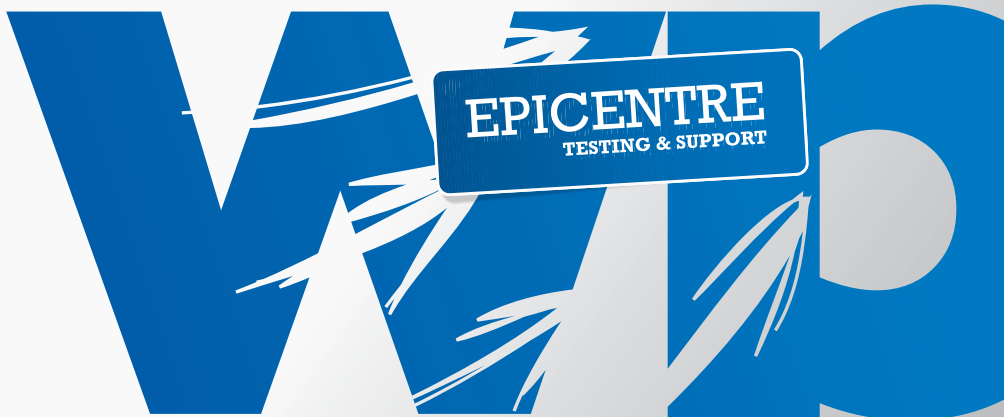


Effective software testing

An epicentre white paper



Visit us at www.epicentre.co.uk

This document is the property of Epicentre and must not be copied in whole or part, without the consent of the Company © 2010

Effective software testing

Effective software testing is crucial to delivering high-quality e-learning. Yet for many, testing is an undervalued and poorly performed task. Even those who do value testing sometimes fly in the face of wisdom and reduce testing time.

Lodged tightly between the main development phase and the delivery deadline, any delays in development put the testing phase under pressure. Something has to give, and it's usually in the form of testing days rather than a deadline extension.

There are other reasons for the pressures on testing. Project complexities and challenges can be great, time short and budgets low. The urge to cut back on something is always there, and whether in e-learning, web or any other software, it is very often the case that testing gets cut first. After all "*We can test it ourselves*"; "*The content reviewers*

can find bugs"; "*The code we're reusing is already stable*", and so on.

But as any tester will know, while testing is the first to get cut, it is also the first to receive knee-jerk reactions from managers when irate customers start calling. This usually takes the form of "*Why on earth didn't the test team find this?*" or "*Are we employing a bunch of monkeys as testers?*". It's also usually the case that testing had actually put in repeated requests for more time which were repeatedly ignored. Ask any tester: It's the story of their life!

Of course it doesn't have to be like that. There are ways and means of achieving successful testing and high-quality deliverables while still handling the pressures and complexities of modern projects successfully. In this updated White Paper we look at how to achieve management buy-in for testing, how to adopt a more cost effective and structured test process and when to call in the experts.

The need for rigorous testing

By way of an introduction to testing, it is important to realise just how great the scope for error is.

An often used example throughout the testing industry comes from *The Art of Software Testing* by Glenford Myers, which points to a simple program containing 20 lines of code, including a loop and three IF statements. It was calculated there were 100 trillion execution paths, which would take a billion years to test! Consider that traditional e-learning features such as question and test sections, scoring and user tracking all require fairly complex programming, and the scope for error in e-learning is clearly huge. It takes a well-trained testing professional to determine how such products should be tested.

The effects of inadequately tested e-learning can be devastating. In IBM/Rockwell's landmark study on the economics of testing, they discovered that a bug found post-release costs on average 100 times more to fix than a bug found early on in development. That cost could include helpdesk support, users unable to complete training, negotiation over re-work with developers, additional re-work cycles, and another internal rollout of the training product. For a high-profile e-learning course in a large organisation, this cost can run into hundreds of thousands. There is overwhelming evidence regarding the cost effectiveness of testing. Yet we see all too often during this crucial stage of quality control that:

- Testing is done in an adhoc manner
- Testing is done late in the development cycle

- Corners are cut by reducing the amount of testing time
- Testing is performed on the cheap using junior, unskilled staff

The pressures on testing are getting ever greater with the rise in popularity of 'rapid e-learning', with all the inherent time and budget challenges that brings. This White Paper looks at all these issues and will help you get your e-learning programme off on the right foot.

The value of testing

Getting developers and managers to accept The value of testing is often the biggest hurdle for the test team to overcome, and different approaches need to be used when dealing with managers and developers.

Gaining management buy-in for testing is fairly straightforward. They need to be made aware that testing is simply an exercise in risk management.

The risk of delivering e-learning with serious faults may result in:

- Ineffective training time
- Staff having to perform training more than once
- Unachieved accreditation
- Decreased perception of training within the organisation
- Loss of reputation

Buy-in from the development team can be trickier: Life on the development coal face is complex and challenging. As far as the development team are concerned, the purpose of testing is less spectacular, but no less important - it is to find problems so that they can be fixed, to deliver a product that meets its requirements.

Put simply, testing is a necessity. Software is highly likely to have faults, and the development team need to know information about its reliability so that decisions can be made regarding its readiness for delivery.

How much of the development budget?

This is the eternal question asked of testing professionals and is never easy to answer. In the wider software industry, the answer is anything between 10% and 60% of development time, and directly related to the risk

associated with shipping bugs. The 60% end of the spectrum is for safety-critical software such as life support or air traffic control systems, while those developing low-risk software will populate the lower end of the spectrum.

Clearly e-learning is at the low end of this spectrum. While a programme manager may dispute that in terms of training and business risks, we need to be clear that no-one is going to die as a result of any courseware errors! It is our belief that with the right buy-in and processes in place, 10-15% of development time is a good aim for e-learning programmes if the team wishes to be truly confident about the quality of the programme and that it has been tested to the best of the test team's ability.

Early involvement

One of the biggest issues in managing testing risk is managing the time and budget allocated to the task of testing. One of the most effective ways of avoiding suffering cutbacks in testing is for the test team to have 'early involvement' in the development process. As we have seen, bugs

found early are cheaper to fix, yet many development teams test late and expensively.

Early involvement means that the test team can plan for testing effectively. As a result of this, when the product is ready for testing the test team can hit the ground running, knowing exactly what they need to do. This doesn't just mean writing a test plan early on in the project. It also implies that:

- The test team is represented at key meetings
- The lead tester is considered a core member of the project team
- The lead tester helps to set achievable testing budgets and timeframes
- The test team have had a chance to write a high-level test plan and detailed test scripts
- The test team are adequately familiarised with the product before testing starts in earnest

Project stakeholders can help manage the risk of low-quality deliveries by insisting that testing is not cut back; a good way to

do this is simply to ask to see the test plan and regular testing progress reports.

ROI of testing

Testing addresses your business risk and provides a valuable and objective measure of confidence in your e-learning. It is closely aligned with the goals of both the client and developer organisations.

Testing is an investment that reduces cost in several ways: It reduces helpdesk costs, ensures you get maximum value from your e-learning and keeps your e-learning cost effective.

The earlier testing is done, the greater the return on investment. IBM/Rockwell's study on the economics of testing demonstrated that a bug found and fixed early may cost just a few pounds to fix; the same bug found after coding will cost hundreds of pounds to fix. If found by the client it will likely cost thousands of pounds to fix and will also start costing the client money. Finally, if serious bugs are released in a product then the developer's entire

project budget or profit can be burned up along with the client facing extremely high costs, with possible long-term damage to reputations.

To determine the ROI of testing, consider the cost of testing and offset this against the cost of failure.

The cost of testing involves testing hardware costs, software costs, personnel costs and general admin costs. Each of these areas will have separate implementation, operational and maintenance costs. The developer will incur this cost, but typically this will be passed on to the client organisation in the development budget.

The cost of failure includes helpdesk support, high-level client/developer liaison, new production rework cycles, more testing at the client organisation to double check the rework, ineffective training time and staff repeating training.

The following template can be used to determine an approximate ROI for testing your own e-learning courses:

The cost of testing

Testing hardware costs (PCs, peripherals, server, etc.)

Testing software costs (operating systems, disc imaging software)

Personnel costs (e.g. 2 staff on £25,000 for 3 month project duration)

General admin costs (systems support, accounts overhead and office furniture for testing staff)

Total:

The cost of failure

Delay of launch

More management time

Couriers, communications and delivery media

Damage to e-learning initiative

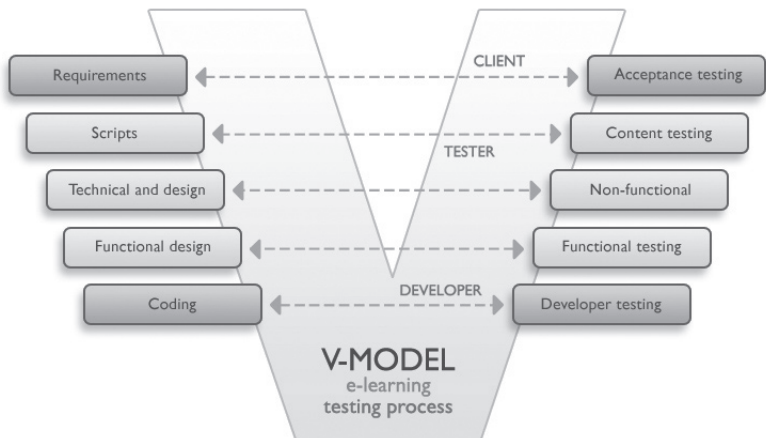
Support costs (cost per call)

Another round of acceptance testing

Total:

The testing process

The e-learning testing process can be split into a number of separate phases. The following table shows the traditional testing 'V-model' process as applied to an e-learning project.



Developer testing

Any developer worth their salt will thoroughly check their own work before passing it on to the test team. Developer testing means bugs are found and fixed as early and cheaply as possible. For developers it is an issue of reputation - nobody wants to get a name for writing buggy code (although as many a tester will acknowledge, they often do). However, such a reputation can often be rectified simply by fostering a 'right first time' attitude.

Peer review of code is also a powerful tool. Many programming teams have moved towards more agile processes which encourage more rigorous testing of code by writing tests into the code itself to exercise each piece of functionality as it is developed. This is called 'test-driven development' and is a key component of the agile programming methodology which advocates widely claim leads to improved code quality.

Functional testing

Functional requirements are those that can be explicitly defined by the client and developer, and relate to how the product is expected to work. Typical functional requirements would include things like 'user must login to system', 'user must choose module from list' and 'user must do pre-test at start of module'. These requirements should be specified in a functional design or specification document, which would be used as the basis for a detailed test plan.

The functionality of the product should be rigorously tested on at least one key platform of the required environment. The functionality test covers such areas as installation, uninstallation, navigation, audio, video, animations, quizzes, user tracking and more. This rigorous testing should be complemented by a phase of 'exploratory' or 'destructive' testing to try to break the program in unforeseen ways.

Your users need to be able to use your product on their

computer, and there are a lot of different software and hardware configurations in the marketplace. The first step is to know your target market, and perform research on the most likely configurations that your software will be used on, including hardware, operating systems and browsers.

If you are running e-learning at a large organisation, the chances are that you will use standard configurations on your laptops and desktop workstations. A multinational may use standard desktops in each country, to which native language operating systems will be thrown into the mix. However, it is also possible that a mix of hardware is in use, spanning several generations of technology, some of which may even be obsolete. This all needs to be taken into consideration when determining the project's requirements.

A more challenging scenario is when your e-learning is intended for home or mobile use – and the multitude of operating systems, mobile devices, hardware and browsers installed in the home - at which

point specialist compatibility testing services come into their own.

There are several design considerations to make in this scenario. You need to be able to second-guess the most popular configurations with some good research on your target market. You should also use a feature in the course launch page which detects the user's configuration and informs them if it is not suitable for the course they are trying to use. The test team should validate any claims to work on whatever configurations you decide to support.

Browser compatibility is an important area. There are many versions of Internet Explorer, Firefox, Safari, Chrome and other browsers. For each version, there are usually a range of language versions too. There are many hundreds of possible browsers out there, most of which have their own idiosyncrasies and will display HTML content slightly differently. Knowing your target market is therefore crucial in keeping development and

testing costs down. In practice, there are usually only a few supported browsers for e-learning aimed at public and private sector organisations where platforms are fairly well controlled, although very large organisations often have lots of legacy platforms from bygone days and require support for a wider range of browsers.

Operating System compatibility

is a similar story. For all the versions of Windows, Linux and Mac OS, each has multiple language versions available. An extensive operating system archive will stretch to hundreds of CDs.

There are ways around this. Some test labs use disk imaging software such as Ghost to enable hard disk 'images' to be loaded up from a bank of pre-configured test environments. The scenario below shows the setup at Epicentre, where the test lab contains user workstations simply consisting of keyboard, video and mouse, connected via switches to a bank of machines. This means that one workstation can access a wide array of hardware and

software configurations at the click of a button.

The amount and variety of possible configurations is impossibly vast, far greater than most development companies can either purchase or store, so it makes real sense to outsource compatibility testing if your target market is a varied one.

Hardware compatibility

is also a minefield. Turning the pages of any PC magazine will reveal dozens of popular manufacturers of PCs. But also consider the numerous types of processors manufactured by the likes of Intel and AMD, which go back years. Motherboard manufacturers, graphics and sound card manufacturers, CD and DVD drive manufacturers – the list reaches hundreds of names. And all these components are mixed up into different configurations in off-the-shelf and self-built machines from low-end home and office PCs to high-end gaming PCs.

Non-functional testing

Non-functional requirements are those that place some kind of constraint on the development team. Non-functional requirements have increased massively in recent years as e-learning projects have shifted to using web technologies; this has led to significant changes in associated project risk. Typical non-functional requirements cover e-learning standards, accessibility, usability, performance and security.

E-learning specifications and standards

Nearly all e-learning programmes being commissioned are now required to conform to one or more e-learning specifications and standards, typically SCORM 1.2 and 2004. Other standards from AICC and IEEE are required from time to time. These standards generally address the key areas of interoperability with Learning Management Systems (LMS) and the reusability of learning content.

There are various tools available to help test against these standards. However, this type of validation testing should be conducted by an e-learning testing specialist.

LMS/VLE interoperability

There are a large number of commercial and open source Learning Management Systems (LMS) and Virtual Learning Environments (VLE) currently available, providing various solutions for managing online learning. Most claim to be SCORM compliant and so should take any SCORM content happily. In practice, they all implement the SCORM standard slightly differently so there is no alternative to manually testing course upload, launch and tracking functionality on your target LMS/VLE.

Accessibility

Accessibility testing is vital to verify that your course can be used by people with a variety of disabilities. In some countries, people with accessibility needs make up 8-10% of the population. 52

million people have cognitive, visual, hearing and physical disabilities in the US alone, and can benefit enormously from e-learning which conforms to recommended accessibility standards.

Accessibility testing needn't be an expensive process as tool support is generally good. In particular, the IE Accessibility toolbar and Firefox Web Developer toolbar both provide useful suites of accessibility validation tools for standards compliance, colour blindness and more.

Tool support cannot be completely relied on however, and manual tests should be conducted using screen readers such as Jaws and Supernova, and screen magnifiers such as WindowsEyes.

More extensive tests should be conducted by disabled users themselves. No amount of testing tools and assistive technology tools can replace the knowledge and thoughts of a person accumulated through years of experience of using badly designed websites and software.

Usability

Usability testing is typically conducted on either prototypes – paper or screen based – or complete products. There are many techniques for usability testing, from ad hoc and informal to structured and highly formal. Among the most cost-effective methods that can be conducted in a reasonable time by your own design or test team are taped user observations in which the user thinks out loud as they use the program and try to perform some set task. Taped, group discussions around agreed topics at the end of a user trial are also useful.

The typical deliverable for usability tests includes a full written report with detailed recommendations for improving usability. This type of testing should be managed by an HCI (Human Computer Interaction) design expert with formal training in multimedia and HCI design.

Performance

Performance bottlenecks can occur at many points between the user and the server, including the company LAN, network hubs and switches, firewalls, internet connections, ISPs, internet exchanges and so on. The causes are generally high volumes of users that were not planned for. Performance testing is a counter measure, which can prevent potentially embarrassing and costly failures.

A common scenario is to have your LMS on an extranet or external host so that staff can access it from outside the office. You may find your external internet link is a bottleneck. We know of a public sector organisation with tens of thousands of staff which runs its e-learning through a 64Kbps internet connection. The resulting user experience is enough to put people off e-learning for life! Well scoped out requirements, a sensible hardware specification and adequate performance testing can prevent these problems before they occur.

Performance testing is a specialist task and the objectives should be clear to the test team. These may include performance tuning, verifying download times, determining response times, determining maximum numbers of users and investigating how loads greater than anticipated are handled.

This need not be a costly exercise as there are powerful open source performance testing tools available. However, it is a specialist discipline and appropriately skilled and experienced staff should be used.

Security

There are several security issues affecting the design of e-learning, such as use of cookies, writing user data to hard drives, use of browser popups, players and plugins, antivirus and firewall software running on end-user PCs, and plugin configuration such as Flash Player security settings.

Most organisations have security policies, which may

place significant constraints for e-learning designers and architects, in particular large organisations with centralised, remote systems support. It is vital that these are captured early in the project process.

Server security and privacy of personal data is also an issue. Learning Management Systems may store personal details of staff, which would be protected under the Data Protection Act. These servers obviously need to be secure, so security testing is a fundamental part of the test process.

Post-deployment availability

When e-learning is delivered and running nicely on a web server, who is responsible for ensuring that it stays that way? You may need to arrange a service level agreement (SLA) to cover course availability.

To this end, reliability and availability testing should be performed so that you have objective, quantifiable data that tells you your course is performing properly. This may be a simple task using website

monitoring tools, or may be a complex task performed by testing specialists; the needs are defined by the nature and profile of your e-learning course usage.

Content testing

Content testing is a high-priority issue in e-learning and can be split into on-screen and script tests.

On-screen content testing includes:

- Image validation
- Font validation
- Animation, audio and video validation
- Style guide adherence
- Text overruns
- Language testing
- Reviewing assets and content against the written script document

Script testing includes:

- Spell checking (by the script writer)
- Proof-reading (by someone independent of the project)
- Subject matter expert (SME) review

Acceptance testing

Project stakeholders should have someone perform a test to make sure they received what they paid for. This type of basic test to check the e-learning meets its high-level requirements is known as an acceptance test. If an agreed acceptance test fails then the course should be sent back to the supplier for re-work.

Managing risk through testing

If everyone on the test team, from test manager to tester, is aware of the relationship between testing and risk, then a real synergy is created between the test team and the goals of the project, the development company and the project's stakeholders.

- Each module assesses pre-course status and guides the users to the parts of the course of the most value to them
- Each module checks the user's understanding of all the learning objectives in that module
- Users' scores and progression through the course, along with the time taken to complete the module, is collected using a LMS.

The scope for error in e-learning

Consider the following description of a typical e-learning course:

- The course provides one hour of learning in modules
- Each module allows users to practise their skills and knowledge in interactive scenarios that are customised to mirror the real situations they will face
- Each module is topped and tailed by a quiz

This is a fairly typical example of an e-learning course. All this functionality represents a far more complex program than Myers' twenty lines of code with almost unlimited paths through the program. However, with a well-planned test process conducted by a dedicated and experienced test team, you can help ensure that your product is tested effectively, and significantly reduce the risk of any of your users finding bugs in the finished product.

Effective test planning

As we have demonstrated, testing helps to reduce risk substantially. Good test documentation processes and tool support during the planning stage is invaluable.

The test plan

The test plan is a high-level overview of the proposed test process. Among other things, it should contain an executive summary, overall approach, features to be tested, test deliverables, test environment, responsibilities and schedule.

For more information, refer to the IEEE Standard for Software Test Documentation (IEEE 829), available from the IEEE website.

Of particular importance, the test plan should identify the features of the e-learning course to be tested. With each of these features, the risk of failure is used to prioritise the testing. The consequence of the failure of these tests needs to be made as clear as possible to the project's stakeholders, so that the time and effort

required to test is bought into at all levels. In prioritising the tests through risk, the team has access to valuable information regarding how much testing should be done. In marking these prioritised tests as complete the project team is able to make all-important delivery decisions with confidence.

The schedule section also has a great effect on the success of the testing. The Holy Grail for test teams the world over is the phrase 'early involvement!' Testing performed late is expensive and ineffective; testing performed early is cost effective and an efficient use of resources. Testing early, with a good test plan, will make a major improvement towards delivering a quality product.

While the test plan is high-level, a more low-level and detailed test plan should be used by the testers as their brief throughout the test execution phase. This detailed plan is often called a test script. In its simplest form it would be an Excel worksheet on which the tester passes or fails each item as they progress

through the plan. Excel is better than Word for this task as it provides more scope for generating test metrics such as completion and pass/fail statistics.

Tool support

There is strong tool support in this area. High-end commercial tools such as TestDirector are rich in functionality and integrate with many other systems such as test automation tools, bug databases and requirements tools. At the other end of the scale there are open source tools which provide basic test management functionality such as test case creation, results and history tracking, management reports, data import and export and integration with bug databases.

Epicentre recently conducted an evaluation of open source test management tools. A shortlist of six tools was created:

- Bugzilla Testopia
- QA Project Manager
- QATraq

- RTH
- Salome-TMF
- Testlink

Each tool had the following attributes:

- Professional looking interface
- Good documentation
- Active developer community
- Stable versions released
- English language
- Referenceable adopters
- Supported by a stable organisation
- Third-party reviews published

We found that TestLink was the best of these. With over 50,000 downloads this is one of the most popular open source test management tools. It enables the creation, management and organisation of test cases into test plans, and mapping of test cases to software requirements. Setup is easy - we downloaded

and installed TestLink in 5 minutes.

TestLink integrates with the main open source defect trackers and can do requirements management/tracing as well as test case management. With good documentation, high and ongoing development activity, and major iterations every few months, a strong support community has become established. For a good introduction to online test management tools, you would be hard pressed to find a better alternative to this free and open source solution.

The runners up in our evaluation were Testopia (great if you already use the Bugzilla defect tracking system) and Salome TMF (if you don't mind installing and maintaining Java applications).

Bug reporting

During test execution, each individual tester has an important role to play in managing the business risk associated with software

failure. Testers report bugs so that they can be fixed, yet often developers will sideline reported bugs, deferring them to the next release, or marking them with 'ignore' or 'as designed'. If having lots of sidelined bugs sounds familiar to you, your testing process is probably not as effective as it should be and it is likely that bugs are slipping through the net to the release environment. Therefore, when writing each bug report, the tester must outline the most serious consequence of the bug in order to maximise the chances of it being fixed effectively.

There is plenty of tool support in the bug reporting area, in fact online bug tracking systems are pretty much *de rigueur* these days. Popular commercial tools include TestDirector, Jira, Tracker and TestTrack Pro, while the more well established open source tools include Bugzilla, Mantis, BugTracker.NET, Flyspray, BugNet and itracker.

Independent testing

Why development staff can't test

Good development staff – programmers, script writers and designers – are not necessarily good testers. Indeed, they generally tend to underestimate testing and overlook mistakes due to over familiarisation with the project.

A tester is more removed and, most importantly, has a different frame of mind. The tester will be actively looking for failure whereas the development staff will be verifying that their program works correctly. It is the former mindset that will find the most bugs.

The skills that testers and development staff require are also very different and often in opposition to each other. While the development staff have intimate product knowledge and expertise, the testers see only the interface and can disassociate themselves from

the inner workings of the product. They also often come to the project with a fresh perspective.

The tester is well positioned to model user behaviour. While testers appreciate the nuances of user behaviour, how often have you heard programmers nursing a belief that “*users are stupid*” or “*why would anybody do that?*”

Finally, development staff will often come across a valid bug but automatically think ‘as designed’ and would not report it as a bug, even though it could seriously hinder the end-user experience. Bugs reported by development staff rarely conform to bug reporting style guides, which often cause problems when testers come to retest them.

When to outsource testing

There are four situations when consideration should be given to test outsourcing:

1. **To augment your current QA effort.** Choose an outsourcing company whose services and expertise complement your own domain knowledge.
2. **To gain access to extra resources.** Many development teams simply don't have the resources required to test multiple simultaneous projects and releases, or have sudden resource shortages. Outsourcing reduces the effort of hiring extra staff to ramp up your testing. A long-term relationship with a test outsourcing team will bring the advantage that they already know your products and house style.
3. **To gain an independent view.** When your team has been working on a product for several months, a fresh set of eyes is an

invaluable addition. If you are applying for Windows logo conformance you will also be required to use an accredited company.

4. **Time considerations.** Test outsourcing can be done across time zones. For example, a US developer using a UK outsourcing company can upload a course at the end of their working day, the UK company can test it during the course of their normal UK working day, and send the bug reports back to the US in time for breakfast.

A successful outsourcing project will satisfy your quality, budget and time criteria and make a significant contribution to your overall efficiency. The outsourcing supplier should set realistic goals and expectations, provide regular progress reports, and either daily or real-time bug reports. As with development test teams, the golden rule of 'test early, test cheap' still applies to test outsourcing.

Epicentre.co.uk

Epicentre is the comprehensive software testing facility offered by Epic, the UK's leading e-learning provider. We provide:

- Twenty years' experience of testing
 - Dedicated account manager, project manager and test team for each project
 - Extensive experience in LMS interoperability, e-learning standards and web accessibility
 - Ability to resource both personnel and technology on demand
 - Demonstrable quality and value for money
 - Highly developed process-driven approach with ISO9001:2000 accreditation
 - Strong financial position and resource base
 - Ability to respond pro-actively to urgent deadlines
- Test Lab consisting of a wide range of hardware configurable to virtually any known operating system.

Our services include:

- E-learning standards testing
- Educational publishing test experts
- LMS interoperability and compatibility
- Web accessibility
- Functional testing
- Compatibility testing
- Localisation
- Usability testing
- Mobile/Handheld testing
- Network testing

For more information, please visit Epicentre at www.epicentre.co.uk or call us on 01273 728686.

Appendix

Testing for Mobile Platforms and Handheld Devices

The recent surge in popularity of smartphones, PDAs and other portable handheld devices such as the Nintendo DS and iPod, means that people now have virtually constant access to email and the internet, and therefore documents, training and learning resources.

This has opened up a whole new marketplace for mobile learning allowing users to tailor their use of e-learning to fit their lifestyle, whether they choose to work in the office, out in the field, at home or even when commuting on the train.

With nearly 200 million smartphones sold worldwide in 2009 alone, the range of portable device platforms is fast becoming more diverse than that of the desktop computer market. Because of the unique and disparate nature of mobile technology, developers often take a far more individual

approach to application design for these devices than with traditional PC software where many standardised practices and approaches to usability have been developed over the years. Standards for mobile devices certainly exist but due to the diversity of technologies, developers often have to take a 'per platform' approach to application design.

Combined with the shorter development cycles and time-to-market that mobile applications are generally given, this generalised approach to standards means that an individual attitude must also be applied when testing mobile applications. Software developers simply can't afford not to test their products and applications across the range of mobile devices in use today if they want to be sure of customer satisfaction and therefore maintain a reliable and reputable brand.

Before you can begin devising an effective test plan for your mobile application, you need to first establish whether your

product is aimed at a specific area of the mobile market or if it is for a more generic user base. If you know your target audience uses a select range of platforms and devices, you will likely want to test directly on these specific hardware units or by running software emulators to reproduce the operating environment as closely as possible.

However, in some cases the sheer range of hardware and software options combined with the shorter development cycles for mobile apps means it may be impractical to test on every possible combination that your audience could be using. In these instances, testing should at least cover the more popular platforms in mainstream handheld use. The most recent figures for 2009 show the following mobile operating systems to be the most common in order of smartphone market share:

- Symbian 50.3%
- RIM BlackBerry 20.9%
- Apple iPhone 13.7%
- Windows Mobile 9%
- Google Android 2.8%

Within these platforms, there tend to be 'families' of similar devices from the manufacturers which can help eliminate the need to test on specific individual devices if time or resources are limited. You could also conduct short user surveys to try and ascertain the more frequently used platforms of your target audience.

Once you've decided on the range of hardware and operating systems to test on, you can begin to consider the factors that commonly affect mobile application design. The process of testing applications on handheld devices is in many ways similar to standard software testing procedures, i.e. covering core functionality and compatibility of the application or e-learning. But you should also be thinking about what may need specific attention in terms of the technology that is unique to mobile devices. The following are some basic things to consider when testing applications for mobile and handheld platforms:

Porting

If your product is to be ported to many different mobile platforms, you need to be aware that these platforms

may not only differ in operating system and software architecture but in physical technology as well. Simply converting code or operational procedures may not be enough to make the application fully functional across a diverse range of platforms; it may also involve a complete redesign of the user interface and core design.

Screen layout and web page rendering

Although most portable devices include native web browsers which render HTML pages reasonably consistently, if your application is web based, you may see some slight differences between mobile platforms and their more common desktop counterparts. In less extreme instances this may only have a slight aesthetic effect on visual formatting but it could potentially render web pages hard to read or in extreme cases, completely inaccessible.

In general, try to keep to horizontal layouts and don't have too much side-by-side content. Many smartphones also include accelerometers - automatic orientation recognition which resizes the

display depending on whether the phone is held horizontally or vertically and this may also have an effect on layout and functionality.

Also, users may be browsing with images turned off to save on download time and connection charges so make sure to use relevant and descriptive alt tags for any graphics that convey important information.

Browser plug-ins

Third party plugins such as Java and Flash-based functionality can also be affected as mobiles may be running lightweight versions of these multimedia formats (e.g. Java ME and Flash Lite) so it shouldn't be assumed they will work correctly without extensive testing.

Input device and usability

One restriction common to the majority of portable devices is their lack of space for a fully functioning keyboard, usually requiring an alternative method of input. These can include numeric keypad, touch screen, stylus, dial-wheel, jog-wheel and other variations on miniature QWERTY keyboards.

Combined with the much reduced availability of screen real estate, special consideration must be given to simple, quick and functional navigation methods. With the increase of touch screen technology, consider the width of the human finger when designing buttons and be sure to leave space around links. Don't repeat navigation links on every page – breadcrumb trails can serve as a space-saving alternative for quick navigation.

Usability of portable devices shares much of the same design principles of designing for accessibility; you need to ensure that buttons, text entry fields, menus and navigation items are all functional and ergonomically usable across a wide range of devices. The use of ordered lists and designated access keys can help to facilitate effective navigation.

In general, people using mobile applications and websites are less interested in fancy design and slow loading graphics and more interested in being able to access information and resources quickly and easily so keep things simple wherever possible.

Bandwidth and connection speeds

Phones and portable devices connect to the internet and communication networks through a variety of standards, depending on location and network availability. The list of standards and protocols includes GSM, 3G, WiFi, GPRS, EDGE, WCDMA and HSDPA among others.

This means that some users may be connecting their mobile devices via high-speed WiFi connections at home or work, and others through slower and less stable cellular connections when out on the move. In some situations a combination of multiple network connections may be required for each individual user depending on their location.

You therefore need to make sure that your application works just as well regardless of network carrier, connection speed and reliability, especially if it involves interoperability and sending and storing user data. Nothing is more likely to cause frustration to a user than having entered large amounts of information only to find it lost or corrupted in transmission.

Other connectivity issues worth considering are whether your application depends on GPS positioning data or being able to automatically detect which network access points are available depending on a user's location.

Data and network security

If your application involves transmitting and receiving personal and private user data, it's of paramount importance that the handling of this information is treated securely and with confidentiality. This means using secure encryption methods to send and receive data as well as ensuring relevant certificates and server security procedures are in place for any personal information that will be stored on a long-term basis.

Memory footprint and leakage

Another way that mobile devices differ greatly from desktop computers is in the amount of available memory and drive space for storing and working with applications and data. Although memory cards, drive space and RAM for mobile devices has generally increased to capacities of gigabytes rather than megabytes in recent years, handheld

devices still aren't generally afforded the luxury of capacity as desktop PCs.

This lack of working RAM and free drive space can often cause unexpected behaviour, out-of-memory errors or even application crashes when programs cause memory leaks or leave behind unwanted files when they're finished, uninstalled or deleted.

Many end-users also adopt a multitasking approach with mobile devices such as leaving email clients or navigation applications open in the background, so you need to ensure your product doesn't cause conflicts or monopolise memory or system resources.

Battery usage

If your application includes any device-intensive processes (e.g. constant searching for Bluetooth, GPS or Wi-Fi connections, vibration alerts, sound or video streaming, camera functions, push mail etc.) it may cause reduced battery life which your users won't thank you for.

Make sure resource-hungry processes are kept to a minimum or give users

the option to customise preferences such as screen brightness, multimedia settings, connectivity options, alert frequency etc.

The evolution of mobile technology

Mobile learning is increasing at a substantial rate; the use of smartphones, PDAs and other handheld devices to access resources and shared learning is affordable, convenient and flexible, giving the user a greater choice of when and where to access information. This in turn increases the likelihood of users actively wanting to learn through this medium.

However, in relative terms, current users of mobile and handheld devices are much less experienced in using this form of technology to access information than traditional computer-based users. Therefore the likelihood of them being disappointed or frustrated with problematic or bug-ridden software is significantly higher than with standard learning applications and resources.

Mobile and handheld devices are complicated pieces of

consumer equipment that are tailored to fit the requirements of current communication technology. As this technology rapidly evolves, the market will become more shaped and standardised by considerations such as battery life, network connectivity, user interfaces, size and form factor, media formats etc.

As users become more familiar with mobile applications and technology, they will standardise their ways of working with them as they have with desktop technology. Therefore it's essential that your products are considered sufficiently robust and well implemented across the increasingly diverse range of mobile platforms in order for your product to thrive and enhance your brand's reputation with your customers.

Other epic e-learning white papers

E-learning benefits

E-learning: Return on investment
Organisational benefits

Subjects

Induction and e-learning
Compliance and e-learning
Softskills and e-learning
Healthcare and e-learning
E-learning for IT systems

Learning and design

Blended learning
Blended learning in practice
Use of media in e-learning
Learning design for e-learning
Usability in e-learning
Localisation and e-learning
Build, Buy or Both?
Learner Centred Design

Learning

Assessment and e-learning
The psychology of e-learning
Motivation in e-learning
Pedagogy and e-learning
Informal learning
Personalisation and e-learning

Innovation

Simulations and e-learning
Blogs
Web 2.0

Delivery

Change management & e-learning
E-tutoring

Technology

Open Source and e-learning
Reusable learning objects
Effective software testing

Standards

Standards in e-learning
Accessibility and e-learning

To order white papers, please email:

contactenquiries@epic.co.uk or
telephone +44 (0) 1273 728686.

To be notified of new white papers, and get the freshest thinking in e-learning, sign up for the regular Epic e-newsletter at www.epic.co.uk.



This document is the property of
Epic Performance Improvement Ltd. and
must not be copied in whole or part,
without the consent of the Company.
© 2010