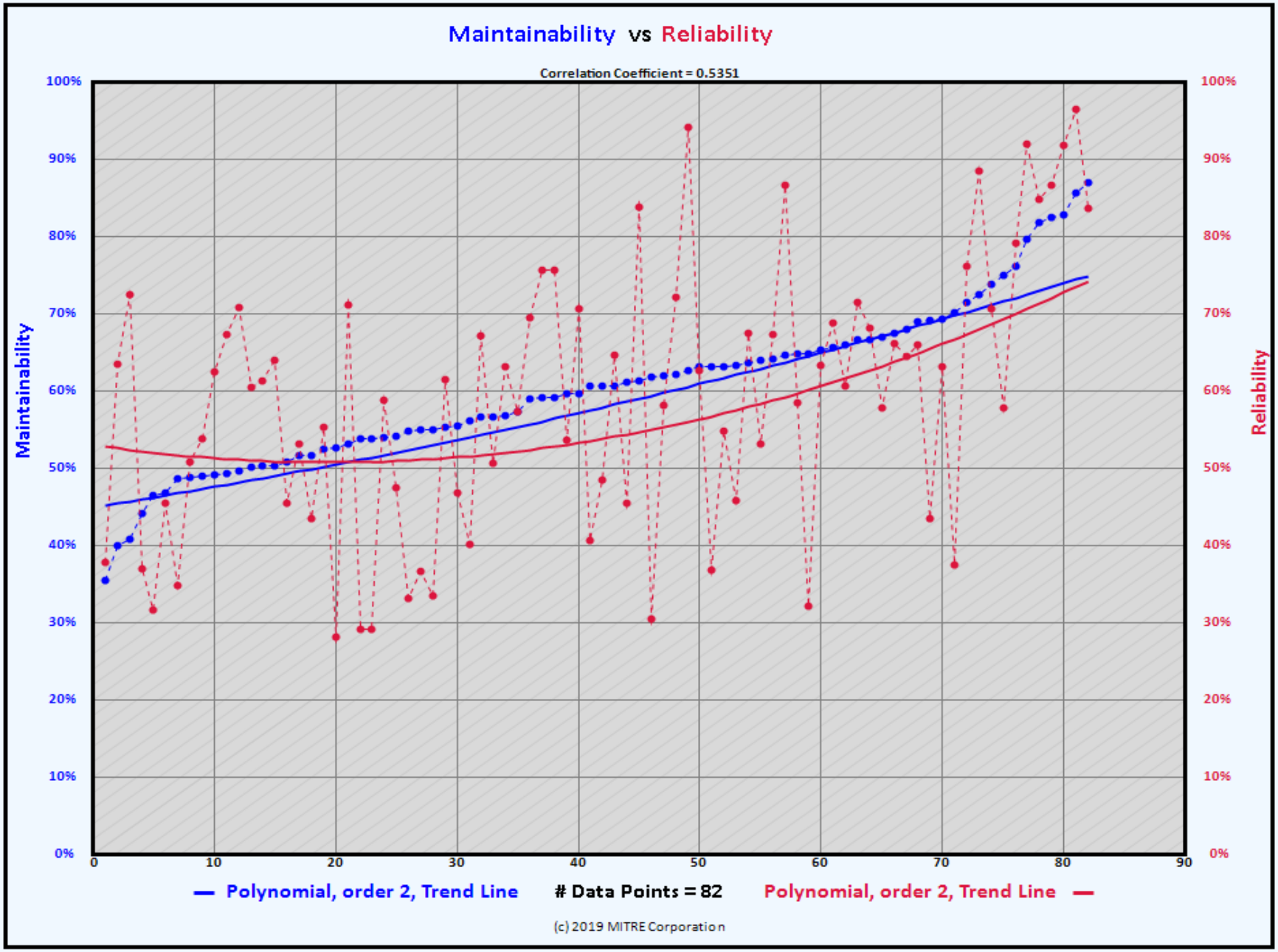
© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Maintainability vs Reliability

As Maintainability increases (becomes easier to maintain) Reliability increases (becomes more reliable)

- **True**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

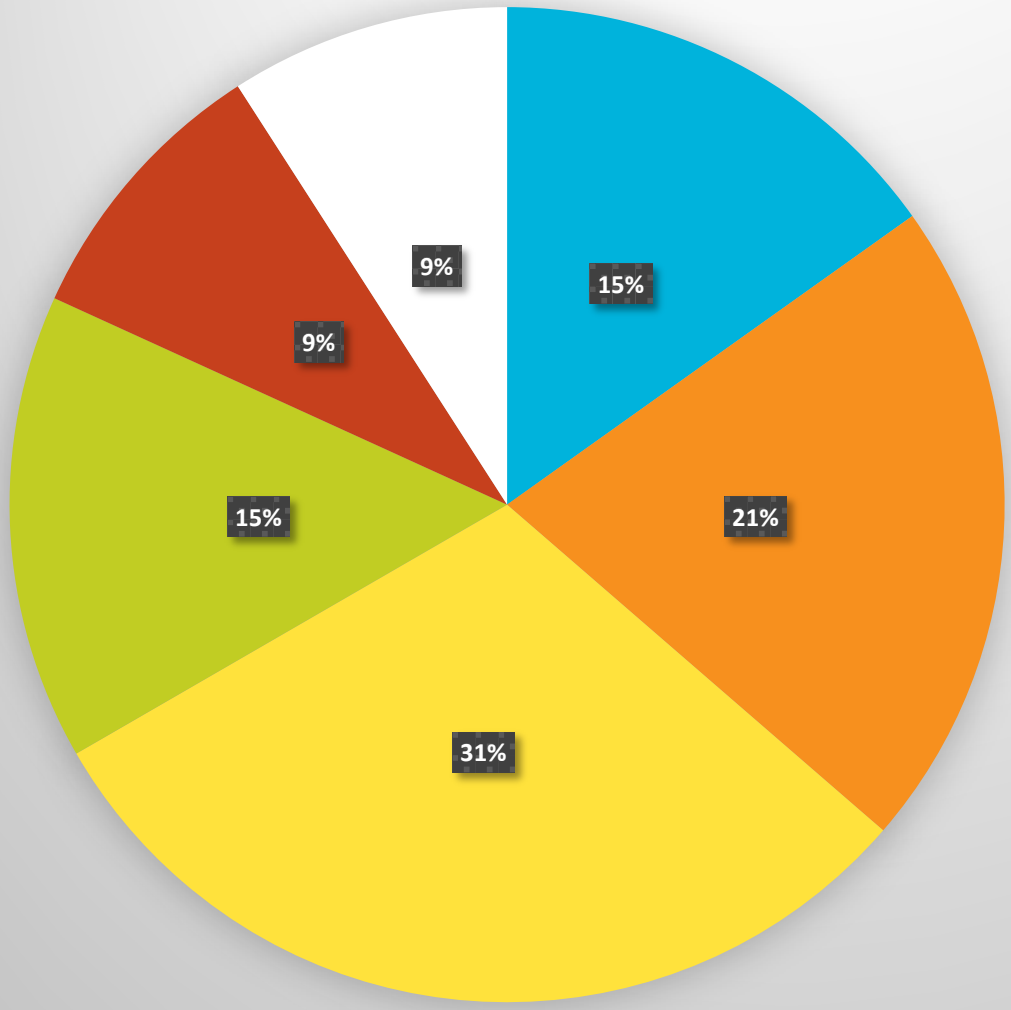
Maintainability + Reliability vs Scalability

As Maintainability and Reliability increase so does Scalability

- **Unknown**
- **Why?**
  - What is the industry accepted definition of Scalability in software?

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

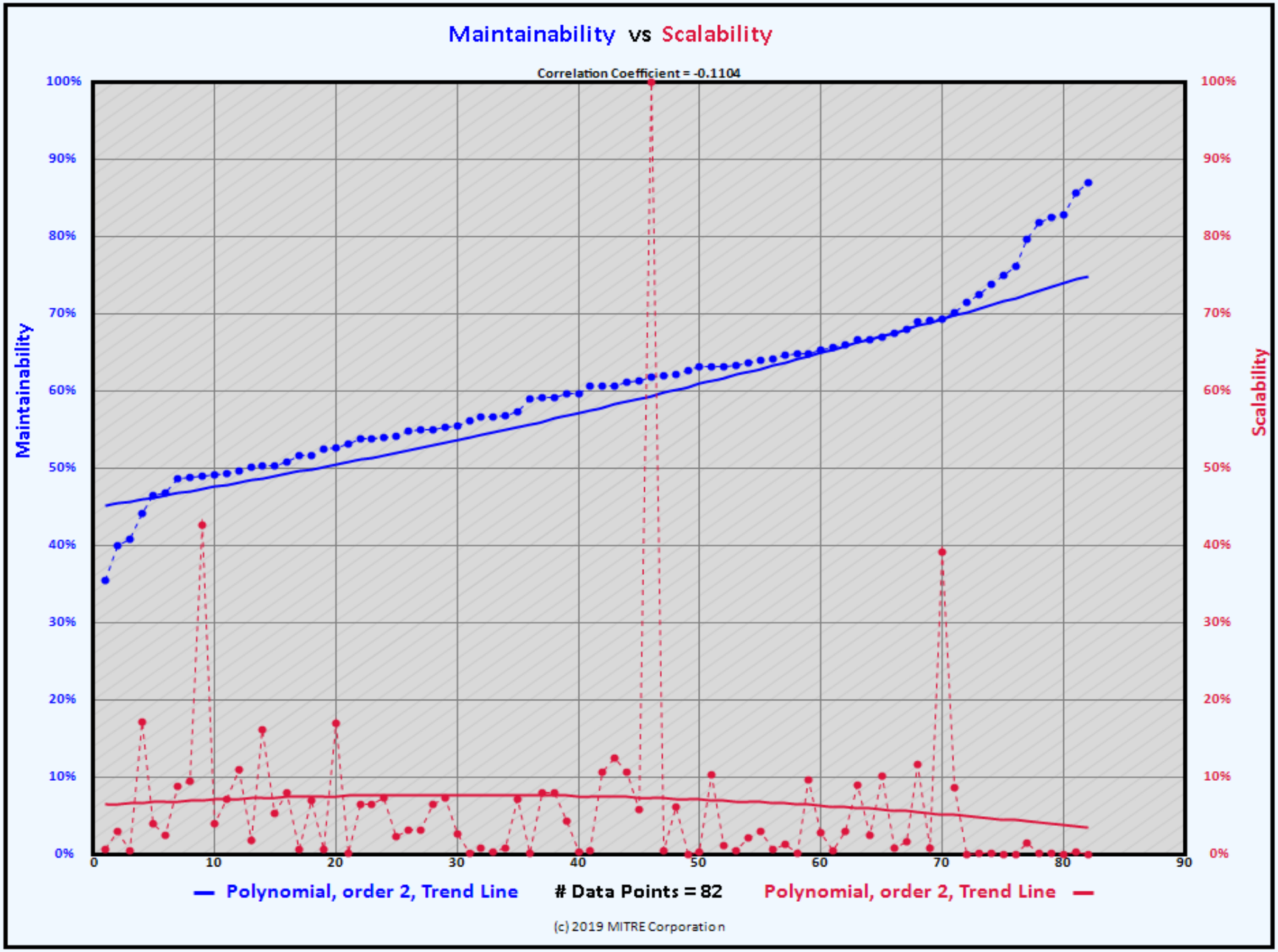
Scalability (proposed)



- Complexity
- Architecture
- Programming Practices
- Naming Conventions
- Documentation

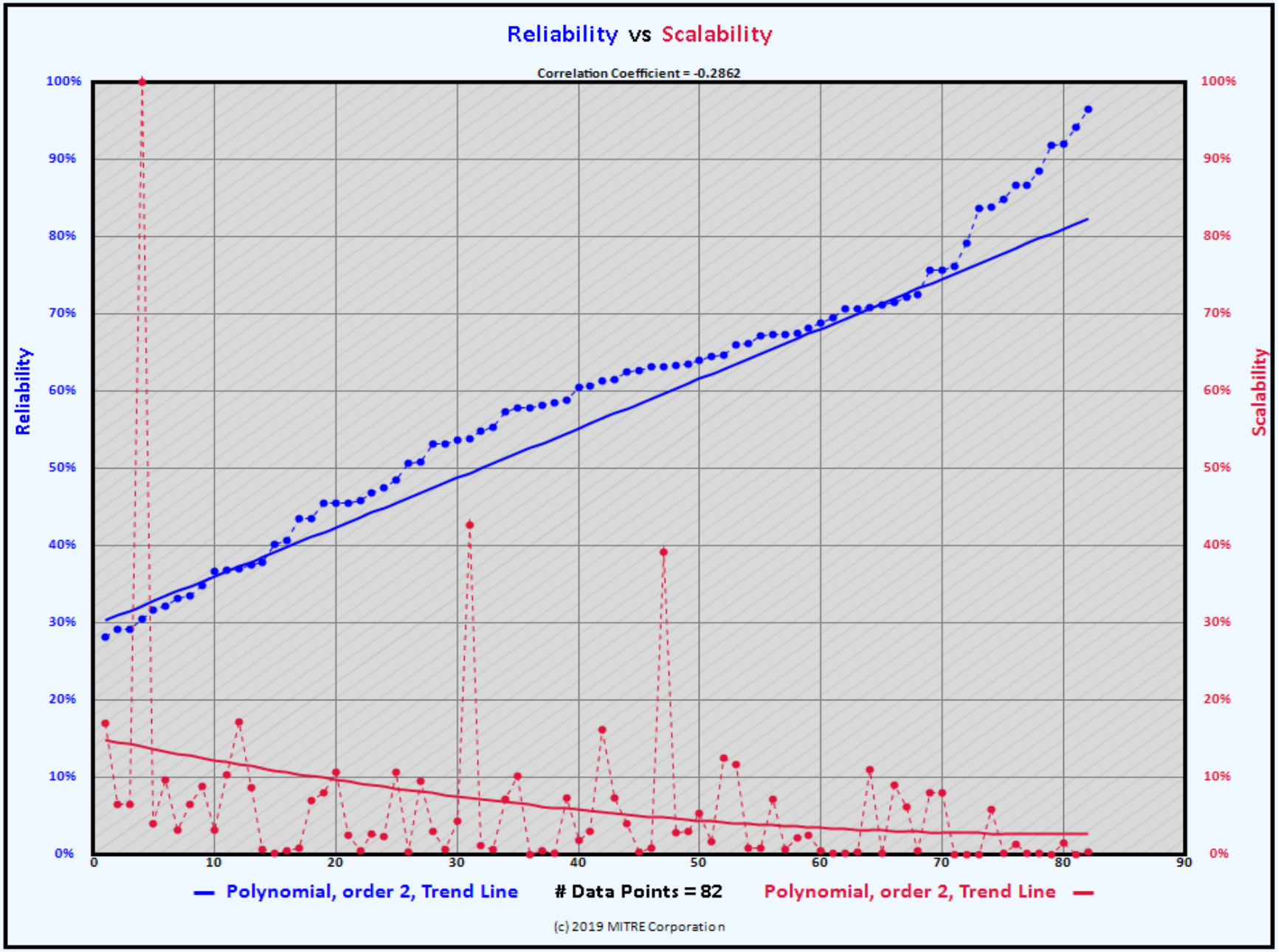
© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



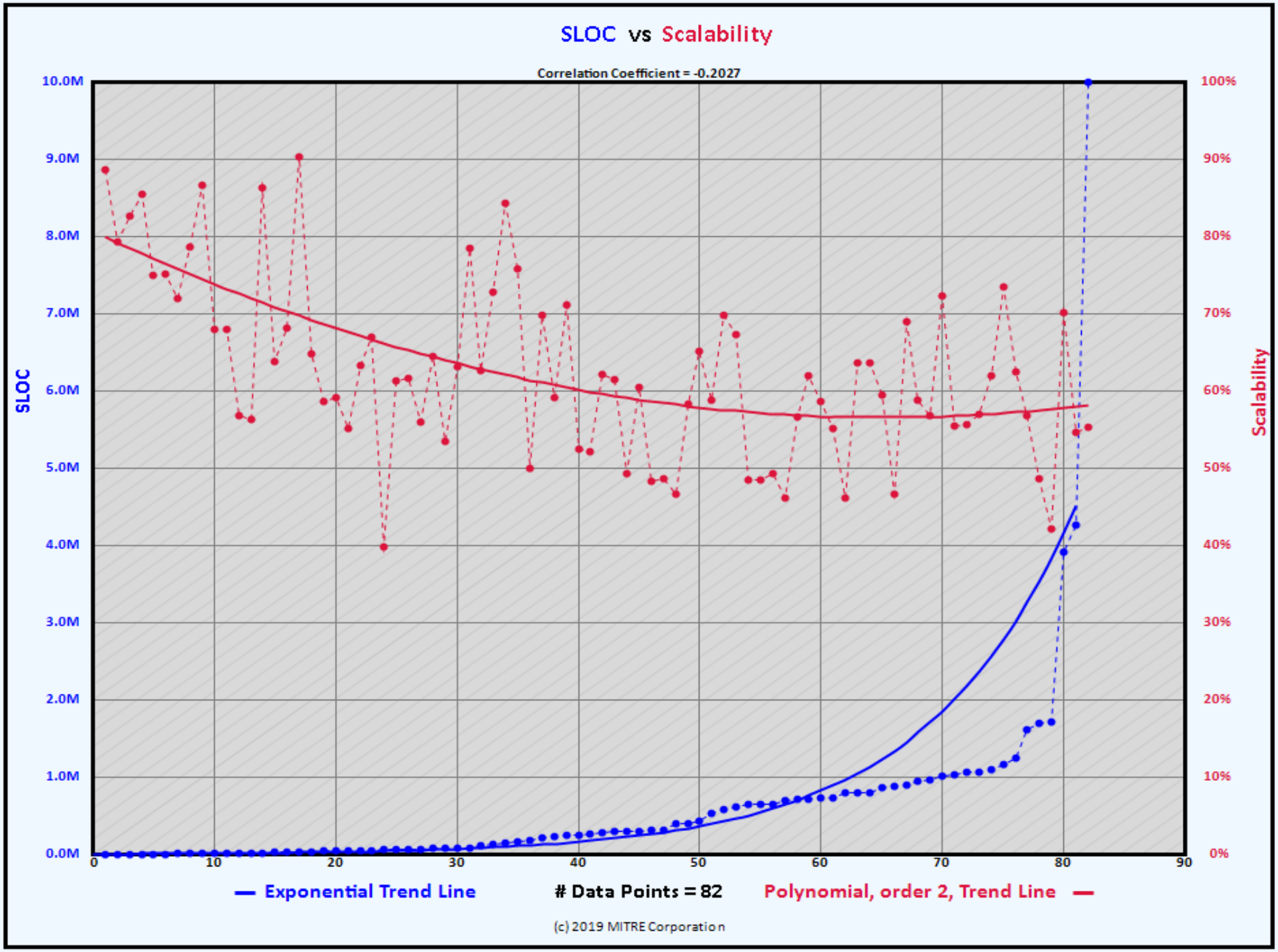
© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

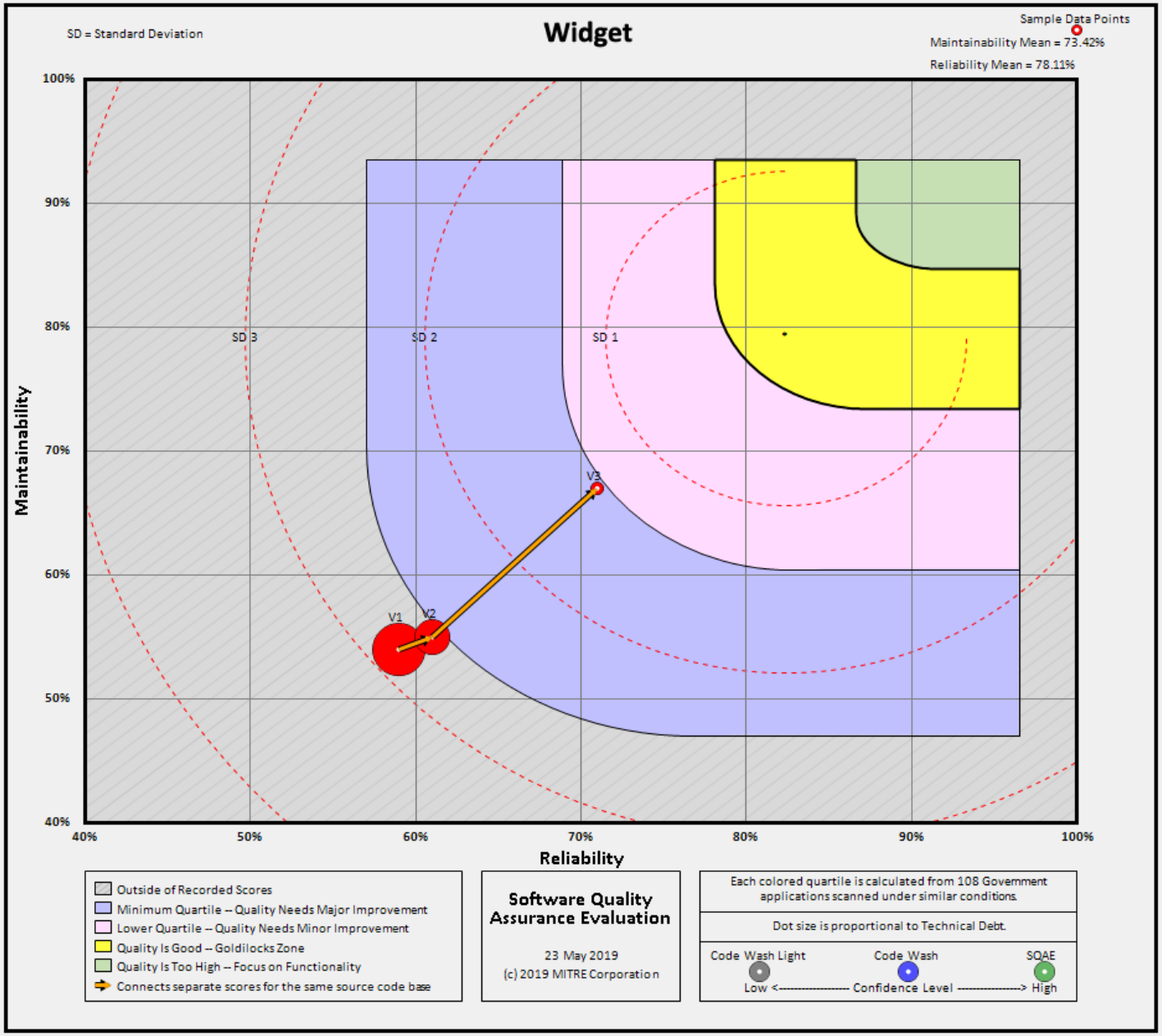
Scanning vs Quality

Scanning source code improves the quality of the source code

- **False**

Yes, it is a trick question

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Cost vs SDLC

80% of the cost of a project over its entire lifecycle is expended in the Maintenance Phase

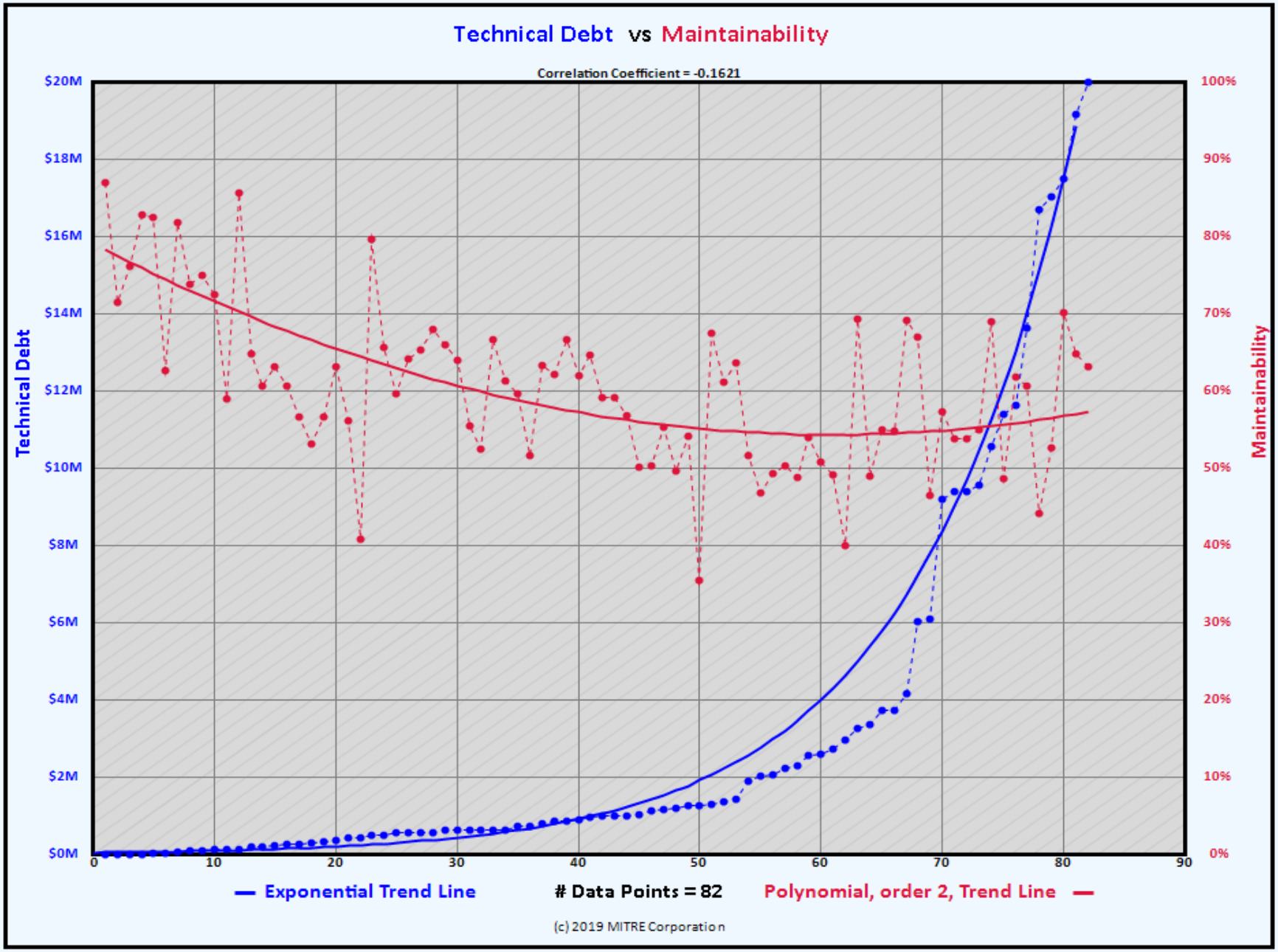
- **True**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Technical Debt is Just Bad

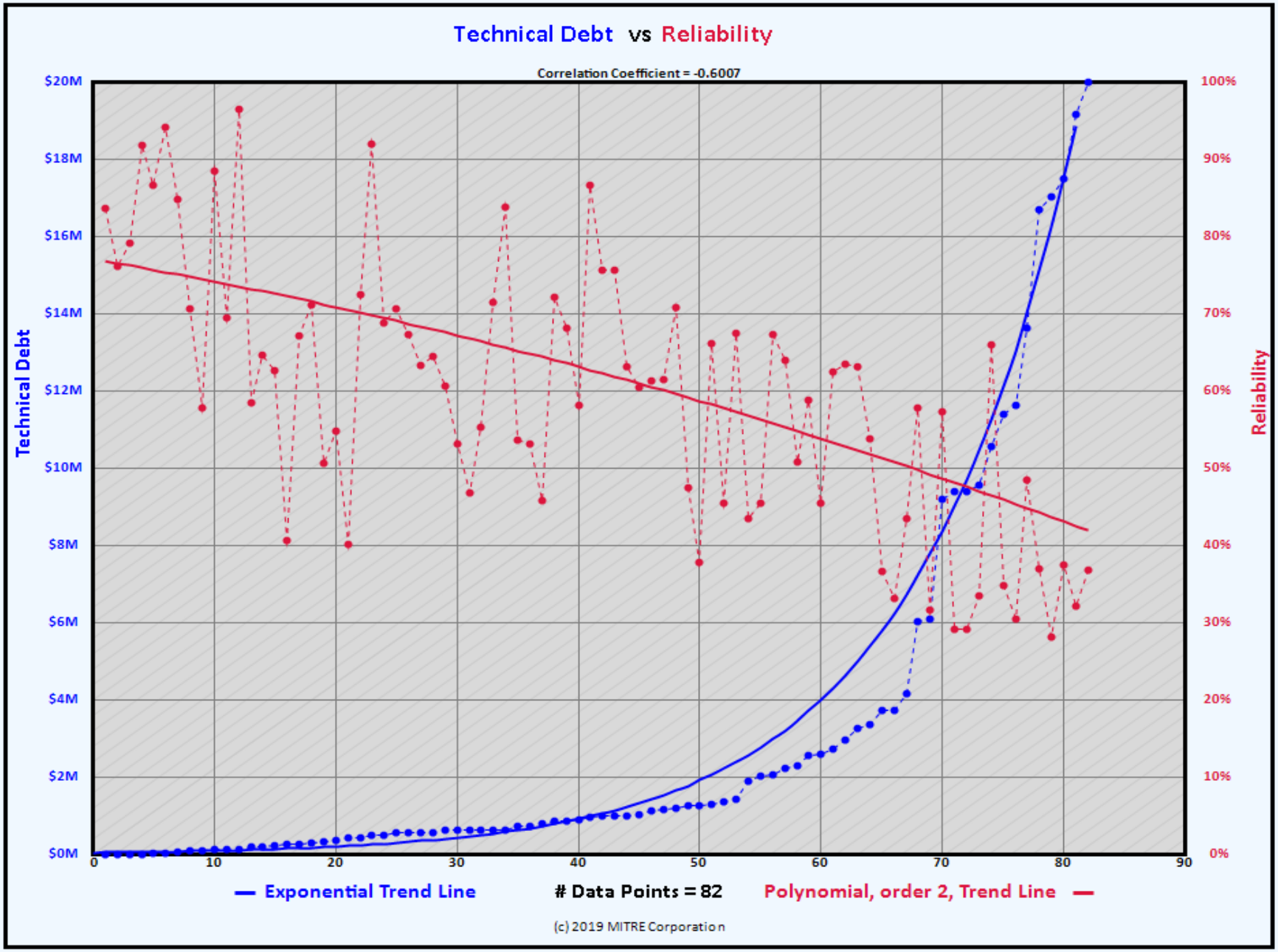
- **No argument here**

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2



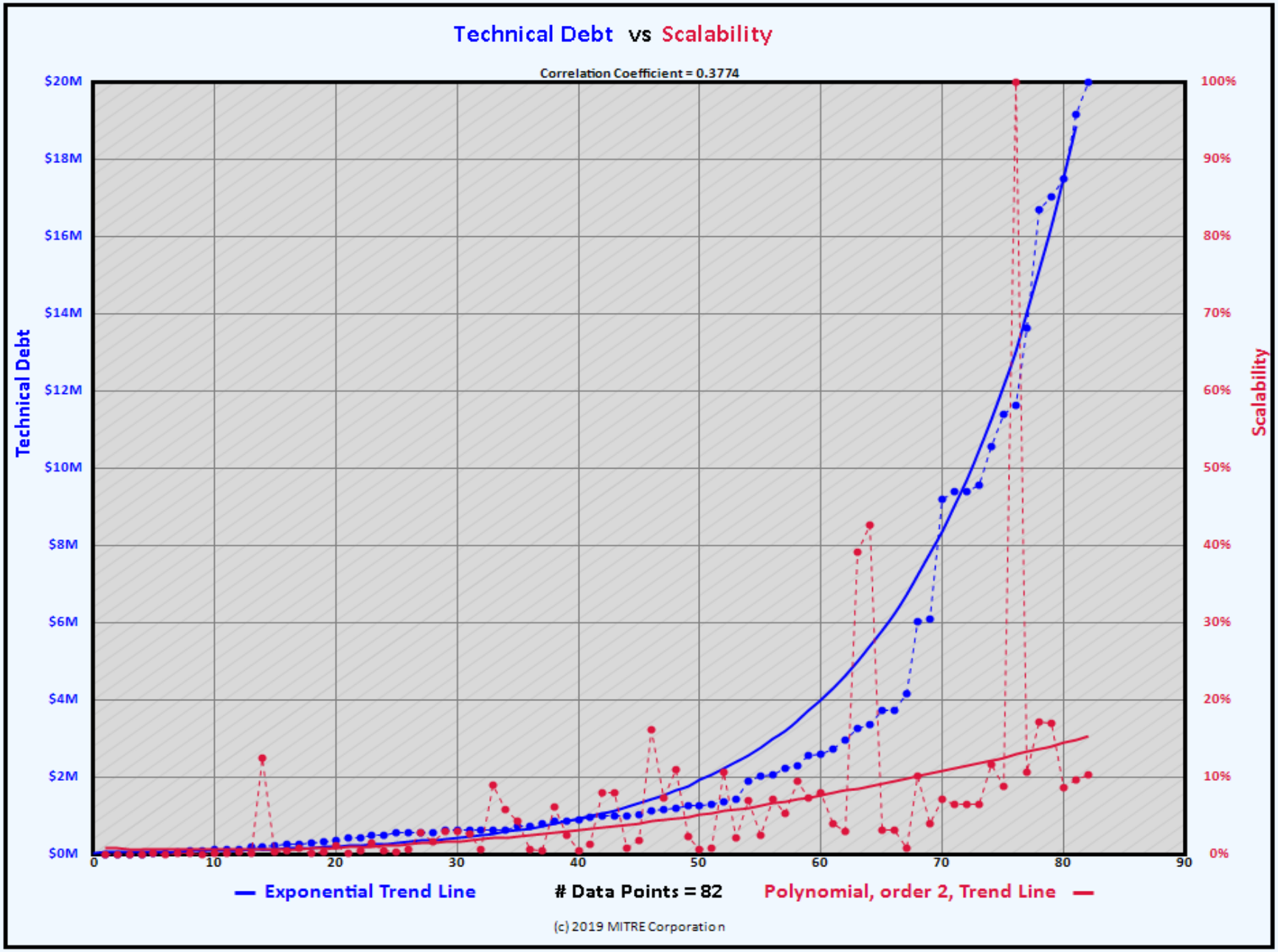
© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE



© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

Agile vs the CVE Count

Wait!

The Start of the Agile development Model

CVEs
1/1999 – 1/2019

While Agile started then, it has nothing to do with the number of vulnerabilities discovered AND fixed!

During the reign of Agile the Vulnerability rate significantly increased!

We can

say that during the reign of Agile we found and fixed more vulnerabilities, but that had nothing to do with Agile …

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

MITRE

What is Next?

- Collect more data
- Publish a paper on the data we have and again when we have more data

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2

Questions?

© 2019 The MITRE Corporation. All rights reserved. Approved for Public Release; Distribution Unlimited. Case No: 19-01876-2