Volume 5, Issue 11
November / December 2002—
The Software Quality Advisor

The Software Quality Advisor Online

Testing COTS-based Applications

In this article, we will examine the challenges and solution strategies in testing Commercial Off-the-Shelf software applications.

We will look at issues such as:

- Vendor test responsibilities
- Customer test responsibilities
- How to know the quality of vendor testing
- How to know what to test when a package has a huge amount of functionality

This article will also reference a downloadable COTS testing workbench, which you can use as a basis for your own COTS testing process.

Download the COTS Testing Workbench

Inside this issue:

Testing COTS-based Applications

Randall W. Rice, CSQA, CSTE

Commercial Off-the-Shelf (COTS) software is becoming an ever-increasing part of organizations' total IT strategy for building and delivering systems. A common perception held by many people is that since a vendor developed the software, much of the testing responsibility is carried by the software vendor. However, people are learning that as they buy and deploy COTS-based systems, the test activities are not necessarily reduced, but shifted to other types of testing not seen on some in-house developed systems.

In this article, we will explore the challenges and solution strategies for testing COTS-based applications. We will also see a process for testing COTS-based applications.

A Case Study

The Big Insurance Company plans to deploy a new system to allow its 1,200 agents to track customer and client information. Instead of writing its own application, the company has chosen to buy a site license of a popular contact management application. The solution appears to be cost-effective, as the total cost of the software to be deployed to all agents will be about $100,000 as compared to an in-house development estimate of $750,000. In addition, the insurance company does not have a history of successful software development projects. There are, however, some considerations that the company realized after they made the purchase decision:

- New versions of the application will be released each year.
- An annual maintenance fee will be required for vendor support.
- The interfaces between the contact management software and the office suite currently being used are not as seamless as originally thought.
- There are some computers being used by agents that are too old or too new for the software
- Each agent has an existing client contact database of about 1,000 people, but the data is stored in a variety of products and database formats.

In planning the purchase and deployment of the application, the project manager allocated ample time to perform a pilot deployment with 10 field agents using a

(Continued on page 2)

Book Review—*Questioning Extreme Programming* by Pete McBreen

Overview

This book examines Extreme Programming and asks very probing questions about an approach that has generated a lot of ardent followers and a lot of controversy.

What I Liked About the Book

- **The Author's Background**

Pete McBreen is a great choice to author this book because he is not so far out on the process-driven spectrum as to immediately cause people to dismiss his ideas. McBreen's previous work, *Software Craftsmanship*, is a thought-provoking book on software development without heavy processes.

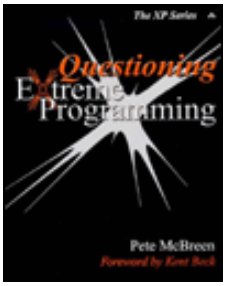
McBreen has a firm grasp on writing clean code that works, and has the experience to back it up.

- **The Author's Approach**

McBreen does not immediately dive into a critique of XP, but first lays out a great discussion on methodologies, what they do, and provides a survey of the more commonly used software development approaches.

Only after laying a solid foundation does McBreen examine XP one tenet at a time.

Actually, I found the discussion on methodologies so well presented that I would recommend the book just for the purpose of understanding development approaches.



- **The Scope of Coverage of XP Topics**

(Continued on page 2)

Book Review—*Questioning Extreme Programming*

(Continued from page 1)

McBreen takes the major tenets of XP one at a time and examines them in light of their origin, real-world projects, and their meaning for the XP.

- **Adequate detail**

McBreen makes his case with enough detail to cause people to take his questions and observations about XP seriously. I have seen articles about XP that criticized the approach, but failed to back up their points with substantial information.

- **Kent Beck's Foreword**

I think it says a lot for an author to write the forward of a book that critiques their work. Beck makes it clear that he doesn't agree with all of McBreen's conclusions, but he also welcomes people to think about the way they develop software.

Audience

The overall audience for this book is those people who have heard about XP and wonder about its viability in their environment and with their projects. The book is suited for people in the following roles:

- Senior IT management
- Software developers
- Project leaders and team leaders
- Software architects
- QA analysts
- Testers

Topics and Outline

Some of the questions explored in this book include:

- Is the cost of change really low?
- Does XP do proper testing?
- Does XP make sense?
- Is XP a return to the dark ages?
- What can other approaches learn from XP?
- Do you need process improvement or process change ?
- Why are developers so zealous about adopting XP?
- Is XP suitable for your projects?
- What is the next step after Extreme Programming?

The book addresses:

Pt. I Introduction

Ch. 1 XP: Hype or Hyper Productive

Pt. II What Is a Methodology?

(Continued on page 5)

Testing COTS-based Applications

(Continued from page 1)

variety of computers. Initial feedback from the 10 agents indicated that the new application worked correctly, but some tasks were hard to understand and perform. Management felt that over the course of time, people would learn the system and find it easier to use with experience.

The deployment plan was to have all agents download and install the new application over a weekend. Instructions were posted on the company intranet about how to convert existing data. A help line was established to provide support to the agents.

On deployment weekend, 98% of the agents downloaded the new software and installed it on their notebook computers. About 20% of the agents had problems installing the software due to incompatibilities with hardware and operating systems. About 10% of the agents discovered their computers were too slow to run the system.

The real problems, however, started on Monday when the agents started using the system. Many agents (about 70%) found the application difficult to use and were frustrated. In addition, all of the agents found that the new application could not perform some of the functions the old contact databases would. Fortunately, many of the agents kept their old contacts database.

After four weeks, the company decided to implement another product, but this time more field testing was performed, other customers of the product were referenced, and more extensive testing was performed for interoperability, compatibility, correctness, and usability. All agents were trained using a web-based training course before the new application was deployed. The second deployment was a huge success.

In the first project, the results were:

- A loss of time and productivity for the agents
- A loss of credibility for the project team and the IT department
- A loss of sales as agents could not use the system to follow-up with prospects quickly

In the second deployment,

- The initial product was more usable and had more useful features
- Agents were trained to avoid confusion in how to use the product
- Testing was more complete, which gave a higher level of confidence in deploying the application

In comparing the deployments, the company learned that:

- Application features are just one aspect of the product's quality.
- End-users must understand how to use the product.
- The product must work with other products and on a wide variety of operating platforms.
- Although the vendor tested the product, the customer has responsibility to test items the vendor can't test.
- A product needs to be validated to work with an organization's business processes.

This case study is not a true story, but it is based in representative projects I have seen in acquiring and deploying COTS products. From this example, we can see the need for testing, but what are the issues in COTS testing and how do we solve them?

Unique Challenges of Testing COTS-based Applications

Challenge #1 - COTS is a Black Box

The customer has no access to source code in COTS products. This forces testers to adopt an external, black-box, test approach. Although black-box testing is certainly not foreign to testers, it limits the view and expands the scope of testing. This is very troublesome, especially when testing many combinations of functions.

(Continued on page 3)

Testing COTS-based Applications

(Continued from page 2)

Functional testing is redundant by its very nature. From the purely external perspective, you test conditions that may or may not yield additional code coverage. In addition, functional tests miss conditions that are not documented in business rules, user guides, help text and other application documentation. The bottom line is that in functional testing, you can test against a defined set of criteria, but there will likely be features and behavior that the criteria will not include. That's why structural testing is also important. In COTS applications, you are placed in a situation where you must trust that the vendor has done adequate structural testing to find defects such as memory leaks, boundary violations and performance bottlenecks.

Although black-box testing is certainly not foreign to testers, it limits the view and expands the scope of testing. This is very troublesome, especially when testing many combinations of func-

Solution Strategies

Avoid complex combinations of tests and the idea of "testing everything." Instead, base tests on functional or business processes used in the real world environment. The initial tendency of people in testing COTS applications is to start defining tests based on user interfaces and all of the combinations of features. This is a slippery slope which can lead to many test scenarios, some meaningful and others with little value.

Challenge #2 - Lack of Functional and Technical Requirements

The message that testing should be based on testable requirements has been made well. Requirements-based testing has been taught so much, however, that people are forgetting about how to test when there are no requirements or to take other approaches on testing. Testing from the real-world perspective is called validation, and validation is the kind of testing that is primary in a customer or user's test of a COTS product.

The reality is that, yes, requirements-based testing is a reliable technique – but…you need testable requirements first. In COTS you may have defined user needs, but you do not have the benefit of documents that specify user need to the developer for building the software. In fact, the developer of the software may not have the benefit of documented requirements for tests, either. For the customer, this means you have to look elsewhere for test cases, such as:

- Exploring the application
- Business processes
- User guides

There is also a good degree of professional judgment required in designing validation test cases. Finding test cases is one thing. Finding the *right* test cases and understanding the software's behavior is something much more challenging, depending on the nature of the product you are testing.

Solution Strategies

- Design tests that are important to how you will use the product. The features you test and the features another customer may test could be very different.
- Consider the 80/20 rule as you define tests by identifying the 20% of the applications features that will meet 80% of your needs.

Challenge #3 - The Level of Quality is Unknown

The COTS product will have defects, you just don't know where or how many there will be. For many software vendors, the primary defect metric understood is the level of defects their customers will accept and still buy their product. I know that sounds rather cynical, but once again, let's face facts. Software vendors are in business to make a profit. Although perfection is a noble goal and (largely) bug-free software is a joy to use, a vendor will not go to needless extremes to find and fix some defects. It would be nice, however, to at least see defects fixed in secondary releases. Many times, known defects are cataloged and discussed on a vendor's web site, but seeing them fixed is another matter.

This aspect of COTS is where management may have the most unrealistic expectations. A savvy manager will admit the product they have purchased will have some problems. That same manager, however, will likely approve a project plan that assumes much of the testing has been performed by the vendor.

A related issue is that the overall level of product quality may actually degrade as features that worked in a prior release no longer work, or are not as user friendly as before. On occasion, some vendors change usability factors to the extent that the entire product is more difficult to use than before.

Solution Strategies

- Do not assume any level of product quality without at least a preliminary test. A common strategy is not to be an early customer of a new release. It's often wise to wait and see what other users are saying about the product. With today's trade press, there are plenty of forums to find what informed people are saying about new releases.
- Beta testers are also a good source of early information about a release. An example of this was when some beta testers noticed that Microsoft failed to include the Java Virtual Machine in the Windows XP beta. Prior to the revelation, Microsoft had not indicated their intention. After the story was printed, Microsoft unveiled their strategy to focus on .Net. Just recently, a judge ruled that Microsoft must include Sun's Java Virtual Machine in Windows and Internet Explorer. For more on this story, go to http://computerworld.com/developmenttopics/development/java/story/0,10801,77050,00.html.

Challenge #4 - Unknown Processes and Methods

Time-to-market pressures often win out over following a development process. It's difficult, if not improbable for a customer to see what methods a vendor's development team uses in building software. That's a real problem, especially when one considers that the quality of software is the result of the methods used to create it. Here are some things you might like to know, but probably will not be able to find out:

- Were peer reviews used throughout the project?
- How experienced are the developers?
- Which phases of testing were performed?

(Continued on page 6)

Links

Hackproofing Your Web Applications - Or, How I Learned to Start Worrying and Love Security
http://www.linux-mag.com/2002-09/security_01.html

Planning for a Metro-Area Armageddon - Regional data center fail-over sites just won't cut it in a post-9/11 world.
http://www.computerworld.com/securitytopics/security/story/0,10801,76430,00.html

FBI: Hacker stole 80,000 credit cards
http://www.cnn.com/2002/TECH/internet/12/09/israel.hacker.ap/index.html

Audio files figure in latest Microsoft vulnerability
http://www.computerworld.com/securitytopics/security/story/0,10801,76935,00.html

Going Mobile, Staying Secure
http://www.computerworld.com/computerworld/records/whitepapers/ciscosuppv2.qxd.pdf

Making Passwords Passe
http://www.computerworld.com/securitytopics/security/story/0,10801,60446,00.html

Four Personal Firewalls Reviewed
http://www.computerworld.com/securitytopics/security/story/0,10801,60821,00.html

24 Hours Not Enough? Work Smarter for Your Business
http://www.officemax.com/max/solutions/product/article.jsp?BV_UseBVCookie=yes&expansionOID=-536892514&edOID=536979981&cArea=bus

Teletips for Handling Outgoing Calls
http://www.officemax.com/max/solutions/product/article.jsp?BV_UseBVCookie=yes&expansionOID=-536892514&edOID=536948610&cArea=bus

Effective Ways to Improve Your Small Business Web Site
http://www.officemax.com/max/solutions/product/article.jsp?BV_UseBVCookie=yes&expansionOID=-536892514&edOID=536915616&cArea=bus

Quotes

"Equal opportunity means everyone will have a fair chance at being incompetent."
Laurence J. Peter (1919 - 1988)

"Everyone rises to their level of incompetence."
Laurence J. Peter (1919 - 1988), "The Peter Principle"

"The incompetent with nothing to do can still make a mess of it."
Laurence J. Peter (1919 - 1988)

"Nothing astonishes men so much as common sense and plain dealing."
Ralph Waldo Emerson (1803 - 1882), 'Art,' 1841

"If an idea's worth having once, it's worth having twice."
Tom Stoppard (1937 -)

"Common sense is the collection of prejudices acquired by age eighteen."
Albert Einstein (1879 - 1955) (attributed)

"Great spirits have always found violent opposition from mediocrities. The latter cannot understand it when a man does not thoughtlessly submit to hereditary prejudices but honestly and courageously uses his intelligence."
Albert Einstein (1879 - 1955)

"It has yet to be proven that intelligence has any survival value."
Arthur C. Clarke (1917 -)

"So far as I can remember, there is not one word in the Gospels in praise of intelligence."
Bertrand Russell (1872 - 1970)

"There is nobody so irritating as somebody with less intelligence and more sense than we have."
Don Herold

"Try not to become a man of success but rather to become a man of value."
Albert Einstein (1879 - 1955)

Questions From the e-Mail Bag

Q: What is the difference between client/server and web testing?

A: There are many differences, but here are a few major things to pay attention to. Of course, there are many ways to implement a web application (on the world wide web, intranets, extranets, etc.).

1. An unpredictable user audience.

You don't know for sure many times who your users are, how much computer usage experience they have, when they will show up on your site, where they will go, and how they will use your web site or application.

This means that for testing web apps you will need to perform usability testing based on your potential user audience. You will need testers that match the profile of your intended users the best you can define them. So, if your site is aimed at 13 to 18 year old boys, your usability testers need to be 13 to 18 year old teenage boys.

For more information on web usability, check out:

www.useit.com

2. An unpredictable user interface.

With client/server apps, you have a consistent user interface (UI) in terms of appearance and functionality. On the web, your UI is a browser, which varies by vendor,

(Continued on page 5)

Questions from the Mail Bag

(Continued from page 4)

version, and O/S platform.

This means you will need to perform compatibility testing on the platforms most likely to be used by your users.

3. Security issues

The web gives people an opportunity to gain access to your information through web servers, firewalls, viruses, web apps and also non-intrusion methods, such as denial-of-service attacks. Client/server applications had security issues, but not to this extent.

You will need to verify the existence of web security policies and protocols, determine the compliance to them, and also validate their effectiveness.

4. Performance Issues

True, we've always done load testing, but on the web the audience can be much larger and more difficult to predict. Tools are required to perform this type of testing with any level of reliability.

5. Rapid testing timeframes

This is what many people refer to as testing on "internet time." In these short time frames, changes are often introduced quickly and without a process of configuration management. Therefore, a risk in web applications is that testing may be shortcut and that things might be missed during implementation.

There are certainly other differences, but one thing that both have in common is correctness. No matter how fast, usable and secure a site may be, it still needs to be correct.

Q: What is the difference between verification and validation?

A: Verification is testing based on specifications. It is a producer-oriented view of testing and focuses on ensuring a product is built as designed.

Examples of verification could be reviews, inspections, unit testing, integration testing and system testing.

Validation is testing based on real-world usage. Validation is user or customer focused and seeks to ensure that the right product has been build or bought.

An example of validation is user acceptance testing that is not based on requirements, but on how the users will apply the application.

Book Review—*Questioning Extreme Programming*

(Continued from page 2)

Ch. 2 What Do Methodologies Optimize?
Ch. 3 What Are XP Projects Scared Of?
Ch. 4 What Do Other Methodologies Consider Important?
Ch. 5 What Is Important for Your Projects?

Pt. III Questioning the Core XP Practices

Ch. 6 Planning Incremental Development
Ch. 7 Truly Incremental Development
Ch. 8 Are We Done Yet?
Ch. 9 Working at This Intensity Is Hard
Ch. 10 Is That All There Is to Extreme Programming?

Pt. IV Questioning XP Concepts

Ch. 11 The Source Code Is the Design?
Ch. 12 Test First Development?
Ch. 13 Large-Scale XP?
Ch. 14 Is the Cost of Change Really Low?
Ch. 15 Setting the Dials on Ten
Ch. 16 Requirements: Documentation or a Conversation?
Ch. 17 Is Oral Documentation Enough?
Ch. 18 Playing to Win?

Pt. V Understanding the XP Community

Ch. 19 Really Strange Sayings
Ch. 20 Feel the Hostility; Experience the Joy
Ch. 21 Transitioning Away From Extreme Programming

Pt. VI Your Choice

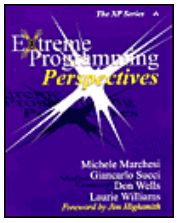
Ch. 22 Is XP for You?
Ch. 23 Do You Have a Suitable First Project?

Great Quote

"One way of looking at a methodology is that it is the way an organization armors itself against past failures. Although good organizations celebrate and learn from failure, in practically every organization, making the same mistake twice is a career-limiting move. Seen in this light, what a methodology does is make it hard to repeat dumb, career-limiting mistakes."

Good Companion Books

Extreme Programming Perspectives, by Giancarlo Succi,James Donovan Wells,Michele Marchesi,Laurie Williams



Summary

I can highly recommend this book to anyone in an organization that is considering agile methods, in particular XP. When it comes to software development methodologies, a common practice is to have high expectations of the latest approach. Seldom are the questions asked concerning applicability to a particular environment and organization. This book will lead you to ask the right questions.

Reviewed by Randall W. Rice

Testing COTS-based Applications

(Continued from page 3)

- Which types of testing were performed?
- Are test tools used?
- How are defects tracked?
- How do developers collaborate on projects?
- How are product features conceived and conveyed to developers?
- What type of development methodology is used?
- Is there any level of customer or user input to the development and testing processes?

Solution Strategies

- This is a tough issue to deal with because the vendors and their staffs do not want to reveal trade secrets. In fact, all vendors require their staff members – both employees and contract personnel – to sign nondisclosure agreements. Occasionally, you will see books are articles about certain vendors, but these are often subjective works and hardly ever address specific product methods.
- Independent assessments may help, but like any kind of audit or review, people know what to show and what to hide. Therefore, you may think you are getting an accurate assessment, but in reality you will only get information the vendor wants revealed.

Challenge #5 - Compatibility Issues

Software vendors, especially those in the PC-based arena, have a huge challenge in trying to create software that will work correctly and reliably in a variety of hardware and operating system environments. When you also consider peripherals, drivers, and many other variables, the task of achieving compatibility is impossible. Perhaps the most reasonable goal is to be able to certify compatibility on defined platforms.

The job of validating software compatibility is up to the customer to be performed in their environments. With the widely diverse environments in use today, it's a safe bet to assume that each environment is unique at some point.

Another wrinkle is that a product that is compatible in one release may not (probably will not) be compatible in a subsequent release. Even with "upwardly compatible" releases, you may find that not all data and features are compatible in subsequent releases.

Finally, be careful to consider compatibility between users in your organization that are using varying release levels of the same product. When you upgrade a product version, you need a plan that addresses:

- When users will have their products upgraded
- Which users will have their products upgraded
- Hardware and other upgrades that may be needed
- Data conversions that may be needed
- Contingency plans in case the upgrade is not successful

Solution Strategies

- Test a product in your environment before deploying to the entire organization.
- Have an upgrade plan in place to avoid incompatibility between users of the same product.

Challenge #6 - Uncertain Upgrade Schedules and Quality

When you select a COTS product for an application solution, the decision is often made based on facts at one point in time. Although the current facts about a product are the only ones that are known and relevant during the acquisition process, the product's future direction will have a major impact in the overall return on investment for the customer. The problem is that upgrade schedules fluctuate greatly, are impacted by other events such as new versions of operating systems and hardware platforms, and are largely unknown quantities in terms of quality.

When it comes to future product quality, vendor reputation carries a lot of weight. Also, past performance of the product is often an indicator of future performance. This should be a motivator for vendors to maintain high levels of product quality, but we find ourselves back at the point of understanding that as long as people keep buying the vendor's product at a certain level of quality, the vendor really has no reason to improve product quality except for competing with vendors of similar products.

Solution Strategies

- Keep open lines of communication with the vendor. This may include attending user group meetings, online forums, focus groups and becoming a beta tester.
- Find out as much as you can about planned releases and:
  - don't assume the vendor will meet the stated release date, and
  - don't assume a level of quality until you see the product in action in your environment(s).

Challenge #7 - Varying Levels of Vendor Support

Vendor support is often high on the list of acquisition criteria. However, how can you know for sure your assessment is correct? The perception of vendor support can be a subjective one. Most people judge the quality of support based on one or a few incidents.

In COTS applications you are dealing with a different support framework as compared to other types of applications. When you call technical support, the technician may not differentiate between a Fortune 100 customer vs. an individual user at home.

Furthermore, when you find defects and report them to the vendor, there is no guarantee they will be fixed, even in future releases of the product.

Solution Strategies

- Talk to other users about their support experiences, keeping in mind that people will have a wide variety of experiences, both good and bad.
- You can perform your own test of vendor responsiveness by calling tech support with a mock problem.

Challenge #8 - Difficulty in Regression Testing and Test Automation

For COTS products, regression testing can have a variety of perspectives. One perspective is to view a new release as a new version of the same basic product. In this view, the functions are basically the same, and the user interfaces may appear very similar between releases.

Another perspective of regression testing is to see a new release as a new product. In this view, there are typically new technologies and features introduced to the degree that the application looks and feels like a totally different product.

The goal of regression testing is to validate that functions work correctly as they did before an application was changed. For COTS, this means that the product still meets your needs in your environment as it did in the previous version used. Although the functions may appear different at points, the main concerns are that:

- Features you use often have not been dropped

(Continued on page 7)

Testing COTS-based Applications

(Continued from page 6)

- Performance has not degraded
- Usability factors have not degraded
- New features do not distract from core application processes
- New technology does not require major infrastructure changes

It's hard to discuss regression testing without discussing test automation. Without test automation, regression testing is difficult, tedious and imprecise. However, test automation of COTS products is challenging due to:

- Changing technologies between releases and versions
- Low return on investment
- The large scope of testing
- Test tool incompatibility with the product

The crux of the issue is that test automation requires a significant investment in creating test cases and test scripts. The only ways to recoup the investment are:

- Finding defects that outweigh the cost of creating the tests
- Repeating the tests enough times to outweigh the manual testing effort

While it is possible that a defect may be found in the regression testing of a COTS product that may carry a high potential loss value, the more likely types of defects will be found in other forms of testing and will relate more to integration, interoperability, performance, compatibility, security and usability factors rather than correctness. This leaves us with a ROI based on repeatability of the automated tests. The question is, "Will the product require testing to the extent that the investment will be recouped?" If you are planning to test only one or two times per release, probably not. However, if you plan to use automated tools to test product performance on a variety of platforms, or to just test the correctness of installation, then you may well get a good return on your automation investment.

For the scope concern, much of the problem arises from the inability to identify effective test cases. Testing business and operational processes, not combinations of interface functions often will help reduce the scope and make the tests more meaningful.

Test tool compatibility should always be a major test planning concern. Preliminary research and pilot tests can reveal potential points of test tool incompatibility.

Solution Strategies:

- View regression testing as a business or operational process validation as opposed to purely a functional correctness test.
- Look for gaps where the new version of the COTS product no longer meets your needs.
- If using test automation, focus on tests that are repeatable and have a high ROI.
- Perform pilot tests to determine test tool compatibility.

Interoperability and Integration Issues

When dealing with the spider web of application interfaces and the subsequent processing on all sides of the interfaces, the complexity level of testing interoperability becomes quite high. Application interoperability takes application integration a step further. While integration addresses the ability to pass data and control between applications and components, interoperability addresses the ability for the sending and receiving applications to use the passed data and control to create correct processing results. It's one thing to pass the data, it's another thing for the receiving application to use it correctly.

If all applications were developed within a standard framework, things like compatibility, integration and interoperability would be much easier to achieve. However, there is a tradeoff between standards and innovation. As long as rapid innovation and time-to-market are primary business motivators, standards are not going to be a major influence on application development.

Some entities, such as the Department of Defense, have developed environments to certify applications as interoperable with an approved baseline before they can be integrated into the production baseline. This approach achieves a level of integration, but limits the availability of solutions in the baseline. Other organizations have made large investments in interoperability and compatibility test labs to measure levels of interoperability and compatibility. However, the effort and expense to build and maintain test labs can be large. In addition, you can only go so far in simulating environments where combinations of components are concerned.

Solution Strategies:

- Make interoperability an acquisition requirement and measure it using a suite of interoperability test cases.
- Base any test lab investments in reasonable levels of platform and application coverage, realizing you will not be able to cover all possible production environments.
- Prioritize interoperability tests to model your most critical and most often used application functions.
- Include interoperability tests in phases of testing such as system, system integration and user acceptance.

Summary

Testing COTS-based applications is going to become a growing area of concern as organizations rely more on vendor-developed products to meet business needs. Just because a vendor develops the product does not relieve the customer from the responsibility of testing to ensure the product will meet user and business needs. In addition, the product may work in some environments but not others.

Testing COTS products relies heavily on validation, which seeks to determine the correctness and fitness of use based on real-world cases and environments as opposed to documented specifications. Although aspects of the COTS product may be described in business needs and acquisition criteria, many tests of the product will likely be based in a customer's daily work processes.

Successfully testing COTS products is possible, but requires a different view of risk, processes, people and tools.

The Software Quality Advisor Online

November/December, 2002

P.O. Box 891284
Oklahoma City, OK 73189

405-793-7449
405-793-7454 Fax
rrice@riceconsulting.com

"Test everything. Hold onto the good."
I Thessalonians 5:21

We're on the Web!
www.riceconsulting.com

January 2003 Issue:

Ten Things to Think About for Software Quality in 2003 by Randall W. Rice

Book Review — Test-Driven Development by Kent Beck

Training at Your Desk, When You Need It!

Visit the Structured User Acceptance Testing Online Training Free Demo

This is a practical web-based course to convey effective methods to plan and conduct user acceptance testing. This is one of the few courses available that teaches a non-technical and easily learned process for testing computer systems from a business process perspective.

The content in the web-based course is exactly the same as presented in our popular three-day course. The value added is that you can take the course at your pace and at your location. The content is delivered by streaming slide shows, video clips and exercises. Complete course notes are downloadable in Adobe PDF format.

Available in full (11 modules) and fast-track (5 modules) versions!

Click Here to Go Directly to the Free Demo.

Calendar of Events

The following events in 2003 are still subject to change.

March 2003

Location - Omaha, NB

Topic - Web Testing (2 days)

April 2nd - 4th, 2003

Location - Chicago, IL

Topic - Security Testing for the Enterprise and the Web (3 days)

April 29 - May 1, 2003

Software Technology Conference

Salt Lake City, UT

May 7 - 9, 2003

Location - Kansas City

Topic - TBD

June, 2003

Location - Chicago, IL

Topic - Surviving the Top Ten Challenges of Software Testing Workshop (1 day)

September 10 - 12, 2003

Location - Chicago, IL

Topic - User-oriented Methods for Software Quality (3 days)

December 11 - 12, 2003

Location - Chicago, IL

Topic - Reviews and Inspections

If you have a group of 12 or more people in your city that would like to sponsor a training event, contact Randy Rice at rrice@riceconsulting.com to find out how to book a special presentation.