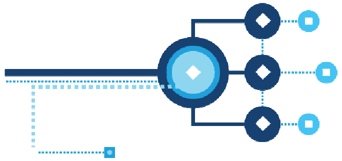
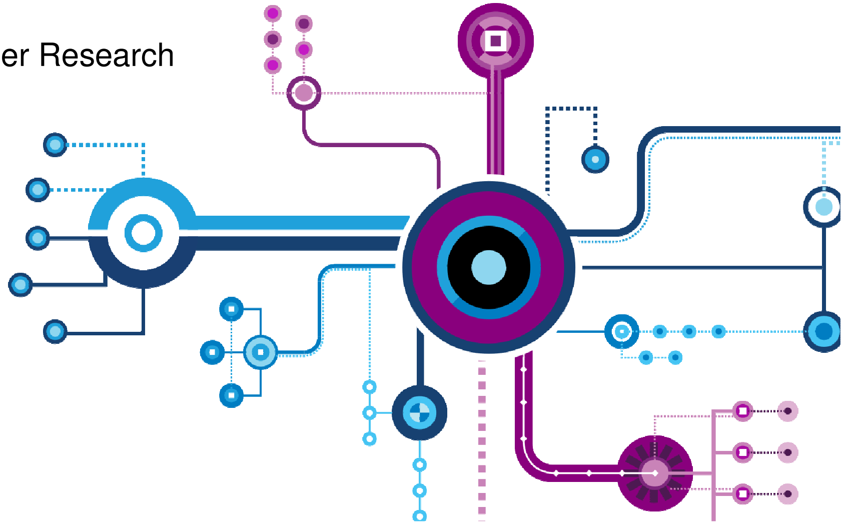


The future of software delivery

Global insights

Diego Lo Giudice - Vice President & Principal Analyst at Forrester Research

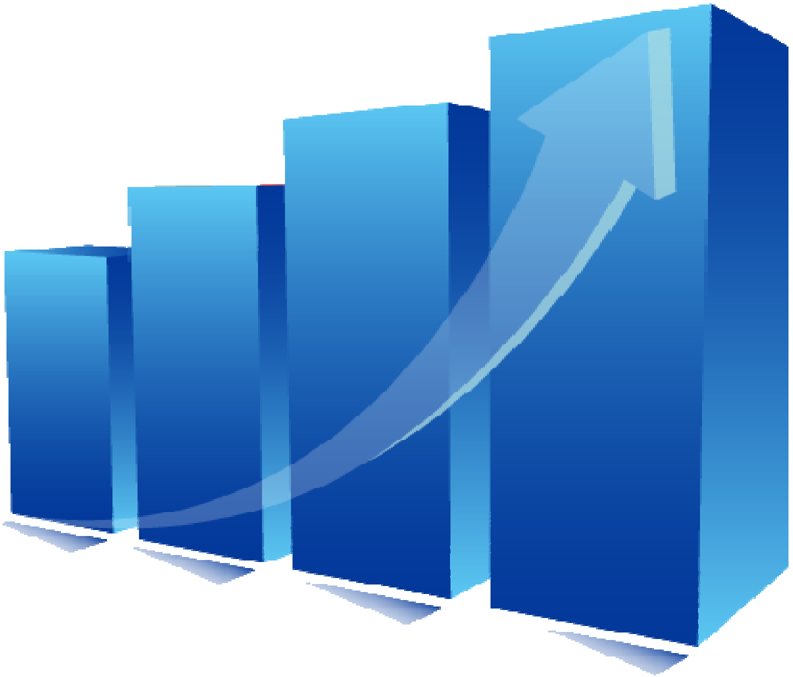
Peter Klenk - Manager Software Technology and Member of the Chief Technology Office (CTO) Council for IBM Rational software



November 2012

Software Quality: A costly problem across all industries

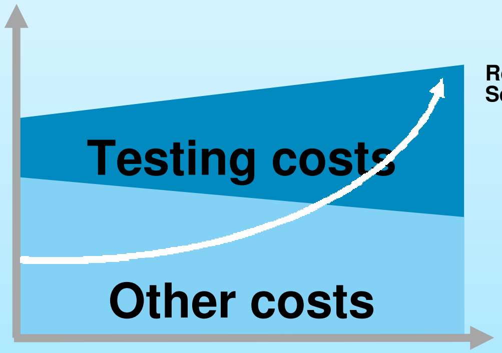
- Software is blamed for ***more major business problems than any other man-made product***.
- Poor software quality has become ***one of the most expensive topics in human history***
  - –***> $150 billion per year in U.S***.
  - –***> $500 billion per year worldwide***.
- Projects cancelled due to poor quality are ***>15% more costly than successful projects*** of the same size and type.



Source: Capers Jones, 2011
Based on 675 companies, 35 government/military groups, 13,500 projects, 50-75 new projects/month, 24 countries, 15 lawsuites

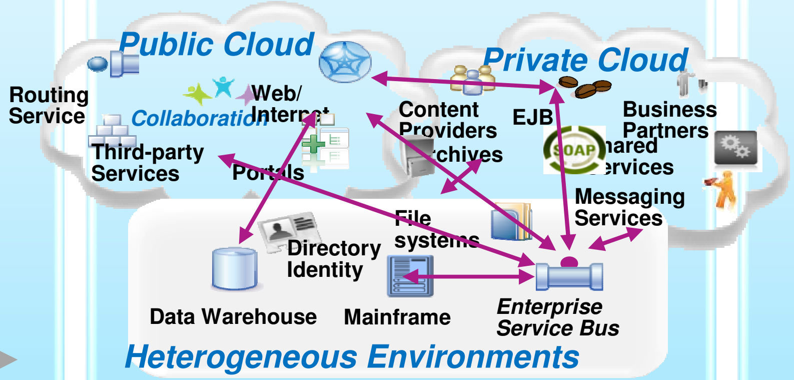
Today’s testing paradigm is impractical

Increasing Cost of Quality



Outsourcing **labor** is no longer a de facto approach as global wages are increasing

Increasing Development Complexity

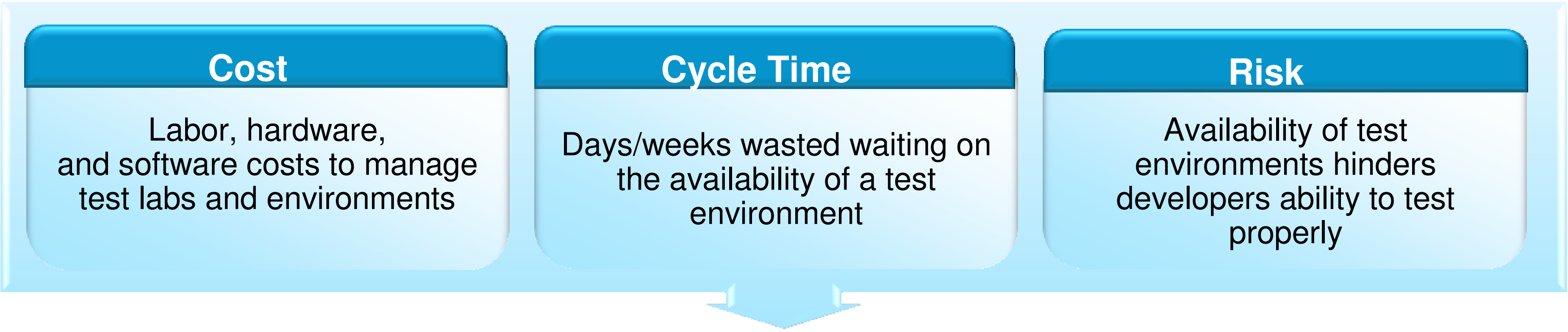


Product and application **complexity** and size are increasing

Balancing Quality and Speed

Productivity is inhibited as test teams can no longer keep up with agile development

Test environment availability is a key inhibitor



Cost

Labor, hardware, and software costs to manage test labs and environments

Cycle Time

Days/weeks wasted waiting on the availability of a test environment

Risk

Availability of test environments hinders developers ability to test properly

- Lots of under-utilized and costly test lab resources
- Development and QA waste a lot of time on unproductive activities: installation, configuration, trial/error, etc.
- A significant portion of the testing effort is pushed late in the process resulting in defects costing 10-100x to fix

The solution today…

Test Lab costs

- Use of hardware-based virtualization or cloud based resources provides partial savings (20-30%)
- Installation and configuration of software is still very labor intensive
- Certain systems cannot leverage hw virtualization, e.g. costly third party services, mainframe applications, proprietary systems

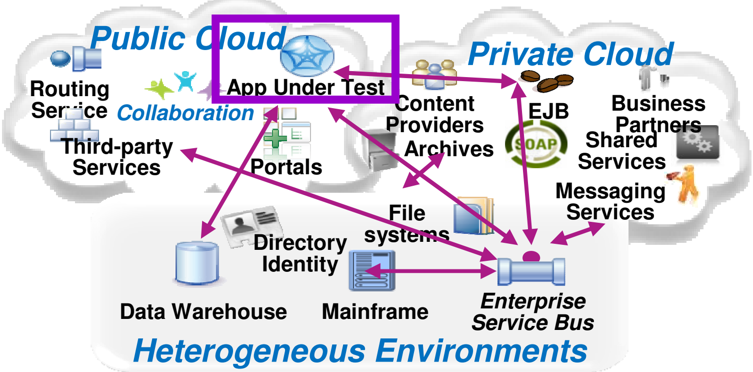
Cycle Time

- Investment in UI test automation has proven to reduce cycle time for regression testing
- Testing new functions still require to have an environment available to develop test scripts
- The time wasted waiting for a test environment is severely reducing the ability to do proper acceptance testing

Risk

- Addressed through better collaboration between development and testing, better test planning, e.g. using Rational Quality Manager
- Too many “trivial” defects are still found late in the process by Quality Assurance teams

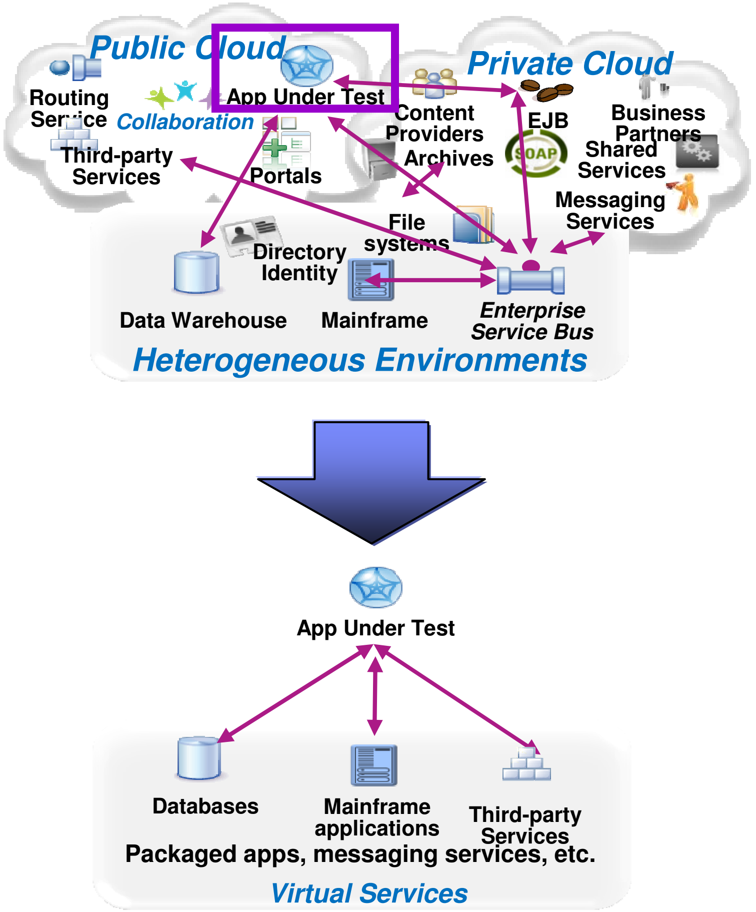
Barriers to complete test environments



- System dependencies are a key challenge in setting up test environments
- Unavailable/inaccessible services: Testing is constrained due to production schedules, security restrictions, contention between teams, or because they are still under development
- Costly 3rd party access fees: Developing or testing against Cloud-based or other shared services can result in costly usage fees
- Impractical hardware-based virtualization: Systems are either too difficult (mainframes) or remote (third-party services) to replicate via traditional hardware-based virtualization approaches

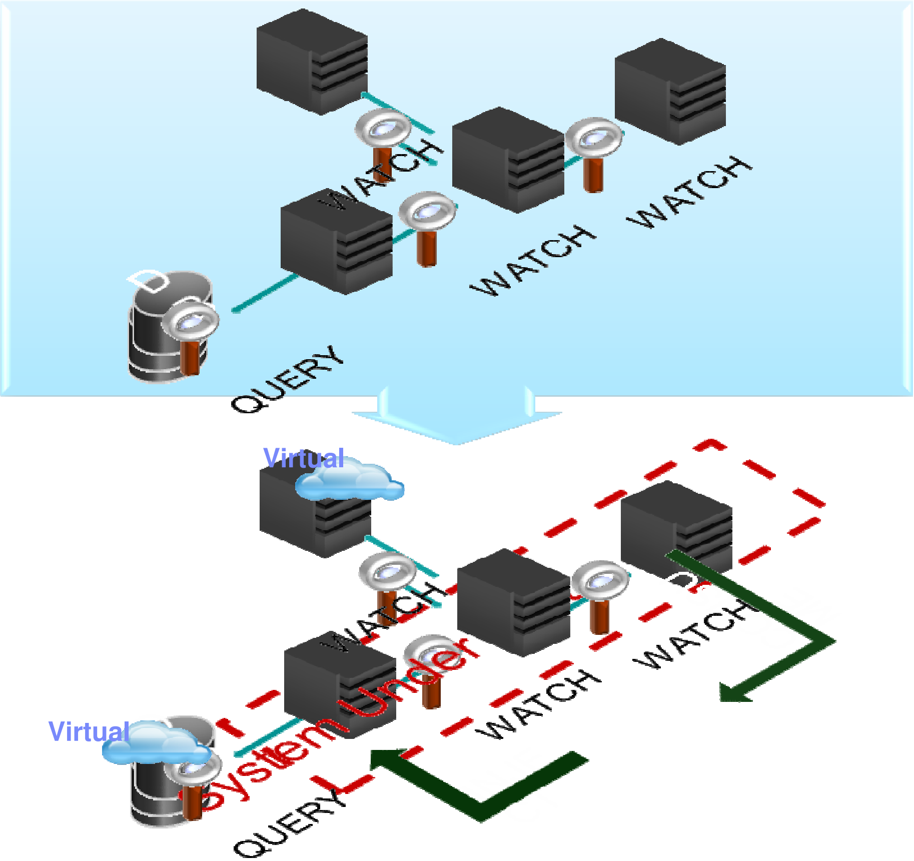
Introducing Service Virtualization

- Virtual components simulate the behavior of an entire application or system during testing
- Virtual components run on commodity hardware, private cloud, public cloud
- Each developer and tester can easily have their own test environment
- Developers and testers continue to use current testing procedures and tools (manual, performance, UI test automation, etc.)
- Highly complementary to service-layer (integration) test automation

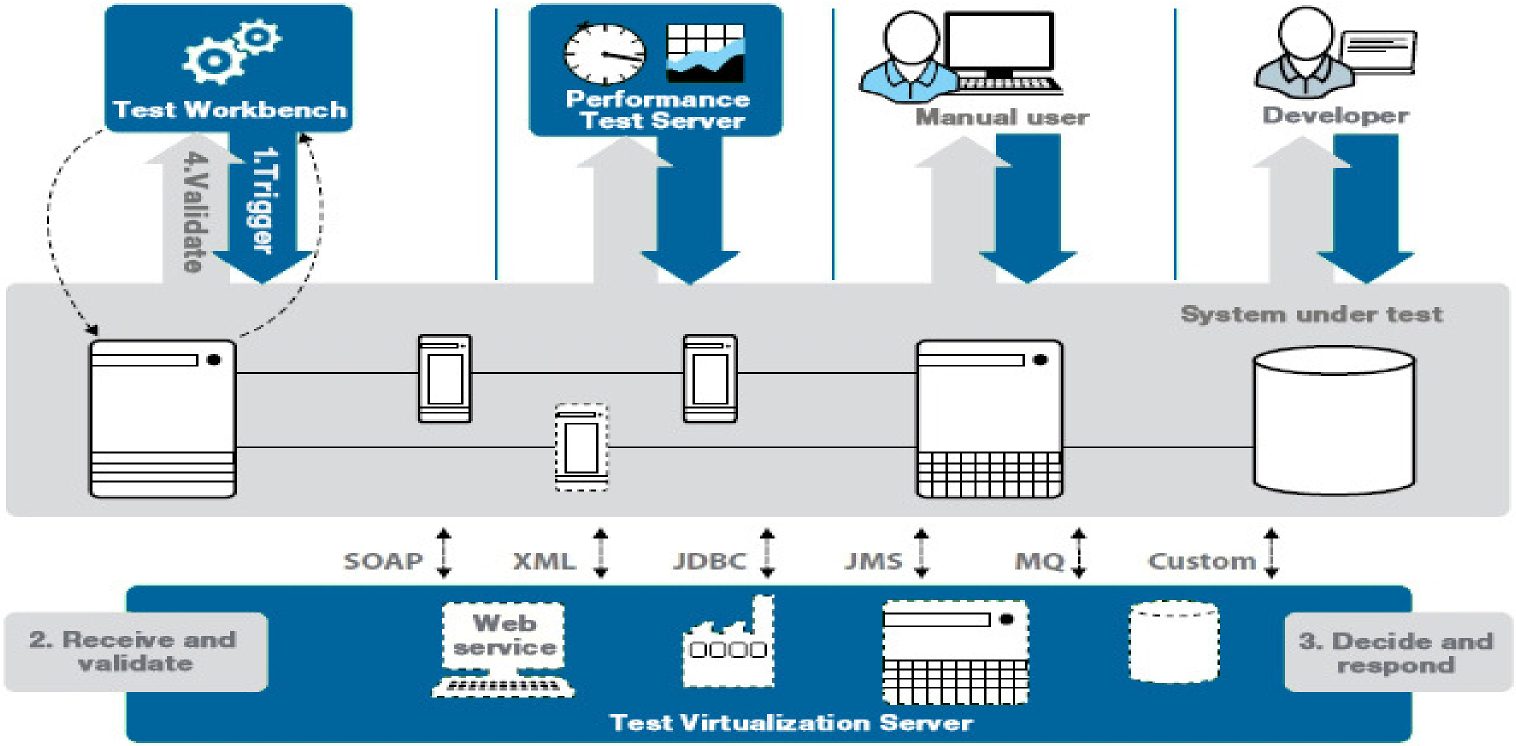


Service Virtualization how-to

- Virtual components can be created from
  - Service specifications or,
  - From recording actual traffic to existing services/applications
- Virtual components can be further customized
  - To simulate simple to complex behaviors
  - To simulate latency, performance profiles, etc.
- Virtual components are published for consumption by developers and testers
  - Testing can start earlier: Testers can now create their tests against virtual services
  - Systems can be incrementally tested as sub-systems become available



Service Virtualization in context



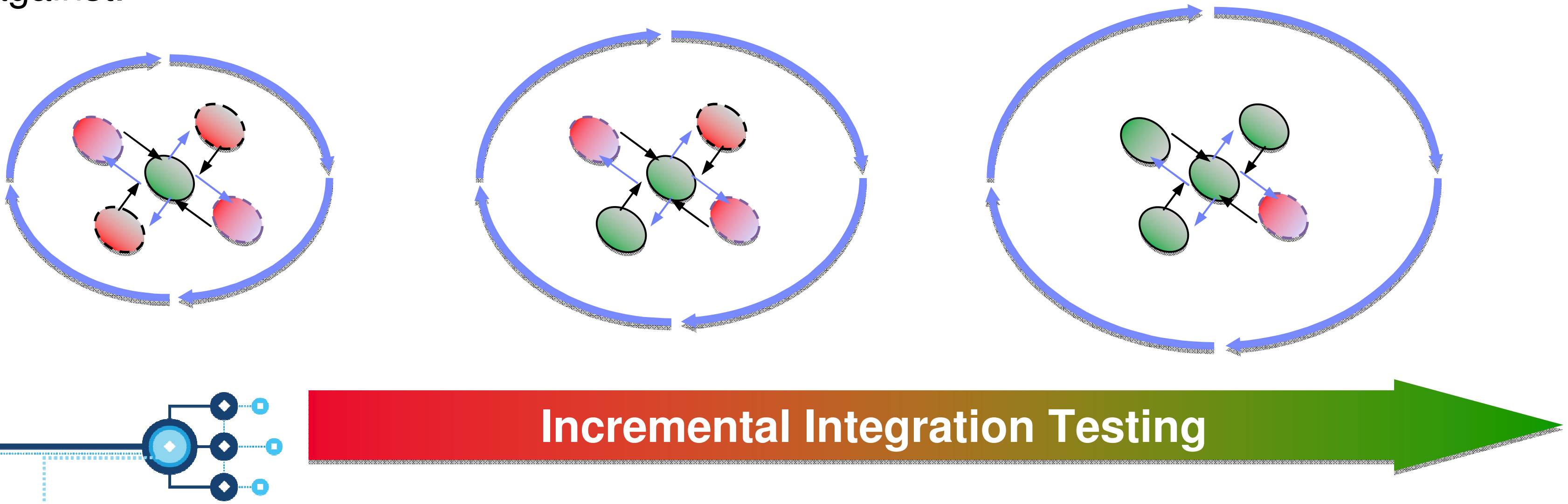
Service Virtualization enables continuous integration testing

✓ Services, applications, systems are introduced into the continuous integration cycle in a prioritized, controlled fashion.

✓ Units not yet available are simulated and tested against.

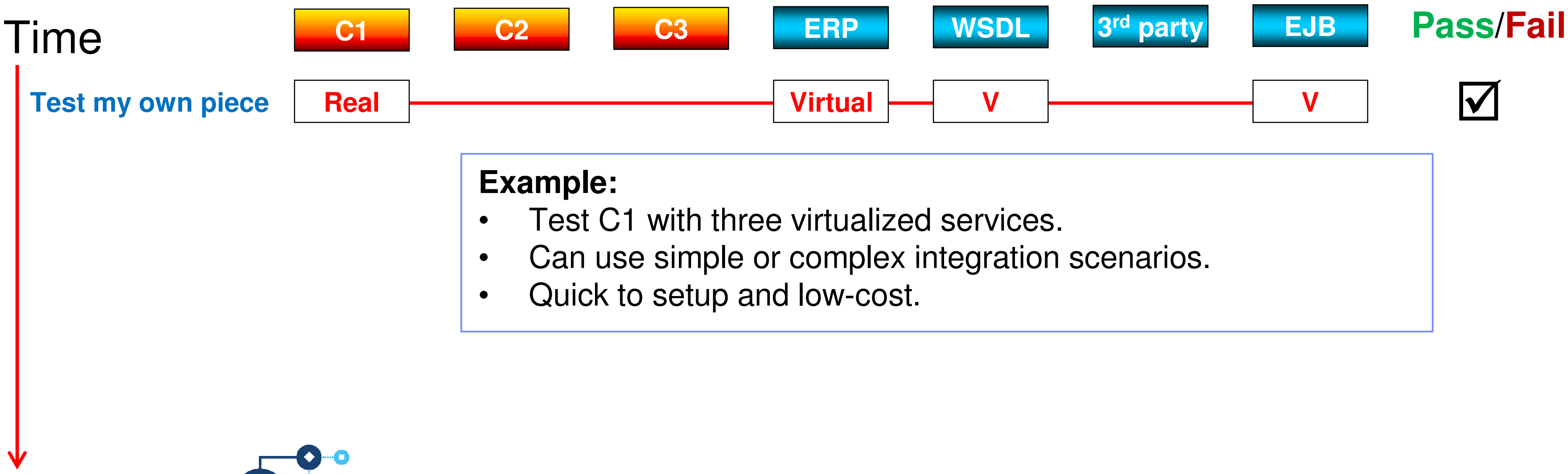
🟢 Actual Component

🔴 Virtual Component



Continuous Integration Testing

- Integration Testing requires components that may not be ready/available yet, or expensive to use – Service Virtualization enables replacing them with a virtual component.
- Services, applications, systems are introduced into the continuous integration cycle in a prioritized, controlled fashion.



Time	C1	C2	C3	ERP	WSDL	3rd party	EJB	Pass/Fail
Test my own piece	Real			Virtual	V		V	☑

Example:

- Test C1 with three virtualized services.
- Can use simple or complex integration scenarios.
- Quick to setup and low-cost.

Continuous Integration Testing

- Integration Testing requires components that may not be ready/available yet, or expensive to use – Service Virtualization enables replacing them with a virtual component.
- Services, applications, systems are introduced into the continuous integration cycle in a prioritized, controlled fashion.

Time	C1	C2	C3	ERP	WSDL	3rd party	EJB	Pass/Fail
Test my own piece	Real			Virtual	V		V	☑
Integrate w/another	Real	R		V	V		V	☒

- C2 introduced some defects – *further testing is blocked!*

Continuous Integration Testing

- Integration Testing requires components that may not be ready/available yet, or expensive to use – Service Virtualization enables replacing them with a virtual component.
- Services, applications, systems are introduced into the continuous integration cycle in a prioritized, controlled fashion.

Time	C1	C2	C3	ERP	WSDL	3rd party	EJB	Pass/Fail
Test my own piece	Real			Virtual	V		V	☑
Integrate w/another	Real	R		V	V		V	☒
Failures won't slow me down!	Real	V		V	V		V	☑

- C2 introduced some defects – *replace it with a virtual service!*
- A defect in C2 doesn't stop testing of those who depend on it!
- Quick to setup and low-cost.

Continuous Integration Testing

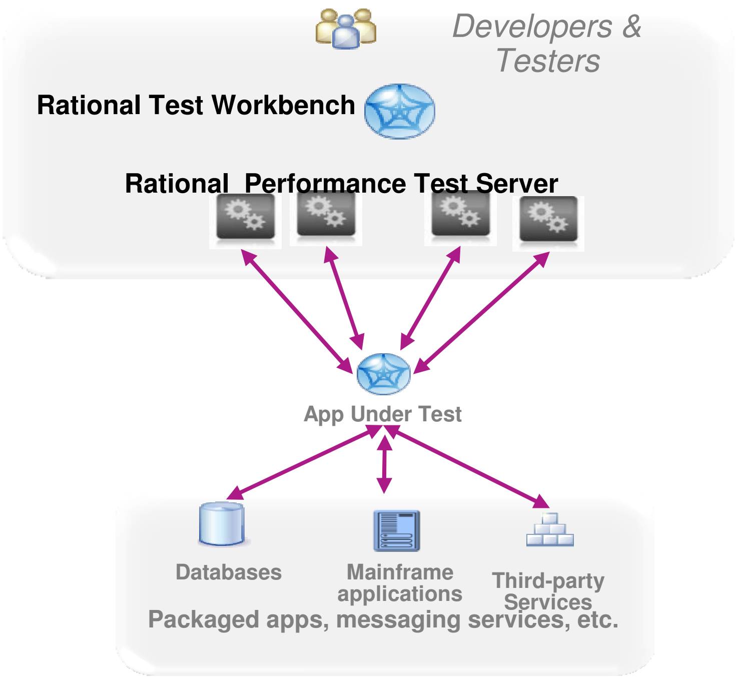
- Integration Testing requires components that may not be ready/available yet, or expensive to use – Service Virtualization enables replacing them with a virtual component.
- Services, applications, systems are introduced into the continuous integration cycle in a prioritized, controlled fashion.

Time	C1	C2	C3	ERP	WSDL	3rd party	EJB	Pass/Fail
Test my own piece	Real			Virtual	V		V	☑
Integrate w/another	Real	R		V	V		V	☒
Failures won't slow me down!	Real	V		V	V		V	☑
Negative tests	Real	R		V'	V		V	☑

- Virtual components can easily be modified to support negative testing.

IBM Rational Service Virtualization Solution

- **Rational Test Workbench** is a desktop solution that enables testers/developers to:
  - Capture and model virtual services
  - Test services and applications long before their user interfaces becomes available and do integration testing (SOA, BPM)
  - Delivers Functional, Performance, and Integration Testing
- **Rational Test Virtualization Server** is a server solution that:
  - Provides a central environment to virtualize heterogeneous hardware, software and services to provide 24x7 testing capabilities
  - Reduces infrastructure costs of traditional testing environments
  - Virtual Services can be built from the interface definition of the system for a wide variety of protocols, including HTTP, web services, SOA, JMS, TIBCO, IBM WebSphere MQ, Oracle, etc.
- **Rational Performance Test Server** enables Rational Test Workbench users to reuse test scripts to drive performance testing
  - Can be used in combination with Virtual Services
  - Probe for identification of system bottlenecks



Rational Test Virtualization Server

Supported Environments & Technologies

Messaging Protocols

- ActiveMQ
- Email (SMTP, IMAP)
- Files
- TCP, FTP/S, HTTP/S
- JMS (JBOSS et al)
- IBM WebSphere MQ
- JBoss MQ
- SAP IDoc, BAPI, RFC & XI/PI
- Software AG's IB & IS
- Solace
- Sonic MQ
- TIBCO Rendezvous, Smart Sockets & EMS
- Custom

SOA, ESB, Others

- CentraSite
- Oracle Fusion
- SCA Domain
- Software AG IS, BPMS
- Sonic ESB
- TIBCO ActiveMatrix
- UDDI
- Web Services
- WebSphere RR
- WSDL

- BPM
- Databases
- Log Files

Message Formats

- .Net Objects
- Bytes
- COBOL Copybook
- ebXML
- EDI
- Fixed Width
- HL7
- IATA
- Java Objects
- MIME
- OAG
- SOAP
- Software AG Broker Docs
- SWIFT
- TIBCO ActiveEnterprise
- XML (DTD, XSD, WSDL)
- Custom

Customer Results

INTERNATIONAL PAPER

- After an acquisition, needed to get off rented infrastructure
- Move to webMethods as fast as possible
- Regression testing essential
- Stubbing of systems while they move over critical systems
- GH Tester performed all required functions quickly and easily
- Fully integrated in six months, two months early
- Saved significant rental costs

Leading global financial services firm, assets of $2 trillion+

- Bought next generation payments system
- Impact = organizational heart transplant
- Disparate, legacy formats
- Stubbed third party systems, otherwise unavailable for testing
- Reduced 10 days of manual testing to 10 minutes
- Saved >$7 million so far
- “Project would have been impossible without the tool”

€30 billion international supermarket operator

- Upgrade to webMethods 8
- “You need 30,000 hours to test the new environment”
- GH Tester + GCS = 4,000 hours
- Focus on business & volume critical components
- New testing strategy for all developments
- Cut time and costs but NOT quality

Green Hat technology is a game changer for Quality Management

Test Lab costs

- Test lab infrastructure costs can be reduced by up to 90%
- Labor involved in setting up test environments can be reduced by 80%+
- Reduced or eliminated the cost of invoking 3rd party systems for non-production use, fee-based web services

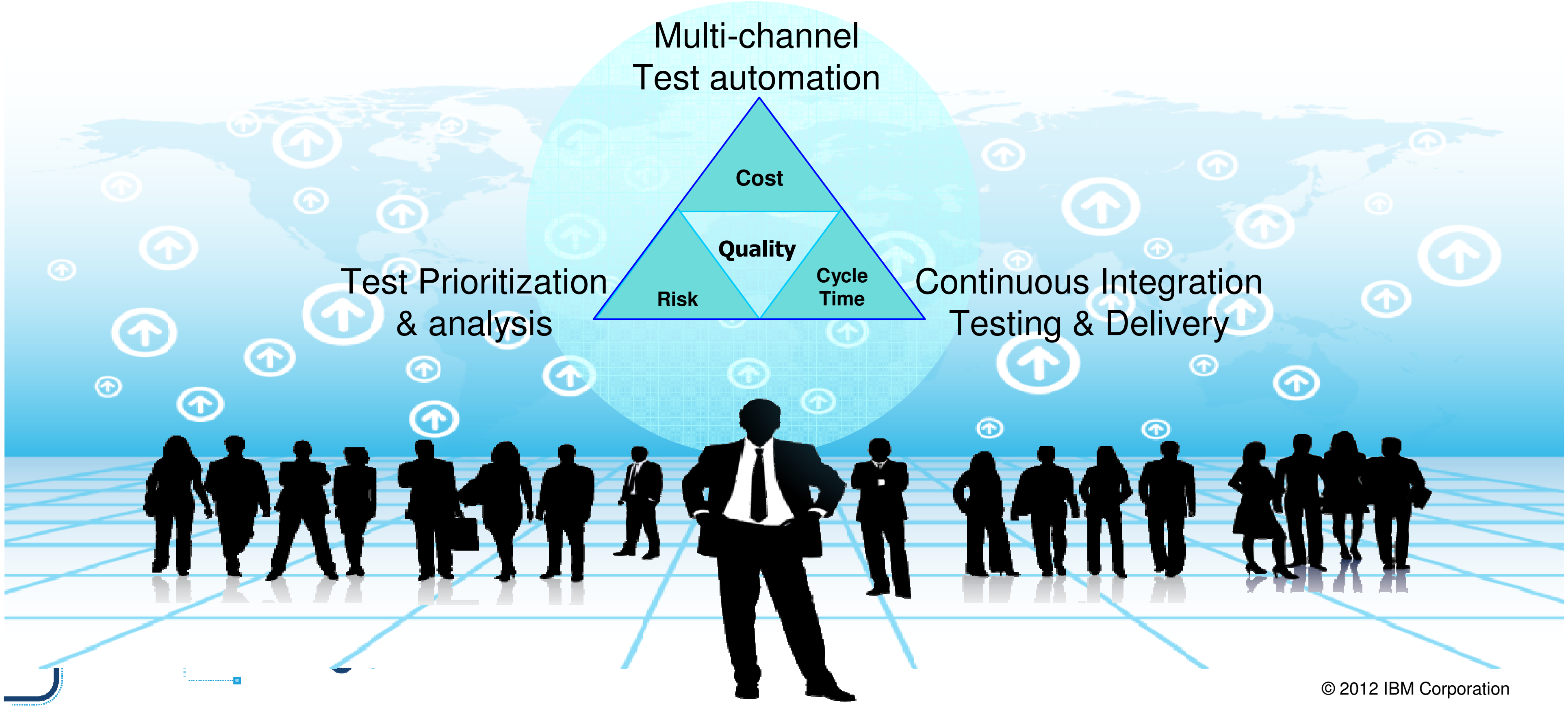
Cycle Time

- Test environments can be configured in minutes vs weeks
- More testers can be focused on testing, rather than configuring test environments
- More regression testing can be done independently from the User Interface, during development

Risk

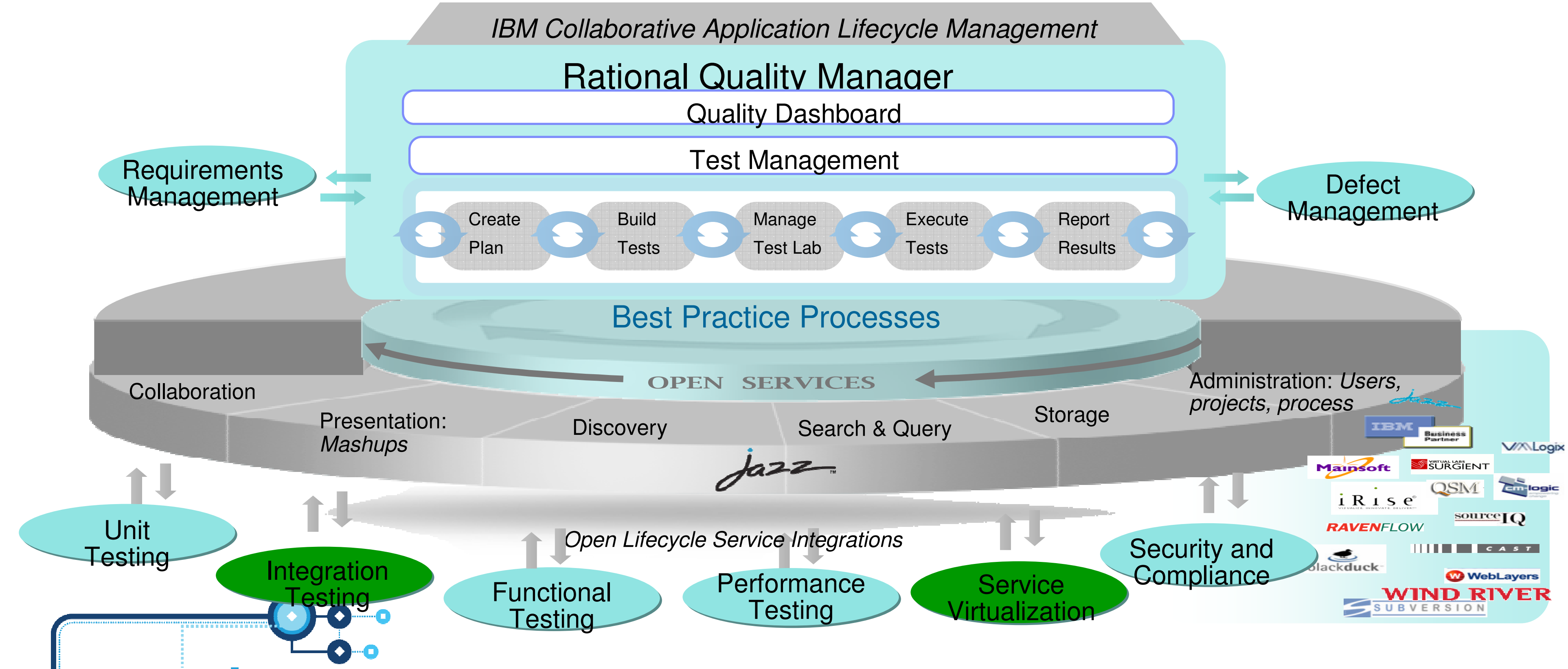
- Developers have the means to test software earlier at the Service/API level
- Large teams working on different parts of an application or system can effectively do parallel development by virtualizing different parts of the system

What is required to effectively drive Quality Management?


Multi-channel
Test automation
Cost
Quality
Risk
Cycle Time
Test Prioritization & analysis
Continuous Integration Testing & Delivery

IBM Rational Quality Management

The leading Agile Quality Management Solution



© 2012 IBM Corporation