The $$ of Poor SW Quality
In the US: A 2020 Report

Herb Krasner
hkrasner@utexas.edu
Date: January 27, 2021
Abbreviations Used

- CPSQ = Cost of Poor Software Quality
- B = billion
- T = trillion
- US = United States of America
- LOC = lines of code
- SW = software
- OSS = Open Source SW
- FP = function point (another measure of SW size)
Introduction to the report

• **Purpose:** to inform and inspire our readers to seek CPSQ knowledge within their own organizations

• **Method:** a unique analysis, synthesis and extrapolation of 88 existing sources of available online information, mixed with some expert knowledge about software and its quality
  – Builds on the 2018 report – basic definitions there –> what is SW quality, Cost of SW Quality model, good vs poor SW quality

• **Result:** a 1st order approximation of the magnitude of this huge and somewhat unrecognized problem
Iceberg Model of CPSQ

Hidden Costs = 6 to 50 times Observable Costs

Direct/Observable Costs:
- Stock loss/lawsuits/lost revenues
- Service outages
- Warranties/Concessions
- Customer problem reports

Indirect/Hidden Costs:
- Delays
- Overtime
- Fixing bugs
- Off track projects
- Technical debt
Summary of Cost Estimates

Technical Debt
$1.31 T (principal only)

Operational Failures
$1.56 T

Unsuccessful Dev. Projects
$260 B

Legacy Systems
$520 B

Finding & fixing defects
$607 B

Cybersecurity Failures
(incl. data breaches)

US GDP for 2020 is ~$20 T
US IT labor base for 2020 is ~$1.4 T

Total CPSQ - $2.08 T

Legacy Systems
$520 B

Operational Failures
$1.56 T

Unsuccessful Dev. Projects
$260 B

Finding & fixing defects
$607 B

Cybersecurity Failures
(incl. data breaches)

US GDP for 2020 is ~$20 T
US IT labor base for 2020 is ~$1.4 T

Total CPSQ - $2.08 T

© 2021 Consortium for Information and Software Quality (CISQ) www.it-cisq.org
Operational Failures: $1.56 Trillion

The Tricentis Software Fail Watch, 5th Ed. reported 606 major software failures from 2017, causing a total loss of $1.7 trillion in assets at 314 companies. This averages out to $2.8 billion per failure.
Failure causality: software flaws

Trends that magnify the impact of software flaws, driving failure costs up:
• 100+ billion new LOC produced worldwide each year -> 25 bugs per 1000 LOC injected on average
• 96 zettabytes of digital data now stored (up from 16 in 2016)
• Growth of cybercrime – ransomware in US cost $9B; $20B worldwide in 2021
• Increasing Digital Transformation: spreading the effects of a software malfunction across the entire value chain.
• Growth of Systems of Systems: expanding complexity exponentially and concealing the triggers for huge failures in a thicket of cross-system interactions.
• Increased Competition: especially online, has prioritized speed-to-business over operational risk and corrective maintenance costs, a huge gamble for systems not designed to expect and manage failures.

1 zettabyte is equal to 1 trillion gigabytes

Broad Recommendations:
• Prevent bugs, flaws, weaknesses, vulnerabilities from being created and fielded
• Find and fix bugs early
• Measure quality
• Adopt high quality development practices
• Analyze potentially flawed components (e.g. OSS)
Cybercrime losses increasing

Cybercrime will cost companies worldwide an estimated $6 trillion annually by 2021, up from $3 trillion in 2015.

This is the greatest transfer of economic wealth in history.¹

¹ Embroker, 11/20/2020
Cybercrime losses increasing

US FBI IC3 Cybercrime Trends

Total complaints
- 2011: 314,246
- 2019: 467,361

Total losses (rounded to the nearest million)
- 2011: $485,000,000
- 2019: $3,500,000,000

IC3 – Internet Crime Complaint Center
Unsuccessful Projects: $260 Billion

IT Project Outcomes
Based on CHAOS 2020: Beyond Infinity Report

Percentage of IT projects

- Challenged
- Failed
- Successful
Reduce # of unsuccessful projects

- For projects of large size ($10^4$ FPs) and above, low-quality projects are 5-6X more likely to be cancelled than high-quality projects.
- Project cancellation rates by size and quality level:

<table>
<thead>
<tr>
<th>Function Points</th>
<th>High Quality</th>
<th>Low Quality</th>
<th>X-factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>.02</td>
<td>.07</td>
<td>3.3</td>
</tr>
<tr>
<td>1000</td>
<td>.05</td>
<td>.16</td>
<td>3.2</td>
</tr>
<tr>
<td>$10^4$</td>
<td>.07</td>
<td>.45</td>
<td>6.4</td>
</tr>
<tr>
<td>$10^5$</td>
<td>.12</td>
<td>.65</td>
<td>5.4</td>
</tr>
</tbody>
</table>

1 FP = ~ 100 LOC

Broad recommendations:
- define what quality means for a specific project and then focus on achieving measurable results against stated quality objectives
- use known best practices & tools for achieving high quality
- don’t compromise quality for speed to operation

1. Jones and Bonsignor, 2012
Problem: after decades of operation, they may have become less efficient, less secure, brittle, incompatible with newer technologies and systems, and more difficult to support due to loss of knowledge and/or increased complexity or loss of vendor support.

- 70-75% of the IT budget
- 80% of the cost of ownership
- Slight decline in CPSQ due to shifting priorities, work force losses
Legacy systems: broad recommendations

Modernization is not always straightforward. The approach depends on the priority of problems to be solved – functionality, performance, obsolete technology, inflexible architecture.

Several strategies are available to improve CPSQ and COO going forward:

- **Encapsulate** - to hide/isolate details (create APIs and containerization)
- **Rehost** – to move systems off the mainframe and migrate them to new hardware or the cloud
- **Replatform** - to speed up with new hardware
- **Repair** – to fix the bugs, maintain
- **Refactor** - to reduce technical debt
- **Rearchitect** - to adapt to a new platform or technology
- **Rebuild** – to fine tune it
- **Replace** – with new or SaaS solutions

- All these strategies are enabled by overcoming the lack of understanding and knowledge of how the system works internally.
- Any tool which helps identify weaknesses, vulnerabilities, failure symptoms, defects and improvement targets is useful
- Benchmarking the health status of a legacy system is a good starting point.
- Detailed blueprints of system connectivity are useful for modernizing architectures that have degraded over time.
Finding and Fixing Bugs: $607 B

1. Find, record and prioritize
2. Replicate and create a test case
3. Determine root cause
4. Develop and fix
5. Try to break the fix
6. Prove the fix works
7. Repeat steps 2-6 as needed until done
8. Record the fix details
9. Distribute the fix as needed

Prevent first then focus on where the $$$ are spent in the above process to reduce CPSQ and improve productivity
Technical Debt: $1.31 Trillion + Interest

- ~1.655 trillion LOC exists worldwide and 513 billion in the US.
- code growth is now ~100 billion new LOC per year, or ~7% growth per year.
Technical Debt: $1.31 Trillion + Interest

How do you manage technical debt?
- Understand, Elevate, Repair, Repay
IBM study\(^1\) of 363 EU SW organizations

<table>
<thead>
<tr>
<th>Performance Factor</th>
<th>Top 10%</th>
<th>Bottom 10%</th>
</tr>
</thead>
<tbody>
<tr>
<td>Productivity (FPs/mo.)</td>
<td>25</td>
<td>5</td>
</tr>
<tr>
<td>Delivered quality (% defects removed)</td>
<td>&gt;95%</td>
<td>&lt;50%</td>
</tr>
<tr>
<td>Cost/Schedule Performance</td>
<td>&lt;= 10%</td>
<td>&gt;40% over</td>
</tr>
<tr>
<td>Post delivery maintenance costs (within 1st yr.)</td>
<td>&lt;1% (of total dev. effort)</td>
<td>&gt;10%</td>
</tr>
</tbody>
</table>

different behaviors

Leaders/C-Suite level
• Establish quality as a 1st-class citizen -> security+
• Ask better questions: externally and internally
• Measure SW quality & CPSQ in your organization

Teams/projects
• Strive for high performance
• Use best practices & tools
• Define & track quality objectives
• Avoid arbitrary and unrealistic schedules or constraints

Individuals
• Learn and grow a disciplined approach
• Don’t be afraid of quality metrics
• Use existing knowledge sources of bug pattern and structural quality flaws
DevQualOps Concept Model

Continuous Evolution

Agile Development Cycle

Continuous Integration

Dev to Ops and back cycle

Continuous Evolution

Configure/install

Continuous Deployment

Operations

Do not release decision

Quality engineering gates

Continuous Delivery

Validate

Continuous Quality Engineering

Preproduction

Plan

Release

Monitor/measure