Measuring and Managing Technical Debt

Dr. Bill Curtis
SVP & Chief Scientist, CAST Research Labs
Director, Consortium for IT Software Quality

The Technical Debt Metaphor

**Technical Debt** — the future cost of defects remaining in code at release, a component of the cost of ownership

- **Business Risk**
  - Opportunity cost
  - Liability from debt

- **Technical Debt**
  - Interest on the debt
  - Principal borrowed

**Opportunity cost**—benefits that could have been achieved had resources been put on new capability rather than retiring technical debt

**Liability**—business costs related to outages, breaches, corrupted data, etc.

**Interest**—continuing IT costs attributable to the violations causing technical debt, i.e., higher maintenance costs, greater resource usage, etc.

**Principal**—cost of fixing problems remaining in the code after release that must be remediated

Structural quality problems in production code


**Uses of Technical Debt Metaphor**

- **Calculating Cost Of Ownership**
- **Assessing Business Risk**
- **Estimate of Technical Debt**
- **Managing Portfolio Quality**
- **Explaining IT Cost of Quality**

**Inputs for Estimating Principal**

<table>
<thead>
<tr>
<th>Data source</th>
<th>Inputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Test results and static analysis</td>
<td>Application quality problems</td>
</tr>
<tr>
<td>Historical data on maintenance</td>
<td>Hours to correct problems</td>
</tr>
<tr>
<td>IT or supplier finance records</td>
<td>Developer's burdened hourly rate</td>
</tr>
</tbody>
</table>

Technical Debt Principal
Analyzing Structural Quality at System Level

CAST Application Intelligence Platform

Language Parsers
- Oracle PL/SQL
- Sybase T-SQL
- SQL Server T-SQL
- IBM SQL/PSM
- C, C++, C#
- Pro C
- Cobol
- COBOL
- Visual Basic
- VB.Net
- ASP.Net
- Java, J2EE
- JSP
- XML
- HTML
- Javascript
- VBScript
- PHP
- PowerBuilder
- Oracle Forms
- PeopleSoft
- SAP ABAP, Netweaver
- Tibor
- Business Objects
- Universal Analyzer for other languages

Application Analysis
Detected Violations
- Expensive operation in loop
- Static vs. pooled connections
- Complex query on big table
- Large indices on big table
- Empty CATCH block
- Uncontrolled data access
- Poor memory management
- Opened resource not closed
- SQL injection
- Cross-site scripting
- Buffer overflow
- Uncontrolled format string
- Unstructured code
- Misuse of inheritance
- Lack of comments
- Violated naming convention
- Highly coupled component
- Duplicated code
- Index modified in loop
- High cyclomatic complexity

Quality Measurements
- Performance
- Robustness
- Security
- Transferability
- Changeability

CAST Confidential

Appmarq Repository

- Industry-leading repository on structural quality
  - 745 Applications
  - 160 Companies, 14 Countries
  - 321,259,160 Lines of Code; 59,511,706 Violations

CAST Confidential
“Even when measured with a conservative formula, the amount of technical debt in most business applications is formidable... estimates of [technical debt] can be a powerful tool to aid management in understanding and controlling IT costs and risks.”

Conservative estimate: $3.61 per LOC

- 70% of Technical Debt is in IT Cost (Transferability, Changeability)
- 30% of Technical Debt is in Business Risk (Robustness, Performance, Security)
- Proportions are generally consistent across technologies
Rethinking Productivity Measurement

**Release Productivity**

\[
\text{Volume of code developed, modified, or deleted} \div \text{Total effort expended on the release}
\]

**Productivity baseline**

a value in a monotonically declining function that compares the amount of product produced to the effort required to produce it

... unless you take action

---

Technical Debt = Carry-forward Rework
Quality-Adjusted Productivity

Quality-Adjusted = Release productivity \times f(\text{Technical debt})

Release Productivity should be adjusted for:
1. Effort shifted forward for fixing functional defects added in this release
2. Effort shifted forward for fixing structural defects added in this release
3. Future effort caused by maintainability problems added in this release

Stronger relationship to:
- Architectural integrity
- Total Cost of Ownership
- Business risk

Manage Technical Debt to Manage Productivity

<table>
<thead>
<tr>
<th>IT Executives</th>
<th>Application Managers</th>
<th>Developers</th>
<th>Build/Release/QA/Al Center</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Step 1</strong></td>
<td><strong>Step 2</strong></td>
<td><strong>Step 3</strong></td>
<td><strong>Step 4</strong></td>
</tr>
<tr>
<td>Set policy and quality priorities</td>
<td>Set reduction targets &amp; plans</td>
<td>Measure Technical Debt</td>
<td>Plan actions for remediation</td>
</tr>
<tr>
<td><strong>Step 5</strong></td>
<td><strong>Step 6</strong></td>
<td><strong>Step 7</strong></td>
<td></td>
</tr>
<tr>
<td>RemEDIATE violations</td>
<td>Track results</td>
<td>Report to the business</td>
<td></td>
</tr>
</tbody>
</table>
Translating Tech Debt to Business Measures

Software attribute | Operational problems | Business measure
---|---|---
Robustness | Outages, slow recovery | Availability
Performance | Degraded response | Work efficiency
Security | Breaches, Theft | Data protection
Transferability | Lengthy comprehension | IT productivity
Changeability | Excessive effort | Delivery speed

Technical debt

Managing Structural Quality in Telecom

Measured impact in a complex enhancement-heavy environment

CLIENT STUDY OVER 24 MONTHS

Before structural quality analysis and management

New critical violations

System test defects

Releases

CAST Confidential

CAST Confidential
**Benefit of Tech Debt Reduction in Banking**

- $2,500,000 loss
- $60,000 loss
- $3,000 loss

**Tech Debt Reduction and Incident Rate**

Correlation of maintenance effort with incident tickets across 20 customers of a global system integrator

Increase of TQI by 0.24 = decrease in maintenance activity by 50%