Advances in Measuring Software Quality and Technical Debt

Dr. Bill Curtis
Executive Director, CISQ

International Standards for Automating Software Size and Structural Quality Measurement

Modern Apps Are a Technology Stack

1. Unit Level
   - Code style & layout
   - Expression complexity
   - Code documentation
   - Class or program design
   - Basic coding standards
   - Developer level

2. Technology Level
   - Single language/technology layer
   - Intra-technology architecture
   - Intra-layer dependencies
   - Inter-program invocation
   - Security vulnerabilities
   - Development team level

3. System Level
   - Multiple languages
   - Architectural compliance
   - Risk propagation
   - Application security
   - Resiliency checks
   - Transaction integrity
   - Function points
   - Integration quality
   - Data access control
   - SDK versioning
   - Calibration across technologies
   - IT organization level
What Is CISQ?

CISQ is chartered to specify measures of software size and quality that can be automated from source code, and promote them through OMG and other international standards organizations.

CISQ/OMG Standards Process

- Automated Function Points
- Reliability
- Performance Efficiency
- Security
- Maintainability

OMG

OMG Approved Standards

ISO Fast Track

Deployment Workshops
### CISQ Structural Quality Measures

<table>
<thead>
<tr>
<th>Category</th>
<th>Weaknesses</th>
<th>Example Architectural and Coding Weaknesses Included in the CISQ Measures</th>
</tr>
</thead>
</table>
| Security       | 22 (Top 25 CWEs) | • SQL injection  
• Cross-site scripting  
• Buffer overflow  
• Empty exception block  
• Unreleased resources  
• Circular dependency |
| Reliability    | 29         | • Expensive loop operation  
• Un-indexed data access  
• Unreleased memory |
| Performance Efficiency | 15 | • Excessive coupling  
• Dead code  
• Hard-coded literals |
| Maintainability| 20         |                                                                 |

An international team of experts selected the weaknesses to include in CISQ measures based on the severity of their impact on operational problems or cost of ownership.

Only weaknesses considered severe enough that they must be remediated were included in the CISQ measures.

CISQ Structural Quality measures are currently being extended to embedded systems software.

### 22 (of Top 25) CWEs Form the CISQ Security Measure

- CWE-22 Path Traversal Improper Input Neutralization
- CWE-78 OS Command Injection Improper Input Neutralization
- CWE-79 Cross-site Scripting Improper Input Neutralization
- CWE-89 SQL Injection Improper Input Neutralization
- CWE-120 Buffer Copy without Checking Size of Input
- CWE-129 Array Index Improper Input Neutralization
- CWE-134 Format String Improper Input Neutralization
- CWE-252 Unchecked Return Parameter of Control Element Accessing Resource
- CWE-327 Broken or Risky Cryptographic Algorithm Usage
- CWE-396 Declaration of Catch for Generic Exception
- CWE-397 Declaration of Throws for Generic Exception
- CWE-434 File Upload Improper Input Neutralization
- CWE-456 Storable and Member Data Element Missing Initialization
- CWE-606 Unchecked Input for Loop Condition
- CWE-667 Shared Resource Improper Locking
- CWE-672 Expired or Released Resource Usage
- CWE-681 Numeric Types Incorrect Conversion
- CWE-706 Name or Reference Resolution Improper Input Neutralization
- CWE-772 Missing Release of Resource after Effective Lifetime
- CWE-789 Uncontrolled Memory Allocation
- CWE-798 Hard-Coded Credentials Usage for Remote Authentication
- CWE-835 Loop with Unreachable Exit Condition ('Infinite Loop')
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

ISO/IEC 25010 — Software Product Quality

- Functional Suitability
  - Functional appropriateness
  - Accuracy
  - Compliance
- Reliability
  - Maturity
  - Availability
  - Fault tolerance
  - Recoverability
  - Compliance
- Performance Efficiency
  - Time behavior
  - Resource utilization
  - Compliance
- Operability
  - Appropriateness
  - Recognizability
  - Learnability
  - Ease of use
  - Attractiveness
  - Technical Accessibility
  - Compliance
- Security
  - Confidentiality
  - Integrity
  - Non-repudiation
  - Accountability
  - Authenticity
  - Compliance
- Compatibility
  - Co-existence
  - Interoperability
  - Compliance
- Maintainability
  - Modularity
  - Reusability
  - Analyzability
  - Changeability
  - Modification
  - Stability
  - Testability
  - Compliance
- Portability
  - Adaptability
  - Installability
  - Replaceability
  - Compliance

CISQ automated structural quality measures are highlighted in blue

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
  - Lost opportunity
  - Liability from debt

- Technical Debt
  - Interest on the debt
  - Principal borrowed

- Structural quality problems in production code

Lost opportunity — 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, and other damaging incidents

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

Technical Debt by Quality Characteristic

- **70% of Technical Debt is in IT Cost** (Transferability, Changeability)
- **30% of Technical Debt is in Business Risk** (Robustness, Performance, Security)
- Health Factor proportions are mostly consistent across technologies


CISQ-like Measures Predict Incidents & Costs

**Correlation of Total Quality Index and log of incidents** for 21 applications in a large global system integrator

- **$R^2 = .34$**
  - Total Quality Index accounts for 1/3 of variation in incidents
  - Increase in Total Quality Index of .24 decreased corrective maintenance effort 50%

$R^2 = .34$

Log of tickets

Corrective Maintenance

Total Quality Index

Log of ticket count

**Log of tickets**

**Linear (Log of tickets)**
Automated Technical Debt Measure

Sum of efforts-to-fix for all instances of each weakness

Weighted effort-to-fix for each instance of a weakness

Reliability weaknesses
Security weaknesses
Performance weaknesses
Maintainability weaknesses

Automated Technical Debt

Predict effort for corrective maintenance
Predict cost of corrective maintenance

Roadmap for CISQ Measures

Automated Function Points
Must measure functional and non-functional code segments

Automated Enhancement Points
Must add future effort to fix bugs into productivity

Quality-Adjusted Productivity
Must estimate the corrective costs in future releases

Automated Technical Debt

Extensions to Embedded Software

Four Quality Characteristic Measures
Application Certification Using CISQ

- **CISQ measures**
- **CISQ-conformance assessment**
- **Technology vendors**
- **CISQ-conformant technology**
- **CISQ service process**
- **Vendor authorized service providers**
- **CISQ-conformant service process**

- **CISQ/OMG**
  - only assess vendor conformance
  - do not certify applications
  - program initiates in 2017

- **Service providers**
  - use CISQ-conformant technology
  - in a CISQ-conformant service process
  - to provide application certifications

Deploying CISQ Measures

- **CISQ measures**
- **OMG standards**
- **ISO standards**

- **Federal IT Policy**
  - Regulations
    - Sec. & Exch. Com.
    - State of Texas
  - System acquisition
    - US State Dept.
    - Gen. Serv. Admin.

- **Corporate IT Policy**
  - Third party Contracts
  - Benchmarks
    - Sec. & Exch. Com.
    - State of Texas
    - Gen. Serv. Admin.
As a greater portion of mission, business, and safety critical functionality is committed to software-intensive systems, these systems become one of, if not the largest source of risk to enterprises and their customers. Since corporate executives are ultimately responsible for managing this risk, we establish the following principles to govern system development and deployment.

1. **Engineering discipline in product and process**
2. **Quality assurance to risk tolerance thresholds**
3. **Traceable properties of system components**
4. **Proactive defense of the system and its data**
5. **Resilient and safe operations**