PRESENTED BY RED HAT

LEARN. NETWORK.
EXPERIENCE OPEN SOURCE.

www.theredhatsummit.com

Evidence-based Application Development

Mark Tomlinson, LoadRunner Product Manager, HP Software

Robb Greathouse, Principal Solution Architect, Red Hat

Can IT matter?

Can IT Matter?

Short answer: YES.

To do this, IT must deliver competitive advantage to the business, continually – dynamically.

A competitive advantage is defined as:

- Becoming better, faster or cheaper.
- Increased revenue opportunity or margin.
- Creating new products, or markets.

What if your IT can't deliver a competitive advantage?

Can IT Matter?

Average IT does not matter – little, or no advantage.

With poor or average IT…what are your options:

- Don’t compete with technological solutions
- Cut the IT budget to improve ROI
- Suffer with poor technology

GREAT IT does matter – hugely competitive advantage.

What is Great IT?

Great IT enables the business to do what it does:

- by innovating existing ways of doing business
- by creating new markets or channels to market
- by enabling entirely new businesses

How do you get Great IT?

1) Align efforts to the big challenges facing the business.

2) Help the business adapt to pace of change:

- Changing competition
- Changing cost structures
- Changing regulations

3) Implement evidence-based practices for development

Evidence-based Development

Understanding “Evidenced-based” origins

Evidence-based Medicine:

“…aims to apply the best available evidence gained from the scientific method to medical decision making….[and that] healthcare professionals should make ‘conscientious, explicit, and judicious use of the current best evidence’ in their everyday practice.”

Evidence-based Practice:

“promotes the collection, interpretation, and integration of valid, important, and applicable patient-reported, clinician-observed, and research-derived evidence”

Understanding Evidence-based Development

Evidence-based Development:

"…aims to apply the best available evidence gained from the development methods to technical decision making….[and that] IT professionals should make 'conscientious, explicit, and judicious use of the current best evidence' in their everyday practice."

Evidence-based IT Practices:

"promotes the collection, interpretation, and integration of valid, important, and applicable business-reported, engineer-observed, and externally-derived evidence."

Critical Thinking about Evidence

In Development we are constantly surrounded by evidence:

- The technology itself – the physical evidence
- Business requirements and user stories
- Results from debugging, testing, monitoring
- Feedback from customers or business analysts
- External sources, papers and publications

All evidence has limitations:

- Sources can be outdated or premature
- Sources can be unreliable – biased or false

Integrating Evidence as part of Engineering

Some engineers are more evidence-aware than others:

- Newbie engineers start with learning basics
- Average engineers must expand awareness
- Top engineers leverage their awareness

EBD supports continuous improvement.

Engineers cannot improve without considering what works and what doesn't work – without considering the *evidence*.

Evidence Collection and Assessment

Traditional Testing is limited evidence collection:

- Creates structured and specific results
- Narrowly focused on requirements or use cases
- Typically performed out-of-band from development
- Typically limited coverage – less than 100%

Exploratory Testing is closer to EBD practices:

- Dynamic evidence assessment
- Faster and more dynamic testing practices
- Coverage still limited to test time and skills/talents

How does EBD build quality into the process?

- Evidence is continuously gathered and assessed
- Critical evaluation of the evidence from the start
- Inherently includes 100% coverage
- Focuses on what works – highlights limitations.
- Reduces turnaround times – re-work, fixes, etc.
- Reduces number of inputs – reduced effort/risk
- Increases reusability – boosting test automation ROI
- Changes the way we think about testing and quality

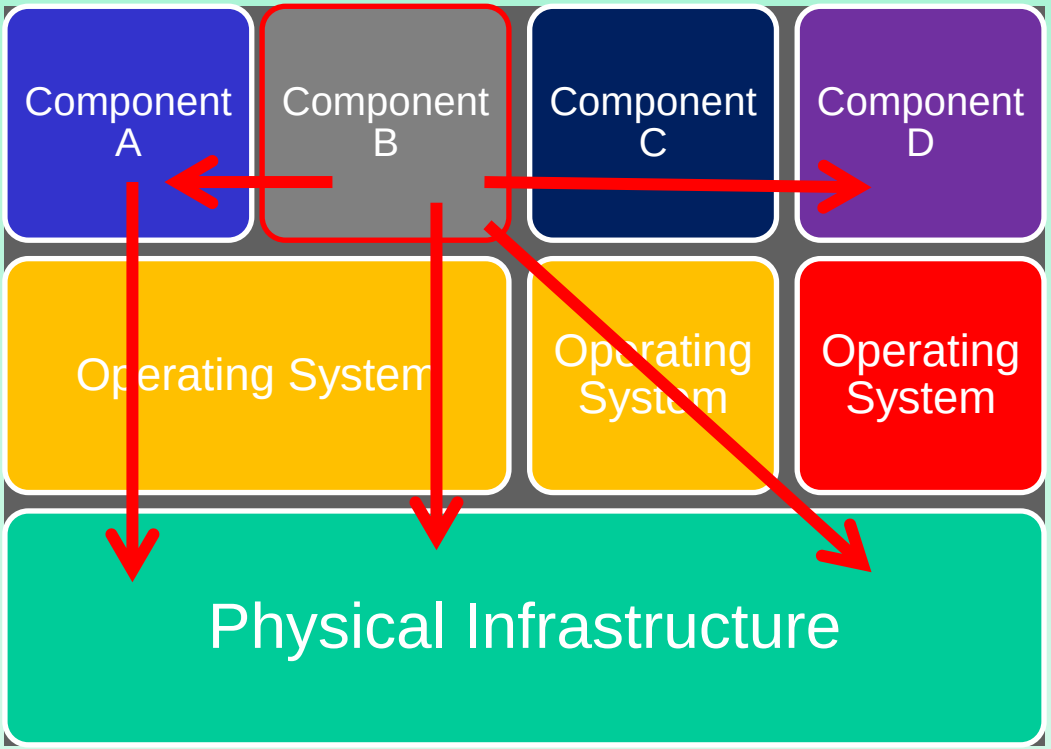
Applying Evidence as part of Engineering

Components are dependent on the underlying infrastructure

Existing Components are unchanged, but still consume resources

Evaluating evidence in real-time is faster and cheaper

- **Objective**: build component B
- **Confirm** the capacity and evidence about dependent components
- **Confirm** required capacity for Component A
- **Build and Validate** Component B processing
- **Catalog** new evidence (test results) for Component B



Implementing Evidence-based Development

Suggestions for implementing EBD

EBD requires a change to the way you think about testing:

- Testing now becomes “critical thinking about evidence”
- Evidence discovery replaces traditional test design
- Testing occurs across the entire app lifecycle, at anytime
- Testing becomes a shared activity, not an isolated practice
- Large testing scopes must be divided into smaller units
- Test automation becomes easier and more essential

Because evidence evaluation takes time you need smaller and more intelligently flexible testing units. Micro-tests.

Micro-testing is essential to EBD success

Micro-tests are really small units of testing:

Robb sez: "Just to start, think small. Then think smaller."

Micro-tests are faster, cheaper and easier to write

- Limited inputs and outputs and verification
- Limited variations to the logic and processing

Micro-tests are easier to combine into test suites

- Well-suited to modern test automation solutions

They are defined to test very basic operations:

- Search, Select, Create, Update and Delete

How to understand Micro-testing

Micro-tests are operationally separate component tests which have very little variation and are easily standardized into template-driven test design.

- **Search:** tests only vary by security, size, criteria and return object(s)
- **Select:** tests only vary by return type
- **Create:** varies by security, validation, business logic and notifications
- **Update:** varies by security, validation and business logic
- **Delete:** varies by security, validation and business logic

Applications built of reused components can be developed and tested more quickly because the only adjustment required will be for the exceptional variations of the implementation.

Roles and responsibilities change with EBD adoption

- Architects ratify their system design with external evidence
- Business Analysts define requirements based on evidence
- Developers build according to known and unknown evidence
- Testers validate system behavior in the context of evidence
- End-users generate strong evidence of failure and/or success
- Managers enforce evidence-based discipline and practices

Since evidence is ubiquitous and shared, everyone is enabled to and capable of capturing and evaluating evidence at all times.

Processes and measures change with EBD adoption

- Adopt “agile-like” development practices – be smaller, faster
- Develop standards for collection and cataloging of evidence
- Gather requirements continuously, dynamically
- Update essential operating components quickly, reliably
- Measuring performance metrics throughout lifecycle
- Collect metrics on reuse of components and evidence
- Measure real-world impacts – go beyond user acceptance

Measure what works for the business advantage, and learn from what’s not working. Adapt and improve, quickly.

Summary

- IT has reached a new low in competitive advantage
- When IT doesn't matter, focus is on reducing budget
- Development and testing practices continue to fail
- EBD is one way to restore IT competitive advantage
- Testing methods must adapt to a new standard
- Implementing EBD means changes to roles
- Implementing EBD means changes to processes

Questions?

FOLLOW US ON TWITTER

www.twitter.com/redhatsummit

TWEET ABOUT IT

#summitjbw

READ THE BLOG

http://summitblog.redhat.com/