Measures for DevOps Pipelines

Evolving Software Measurement
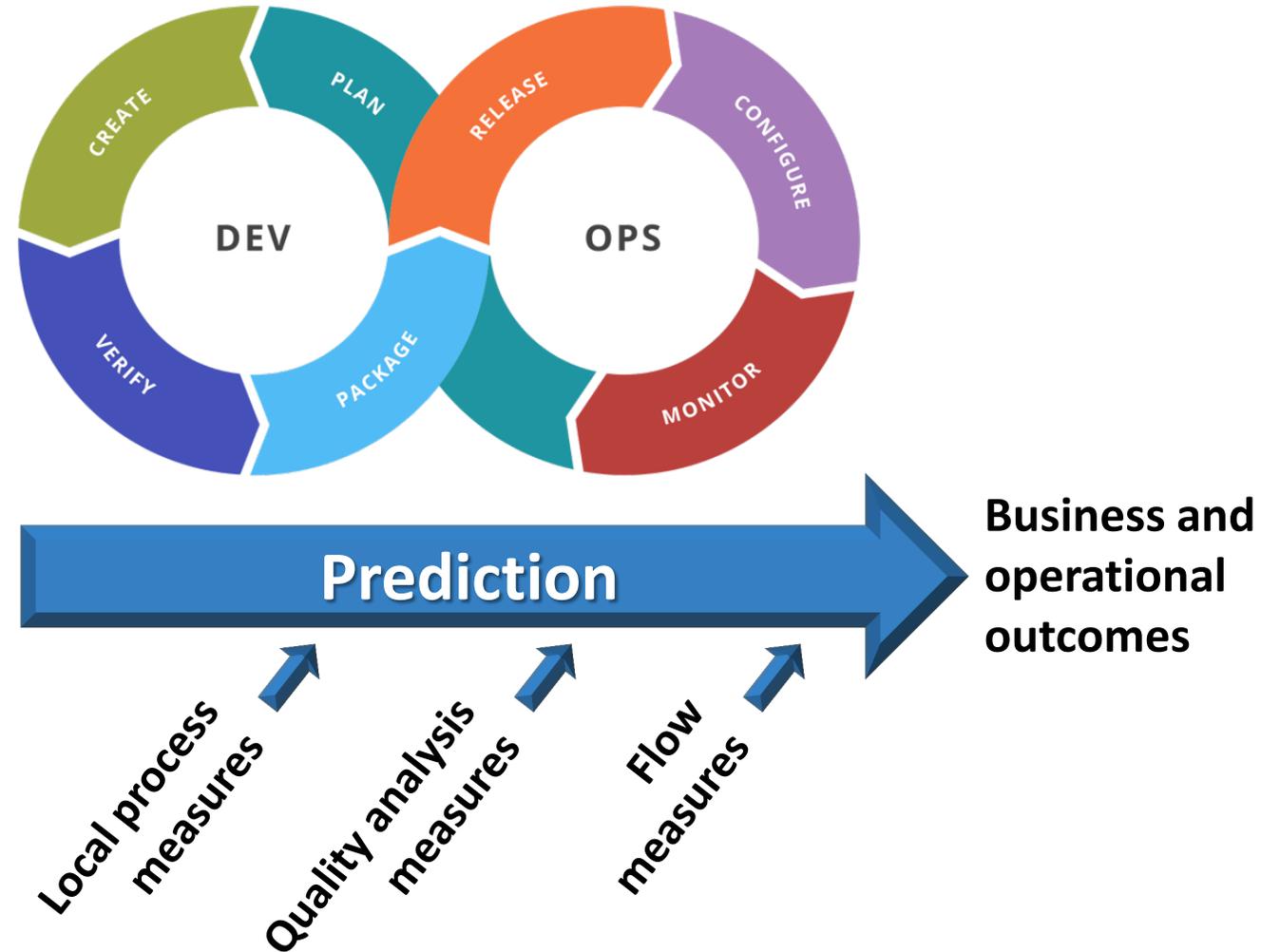
Dr. Bill Curtis

Executive Director
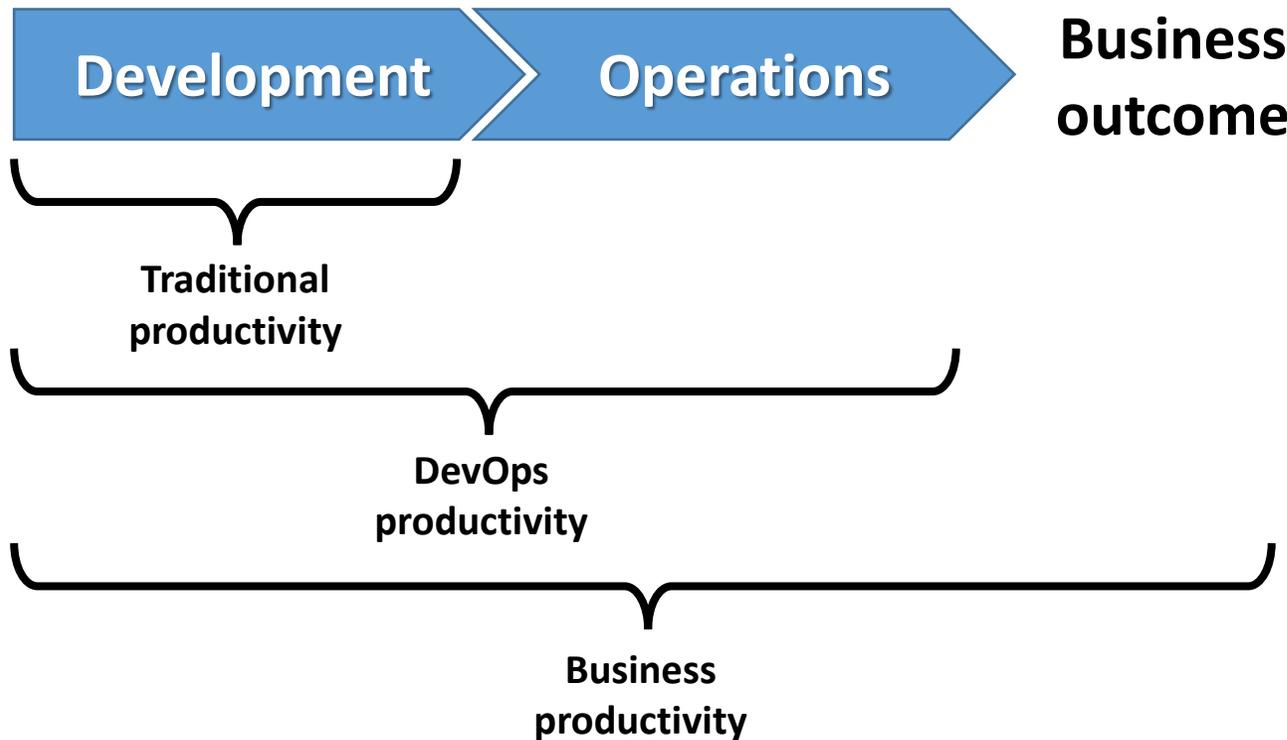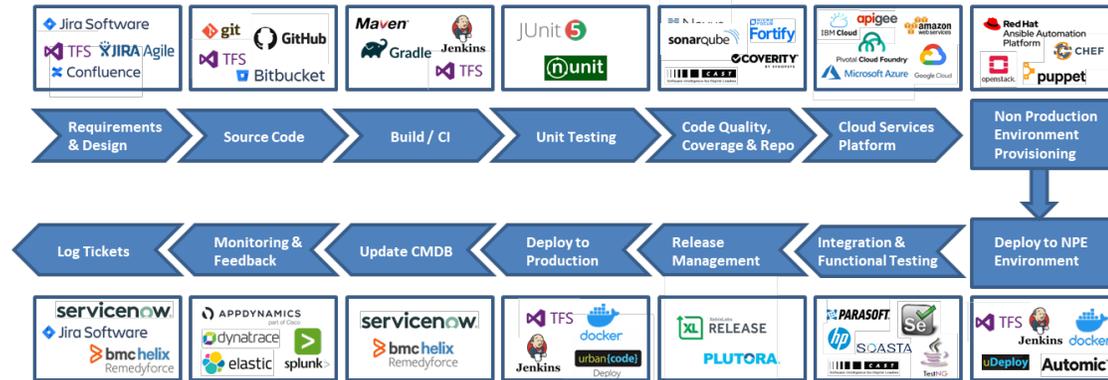
CISQ Consortium for Information and Software Quality

CISQ
Consortium for Information & Software Quality ™

DevOps Measurement Framework

- Value chain philosophy

- Business outcome measures

- Flow measures for cycle time, waiting time, productivity, etc.

- Architecture measures for flow optimization, blockages, etc.

- Local measures aggregated across flow to predict outcomes

DEV — CREATE, PLAN, VERIFY, PACKAGE

OPS — RELEASE, CONFIGURE, MONITOR

Prediction

Business and operational outcomes

Local process measures

Quality analysis measures

Flow measures

CISQ
Consortium for Information & Software Quality ™

Jira Software / TFS JIRA Agile / Confluence	git GitHub / TFS Bitbucket	Maven Gradle Jenkins / TFS	JUnit 5 nunit	sonarqube Fortify COVERITY CAST	apigee amazon webservices / IBM Cloud Pivotal Cloud Foundry / Microsoft Azure Google Cloud	Red Hat Ansible Automation Platform / CHEF openstack puppet
Requirements & Design	Source Code	Build / CI	Unit Testing	Code Quality, Coverage & Repo	Cloud Services Platform	Non Production Environment Provisioning

Log Tickets	Monitoring & Feedback	Update CMDB	Deploy to Production	Release Management	Integration & Functional Testing	Deploy to NPE Environment
servicenow Jira Software bmc helix Remedyforce	APPDYNAMICS part of Cisco dynatrace elastic splunk	servicenow bmc helix Remedyforce	TFS docker Jenkins urban{code} Deploy	XL RELEASE PLUTORA	PARASOFT hp Se SOASTA CAST TestNG	TFS Jenkins docker uDeploy Automic

Development ➤ **Operations** ➤ **Business outcome**

Traditional productivity

DevOps productivity

Business productivity

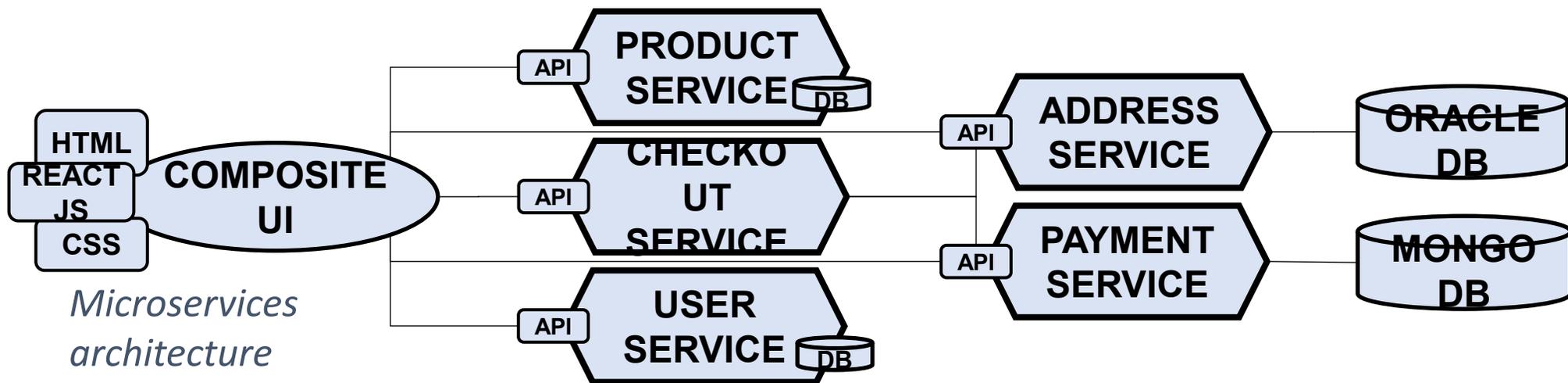- **Flow measures:**
  ➤ cycle time
  ➤ work in process
  ➤ throughput

- **Support issues:**
  ➤ waiting time
  ➤ blockages
  ➤ arrival rate
  ➤ departure rate

Measuring Architectural Impact on DevOps

Theory

Microservices architecture

Reality

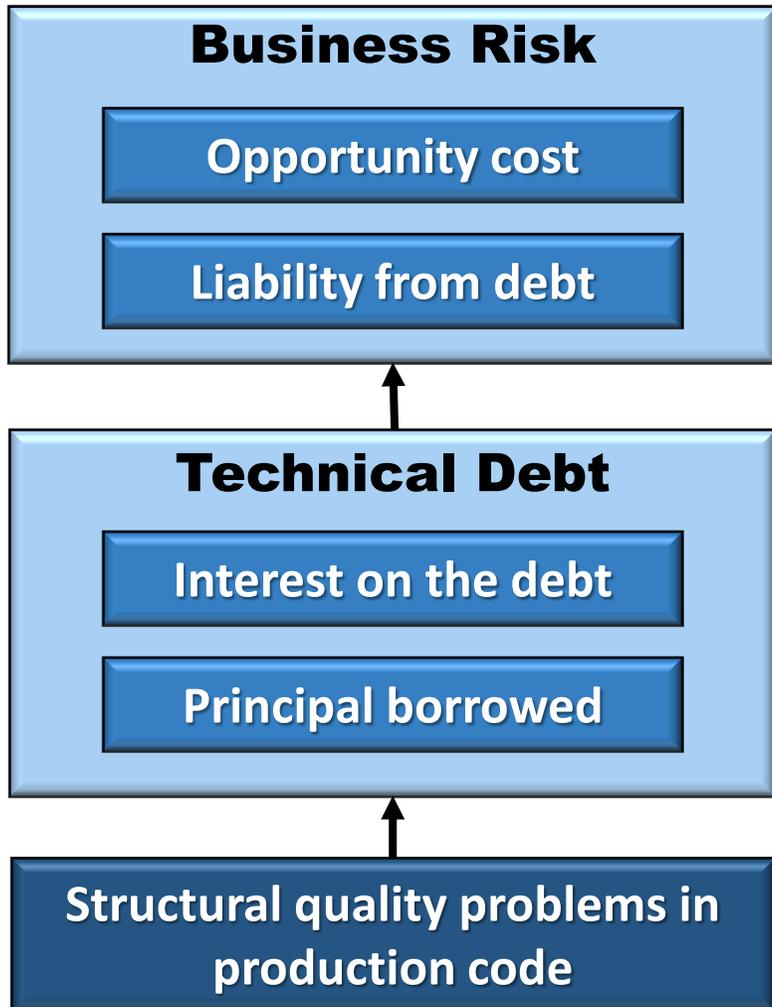*Legacy architecture*

Extracting communities of cohesive functionality
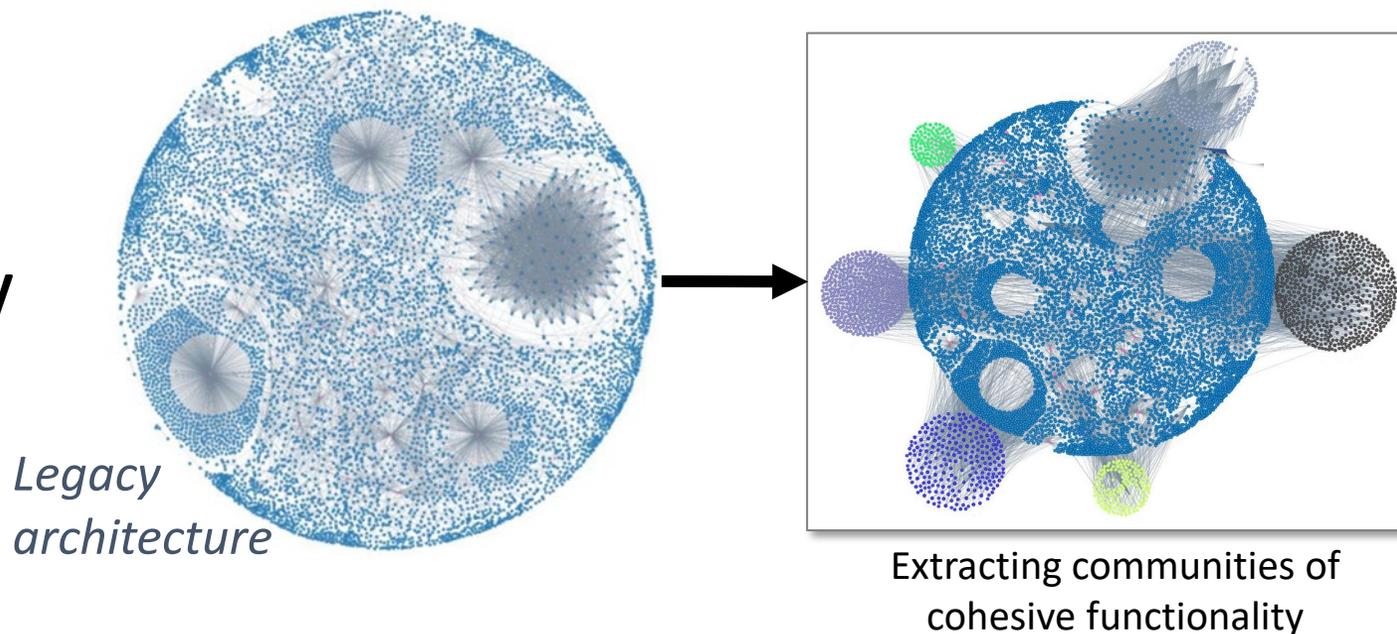
Potential Measures:

- Architectural complexity
- % app potential for micro-services
- # of architectural rules violated
- Rate of architectural degradation
- Flow impedance of dependencies
- Actual vs potential flow rate

CISQ
Consortium for Information & Software Quality ™

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**

Structural quality problems in production code

Opportunity cost—potential benefits from new capability rather than retiring technical debt → • % work in progress

Liability—business costs related to outages, breaches, corrupted data, etc. → • business outcome

Interest—continuing IT costs attributable to weaknesses causing technical debt, i.e, higher maintenance costs, greater resource usage, etc. → • blockages
• throughput
• cycle time

Principal—cost of fixing problems remaining in the code after release that must be remediated → • % work in progress

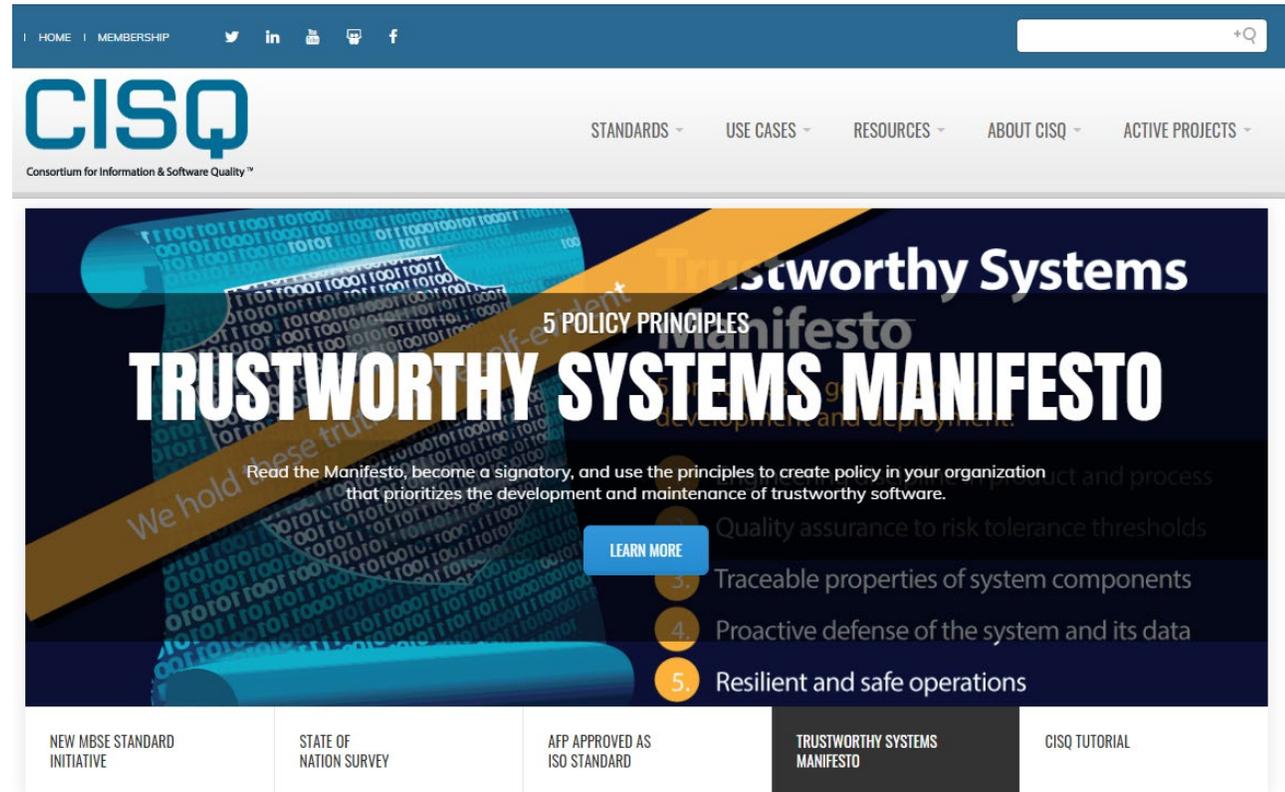# Join CISQ, use the standards, contact us to learn more!

Dr. Bill Curtis
Executive Director
bill.curtis@it-cisq.org

Tracie Berardi
Program Director
tracie@omg.org

Send feedback on the Data Privacy and Protection Measure to info@it-cisq.org

www.it-cisq.org