Software QA/Test Resource Center

Software QA and Testing Frequently-Asked-Questions (Parts 1 and 2)

Software QA and Testing Resources -- Software QA and Testing Tools -- Web Site Test Tools and Site Management Tools

Software QA and Testing Bookstore

|TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Software QA and Testing Resource Center Table of Contents

- What makes a good test engineer?
- What makes a good Software QA engineer?
- What makes a good QA or Test manager?
- What's the role of documentation in QA?
- What's the big deal about 'requirements'?
- What steps are needed to develop and run software tests?
- What's a 'test plan'?
- What's a 'test case'?
- What should be done after a bug is found?
- What is 'configuration management'?
- What if the software is so buggy it can't really be tested at all?
- How can it be known when to stop testing?
- What if there isn't enough time for thorough testing?
- What if the project isn't big enough to justify extensive testing?
- What can be done if requirements are changing continuously?
- What if the application has functionality that wasn't in the requirements?
- How can QA processes be implemented without stifling productivity?
- What if an organization is growing so fast that fixed QA processes are impossible?
- How does a client/server environment affect testing?
- How can World Wide Web sites be tested?
- How is testing affected by object-oriented designs?

Other Software QA and Testing Resources

- Top 5 List
- Software QA and Testing-related organizations
- Links to QA and Testing-related Magazines/Publications
- General Software QA and Testing Resources
- Web QA and Testing Resources
- Web Security Testing Resources
- Web Usability Resources

Software QA and Test Tools

- Test tools
- CM tools and PM tools
- Web site test and management tools

Web Site Test Tools and Site Management Tools

- Load and performance test tools

- Java test tools
- HTML Validators
- Link Checkers
- Free On-the-Web HTML Validators and Link Checkers
- PERL and C Programs for Validating and Checking
- Web Functional/Regression Test Tools
- Web Site Security Test Tools
- External Site Monitoring Services
- Web Site Management Tools
- Log Analysis Tools
- Other Web Test Tools

Software QA and Testing Bookstore

- Software Testing Books
- Software Quality Assurance Books
- Software Requirements Engineering Books
- Software Metrics Books
- Configuration Management Books
- Software Risk Management Books
- Software Engineering Books
- Software Project Management Books
- Technical Background Basics Books
- Other Books

About the Software QA and Testing Resource Center and its author

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

Software QA and Testing Frequently-Asked-Questions, Part 1

What is 'Software Quality Assurance'?
Software QA involves the entire software development PROCESS - monitoring and improving the process, making sure that any agreed-upon standards and procedures are followed, and ensuring that problems are found and dealt with. It is oriented to 'prevention'. (See the Bookstore section's 'Software QA' category for a list of useful books on Software Quality Assurance.)

Return to top of this page's FAQ list

What is 'Software Testing'?
Testing involves operation of a system or application under controlled conditions and evaluating the results (eg, 'if the user is in interface A of the application while using hardware B, and does C, then D should happen'). The controlled conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things

don't happen when they should. It is oriented to 'detection'. (See the Bookstore section's 'Software Testing' category for a list of useful books on Software Testing.)

- Organizations vary considerably in how they assign responsibility for QA and testing. Sometimes they're the combined responsibility of one group or individual. Also common are project teams that include a mix of testers and developers who work closely together, with overall QA processes monitored by project managers. It will depend on what best fits an organization's size and business structure.

Return to top of this page's FAQ list

What are some recent major computer system failures caused by software bugs?

- In early 2000, major problems were reported with a new computer system in a large suburban U.S. public school district with 100,000+ students; problems included 10,000 erroneous report cards and students left stranded by failed class registration systems; the district's CIO was fired. The school district decided to reinstate it's original 25-year old system for at least a year until the bugs were worked out of the new system by the software contractors.
- In October of 1999 the $125 million NASA Mars Climate Orbiter spacecraft was believed to be lost in space due to a simple data conversion error. It was determined that spacecraft software used certain data in English units that should have been in metric units. Among other tasks, the orbiter was to serve as a communications relay for the Mars Polar Lander mission, which failed for unknown reasons in December 1999. Several investigating panels were convened to determine the process failures that allowed the error to go undetected.
- Bugs in software supporting a large commercial high-speed data network affected 70,000 business customers over a period of 8 days in August of 1999. Among those affected was the electronic trading system of the largest U.S. futures exchange, which was shut down for most of a week as a result of the outages.
- In April of 1999 a software bug caused the failure of a $1.2 billion military satellite launch, the costliest unmanned accident in the history of Cape Canaveral launches. The failure was the latest in a string of launch failures, triggering a complete military and industry review of U.S. space launch programs, including software integration and testing processes. Congressional oversight hearings were requested.
- A small town in Illinois received an unusually large monthly electric bill of $7 million in March of 1999. This was about 700 times larger than it's normal bill. It turned out to be due to bugs in new software that had been purchased by the local power company to deal with Y2K software issues.
- In early 1999 a major computer game company recalled all copies of a popular new product due to software problems. The company made a public apology for releasing a product before it was ready.
- The computer system of a major online U.S. stock trading service failed during trading hours several times over a period of days in February of 1999 according to nationwide news reports. The problem was reportedly due to bugs in a software upgrade intended to speed online trade confirmations.
- In April of 1998 a major U.S. data communications network failed for 24 hours, crippling a large part of some U.S. credit card transaction authorization systems as well as other large U.S. bank, retail, and government data systems. The cause was eventually traced to a software bug.
- January 1998 news reports told of software problems at a major U.S. telecommunications company that resulted in no charges for long distance calls for a month for 400,000 customers. The problem

went undetected until customers called up with questions about their bills.

- In November of 1997 the stock of a major health industry company dropped 60% due to reports of failures in computer billing systems, problems with a large database conversion, and inadequate software testing. It was reported that more than $100,000,000 in receivables had to be written off and that multi-million dollar fines were levied on the company by government agencies.
- A retail store chain filed suit in August of 1997 against a transaction processing system vendor (not a credit card company) due to the software's inability to handle credit cards with year 2000 expiration dates.
- In August of 1997 one of the leading consumer credit reporting companies reportedly shut down their new public web site after less than two days of operation due to software problems. The new site allowed web site visitors instant access, for a small fee, to their personal credit reports. However, a number of initial users ended up viewing each others' reports instead of their own, resulting in irate customers and nationwide publicity. The problem was attributed to "...unexpectedly high demand from consumers and faulty software that routed the files to the wrong computers."
- In November of 1996, newspapers reported that software bugs caused the 411 telephone information system of one of the U.S. RBOC's to fail for most of a day. Most of the 2000 operators had to search through phone books instead of using their 13,000,000-listing database. The bugs were introduced by new software modifications and the problem software had been installed on both the production and backup systems. A spokesman for the software vendor reportedly stated that 'It had nothing to do with the integrity of the software. It was human error.'
- Software bugs caused the bank accounts of 823 customers of a major U.S. bank to be credited with $924,844,208.32 each in May of 1996, according to newspaper reports. The American Bankers Association claimed it was the largest such error in banking history. A bank spokesman said the programming errors were corrected and all funds were recovered.
- Software bugs in a Soviet early-warning monitoring system nearly brought on nuclear war in 1983, according to news reports in early 1999. The software was supposed to filter out false missile detections caused by Soviet satellites picking up sunlight reflections off cloud-tops, but failed to do so. Disaster was averted when a Soviet commander, based on a what he said was a '...funny feeling in my gut', decided the apparent missile attack was a false alarm. The filtering software code was rewritten.

Return to top of this page's FAQ list

Why is it often hard for management to get serious about quality assurance?

Solving problems is a high-visibility process; preventing problems is low-visibility. This is illustrated by an old parable:

In ancient China there was a family of healers, one of whom was known throughout the land and employed as a physician to a great lord. The physician was asked which of his family was the most skillful healer. He replied,

"I tend to the sick and dying with drastic and dramatic treatments, and on occasion someone is cured and my name gets out among the lords."

"My elder brother cures sickness when it just begins to take root, and his skills are known among the local peasants and neighbors."

"My eldest brother is able to sense the spirit of sickness and eradicate it before it takes form. His name is unknown outside our home."

Return to top of this page's FAQ list

Why does software have bugs?

- miscommunication or no communication - as to specifics of what an application should or shouldn't do (the application's requirements).
- software complexity - the complexity of current software applications can be difficult to comprehend for anyone without experience in modern-day software development. Windows-type interfaces, client-server and distributed applications, data communications, enormous relational databases, and sheer size of applications have all contributed to the exponential growth in software/system complexity. And the use of object-oriented techniques can complicate instead of simplify a project unless it is well-engineered.
- programming errors - programmers, like anyone else, can make mistakes.
- changing requirements - the customer may not understand the effects of changes, or may understand and request them anyway - redesign, rescheduling of engineers, effects on other projects, work already completed that may have to be redone or thrown out, hardware requirements that may be affected, etc. If there are many minor changes or any major changes, known and unknown dependencies among parts of the project are likely to interact and cause problems, and the complexity of keeping track of changes may result in errors. Enthusiasm of engineering staff may be affected. In some fast-changing business environments, continuously modified requirements may be a fact of life. In this case, management must understand the resulting risks, and QA and test engineers must adapt and plan for continuous extensive testing to keep the inevitable bugs from running out of control - see 'What can be done if requirements are changing continuously?' in Part 2 of the FAQ.
- time pressures - scheduling of software projects is difficult at best, often requiring a lot of guesswork. When deadlines loom and the crunch comes, mistakes will be made.
- egos - people prefer to say things like:

```
  'no problem'
  'piece of cake'
  'I can whip that out in a few hours'
  'it should be easy to update that old code'

 instead of:
  'that adds a lot of complexity and we could end up
     making a lot of mistakes'
  'we have no idea if we can do that; we'll wing it'
  'I can't estimate how long it will take, until I
     take a close look at it'
  'we can't figure out what that old spaghetti code
     did in the first place'

 If there are too many unrealistic 'no problem's', the
 result is bugs.
```

- poorly documented code - it's tough to maintain and modify code that is badly written or poorly documented; the result is bugs. In many organizations management provides no incentive for programmers to document their code or write clear, understandable code. In fact, it's usually the opposite: they get points mostly for quickly turning out code, and there's job security if nobody else can understand it ('if it was hard to write, it should be hard to read').

- software development tools - visual tools, class libraries, compilers, scripting tools, etc. often introduce their own bugs or are poorly documented, resulting in added bugs.

Return to top of this page's FAQ list

How can new Software QA processes be introduced in an existing organization?

- A lot depends on the size of the organization and the risks involved. For large organizations with high-risk (in terms of lives or money) projects, serious management buy-in is required and a formalized QA process is necessary.
- Where the risk is lower, management and organizational buy-in and QA implementation may be a slower, step-at-a-time process. QA processes should be balanced with productivity so as to keep bureaucracy from getting out of hand.
- For small groups or projects, a more ad-hoc process may be appropriate, depending on the type of customers and projects. A lot will depend on team leads or managers, feedback to developers, and ensuring adequate communications among customers, managers, developers, and testers.
- In all cases the most value for effort will be in requirements management processes, with a goal of clear, complete, testable requirement specifications.

(See the Bookstore section's 'Software QA', 'Software Engineering', and 'Project Management' categories for useful books with more information.)

Return to top of this page's FAQ list

What is verification? validation?
Verification typically involves reviews and meetings to evaluate documents, plans, code, requirements, and specifications. This can be done with checklists, issues lists, walkthroughs, and inspection meetings. Validation typically involves actual testing and takes place after verifications are completed. The term 'IV & V' refers to Independent Verification and Validation.

Return to top of this page's FAQ list

What is a 'walkthrough'?
A 'walkthrough' is an informal meeting for evaluation or informational purposes. Little or no preparation is usually required.

Return to top of this page's FAQ list

What's an 'inspection'?
An inspection is more formalized than a 'walkthrough', typically with 3-8 people including a moderator, reader (the author of whatever is being reviewed), and a recorder to take notes. The subject of the inspection is typically a document such as a requirements spec or a test plan, and the purpose is to find problems and see what's missing, not to fix anything. Attendees should prepare for this type of meeting by reading thru the document; most problems will be found during this preparation. The result of the inspection meeting should be a written report. Thorough preparation for inspections is difficult, painstaking

work, but is one of the most cost effective methods of ensuring quality. Employees who are most skilled at inspections are like the 'eldest brother' in the parable in 'Why is it often hard for management to get serious about quality assurance?'. Their skill may have low visibility but they are extremely valuable to any software development organization, since bug prevention is far more cost effective than bug detection.

Return to top of this page's FAQ list

What kinds of testing should be considered?

- Black box testing - not based on any knowledge of internal design or code. Tests are based on requirements and functionality.
- White box testing - based on knowledge of the internal logic of an application's code. Tests are based on coverage of code statements, branches, paths, conditions.
- unit testing - the most 'micro' scale of testing; to test particular functions or code modules. Typically done by the programmer and not by testers, as it requires detailed knowledge of the internal program design and code. Not always easily done unless the application has a well-designed architecture with tight code; may require developing test driver modules or test harnesses.
- incremental integration testing - continuous testing of an application as new functionality is added; requires that various aspects of an application's functionality be independent enough to work separately before all parts of the program are completed, or that test drivers be developed as needed; done by programmers or by testers.
- integration testing - testing of combined parts of an application to determine if they function together correctly. The 'parts' can be code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.
- functional testing - black-box type testing geared to functional requirements of an application; this type of testing should be done by testers. This doesn't mean that the programmers shouldn't check that their code works before releasing it (which of course applies to any stage of testing.)
- system testing - black-box type testing that is based on overall requirements specifications; covers all combined parts of a system.
- end-to-end testing - similar to system testing; the 'macro' end of the test scale; involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.
- sanity testing - typically an initial testing effort to determine if a new software version is performing well enough to accept it for a major testing effort. For example, if the new software is crashing systems every 5 minutes, bogging down systems to a crawl, or destroying databases, the software may not be in a 'sane' enough condition to warrant further testing in its current state.
- regression testing - re-testing after fixes or modifications of the software or its environment. It can be difficult to determine how much re-testing is needed, especially near the end of the development cycle. Automated testing tools can be especially useful for this type of testing.
- acceptance testing - final testing based on specifications of the end-user or customer, or based on use by end-users/customers over some limited period of time.
- load testing - testing an application under heavy loads, such as testing of a web site under a range of loads to determine at what point the system's response time degrades or fails.
- stress testing - term often used interchangeably with 'load' and 'performace' testing. Also used to

describe such tests as system functional testing while under unusually heavy loads, heavy repetition of certain actions or inputs, input of large numerical values, large complex queries to a database system, etc.

- performance testing - term often used interchangeably with 'stress' and 'load' testing. Ideally 'performance' testing (and any other 'type' of testing) is defined in requirements documentation or QA or Test Plans.
- usability testing - testing for 'user-friendliness'. Clearly this is subjective, and will depend on the targeted end-user or customer. User interviews, surveys, video recording of user sessions, and other techniques can be used. Programmers and testers are usually not appropriate as usability testers.
- install/uninstall testing - testing of full, partial, or upgrade install/uninstall processes.
- recovery testing - testing how well a system recovers from crashes, hardware failures, or other catastrophic problems.
- security testing - testing how well the system protects against unauthorized internal or external access, willful damage, etc; may require sophisticated testing techniques.
- compatability testing - testing how well software performs in a particular hardware/software/operating system/network/etc. environment.
- acceptance testing - determining if software is stisfactory to a customer.
- comparison testing - comparing software weaknesses and strengths to competing products.
- alpha testing - testing of an application when development is nearing completion; minor design changes may still be made as a result of such testing. Typically done by end-users or others, not by programmers or testers.
- beta testing - testing when development and testing are essentially completed and final bugs and problems need to be found before final release. Typically done by end-users or others, not by programmers or testers.

(See the Bookstore section's 'Software Testing' category for useful books on Software Testing.)

Return to top of this page's FAQ list

What are 5 common problems in the software development process?

- poor requirements - if requirements are unclear, incomplete, too general, or not testable, there will be problems.
- unrealistic schedule - if too much work is crammed in too little time, problems are inevitable.
- inadequate testing - no one will know whether or not the program is any good until the customer complains or systems crash.
- featuritis - requests to pile on new features after development is underway; extremely common.
- miscommunication - if developers don't know what's needed or customer's have erroneous expectations, problems are guaranteed.

(See the Bookstore section's 'Software QA', 'Software Engineering', and 'Project Management' categories for useful books with more information.)

Return to top of this page's FAQ list

What are 5 common solutions to software development problems?

- solid requirements - clear, complete, detailed, cohesive, attainable, testable requirements that are agreed to by all players. Use prototypes to help nail down requirements.
- realistic schedules - allow adequate time for planning, design, testing, bug fixing, re-testing, changes, and documentation; personnel should be able to complete the project without burning out.
- adequate testing - start testing early on, re-test after fixes or changes, plan for adequate time for testing and bug-fixing.
- stick to initial requirements as much as possible - be prepared to defend against changes and additions once development has begun, and be prepared to explain consequences. If changes are necessary, they should be adequately reflected in related schedule changes. If possible, use rapid prototyping during the design phase so that customers can see what to expect. This will provide them a higher comfort level with their requirements decisions and minimize changes later on.
- communication - require walkthroughs and inspections when appropriate; make extensive use of group communication tools - e-mail, groupware, networked bug-tracking tools and change management tools, intranet capabilities, etc.; insure that documentation is available and up-to-date - preferably electronic, not paper; promote teamwork and cooperation; use protoypes early on so that customers' expectations are clarified.

(See the Bookstore section's 'Software QA', 'Software Engineering', and 'Project Management' categories for useful books with more information.)

Return to top of this page's FAQ list

What is software 'quality'?
Quality software is reasonably bug-free, delivered on time and within budget, meets requirements and/or expectations, and is maintainable. However, quality is obviously a subjective term. It will depend on who the 'customer' is and their overall influence in the scheme of things. A wide-angle view of the 'customers' of a software development project might include end-users, customer acceptance testers, customer contract officers, customer management, the development organization's management/accountants/testers/salespeople, future software maintenance engineers, stockholders, magazine columnists, etc. Each type of 'customer' will have their own slant on 'quality' - the accounting department might define quality in terms of profits while an end-user might define quality as user-friendly and bug-free. (See the Bookstore section's 'Software QA' category for useful books with more information.)

Return to top of this page's FAQ list

What is 'good code'?
'Good code' is code that works, is bug free, and is readable and maintainable. Some organizations have coding 'standards' that all developers are supposed to adhere to, but everyone has different ideas about what's best, or what is too many or too few rules. There are also various theories and metrics, such as McCabe Complexity metrics. It should be kept in mind that excessive use of standards and rules can stifle productivity and creativity. 'Peer reviews', 'buddy checks' code analysis tools, etc. can be used to check for problems and enforce standards.
For C and C++ coding, here are some typical ideas to consider in setting rules/standards; these may or may not apply to a particular situation:

- minimize or eliminate use of global variables.
- use descriptive function and method names - use both upper and lower case, avoid abbreviations, use as many characters as necessary to be adequately descriptive (use of more than 20 characters is not

out of line); be consistent in naming conventions.

- use descriptive variable names - use both upper and lower case, avoid abbreviations, use as many characters as necessary to be adequately descriptive (use of more than 20 characters is not out of line); be consistent in naming conventions.
- function and method sizes should be minimized; less than 100 lines of code is good, less than 50 lines is preferable.
- function descriptions should be clearly spelled out in comments preceding a function's code.
- organize code for readability.
- use whitespace generously - vertically and horizintally
- each line of code should contain 70 characters max.
- one code statement per line.
- coding style should be consistent throught a program (eg, use of brackets, indentations, naming conventions, etc.)
- in adding comments, err on the side of too many rather than too few comments; a common rule of thumb is that there should be at least as many lines of comments (including header blocks) as lines of code.
- no matter how small, an application should include documentaion of the overall program function and flow (even a few paragraphs is better than nothing); or if possible a separate flow chart and detailed program documentation.
- make extensive use of error handling procedures and status and error logging.
- for C++, to minimize complexity and increase maintainability, avoid too many levels of inheritance in class heirarchies (relative to the size and complexity of the application). Minimize use of multiple inheritance, and minimize use of operator overloading (note that the Java programming language eliminates multiple inheritance and operator overloading.)
- for C++, keep class methods small, less than 50 lines of code per method is preferable.
- for C++, make liberal use of exception handlers

Return to top of this page's FAQ list

What is 'good design'?
'Design' could refer to many things, but often refers to 'functional design' or 'internal design'. Good internal design is indicated by software code whose overall structure is clear, understandable, easily modifiable, and maintainable; is robust with sufficient error-handling and status logging capability; and works correctly when implemented. Good functional design is indicated by an application whose functionality can be traced back to customer and end-user requirements. (See further discussion of functional and internal design in 'What's the big deal about requirements?' in FAQ #2.) For programs that have a user interface, it's often a good idea to assume that the end user will have little computer knowledge and may not read a user manual or even the on-line help; some common rules-of-thumb include:

- the program should act in a way that least surprises the user
- it should always be evident to the user what can be done next and how to exit
- the program shouldn't let the users do something stupid without warning them.

Return to top of this page's FAQ list

What is SEI? CMM? ISO? IEEE? ANSI? Will it help?

- SEI = 'Software Engineering Institute' at Carnegie-Mellon University; initiated by the U.S. Defense Department to help improve software development processes.
- CMM = 'Capability Maturity Model', developed by the SEI. It's a model of 5 levels of organizational 'maturity' that determine effectiveness in delivering quality software. It is geared to large organizations such as large U.S. Defense Department contractors. However, many of the QA processes involved are appropriate to any organization, and if reasonably applied can be helpful. Organizations can receive CMM ratings by undergoing assessments by qualified auditors.

```
   Level 1 - characterized by chaos, periodic panics, and heroic
             efforts required by individuals to successfully
             complete projects.  Few if any processes in place;
             successes may not be repeatable.

   Level 2 - software project tracking, requirements management,
             realistic planning, and configuration management
             processes are in place; successful practices can
             be repeated.

   Level 3 - standard software development and maintenance processes
             are integrated throughout an organization; a Software
             Engineering Process Group is is in place to oversee
             software processes, and training programs are used to
             ensure understanding and compliance.

   Level 4 - metrics are used to track productivity, processes,
             and products.  Project performance is predictable,
             and quality is consistently high.

   Level 5 - the focus is on continouous process improvement. The
             impact of new processes and technologies can be
             predicted and effectively implemented when required.


  (Perspective on CMM ratings:  During 1992-1996 533 organizations
  were assessed.  Of those, 62% were rated at Level 1, 23% at 2,
  13% at 3, 2% at 4, and  0.4% at 5.  The median size of
  organizations was 100 software engineering/maintenance personnel;
  31% of organizations were U.S. federal contractors.  For those
  rated at Level 1, the most problematical key process area was
  in Software Quality Assurance.)
```

- ISO = 'International Organisation for Standards' - The ISO 9001, 9002, and 9003 standards concern quality systems that are assessed by outside auditors, and they apply to many kinds of production and manufacturing organizations, not just software. The most comprehensive is 9001, and this is the one most often used by software development organizations. It covers documentation, design, development, production, testing, installation, servicing, and other processes. ISO 9000-3 (not the same as 9003) is a guideline for applying ISO 9001 to software development organizations. The U.S. version of the ISO 9000 series standards is exactly the same as the international version, and is called the ANSI/ASQ Q9000 series. The U.S. version can be purchased directly from the ASQ (American Society for Quality) or the ANSI organizations. To be ISO 9001 certified, a third-party auditor assesses an organization, and certification is typically good for about 3 years, after which a complete reassessment is required. Note that ISO 9000 certification does not necessarily indicate quality products - it indicates only that documented processes are followed.

- IEEE = 'Institute of Electrical and Electronics Engineers' - among other things, creates standards such as 'IEEE Standard for Software Test Documentation' (IEEE/ANSI Standard 829), 'IEEE Standard of Software Unit Testing (IEEE/ANSI Standard 1008), 'IEEE Standard for Software Quality Assurance Plans' (IEEE/ANSI Standard 730), and others.
- ANSI = 'American National Standards Institute', the primary industrial standards body in the U.S.; publishes some software-related standards in conjunction with the IEEE and ASQ (American Society for Quality).
- Other software development process assessment methods besides CMM and ISO 9000 include SPICE, Trillium, TickIT. and Bootstrap.
- See the 'Other Resources' section for further information available on the web.

Return to top of this page's FAQ list

What is the 'software life cycle'?

The life cycle begins when an application is first conceived and ends when it is no longer in use. It includes aspects such as initial concept, requirements analysis, functional design, internal design, documentation planning, test planning, coding, document preparation, integration, testing, maintenance, updates, retesting, phase-out, and other aspects. (See the Bookstore section's 'Software QA', 'Software Engineering', and 'Project Management' categories for useful books with more information.)

Return to top of this page's FAQ list

Will automated testing tools make testing easier?

- Possibly. For small projects, the time needed to learn and implement them may not be worth it. For larger projects, or on-going long-term projects they can be valuable.
- A common type of automated tool is the 'record/playback' type. For example, a tester could click through all combinations of menu choices, dialog box choices, buttons, etc. in an application GUI and have them 'recorded' and the results logged by a tool. The 'recording' is typically in the form of text based on a scripting language that is interpretable by the testing tool. If new buttons are added, or some underlying code in the application is changed, etc. the application can then be retested by just 'playing back' the 'recorded' actions, and comparing the logging results to check effects of the changes. The problem with such tools is that if there are continual changes to the system being tested, the 'recordings' may have to be changed so much that it becomes very time-consuming to continuously update the scripts. Additionally, interpretation of results (screens, data, logs, etc.) can be a difficult task. Note that there are record/playback tools for text-based interfaces also, and for all types of platforms.
- Other automated tools can include:

```
code analyzers - monitor code complexity, adherence to
                 standards, etc.

coverage analyzers - these tools check which parts of the
                 code have been exercised by a test, and may
                 be oriented to code statement coverage,
                 condition coverage, path coverage, etc.

memory analyzers - such as bounds-checkers and leak detectors.

load/performance test tools - for testing client/server
                 and web applications under various load
                 levels.

web test tools - to check that links are valid, HTML code
                 usage is correct, client-side and
                 server-side programs work, a web site's
                 interactions are secure.

other tools - for test case management, documentation
                 management, bug reporting, and configuration
                 management.
```

See the 'Tools' section for test tool listings and the 'Web Tools' section for web site testing tools.

Return to top of this page's FAQ list

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

About the Software QA and Testing Resource Center and its author

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Software QA and Testing Frequently-Asked-Questions Part 2

What makes a good test engineer?

A good test engineer has a 'test to break' attitude, an ability to take the point of view of the customer, a strong desire for quality, and an attention to detail. Tact and diplomacy are useful in maintaining a cooperative relationship with developers, and an ability to communicate with both technical (developers) and non-technical (customers, management) people is useful. Previous software development experience can be helpful as it provides a deeper understanding of the software development process, gives the tester an appreciation for the developers' point of view, and reduce the learning curve in automated test tool programming. Judgement skills are needed to assess high-risk areas of an application on which to focus testing efforts when time is limited.

Return to top of this page's FAQ list

What makes a good Software QA engineer?
The same qualities a good tester has are useful for a QA engineer. Additionally, they must be able to understand the entire software development process and how it can fit into the business approach and goals of the organization. Communication skills and the ability to understand various sides of issues are important. In organizations in the early stages of implementing QA processes, patience and diplomacy are especially needed. An ability to find problems as well as to see 'what's missing' is important for inspections and reviews.

Return to top of this page's FAQ list

What makes a good QA or Test manager?
A good QA, test, or QA/Test(combined) manager should:

- be familiar with the software development process
- be able to maintain enthusiasm of their team and promote a positive atmosphere, despite what is a somewhat 'negative' process (e.g., looking for or preventing problems)
- be able to promote teamwork to increase productivity
- be able to promote cooperation between software, test, and QA engineers
- have the diplomatic skills needed to promote improvements in QA processes
- have the ability to withstand pressures and say 'no' to other managers when quality is insufficient or QA processes are not being adhered to
- have people judgement skills for hiring and keeping skilled personnel
- be able to communicate with technical and non-technical people, engineers, managers, and customers.
- be able to run meetings and keep them focused

Return to top of this page's FAQ list

What's the role of documentation in QA?
Critical. (Note that documentation can be electronic, not necessarily paper.) QA practices should be documented such that they are repeatable. Specifications, designs, business rules, inspection reports, configurations, code changes, test plans, test cases, bug reports, user manuals, etc. should all be documented. There should ideally be a system for easily finding and obtaining documents and determining what documentation will have a particular piece of information. Change management for documentation should be used if possible.

Return to top of this page's FAQ list

What's the big deal about 'requirements'?
One of the most reliable methods of insuring problems, or failure, in a complex software project is to have poorly documented requirements specifications. Requirements are the details describing an application's externally-perceived functionality and properties. Requirements should be clear, complete, reasonably detailed, cohesive, attainable, and testable. A non-testable requirement would be, for example, 'user-friendly' (too subjective). A testable requirement would be something like 'the user must

enter their previously-assigned password to access the application'. Determining and organizing requirements details in a useful and efficient way can be a difficult effort; different methods are available depending on the particular project. Many books are available that describe various approaches to this task. (See the Bookstore section's 'Software Requirements Engineering' category for books on Software Requirements.)

Care should be taken to involve ALL of a project's significant 'customers' in the requirements process. 'Customers' could be in-house personnel or out, and could include end-users, customer acceptance testers, customer contract officers, customer management, future software maintenance engineers, salespeople, etc. Anyone who could later derail the project if their expectations aren't met should be included if possible.

Organizations vary considerably in their handling of requirements specifications. Ideally, the requirements are spelled out in a document with statements such as 'The product shall.....'. 'Design' specifications should not be confused with 'requirements'; design specifications should be traceable back to the requirements.

In some organizations requirements may end up in high level project plans, functional specification documents, in design documents, or in other documents at various levels of detail. No matter what they are called, some type of documentation with detailed requirements will be needed by testers in order to properly plan and execute tests. Without such documentation, there will be no clear-cut way to determine if a software application is performing correctly.

Return to top of this page's FAQ list

What steps are needed to develop and run software tests?
The following are some of the steps to consider:

- Obtain requirements, functional design, and internal design specifications and other necessary documents
- Obtain budget and schedule requirements
- Determine project-related personnel and their responsibilities, reporting requirements, required standards and processes (such as release processes, change processes, etc.)
- Identify application's higher-risk aspects, set priorities, and determine scope and limitations of tests
- Determine test approaches and methods - unit, integration, functional, system, load, usability tests, etc.
- Determine test environment requirements (hardware, software, communications, etc.)
- Determine testware requirements (record/playback tools, coverage analyzers, test tracking, problem/bug tracking, etc.)
- Determine test input data requirements
- Identify tasks, those responsible for tasks, and labor requirements
- Set schedule estimates, timelines, milestones
- Determine input equivalence classes, boundary value analyses, error classes
- Prepare test plan document and have needed reviews/approvals

- Write test cases
- Have needed reviews/inspections/approvals of test cases
- Prepare test environment and testware, obtain needed user manuals/reference documents/configuration guides/installation guides, set up test tracking processes, set up logging and archiving processes, set up or obtain test input data
- Obtain and install software releases
- Perform tests
- Evaluate and report results
- Track problems/bugs and fixes
- Retest as needed
- Maintain and update test plans, test cases, test environment, and testware through life cycle

Return to top of this page's FAQ list

What's a 'test plan'?

A software project test plan is a document that describes the objectives, scope, approach, and focus of a software testing effort. The process of preparing a test plan is a useful way to think through the efforts needed to validate the acceptability of a software product. The completed document will help people outside the test group understand the 'why' and 'how' of product validation. It should be thorough enough to be useful but not so thorough that no one outside the test group will read it. The following are some of the items that might be included in a test plan, depending on the particular project:

- Title
- Identification of software including version/release numbers
- Revision history of document including authors, dates, approvals
- Table of Contents
- Purpose of document, intended audience
- Objective of testing effort
- Software product overview
- Relevant related document list, such as requirements, design documents, other test plans, etc.
- Relevant standards or legal requirements
- Traceability requirements
- Relevant naming conventions and identifier conventions
- Overall software project organization and personnel/contact-info/responsibilties
- Test organization and personnel/contact-info/responsibilities
- Assumptions and dependencies
- Project risk analysis
- Testing priorities and focus
- Scope and limitations of testing
- Test outline - a decomposition of the test approach by test type, feature, functionality, process, system, module, etc. as applicable

- Outline of data input equivalence classes, boundary value analysis, error classes
- Test environment - hardware, operating systems, other required software, data configurations, interfaces to other systems
- Test environment setup and configuration issues
- Test data setup requirements
- Database setup requirements
- Outline of system-logging/error-logging/other capabilities, and tools such as screen capture software, that will be used to help describe and report bugs
- Discussion of any specialized software or hardware tools that will be used by testers to help track the cause or source of bugs
- Test automation - justification and overview
- Test tools to be used, including versions, patches, etc.
- Test script/test code maintenance processes and version control
- Problem tracking and resolution - tools and processes
- Project test metrics to be used
- Reporting requirements and testing deliverables
- Software entrance and exit criteria
- Initial sanity testing period and criteria
- Test suspension and restart criteria
- Personnel allocation
- Personnel pre-training needs
- Test site/location
- Outside test organizations to be utilized and their purpose, responsibilties, deliverables, contact persons, and coordination issues
- Relevant proprietary, classified, security, and licensing issues.
- Open issues
- Appendix - glossary, acronyms, etc.

(See the Bookstore section's 'Software Testing' and 'Software QA' categories for useful books with more information.)

Return to top of this page's FAQ list

What's a 'test case'?

- A test case is a document that describes an input, action, or event and an expected response, to determine if a feature of an application is working correctly. A test case should contain particulars such as test case identifier, test case name, objective, test conditions/setup, input data requirements, steps, and expected results.
- Note that the process of developing test cases can help find problems in the requirements or design of an application, since it requires completely thinking through the operation of the application. For this reason, it's useful to prepare test cases early in the development cycle if possible.

Return to top of this page's FAQ list

What should be done after a bug is found?
The bug needs to be communicated and assigned to developers that can fix it. After the problem is resolved, fixes should be re-tested, and determinations made regarding requirements for regression testing to check that fixes didn't create problems elsewhere. If a problem-tracking system is in place, it should encapsulate these processes. A variety of commercial problem-tracking/management software tools are available (see the 'Tools' section for web resources with listings of such tools). The following are items to consider in the tracking process:

- Complete information such that developers can understand the bug, get an idea of it's severity, and reproduce it if necessary.
- Bug identifier (number, ID, etc.)
- Current bug status (e.g., 'Released for Retest', 'New', etc.)
- The application name or identifier and version
- The function, module, feature, object, screen, etc. where the bug occurred
- Environment specifics, system, platform, relevant hardware specifics
- Test case name/number/identifier
- One-line bug description
- Full bug description
- Description of steps needed to reproduce the bug if not covered by a test case or if the developer doesn't have easy access to the test case/test script/test tool
- Names and/or descriptions of file/data/messages/etc. used in test
- File excerpts/error messages/log file excerpts/screen shots/test tool logs that would be helpful in finding the cause of the problem
- Severity estimate (a 5-level range such as 1-5 or 'critical'-to-'low' is common)
- Was the bug reproducible?
- Tester name
- Test date
- Bug reporting date
- Name of developer/group/organization the problem is assigned to
- Description of problem cause
- Description of fix
- Code section/file/module/class/method that was fixed
- Date of fix
- Application version that contains the fix
- Tester responsible for retest
- Retest date
- Retest results

- Regression testing requirements
- Tester responsible for regression tests
- Regression testing results

A reporting or tracking process should enable notification of appropriate personnel at various stages. For instance, testers need to know when retesting is needed, developers need to know when bugs are found and how to get the needed information, and reporting/summary capabilities are needed for managers.

Return to top of this page's FAQ list

What is 'configuration management'?
Configuration management covers the processes used to control, coordinate, and track: code, requirements, documentation, problems, change requests, designs, tools/compilers/libraries/patches, changes made to them, and who makes the changes. (See the 'Tools' section for web resources with listings of configuration management tools. Also see the Bookstore section's 'Configuration Management' category for useful books with more information.)

Return to top of this page's FAQ list

What if the software is so buggy it can't really be tested at all?
The best bet in this situation is for the testers to go through the process of reporting whatever bugs or blocking-type problems initially show up, with the focus being on critical bugs. Since this type of problem can severely affect schedules, and indicates deeper problems in the software development process (such as insufficient unit testing or insufficient integration testing, poor design, improper build or release procedures, etc.) managers should be notified, and provided with some documentation as evidence of the problem.

Return to top of this page's FAQ list

How can it be known when to stop testing?
This can be difficult to determine. Many modern software applications are so complex, and run in such an interdependent environment, that complete testing can never be done. Common factors in deciding when to stop are:

- Deadlines (release deadlines, testing deadlines, etc.)
- Test cases completed with certain percentage passed
- Test budget depleted
- Coverage of code/functionality/requirements reaches a specified point
- Bug rate falls below a certain level
- Beta or alpha testing period ends

Return to top of this page's FAQ list

What if there isn't enough time for thorough testing?
Use risk analysis to determine where testing should be focused.
Since it's rarely possible to test every possible aspect of an application, every possible combination of events, every dependency, or everything that could go wrong, risk analysis is appropriate to most software development projects. This requires judgement skills, common sense, and experience. (If

warranted, formal methods are also available.) Considerations can include:

- Which functionality is most important to the project's intended purpose?
- Which functionality is most visible to the user?
- Which functionality has the largest safety impact?
- Which functionality has the largest financial impact on users?
- Which aspects of the application are most important to the customer?
- Which aspects of the application can be tested early in the development cycle?
- Which parts of the code are most complex, and thus most subject to errors?
- Which parts of the application were developed in rush or panic mode?
- Which aspects of similar/related previous projects caused problems?
- Which aspects of similar/related previous projects had large maintenance expenses?
- Which parts of the requirements and design are unclear or poorly thought out?
- What do the developers think are the highest-risk aspects of the application?
- What kinds of problems would cause the worst publicity?
- What kinds of problems would cause the most customer service complaints?
- What kinds of tests could easily cover multiple functionalities?
- Which tests will have the best high-risk-coverage to time-required ratio?

Return to top of this page's FAQ list

What if the project isn't big enough to justify extensive testing?
Consider the impact of project errors, not the size of the project. However, if extensive testing is still not justified, risk analysis is again needed and the same considerations as described previously in 'What if there isn't enough time for thorough testing?' apply. The tester might then do ad hoc testing, or write up a limited test plan based on the risk analysis.

Return to top of this page's FAQ list

What can be done if requirements are changing continuously?
A common problem and a major headache.

- Work with the project's stakeholders early on to understand how requirements might change so that alternate test plans and strategies can be worked out in advance, if possible.
- It's helpful if the application's initial design allows for some adaptability so that later changes do not require redoing the application from scratch.
- If the code is well-commented and well-documented this makes changes easier for the developers.
- Use rapid prototyping whenever possible to help customers feel sure of their requirements and minimize changes.
- The project's initial schedule should allow for some extra time commensurate with the possibility of changes.
- Try to move new requirements to a 'Phase 2' version of an application, while using the original requirements for the 'Phase 1' version.

- Negotiate to allow only easily-implemented new requirements into the project, while moving more difficult new requirements into future versions of the application.
- Be sure that customers and management understand the scheduling impacts, inherent risks, and costs of significant requirements changes. Then let management or the customers (not the developers or testers) decide if the changes are warranted - after all, that's their job.
- Balance the effort put into setting up automated testing with the expected effort required to re-do them to deal with changes.
- Try to design some flexibility into automated test scripts.
- Focus initial automated testing on application aspects that are most likely to remain unchanged.
- Devote appropriate effort to risk analysis of changes to minimize regression testing needs.
- Design some flexibility into test cases (this is not easily done; the best bet might be to minimize the detail in the test cases, or set up only higher-level generic-type test plans)
- Focus less on detailed test plans and test cases and more on ad hoc testing (with an understanding of the added risk that this entails).

Return to top of this page's FAQ list

What if the application has functionality that wasn't in the requirements?
It may take serious effort to determine if an application has significant unexpected or hidden functionality, and it would indicate deeper problems in the software development process. If the functionality isn't necessary to the purpose of the application, it should be removed, as it may have unknown impacts or dependencies that were not taken into account by the designer or the customer. If not removed, design information will be needed to determine added testing needs or regression testing needs. Management should be made aware of any significant added risks as a result of the unexpected functionality. If the functionality only effects areas such as minor improvements in the user interface, for example, it may not be a significant risk.

Return to top of this page's FAQ list

How can Software QA processes be implemented without stifling productivity?
By implementing QA processes slowly over time, using consensus to reach agreement on processes, and adjusting and experimenting as an organization grows and matures, productivity will be improved instead of stifled. Problem prevention will lessen the need for problem detection, panics and burn-out will decrease, and there will be improved focus and less wasted effort. At the same time, attempts should be made to keep processes simple and efficient, minimize paperwork, promote computer-based processes and automated tracking and reporting, minimize time required in meetings, and promote training as part of the QA process. However, no one - especially talented technical types - likes rules or bureacracy, and in the short run things may slow down a bit. A typical scenario would be that more days of planning and development will be needed, but less time will be required for late-night bug-fixing and calming of irate customers.
(See the Bookstore section's 'Software QA', 'Software Engineering', and 'Project Management' categories for useful books with more information.)

Return to top of this page's FAQ list

What if an organization is growing so fast that fixed QA processes are impossible?

This is a common problem in the software industry, especially in new technology areas. There is no easy solution in this situation, other than:

- Hire good people
- Management should 'ruthlessly prioritize' quality issues and maintain focus on the customer
- Everyone in the organization should be clear on what 'quality' means to the customer

Return to top of this page's FAQ list

How does a client/server environment affect testing?
Client/server applications can be quite complex due to the multiple dependencies among clients, data communications, hardware, and servers. Thus testing requirements can be extensive. When time is limited (as it usually is) the focus should be on integration and system testing. Additionally, load/stress/performance testing may be useful in determining client/server application limitations and capabilities. There are commercial tools to assist with such testing. (See the 'Tools' section for web resources with listings that include these kinds of test tools.)

Return to top of this page's FAQ list

How can World Wide Web sites be tested?
Web sites are essentially client/server applications - with web servers and 'browser' clients. Consideration should be given to the interactions between html pages, TCP/IP communications, Internet connections, firewalls, applications that run in web pages (such as applets, javascript, plug-in applications), and applications that run on the server side (such as cgi scripts, database interfaces, logging applications, dynamic page generators, etc.). Additionally, there are a wide variety of servers and browsers, various versions of each, small but sometimes significant differences between them, variations in connection speeds, rapidly changing technologies, and multiple standards and protocols. The end result is that testing for web sites can become a major ongoing effort. Other considerations might include:

- What are the expected loads on the server (e.g., number of hits per unit time?), and what kind of performance is required under such loads (such as web server response time, database query response times). What kinds of tools will be needed for performance testing (such as web load testing tools, other tools already in house that can be adapted, web robot downloading tools, etc.)?
- Who is the target audience? What kind of browsers will they be using? What kind of connection speeds will they by using? Are they intra- organization (thus with likely high connection speeds and similar browsers) or Internet-wide (thus with a wide variety of connection speeds and browser types)?
- What kind of performance is expected on the client side (e.g., how fast should pages appear, how fast should animations, applets, etc. load and run)?
- Will down time for server and content maintenance/upgrades be allowed? how much?
- What kinds of security (firewalls, encryptions, passwords, etc.) will be required and what is it expected to do? How can it be tested?
- How reliable are the site's Internet connections required to be? And how does that affect backup system or redundant connection requirements and testing?
- What processes will be required to manage updates to the web site's content, and what are the requirements for maintaining, tracking, and controlling page content, graphics, links, etc.?

- Which HTML specification will be adhered to? How strictly? What variations will be allowed for targeted browsers?
- Will there be any standards or requirements for page appearance and/or graphics throughout a site or parts of a site??
- How will internal and external links be validated and updated? how often?
- Can testing be done on the production system, or will a separate test system be required? How are browser caching, variations in browser option settings, dial-up connection variabilities, and real-world internet 'traffic congestion' problems to be accounted for in testing?
- How extensive or customized are the server logging and reporting requirements; are they considered an integral part of the system and do they require testing?
- How are cgi programs, applets, javascripts, ActiveX components, etc. to be maintained, tracked, controlled, and tested?

Some sources of site security information include the Usenet newsgroup 'comp.security.announce' and links concerning web site security in the 'Other Resources' section.

Some usability guidelines to consider - these are subjective and may or may not apply to a given situation (Note: more information on usability testing issues can be found in articles about web site usability in the 'Other Resources' section):

- Pages should be 3-5 screens max unless content is tightly focused on a single topic. If larger, provide internal links within the page.
- The page layouts and design elements should be consistent throughout a site, so that it's clear to the user that they're still within a site.
- Pages should be as browser-independent as possible, or pages should be provided or generated based on the browser-type.
- All pages should have links external to the page; there should be no dead-end pages.
- The page owner, revision date, and a link to a contact person or organization should be included on each page.

Many new web site test tools are appearing and more than 170 of them are listed in the 'Web Test Tools' section.

Return to top of this page's FAQ list

How is testing affected by object-oriented designs?
Well-engineered object-oriented design can make it easier to trace from code to internal design to functional design to requirements. While there will be little affect on black box testing (where an understanding of the internal design of the application is unnecessary), white-box testing can be oriented to the application's objects. If the application was well-designed this can simplify test design.

Return to top of this page's FAQ list

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

About the Software QA and Testing Resource Center and its author

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Other Software QA and Testing Resources

Top 5 List

- Search comp.software.testing at Deja News - The Deja News site, which can be used to search through past postings to the comp.software.testing newsgroup by entering the newsgroup name in the search page. Postings go back to 1995. (The comp.software.testing searchable archive at jpl.nasa is no longer available.) Or access current discussions in comp.softwar.testing via the web .

- The comp.software.testing FAQ - The comp.software.testing FAQ, maintained by Danny Faught; excellent resource for testing-related conferences, mailing lists, books, periodicals, organizations, and links to other sites.

- STORM - Software Testing Online Resources/MTSU - a well-organized site with listings of many links to software QA and testing-related web sites.

- Cem Kaner's software testing site - Cem Kaner's site about software testing, legal issues, test automation, 'what is a serious bug?', software quality economics, outsourcing, and more. Also links to his 'badsoftware.com' website , a consumer and legal-issues orientation to software quality issues.

- Managing a Software Test Team - Article on issues in managing a test group; by James Bach; from Data Dimensions web site.

Return to top of Resources Listing

Software QA and Testing-related Organizations

Society for Software Quality - Has chapters in Houston, San Diego, and Washington DC area; each with monthly meetings.

SEI - Software Engineering Institute web site; info about SEI technical programs, publications, bibliographies, some online documents, SEI courses and training, links to related sites.

IEEE Standards - IEEE web site; has Software Engineering Standards titles and prices in the Information Technology - Software Engineering section.

American Society for Quality - American Society for Quality (formerly the American Society for Quality Control) web site; geared to quality issues in general, not just Software QA. Searchable site, has some information related to Software QA and their Certified Software Quality Engineer (CSQE) certification program.

QAI - Quality Assurance Institute

Return to top of Resources Listing

Links to QA and Testing-related Magazines/Publications

SEI Interactive - From Carnegie-Mellon U. Software Engineering Institute; each issue has a focus topic such as 'requirements engineering'; includes various articles, regular columns, and other features such as discussion groups/message boards. heavily oriented to software process improvement and quality assurance.

Software Quality Professional Magazine - Published by the American Society for Quality; web site includes table of contents and full text of selected articles.

Software Testing and Quality Engineering Magazine - Web site has full text of each print issue's featured article.

Quality Techniques Newsletter - Monthly online QA and testing newsletter from Software Research, Inc.; includes current and past issues.

TechWeb - CMPMedia's tech info site includes a variety of computer technology news including online versions of Information Week, Network Computing, and Communications Week; searchable. Has some reviews of web site test and management tools. Also includes links to online sites Byte, Data Communications, InternetWeek, Planet IT, TechShopper, Windows Magazine, Computer Telephony, Dr. Dobbs, Intelligent Enterprise, NT Systems, Performance Computing, Software Development, Wall Street and Technology, Web Review, and others.

Return to top of Resources Listing

General Software QA and Testing Resources

(Note: also see the 'Books' section for a listing of books on Software QA, Testing, and related subjects.)

●Risk-Based Testing - Article by James Bach in Software Testing and Quality Engineering Magazine.

●Best Practices Survey - Article titled 'A Worldwide Survey on Best Practices Toward Software Engineering Process Excellence' from Dec 99 issue of ASQ 'Software Quality Professional Magazine'. Includes a listing and survey rankings of 444 'best practices' candidates culled from a variety of software development models.

●Risks Digest - Digest of the 'Forum on Risks to the Public in Computers and Related Systems'. Includes latest issue and archives covering software and system problems, vulnerabilities, disasters; based on the comp.risks newsgroup.

●CMMI-SW to replace CMM - The SEI's CMMI project is developing a product suite that provides a set of integrated products to support process and product improvement and a common assessment methodology for multiple domains, including software engineering. The initial priorities are:
* CMMI-SW for the software-only model
* CMMI-SE/SW for the systems and software engineering integrated model
* CMMI-SE for the systems engineering model
* CMMI-SE/SW/IPPD for the systems, software engineering, and integrated product & process development model
It is intended to support several assessment types including 'quick-look assessments, first assessments, and reassessments'. The CMMI-SW incorporates what was Draft C of Version 2.0 of the Software Capability Maturity Model.

●Software Reliability Engineering for Managers - Good introductory article on 'Software Reliability Engineering' by John Musa.

●Construx Software Resources - Site with many useful resources such as articles like 'Why Projects Fail', 'Software's 10 Essentials', 'Sample Software Project Management Documents', estimation info and resources, various checklists, and Steve McConnell's 'Software Survival Guide' website.

●Sample Test Plans - Sample software system test plan in HTML format, plus links to many other sample test plans or templates.

●CM FAQ - Configuration Management FAQ edited by David Eaton; includes 'What is CM?', 'How should a CM system relate to process enforcement?', etc.

●SR/Institute's Software Quality Hot List - Extensive collection of links to many QA and testing-related articles, resources, etc.

●How To Build Reliable Code - Old but timeless article from 12/95 issue of Byte Magazine.

●Checklist of common GUI problems - Good starting point for GUI test considerations.

●Vendors' Right to Ship Buggy Software Under Fire - Interesting PC World Online 3/98 article concerning software quality. Primarily about Ralph Nader's CPT (Consumer Project on Technology) group and it's issues with the proposed new Uniform Commercial Code Article 2B (In 1999 it was recast as the Uniform Computer Information Transactions Act - UCITA), which would implement new laws in all 50 states concerning software quality. The battle over this controversial proposal is continuing into the latter half of 1999. A discussion of software quality issues and the proposed law is at the CPT web site.

● NASA SW Engineering Lab site - NASA's Software Engineering Lab web site; includes publications such as "Recommended approach to Software Development". Though this publication is several years old, it is an excellent online guide to the SW development process with extenisve sections on System Testing, Acceptance Testing, and other QA and testing issues. Other publications freely available at the site include "Cost and Schedule Estimation Study Report", "C Style Guide", "Software Measurement Guidebook", and "Improving the Software Testing Process in NASA's Software Engineering Laboratory".

● Kerry's Software Testing Web Site - Large well-organized collection of testing-related links with good descriptions of each link. Organized into sections on associations, tools, training, services, reference info, conferences.

● ASQ-Certified Software Quality Engineer (CSQE) - ASQ (American Society for Quality) Certified Software Quality Engineer - information on requirements, outline of required 'Body of Knowledge', listing of study references and more.

● QAI (Quality Assurance Institute) -- Certification Programs - For CQA and CSTE certifications.

● "Negotiating Testing Resources" - Excellent article by Cem Kaner about testing project planning and budgeting; from a 1996 software quality conference.

● Bret Pettichord's Software Testing Hotlist - Web site with links to various test and QA-related info; good list of test automation articles, other useful web sites, info about several good books.

● Software Engineering Resources - Large collection of useful information and links to many other sites and resources, all related to the SW engineering process including project planning and management, metrics, risk analysis, programming methods, OO SW engineering, testing, QA, CM. From R.S. Pressman, author of the book 'Software Engineering, A Practitioner's Approach'.

● Software Technology Review - Software Engineering Institute's technology descriptions listing - summaries of many software terms and technologies such as CORBA, COM, OO Design, Requirements Tracing, etc

● Software Test Coverage Analysis article Article containing a good discussion of test coverage analysis from Bullseye Testing Technology, maker of "C-Cover Test Coverage Analyzer" tool.

● Cleanroom Software Engineering - 'Introduction to Cleanroom Software Engineering for Managers'; long web document, good intro to 'Cleanroom Software Engineering'.

● BugNet Home Page - Lists of commercial software bugs, fixes, news, etc.

● comp.software-engineering FAQ - FAQ from the comp.software-engineering newsgroup; has a lot of useful info and links related to QA and testing. FAQ includes 'What's a function point', What's a bug?', What is 'clean room?', etc.

● Software QA resources list - Large collection of links to QA and testing organizations, literature, conferences, web sites, tools info, articles, etc. from RST Corp.'s web site comp.object FAQ - Extensive FAQ for object oriented subjects; includes some info about object-oriented testing.

CMM ver 1.1 info - CMM ver 1.1 docs (SEI-93-TR-24 & 25) from the source (SEI), in pdf format. Note: Draft C of v2.0 of the CMM is now incorporated in the 'CMMI-SW'.

Return to top of Resources Listing

Web QA and Testing Resources

Crash-Proof Your Web Site - Article on 'What you need to do to keep your site up and running' from ZDNet PC Magazine online; discusses load balancing, redundancy, clustering, caching, etc. Reviews related products.

Avoiding Scalability Shock - Article on 'Five Steps to Managing the Performance of E-business Applications' by Billie Shea from Software Testing and Quality Engineering Magazine.

Top 10 Web Site Performance Issues - Interesting summary of sources of performance problems for web sites, provided by Keynote Systems. Emphasizes connectivity issues such as backbone provider and location. Also see the related Business Site performance summary which lists 40 well-known business web sites along with each site's net access provider and web server software.

Handling and Avoiding Web Page Errors - Three part series from Microsoft site; covers sources of common Web page errors, how to handle run-time script errors, and techniques for avoiding preventable errors.

Testing Web Server Stress - Article from 10/98 issue of LAN Times discussing web site stress testing and server performance monitoring and management.

The NT Application that Wouldn't Die - Interesting article from Fall 1998 Enterprise Development Magazine about NASDAQ's NT web server system that was originally designed for 10,000 hits/day but reached 20 million hits/day (peak load 440 hits/sec among 7 servers). Discusses load balancing approach, issues of server programming, data replication issues, and architecture issues (such as how many database servers per web server).

Testing Database-Driven Web Sites - Article on web testing from 'DBMS Magazine' (now 'Intelligent Enterprise' Magazine).

Scalability Zone - Allaire's site with discussions of scalability issues, useful although oriented toward Allaire's Cold Fusion product.

InternetNews.Com - Large collection of news summaries on current web-related topics, including web development, browsers, servers, e-commerce, java, javascript, hardware, etc. and links to many related publications.

BUILDER.COM - CNet's Web Developer site with info/articles about web authoring/programming, servers, business, HTML standards updates, technology updates, etc.

CGI FAQ - Nick Kew's CGI FAQ - useful background info for those doing web site testing.

WWW FAQ - Maintained by T. Boutell; good resource for html authoring and web sites, includes a few items relevant to testing.

WebSite Performance Analysis - Old but still interesting discussion of web site performance analysis, capacity testing, metrics by Robert Denny; May 1996.

(Note: Also see the Y2K section which has several links to info concerning web and Javascript Y2K problems.)

Return to top of Resources Listing

Web Security Testing Resources

SANS website - Web site of SANS (System Administration, Networking, and Security Institute) web site, a cooperative research and education organization through which more than 96,000 sysadmins, security professionals, and network administrators share lessons learned and solutions.

Security Focus.Com - Site for news, forums, resources, vulnerability info, conference info, tools, etc. related to computer security including web and internet security issues. Search vulnerability database by keywords, date, vendor, version, etc.

The WWW Security FAQ - Web security FAQ maintained by Lincoln Stein and hosted by the W3C Consortium (the folks who set web standards/protocols, etc.)

COAST Security Archive - Purdue University's computer security site; includes extensive collection of links organized by subject to security tools, info resources, etc. Tools list of more than 100 security tools includes many test tools such as CRACK, COPS, IPSend, Tiger, Secure Sun, etc.; all tools listed are available for download from the COAST site.

Microsoft Security Advisor - Microsoft's web site for discussion of security issues for MS products, including their web server products.

V-One Internet Firewalls FAQ - V-One's Internet firewalls primer with basic info re firewalls, proxy servers, various types of security problems, glossary of firewall-related terms.

Mitre's Computer Security web reources - Mitre Corp.'s site with extensive listing of commercial, US Govt., and International security info sites, hacker info sites, FAQ's, firewalls, and more.

Computer Emergency Response Team site - CERT's web site; contains web server security information.

FIRST security site - 'Forum of Incident Response and Security Teams' computer security info web site.

Return to top of Resources Listing

Web Usability Resources

● Web Usability: Why and How - From Jakob Nielsen's 9/98 'User's First' column; how to design and test the usability of a web site. He also has related info at his website at www.useit.com such as 'How Users Read on the Web", 'Costs of User Testing', and 'Differences between Print Design and Web Design'.

● Usability Testing of Advanced Web Concepts - Article on usability testing at Sun web site.

● Site Usability Testing - Interesting article in 10/97 WebReview by Keith Instone about web site usability testing; this is one of his regular "Usability Matters" columns.

● User Interface Engineering - Site of UIE, a UI training and consulting company; info and links re usability testing and a chapter from the book "Web Site Usability" by Jarod Spool.

● Microsoft Usability Home Page - Microsoft's collection of info about their usability labs and related issues; includes info from the 'Human Factors and the Web' conference series.

Return to top of Resources Listing

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

© 1996-2000 by Rick Hower

Software QA and Testing Tools Info

●CM & Problem Management FAQ/Tools Lists - Excellent, comprehensive resource for Configuration Management and Problem Management. Includes CM FAQ, tool descriptions and reviews, links to related web sites. Based on the FAQ from usenet news group comp.software.config-mgmt.

●Test Tools Supplier List - Formerly Brian Marick's listing, now maintained by Danny Faught; listings and summaries of test tools, and links to their web sites.

●The CM Yellow Pages - Configuration Management Today Yellow Pages site; extensive collection of links for CM-related conferences, jobs, commercial and public-domain CM tools, info links, training, product reviews, commercial and public-domain problem tracking systems.

●Web Site Test/Management Tools Information - Software QA/Test Resource Center's listing of more than 170 web site test and management tools. Load testing tools, HTML validators, security test tools, link checkers, site mapping tools, site monitoring services, functional and regression test tools, site administration and maintenance tools, version control, java test tools, etc.

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

Web Site Test Tools and Site Management Tools

- The description of each tool is based on descriptions provided in its web site.
- Check the listed sites for latest product capabilities, platforms/servers/clients supported, etc.
- New additions in each update are added to top of list in each section, for easy browsing by users who check in periodically for newly-listed tools.
- See How can World Wide Web sites be tested? in the FAQ Part 2 for a discussion of web site testing considerations; also there are several articles about web site testing and management in the 'Other Resources' section.

Organization of Web Test Tools Listing

This tools listing has been loosely organized into the categories shown below.
Categories are not well-defined and some tools could have been listed in several categories. (Note that the 'Web Site Management Tools' category includes products that contain: site version control tools, combined utilities/tools, server management and optimization tools, and authoring/publishing/deployment tools that include significant site management or testing capabilities.) Suggestions for category improvement are welcome; see bottom of this page to send suggestions.

Load and Performance Test Tools

●WebSpray - Load testing tool; includes link testing capabilities; can simulate up to 1,000 clients from a single IP address; also supports multiple IP addresses with or without aliases. For Win 98/2000/NT4.0

●TestWorks/Web - Collection of web test tools for capture/playback, load testing, etc. from Software Research, Inc. Includes their XVirtual load generation tool. For Win95/NT and UNIX platforms.

●WebPerformance Trainer - Load test tool emphasizing ease-of-use. Supports all browsers and web servers; simulates up to 200 users per playback machine at various connection speeds; records and allows viewing of exact bytes flowing between browser and server. Modem simulation allows each virtual user to be bandwidth limited. Can automatically handle variations in session-specific items such as cookies, usernames, passwords, and any other parameter to simulate multiple virtual users. For NT, Linux, Solaris, most UNIX variants.

●WebSizr/WebCorder - Load testing and capture/playback tools from Technovations. WebSizr load testing tool supports authentication, cookies, redirects and requires Win95 or NT; WebCorder is a java-based tool requiring JDK 1.0.2 or higher.

●Benchmark Factory - E-commerce load testing tool from Client/Server Solutions, Inc. Includes record/playback, web form processing, user sessions, scripting, cookies, SSL. Also includes pre-developed industry standard benchmarks such as AS3AP, Set-Query, Wisconsin, WebStone, and others. Includes optimized database drivers for vendor-neutral comparisons - MS SQL Server, Oracle 7 and 8, Sybase System 11, ODBC, IBM’s DB2 CLI, Informix. Controlled by a 'Visual Control Center' on NT.

●MS Web Application Stress - Microsoft stress test tool created by Microsoft's Internal Tools Group (ITG) and subsequently made available for external use. Includes record/playback, script recording from browser, SSL, adjustable delay between requests, custom header per request, configurable number of threads, sockets, users, scripts, supports ASP, cookies. Site includes a useful web load/stress testing tutorial. For testing ASP web sites running on NT Server or Win2000; MSIE 4.0 or newer required. Appears to be freeware at present.

●Rational Suite Performance Studio - Rational's client/server and web performance testing tool. 'LoadSmart Scheduling' capabilities allow complex usage scenarios and randomized transaction sequences; handles dynamic web pages. Requires NT for master station; agents can be NT or UNIX.

●FORECAST - Load testing tool from Facilita Software for web, client-server, network, and database systems; for UNIX platforms.

●Zeus Free Web Load Test Tool - Free web benchmarking/load testing tool available as source code; will compile on any UNIX platform. Similar to the tool that ships with Apache 1.3.x web server.

●e-Load - Load test tool from RSW geared to testing web applications under load and testing scalability of E-commerce applications. For use in conjunction with test scripts from their e-Tester functional test tool. Allows on-the-fly changes and has real-time reporting capabilities. Utilizes 'Data Bank Wizard' to vary usernames, passwords, or other data for each virtual user. For Win95/98/NT.

●VeloMeter - Java-based web load test tool, includes source code. Requires JDK 1.1.6 or greater. 2

versions: Free and Pro

●http-Load - Free load test application to generate web server loads, from ACME Software. For Unix.

●Compuware's QALoad - Compuware's QALoad for load/stress testing of database, web, and char-based systems, works with such middleware as: SQLnet, DBLib or CBLib, SQL Server, ODBC, Telnet, and Web. The load test manager runs on Win95/NT and the load test player runs on UNIX and NT platforms.

●Microsoft WCAT load test tool - Web load test tool from Microsoft for load testing of MS IIS on NT.

●Portent Web Load test tool - Loadtesting.com's low-priced web load testing tool. Written in Java; multi-platform; evaluation version available.

●SilkPerformer - Load and performance testing component of Segue's Silk web testing toolset.

●WebART - WebART , from OCLC, Inc. - tool for load testing of up to 100-200 simulated users; also includes functional and regression testing capabilities, and capture/playback and scripting language. Evaluation copy avail. For Win 3.1/95/NT.

●Performix - Rational's (formerly PurePerformix's) Performix web site load testing tool. Supports Unix environments.

●WebLoad - Web load testing tool from Computer Associates (Note: this tool was originally Radview's WebLoad tool and it was distributed by Platinum Technologies, which is now a part of Computer Associates.) For Win95/NT, Solaris, AIX.

●Radview's WebLoad - WebLoad web site load testing tool from Radview Software. Suports recording of SSL sessions, cookies, proxies, password authentication, dynamic HTML; multiple platforms. Evaluation version available. For Win95/NT, Solaris, AIX.

●Astra Loadtest - Mercury's load/stress testing tool; includes record/playback capabilites; integrated spreadsheet parameterizes recorded input to exercise application with a wide variety of data. 'Scenario Builder' visually combines virtual users and host machines for tests representing real user traffic. 'Content Check' checks for failures under heavy load; Real-time monitors and analysis ; for Win3.1/95/NT and Solaris, SunOS, HP-UX, IBM AIX, NCR.

Return to top of web tools listing

Java Test Tools

●Panorama for Java - Contains six integrated java tools; JavaSQA for Object-Oriented software quality measurement; JavaDocGen for Java code static analysis; JavaStructure for Java code structure analysis and diagramming; JavaDiagrammer for Java code logic analysis, control flow analysis and diagramming; JavaTest for test coverage analysis and test case minimization, etc.; and JavaPlayback for GUI operation capture and automatic playback. For Win95/98/NT.

●Java Tool Suite from Man Machine Systems - Includes JStyle, a Java source analyzer to generate code comments and metrics such as inheritance depth, Cyclomatic Number, Halstead Measures, etc; JPretty reformats Java code according to specified options; JCover test coverage analyzer; instruments Java source for analysis of execution trace and coverage of branches, statements, methods, classes, file, and package; the JVerify Java class/API testing tool uses an invasive testing model allowing access to internals of Java objects from within a test script and utilizes a proprietary OO scripting language; and JMSAssert, a tool and technique for writing reliable software.

JProbe Developer Suite - Collection of Java debugging tools; includes JProbe Profiler and JProbe Memory Debugger for finding performance bottlenecks and memory leaks, LProbe Coverage code coverage tool, and JProbe Threadalyzer for finding deadlocks, stalls, and race conditions. ServerSide edition available for web server-side java development. Supports Win95/98/NT and Solaris.

Documentary for Java - Automated source code documentation tool; produces HTML documentation to all classes, methods, etc. in project as well as outher reference information; for Windows, Solaris, HPUX.

Krakatau Metrics for Java - Software metrics tool includes more than 70 OO, procedural, complexity, and size metrics related to reusability, maintainability, testability, and clarity. Has online advisor for quality improvement; text, graphical, and HTML reporting capabilities. For Windows, Solaris, HPUX

Metamata Java Code Analysis Tools - Suite of tools for analysis of Java source code quality, complexity and quality metrics, debugging, and more.

McCabe Visual Test - 'McCabe Visual Testing ToolSet' and 'McCabe Quality ToolSet', with Java coverage and metrics capabilities.

Assure for Java - Java debugging tool - monitors memory, references, object creation, and lock acquisitions. Finds thread deadlocks, stalls, data race conditions, synchronization errors. Results displayed in browser with pointers to source code. For Win95/NT, Solaris.

OptimizeIt - Java language profiling tool, analyzes use of memory and CPU resources to find bottlenecks and to solve Java performance problems. Supports Win95/NT and Solaris.

jTest! - ParaSoft's automated white box Java test tool.

DevPartner for Java - NuMega/Compuware's debugging/productivity tool to detect and diagnose Java bugs and performance problems, thread and event analysis, coverage analysis.

VTune - Intel's performance tuning tool for applications running on Intel processors; includes Java support.

SilkBeans - Segue's Java test tool; part of their Silk web test tool suite.

AppletLoad - Part of Radview's WebLoad tool set; for performance testing of Applets and 'Java-implemented protocols'.

Sun's Java Test Tools - As of February 4, 2000 Sun discontinued accepting orders for these products.

TCAT for Java - Code coverage analyzer and code analysis for Java applets; written in java. For Unix.

● DeepCover for Java - Code coverage tool for Java from RST. For Solaris and Win95/NT.

● TotalMetric - RST's Code metrics and OO metrics tool for Java.

● AssertMate - RST's code assertion toolkit for Java programmers and class level testers.

(Note: some other tools in these listings also handle testing, management, or load testing of java applets, or are planning to add such capabilities. Check listed web sites for current information.)

Return to top of web tools listing

Link Checking Tools

● LinkGuard Online - LinkGuard Online, free on-the-web broken link checking service.

● Xenu's Link Sleuth - Freeware link checker by Tilman Hausherr; supports SSL websites; partial testing of ftp and gopher sites; detects and reports redirected URL; Site Map; for Win95/NT.

● Linkalarm - Low cost on-the-web link checker; free trial period available. Automatically-scheduled reporting by e-mail.

● Theseus - Link checker for Mac; evaluation version available.

● Alert Linkrunner - Link check tool; evaluation version available. For Win95/98/NT.

● I-Control WebWeaver - Link checker; evaluation version available; for Win95/NT.

● RiadaLinx - Link checker; evaluation copy available; for Win95/NT.

● Linkbot - Tetranet's tool for checking links, missing page titles, slow pages, stale content, site mapping; from local drive or remote server. Evaluation copy available; for Win95/NT.

● InfoLink - Link checker program from BiggByte Software; can be automatically scheduled; includes FTP link checking; multiple page list and site list capabilities; customizable reports; changed-link checking; results can be exported to database. Freeware and evaluation versions available. For Win95(?).

● LinkScan - Electronic Software Publishing Co.'s link checker/site mapping tool; capabilities include automated retesting of problem links, randomized order checking; can check for bad links due to specified problems such as server-not-found, unauthorized-access, doc-not-found, relocations, timeouts. Includes capabilities for central management of large multiple intranet/internet sites. Evaluation copy available. For UNIX, and Win98/NT servers. Requires Perl 5.

● CyberSpyder Link Test - Shareware link checker by Aman Software; capabilities include specified URL exclusions, ID/Password entries, test resumption at interruption point, page size analysis, 'what's new' reporting. For Win3.1/95/NT.

● Surf! - Link-testing tool used in Segue's Live Quality-Web Site web test tool suite.

● Big Brother - Shareware link checker for Mac, Linux, Win95/NT.

Return to top of web tools listing

HTML Validators

RealValidator - Shareware HTML validator based on SGML parser by Liam Quinn. Unicode-enabled, supports documents in virtually any language; supports HTML 4.0, HTML 3.2, HTML 3.0, and HTML 2.0; extensible - add proprietary HTML DTDs or change the existing ones; fetches external DTDs by HTTP and caches them for faster validation; HTML 3.2 and HTML 4.0 references included as HTML Help. For Win95/98/NT; MSIE 4.0 or greater, HTML Help 1.1

CSE 3310 HTML Validator - Shareware HTML validator for Win95/NT. (This tool is also integrated into the 'HomeSite' web development/authoring tool.)

Spyglass HTML Validator - Free standalone local-drive HTML validator. Color coded display, errors in list can link directly to error location in page; editable corrections within display; selectable validation levels; for Win95. (Product is no longer supported but is still available at web site.)

(Note: Many of the products listed in the Web Site Management Tools section include HTML validation capabilities.)

Return to top of web tools listing

Free On-the-Web HTML Validators and Link Checkers

WDG HTML Validator - Web Design Group's validator - latest HTML version support, flexible input methods, user-friendly error messages.

MetaMedic - Northern Webs' free on-the-web meta tag checker; professional version available for purchase also.

Web Page 'Purifier' - Free on-the-web HTML checker allows viewing a page 'purified' to HTML 2.0, HTML 3.2, HTML 4.0, or WebTV 1.1. standards.

W3C HTML Validation Service - HTML validation site run by the WWW Consortium (the folks who set web standards); handles one URL at a time; site includes specifications info for several recent HTML standards.

NetMechanic - Link checker and HTML validator by Monte Carlo Software. Type in the site URL to check, then later receive e-mail with the URL of a web page that contains the results. Validator can choose among standard HTML versions or MSIE or Netscape versions or combinations.

Bobby - HTML validator; can validate against AOL browsers, MSIE, Netscape, WebTV, specific HTML standards, and Lynx. Can run validation against multiple standards simultaneously; alos checks page for accessability to users with disabilities.

Doctor HTML - Site with online web page checker by Imagiware. Checks spelling, forms, tables, tag usage, links. Analyze by page or by site. Selectable depth for sites. Also has info about their SiteDoctor product, which checks an entire site.

●EWS Weblint Gateway - Site with online HTML validator; somewhat configurable.

●Web Page Backward Compatibility Viewer - On-the-web HTML checker; will serve a web page to you with various selectable tags switched on or off; very large selection of browser types; to check how various browsers or versions might see a page.

●MindIt - NetMind's 'MindIt' (formerly 'URL-minder') sends e-mail whenever a URL is updated. Can enter many in a maintained list. Can be used to keep track of changes to external linked documents/pages.

Return to top of web tools listing

PERL and C Programs for Validating and Checking

●HTML TIDY - Dave Raggett's free utility available from the WWW Consortium; for automatic fixing of HTML errors, formatting disorganized editing, and finding problem HTML areas. Available as C source code, Java, or binaries available for Win95/98/NT, Amiga, Mac, AIX, Java, Linux, and others.

●SWAN - SubWeb Analyzer, a UNIX shell script that analyzes a local subweb of the web; for maintaining independent subwebs. Collects statistics on a subweb and reports internal inconsistencies; generates a cross reference that enables following links in reverse direction. Used for finding all referencing documents when changing a page or when looking for related documents.

●Linklint - Perl shareware link checker works with Perl 4 or 5 on Win or Unix platforms.

●MOMspider - Multi-Owner Maintenance Spider; link checker. PERL script for a web spider for web site maintenance; for UNIX and PERL 4.036. Utilizes the HTTP 'HEAD' request instead of the 'GET' request so that it does not require retreival of the entire html page. This site contains an interesting discussion on the use of META tags.

●Weblint - PERL script for web page syntax and style checking; for UNIX, Mac, NT, OS/2.

●PERL WWW test tools - Collection of PERL scripts at NASA site for link checking, html validations, etc.

●HTMLchek for awk or perl - Old but still useful HTML 2.0 or 3.0 validator programs for AWK or PERL; site has much documentation and related info.

●SP SGML Parser - Free source code , C++, for SGML parser; can be used for HTML validation; for DOS/UNIX/Win95; this code is the basis for several on-the-web html validators.

Return to top of web tools listing

Web Functional/Regression Test Tools

●Macrobot - Functional and regression test tool for e-business websites from Watchfire.com. Capture/playback capabilities, includes spreadsheet-like data table feature that allows use of varying values for user accounts, order baskets, bank account transactions, credit card types, etc. 'Element testing' feature allows testing for specific page elements including forms, graphics, text, and links. 'Test Plan' feature allows automatic generation of test plans by generating 'plain English' explanations of recording test scripts for tracking web development and QA processes. Requires NT4.0 SP4 or later, Win95/98, MSIE 4.01 SP1 or higher.

●Rational Visual Test - Rational's functional testing tool; includes object testing automation tool which recognizes objects in Java, HTML, DHTML, ActiveX, Visual Basic, Visual C/C++, PowerBuilder, Oracle Developer/2000. For 95/98/NT.

●e-Tester - Web functional/regression test tool from RSW. Includes record/playback, scripting language; can test image maps, frames, forms, links, ActiveX, Applets, VBScript, JavaScript. Includes SiteSpider site element detection and mapping tool, and a 'Data Banks' feature for data-driven automated testing. Supports external custom test logic via External Callout Facility. Special test extensions for Net Dynamics, Web Objects, Cold Fusion, Microsoft, and Silverstream. Evaluation version available. For Win95/98/NT.

●Astra QuickTest - Mercury's product for functional testing of web-based e-business applications; can emulate various browser types and versions and HTTP protocols. For Win95/98/NT.

●Compuware's QARun - QARun for functional/regression testing of web, character-based, and GUI systems; for Win95/NT

●FormAgent - VCI's FormAgent for automated form-filling; can be configured for multiple users. Can be adapted for partial automation of forms-based web testing. Evaluation copy available; for Win95/NT, NS3.0+, MSIE3.x.

●AutoTester Web - AutoTester Corp.'s link check and site test tool; includes scripting capabilities, automated documentation, reporting functionality, and the ability to test Java applets and ActiveX. Works with MSIE or Netscape browsers. For Win95/NT/AIX/Solaris/Tandem/NextStep

●WebART - WebART , from OCLC, Inc., tool for functional, regression, and performance testing. Includes capture/playback and scripting language; can simulate 100-200 users in load testing. Evaluation copy avail. For Win 3.1/95/NT.

●SilkTest - Segue's web testing tool for functional and regression testing; includes capabilities for testing Java applets, HTML, ActiveX, images; works with MSIE, Netscape, includes capture/playback capabilities.

Return to top of web tools listing

Web Site Security Test Tools

●HackerShield - Security scanner from BindView; examines servers, workstations, routers, hubs, printers, and any other devices with an IP address on a network, for security holes that hackers could use to break-in. Scans any device on network regardless of platform; updates are sent via secure email and automatically integrated database of security checks. Runs under NT4.0 with SP 4 or better and MSIE.

●Cyber Attack Defense System - Monitors and analyzes suspicious activity at Internet gateways and public Web servers; takes action to stop cyber attacks including distributed 'Denial of Service' attacks; integrates with third-party OPSEC products such as switches and load balancing solutions; for Solaris and NT.

●ITS4 - Open source software tool from RST Corp. automatically scans C/C++ source code for potential security vulnerabilities; for UNIX and Windows with CygWin.

●HostCheck - Suite of security test and management tools from DMW Worldwide. Checks for presence of sniffers and insecure network configuration, secure file removal by repetitive overwriting of location, checks for pre-existing security problems against vulnerability database, file/directory permissions management and monitoring, password security testing, user management and account security checking, security reporting and test scheduling. For UNIX.

●WebTrends Security Analyzer - Web site tool to detect and fix security problems. Includes periodically- updated Expert knowledge base. For Win95/98/2000/NT

●Secure Scanner - Cisco's product for detecting and reporting on Internet server and network vulnerabilities; risk management; network mapping. For NT or Solaris.

●Inspectorscan - Shavlik Technology's security analysis and reporting tool; for NT.

●Netective Site - Netect's product for analyzing internal and external security vulnerabilities. For Solaris.

●CyberCop SCANNER - NT/Limux security auditing tool from Network Associates for Intranets, web servers, firewalls, and screening routers.

●COAST Security Archive - Purdue University's computer security site; includes extensive collection of links organized by subject to security tools, info resources, etc. Tools list of more than 100 security tools includes many test tools such as CRACK, COPS, IPSend, Tiger, Secure Sun, etc.; all tools listed are available for download from the COAST site.

●CERT FTP Directory of /pub/tools - CERT ftp site with collection of some site security tools; some of these tools can be adapted for use in testing a site's security.

●NukeNabber - For monitoring TCP and UDP ports for intrusions; includes tracer utilities. Freeware from DSI; for Win95/98/NT.

●NetRecon - Axent Technology's tools for testing vulnerabillity of network systems and resources from inside or outside the firewall; exploits IP and non-IP-based protocols; selective targeting capabilities; creates HTML reports; 'Intruder Alert' capability for 7x24 monitoring. Probes any platform type. Runs on NT.

●SAFEsuite - Collection of security products from Internet Security Systems for security assessment, intrusion detection and security management. Includes 'Internet Scanner' for security scans of devices at the network level; 'Database Scanner' for protecting database applications through security policy creation, compliance, and enforcement, available for Oracle, MS SQL Server, and Sybase. 'System Scanner' searches online operation to provide host-based security assessment targeting weaknesses undetectable through network scanning. For NT and UNIX platforms.

●SATAN web security testing tool - Security Administration Tool for Analyzing Networks - for UNIX. Includes tutorial that explains problems found, what its impact could be, and what can be done about it: correct errors in a configuration file, install vendor bugfixes, use other means to restrict access, or disable service. Includes some links to related information.

Return to top of web tools listing

External Site Monitoring Services

●eValid - Service from Software Research, Inc.; benchmark site response hour to hour and day to day; evaluate E-Commerce transactions and uses text validation capabilities. Geographic availability in multiple time zones; simulation of user interaction including type-ins and FORM submittals.

●ActiveTest - Mercury Interactive's hosted Web-based load testing e-service; emulates thousands of users against a staging server; emergency 24-hour turnaround available.

●Inverse Internet Benchmark - Suite of network and site performance, availability, and other monitoring services from Inverse Network Technology; monitoring from multiple locations.

●Keynote Monitoring - Site performance and availability monitoring service from Kenote Systems; service available from single or multiple locations.

●Topaz ActiveWatch - Web site application performance management service from Mercury Interactive.

●WatchMyWeb - Free web site monitoring service from RingSys; web site is tested every thirty minutes. If a problem is detected, three attempts are made at 30 second intervals before sending alarm via email or pager.

●SiteAlert - Service checks as often as every 5 minutes, 24 hours a day; configurable monitoring options include head, get, get all, post; and transx for monitoring users' page by page shopping experience. Detailed reports accessible online via SiteAlert website. Alerts by phone, fax, pager or e-mail.

●SiteSeer - Monitoring service utilizes monitoring locations on major Internet backbones. Tests online transactions and associated processes; receive notification of errors via e-mail or pager; verify contents including specific text or test to ensure specific text not present (i.e., an error message), identify local versus global access problems, daily and weekly customized management reports.

●SM-Web - Web site performance measurement service, measures from the end-user's perspective. Monitors various statistics such as response time, throughput, end-To-end time comparison among 10 select cities, and metrics for e-commerce transactions, searches, and more.

●ServerCheck - NetMechanic's on-the-web tool checks a web server's performance periodically for 8 hours; the resulting report sent by e-mail includes comparisons to other servers. Site also includes a link-checker, HTML validator, spell checker,

● NetMechanic/ZDNet's Site Test Utilities - Same as above ServerCheck on-the-web testing capabilities but from a joint NetMechanic/ZDNet site.

Return to top of web tools listing

Web Site Management Tools

(This section includes products that contain: site version control tools, combined utilities/tools, server management and optimization tools, and authoring/publishing/deployment tools that include significant site management or testing capabilities.)

● BladeRunner - web content design,creation mgmt, publish tool from BroadVision Inc. Enables companies to create, manage and publish 'e-content' for their web enabled applications, using XML as its technology backbone and MS Wordfor content creation.

● WS_FTP Pro - Web publishing tool from Ipswitch; manage, upload, and update websites; automatically resume interrupted transfers; support more than 50 host file systems; drag-and-drop files. Win3.1/95/98/2000/NT

● JetStream - Site management suite for web server monitoring, link checker, real-time client usage analysis, log file analysis, problem determination, performance measurement and load-balancing. Can manage any web server on any platform; if server is either located on same NT system or same network segment with JetStream. For WinNT; is installed as a service and accessed via any standard Java and JavaScript enabled browser.

● A1Monitor - Utility from A1Tech for monitoring availability of web servers. Capabilities include notification by email and automatic reboot of web server. For Win95/NT

● NetLoad - For synchronizing remote web sites with local web files; from Aerosoft Australia. Transfers only new and updated files, removes old files and directories, and creates entire directory structures on your FTP server to match those of your PC. For Win95/98/NT

● LivePage Enterprise - Multi-user XML/HTML content-base management system; has search capabilities, access control, LDAP support, more. For NT/Linux/Solaris

● Ringsys WebMonitor - Web site availability and monitoring tool for Windows NT, Solaris, HP-UX, Linux with support of ITO, Autosys, Unicenter TNG

● WebLog - In addition to logging capabilities, checks search engine placement including position, page, rank on page, click paths. Enhanced logging utilizing custom generated script uploaded to host; includes 'anti-caching technology' ensuring more reliable logging by blocking caching of site's pages by visitor's browsers; 'Active Protect MetaTags' allowing only search engine spiders to see front-page meta-tags; 'Active Protect Images' to prevent other sites from "borrowing" images; and 'Active Deny Access' to prevent particular IP addresses from accessing web site. Requires UNIX, PERL, SSI, on web server and Win95/98/NT client

WebReady Manager - Web postitionfrom analyzer and promotion tool from Monocle Solutions, checks a site's search engine rankings. Requires Win95/98/NT.

Web Automation Manager - Tool written in pure Java helps automate web-based processes, monitor web site and test e-commerce and other web applications; load testing, link checking. Supports cookies, GET-POST-PUT methods, web server authentication. Requires JDK/JRE 1.1.8 or 1.2 (remommended) with HotSpot compiler.

AgentWebRanking - Freeware tool AADSoft to monitor site's search engine position, improve search engine ranks, submit URL's. Searches top engines such as Yahoo, Excite, Alta Vista, InfoSeek, etc. for keywords; can specify search depth. Also has keyword count for pages vs competitor's pages; auto or manual submit of URL's to search engines, meta tag creator. Requires MSIE 4 or 5, Win95/98/NT

Web Site Garage - Services for maintaining/improving web site. Automate site maintenance checks, optimize graphics and analyze traffic. Free single-page on-the-web tune-up; or fee-based services for entire site. Services include Load Time Check, Link Check, Link Popularity Check, Spell Check, HTML Design Check, browser compatibility check for 18 different browsers, platforms and screen sizes; more.

SiteMARC - JAVA-based Site Traffic and Resource Management (SRM) tool from WebManage Technologies; intelligent Load Balancing using a variety of algorithms; Access Control, Content Management, Site Activity Reporting. Customizable real-time reporting.

INetMonitor - Microsoft tool for capacity planning, load monitoring, hardware configuration, and simulation. Included with the Microsoft Press' BackOffice Resource Kit (2nd Edition); will also be included in the forthcoming Microsoft Commercial Internet System (MCIS) 2.0 Resource Kit. Also includes integrated support for a wide variety of protocols, including HTTP 1.0 and 1.1, IMAP4, IRC, LDAP 2.0 and 3.0, MIC, NNTP, POP3, and SMTP; supports cookies and authentication methods used with Microsoft SiteServer. Monitors services and hardware.

EPrise Participant Server - Web site management tool from EPrise, for content and workflow management, access and version control, business rule management within a single platform. Runs on NT or Solaris.

WebKing SiteRuler - Web site management tool from ParaSoft, includes publishing manager, link checker, HTML checker with custom rule selection, orphan file checking, and more. For Win95/98/NT, Linux, Solaris.

WebSite Director - Web-content workflow management system with browser-based interface includes configurable workflow management, e-mail submission of web content, and e-mail notifications; allows defining and applying existing workflow and approval rules to web content management process. For NT, UNIX.

NetDialect - Multilingual web site management solution; uses a scalable Oracle database that reuses translation memory across multiple sites/projects simultaneously. Extracts information kernels from their native html files, then checks extracted information against an accumulating repository of translated sentences, key words or glossaries. Translated strings are automatically populated in the translation editor. Human or machine translation completes the translation process. After review for quality, the native file is automatically reassembled with the translated/updated information.

BazaarAnalyzer Pro - Aquas Software's web management and analysis tool; runs on web server, interface via any java-enabled browser. Manage multiple sites, users, security; customizable web report, real-time alerting for broken links, etc.; visitor analysis. For NT, Solaris, HP, Linux, SGI/IRIX, BSD/BSDI.

Equalizer - Load balancing server appliance and site management tool from Coyote Point Systems. Web based interface for load balancing administration, server failure detection, real-time server monitoring of server response time, number of pending requests, etc.

WebTrends Enterprise Suite - Web site management tool including log analysis, link analysis and quality control, content management and site visualization, alerting, monitoring and recovery, proxy server traffic analysis and reporting.

e-Monitor - Web tool from RSW for 7x24 web site monitoring. Can utilize test scripts created with their e-Tester tool; allows wide range of corrective action and notification responses. Includes a wizard script generator that generates scripts in standard Visual Basic. Evaluation version available. For Win95/98/NT.

HP FireHunter - HP's tool for monitoring, diagnosis, and management of internet services; including performance, fault detection, reporting. Supports a variety of platforms.

HotMetalPro - SoftQuad's web development tool for web site authoring/development/management, includes capabilities link management, site mapping. For Win95/98/NT.

Unicenter TNG w/Web Management Option - Site management application from Computer Associates includes access and security control, monitoring, logging, metrics, server management, network management. For MS and Netscape web servers.

Interwoven Team Site - Web development, version control, access control, and publishing control tool; works with many servers, OS's, and platforms.

Cold Fusion - Allaire's web development environment includes integrated test web server for immediate site testing, automatic database query tester, version control. Evaluation version available. For Win95/NT/Solaris.

Intranet Solutions - VitalSign Software's suite of products for dynamically monitoring faults, response times, congestion, downtime, bottlenecks, timeouts, performance changes for intranet systems. For NT.

Site/C - 'Set-and-forget' utility for periodic server monitoring for web server connection problems, link problems; e-mail notifications, logging capabilities. Evaluation version available; for Win95/NT.

Blueprint - Web site analysis and management tool, includes link checker, site mapper, reporting, statistics, integrated ftp and uploading. For Win95/NT.

WireTap - Continuously monitors and manages network, intranet, and Internet performance; provides real-time web traffic analysis, and monitors web and SQL transaction response times. Initially from Platinum Technologies, now a part of Computer Associates. Supports AIX, HP-UX, Solaris, NT.

PowerMapper - From Electrum Multimedia; for customizable automated site mapping, HTML validation, link checking. Evaluation copy available; for Win95/NT and MSIE 3.0 or later.

●SQA Sitecheck - Rational's web site management tool includes link checker, finds slow pages, simulates both Netscape and MSIE browsers. SSL support. For 95/98/NT.

●SiteScope - Freshwater Software's product for site monitoring and maintenance. Runs on servers and monitors server performance, links, connections, logs, etc. and provides notifications of problems. Includes published API for creating custom monitors. Monitors mimic users end-to-end actions. For NT or Unix.

●SITEMAN - Web site management and editing tool collection from Greyscale Systems. Link checking. Global search and replace. Checks for orphan files. Call HTML editor from within program. Has freeware and shareware versions. Evaluation copy available. For Win 31./95/NT.

●HTML PowerTools - HTML validator, global search-and-replace. Date stamper, spell checker, Meta manager, image tag checker, HTML-to-Text converter, customizable reports. Link checker. Validates against various HTML versions, NS and MSIE extensions; has updateable rulebase. From Talicom. Free 30-day evaluation copy. For Win3.1/95/NT .

●Astra SiteManager - Mercury's web site management tool; scans web site and hilites functional areas with color-coded links and URLs to provide a visual map of site. Site map includes HTML, cgi scripts, applets, etc. Shows broken links, access problems, compares maps as site changes, identifies usage patterns and validates dynamically generated pages. Change management/tracking. Evaluation copy available. For Win95/NT.

●HTML Grinder - Matterhorn Media's Web site mgmt/authoring software w/multi-file find and replace, filename management, TOC-builder, more. Evaluation version available. For Mac.

●Microsoft SiteServer - Web site development, maintenance, management tool includes site mapping, link checker, publishing control, site analysis; formerly MS Backoffice Live:WebMapper and also formerly NetCarta WebMapper until bought out by Microsoft. For MS Internet Server on NT.

●COAST WebMaster - Coast Software, Inc. WebMaster site management tool; for web site file management, link checking, site version comparisons, page download timing and estimating, server log file reporting. Includes HTML editor and file manager, page display verification, global search and replace. Evaluation copy available. For Win95/WinNT.

●BlackBoard Tracker - BlackBoard Software's web project management tool - shareware available via Download.com site in Internet->Site Management section. For tracking web project's code, etc. including program name, info, creation date, related files, comments, documentation, version, status, programmer, changes; reporting capabilities. For Win3.1 or better.

●SiteBoss - Opposite Software's tool for web site management, meta tag management, spell checking, diagnosis, repair, and optimization; global search and replace, HTML-to-text converter, link checker, HTML validatior; integrated HTML editor and browser. Updatable rulebase. Evaluation copy available. For Win95/NT.

●OpenDeploy - Interwoven's configurable control system for deploying from development to production environments. Includes automated deployment, security, and encryption capabilities. Rollback capabilities if used in conjunction with their TeamSite product.

●TeamSite - Interwoven's collaborative web site production control, administration, and management product for enterprise-wide internet and intranet projects. Includes version control, browser interface, comparison capbilitie, file edit merging, variable lock controls. Client side requires NS 3.01+ or MSIE3.01+; server side compatible with many available web servers.

●DynaBase - Inso's web site publishing and development management product for multi-user control, version control, link verification, and site deployment. Capabilities include remote access, muliple editions, deployment. Capabilities include remote access, multiple editions, dynamic publishing.

●StoryServer Web System - Vignette Corporation's product for web site collaborative content, publishing, management, and maintenance. Supports servers on Solaris, NT 4.0. Can optimize pages based on incoming browser, supports various attributes including: CSS, Cookies, DynaHTML, Frames, Java, JavaScript, Lynx, Macintosh, MSIE versions, Netscape versions, Tables, WebTV. Supports Oracle, Sybase, Informix, UC2, SQL Server, via native API's. Supports NS Enterprise Server, Apache, OpenMarket, MS IIS; supports NSAPI to Netscape, ISAPI to Microsoft IIS and Fast CGI to Apache and OMI. Requires Java Runtime Environment.

●eAuthor/Site - Template-based site authoring tool from HyperAct, Inc. with site management capabilities including global search and replace, global spell checking, table of contents creation, index creation, thumbnails, tree-based site map view, more. Evaluation version avail. For Win 95/NT.

●Table of Contents - 'Table of Contents' web site mgmt. tool - creates framed HTML lists of local files. Indexes and shows html files in one frame, view contents in another; can specify which types of file are indexed, and alphabetically sort the entries. Shareware for Win95, NT4.0. Evaluation copy avail.

●TestWorks/Web - Collection of web test tools for capture/playback, load testing, java testing from Software Research, Inc. Includes their CAPBAK/Web for Web test tool, XVirtual load generation tool, and TCAT for Java test coverage tool. For Win95/NT and UNIX platforms.

●MKS Web Integrity - MKS' 'Web Integrity' web object management system for web site maintenance and management; uses web-server-based repository and a java client interface; revision control, publishing control and security, audit trails. Works with Win95/NT Netscape or MS IE, and several Netscape and Microsoft web servers on NT and Unix.

●MS Visual Source Safe - Microsoft's version 5.0 of their software version control tool; includes site change management, web link checker, site mapping; can integrate with MS FrontPage. For Win3.1/95/NT/DOS/UNIX/Mac.

●Microsoft FrontPage - Microsoft's 'Front Page' web site authoring and site management tool; includes site management capabilities, link checking, etc. Evaluation copy available.

●HAHTSite - Haht Software's integrated web site development and management environment, for site authoring, publishing control, team development, link and object management, debugging.

●HomeSite - Allaire's web site authoring/validator tool; page in screen is validated with results displayed below; error info is linked to error in html; file uploading; link checker; thumbnail viewer. For Win95/NT.

●Mortar - Web site development/authoring/validation tool; includes capabilities for multi-file search/replace, custom tags, site mapping, project management. Evaluation version available. For Win95/NT.

●NetObjects Team Fusion - Site authoring/management tool. Visual site structure editor, layout editor, graphics management, staging/publishing control. Evaluation version available. For NT.

●Adobe SiteMill - Adobe's product for web site management, link checking/updating, page authoring and editing; available as add-on for PageMill; for Macintosh.

Return to top of web tools listing

Log Analysis Tools

●HTTPD Log Analyzers list - Most extensive log analysis tool listing on the net - more than 100 listed with short descritions of each, organized by platform: Unix, Win, NT, Mac

●Web Developers Virtual Library Log Analyzer Listing - Smaller log analysis tools listing, includes about 30 tools with descriptions

Return to top of web tools listing

Other Web Test Tools

●Browserola - Browser emulator from Codo Development that allows viewing of HTML in emulations of Netscape, IE, and W3C standards. For Win95/NT

●Enterprise Minder - NetMind's server-based application for keeping track of changing information on extranets/intranets/Internet. Allows each user to have granular control over what is tracked and update frequency. Works with existing browsers and email; works with most NT or Solaris servers. supports secure SSL, Basic and NT LAN Manager authentication and proxy servers.

●WebBug - Debugging tool from Aman Software for monitoring HTTP protocol sends and receives; handles HTTP 0.9/1.0/1.1; allows for entry of custom headers. Freeware.

●WebMetrics - Web usability testing and evaluation tool suite from U.S. Govt. NIST. Source code available. For Unix, Win95/NT.

●SACcat - Network administrative security tool from SAC Technologies - mouse-sized PC-compatible fingerprint reader device for integrating with network security systems. Can be used to control access to network and server management and security.

●MRTG - Multi Router Traffic Grapher - tool utilizing SNMP to monitoring traffic loads on network links; generates reports as web pages with GIF graphics on inbound and outbound traffic. For UNIX and

NT.

●Net.Medic - Performance monitoring tool from VitalSoft, division of Lucent Technologies. Assists in monitoring, isolating, and correcting bottlenecks, problems, and isolating problems to modem, ISP, backbone, or server. Can monitor connections' number of hops, download rates, server vs. network effects, maintain metrics over time such as server availability, connection rates, bytes transferred, etc. The Net.Medic Pro version is geared to webmasters and ISP's, and produces more detail and can perform continuous monitoring. Evaluation version available. For Win95/98/NT.

●Castalia IP Socket Tester - Castalia IP Socket Tester; freeware available via Download.com's site in the Internet->Tools and Utilities section; can test port-specific socket-based IP comms; logging capabilities. For Win95/NT.

●Listing of Web log analysis tools.

Return to top of web tools listing

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Send any comments/suggestions/ideas for the Web Test Tools list to: Rick Hower

---

Software QA/Test Resource Center

Software QA and Testing Resource Center Bookstore
---In Association with Amazon.com---

IN ASSOCIATION WITH
BOOKS, MUSIC & MORE
amazon.com

1. Books are listed below in **10 categories**.
2. **Click on a title** for **more info** about a book (price, availability, reviews, ordering and shipping info, etc.) or to **order** the book from Amazon.com
3. Titles were chosen for **usefulness to working QA or Test Engineers.**

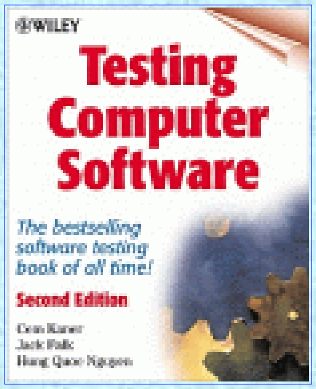
CATEGORIES:

(Note: See bottom of this page to search for any books not listed here.)

Software Testing

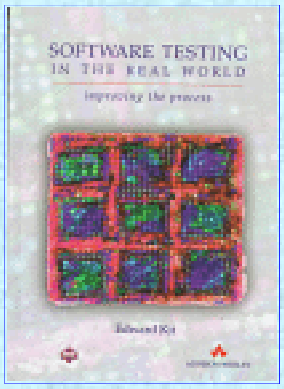
(click on a title for more information or to order)

Top 5 Recommendations:


WILEY
Testing Computer Software
The bestselling software testing book of all time!
Second Edition
Cem Kaner
Jack Falk
Hung Quoc Nguyen

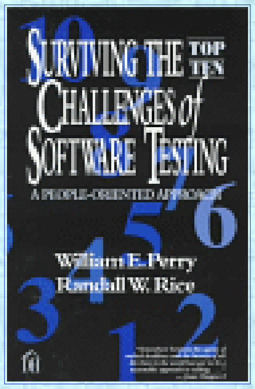
Testing Computer Software, by C. Kaner, et al (1999)

This book has been a standard reference for software testers since it's first edition was published in 1988 and second edition in 1993. Chapters include "The Objectives and Limits of Testing", "Test Case Design", "Localization Testing", "Testing User Manuals", "Managing a Testing Group", and more. The authors are all experienced in software testing and project management, and the book discusses many of the practical and 'human' aspects of software testing. (Note: The 1999 edition is the same as the 1993 edition)



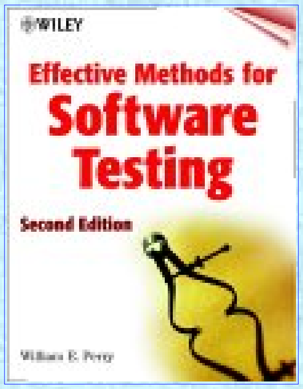
Software Testing in the Real World, by E. Kit (1995)

Excellent introductory book on software testing. Includes chapters such as "The Six Essentials of Software Testing", "Critical Choices: What, When and How to Test", "Testing Tasks, Deliverables, and Chronology", "Software Testing Tools", and more. Useful appendices covering standards, checklists, and more. The writing avoids the heavy, dense feel of many other technical books, and at 252 pages is practical and readable.


SURVIVING THE TOP TEN CHALLENGES of SOFTWARE TESTING
A PEOPLE-ORIENTED APPROACH
William E. Perry
Randall W. Rice

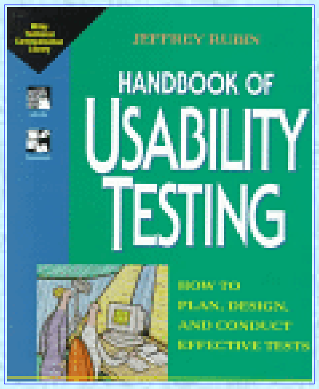
Surviving the Top Ten Challenges of Software Testing: A People-Oriented Approach, by W. Perry, et al (1997)

Addresses many of the non-technical problems that software testing and QA engineers have to contend with; the emphasis is on communication issues. Chapters include "Getting Trained in Testing", "Explaining Testing to Managers", Hitting a Moving Target", "Having to Say No", and more. Author William Perry has authored more than 50 books on software QA and testing, and was previously a member of the Board of Examiners for the Malcolm Baldridge National Quality Awards.



Effective Methods of Software Testing, 2nd Edition, by W. Perry (2000)

An in-depth guide (600 pages) to the basics of software testing methodologies, test planning, techniques, and metrics. Includes worksheets and checklists. Chapters include "Addressing the Software System Business Risk", "Requirements Phase Testing", "Inspecting Test Plans and Test Cases", "Testing Techniques", "Evaluating Test Effectiveness", and more.

 JEFFREY RUBIN HANDBOOK OF USABILITY TESTING HOW TO PLAN, DESIGN, AND CONDUCT EFFECTIVE TESTS	Handbook of Usability Testing: How to Plan Design and Conduct Effective Tests, by J. Rubin (1994) This is an excellent introductory text to one of the less-understood aspect of software testing. It is geared to those without formal training in human factors or usability engineering and without extensive resources for usability testing. Chapters include "Introduction to Usability Testing", "Six Stages of Conducting a Test", "Preparing Test Materials", "Transforming Data into Findings and Recommendations" and more.

Other Books in 'Software Testing' Category:

1. Managing the Testing Process, by Rex Black (1999)
2. Automated Software Testing: Introduction, Management, and Performance by E. Dustin, J. Rashka, J. Paul (1999)
3. Client Server Software Testing on the Desk Top and the Web, by D. Mosley (1999)
4. Handbook of Walkthroughs, Inspections, and Technical Reviews, by D. Freedman and G.Weinberg (1990)
5. The Craft of Software Testing, by B. Marick (1995)
6. Test Process Improvement : A Practical Step-By-Step Guide to Structured Testing, by T. Koomen et al (1999)
7. The Art of Software Testing, by G. Myers (1979)
8. Software Testing Techniques, by B. Beizer (1990)
9. The Complete Guide to Software Testing, by W. Hetzel (1993)
10. Software Testing: A Craftsman's Approach, by P. Jorgensen (1995)
11. Black-Box Testing, by B. Beizer (1995)
12. Testing Client/Server Systems, by K. Bourne(1997)
13. Fatal Defect: Chasing Killer Computer Bugs, by I. Peterson (1996)
14. Testing Safety-Related Software : A Practical Handbook, by S. Gardiner (Editor) (1999)
15. Software Reliability Engineering : More Reliable Software, Faster Development and Testing, by J. Musa (1998)
16. Object Oriented Software Testing: A Hierarchical Approach, by S. Siegel and R. Muller (1996)
17. Analysis and Testing of Distributed Software Applications, by H. Krawczyk(1998)
18. The Art of Testing Network Systems, by R. Buchanan (1996)
19. Software Verification and Validation: A Practitioner's Guide, by S. Rakitin (1997)
20. System Validation and Verification, by J. Grady (1997)
21. Software Testing, by M. Roper(1994)

22. Automating Specification-Based Software Testing, by R. Poston (1996)

Return to top of Book List

Software Quality Assurance

(click on a title for more information or to order)

1. Handbook of Software Quality Assurance, by G. Schulmeyer, et al (

Software QA/Test Resource Center

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Site Index -- key words and key concepts

change management
changing requirements
cleanroom software engineering
client-server testing
CM
CM books
CM FAQ
CM tools
CMM
CMM ratings results 1992-1996
COAST
code quality
code analyzer tools
code comments
code readability
coding standards
coding style
common software project problems
communication problems
comp.software.testing
comp.software.testing FAQ
comparison testing
compatability testing
complexity metrics
computer technical background basics books
configuration management
configuration management
configuration management
configuration management books
configuration management tools
coverage analysis tools
CQE
CSQE
CSQE
CSTE
customer
customer expectations

--D--
deja news
dependencies

design
detection
documentation
documentation
documentation change management

--E--
elder brother
eldest brother
end-to-end testing
error handling
exception handling
e-zines

--F--
FAQ, Software QA/Test
FAQ, comp.software.testing
failures, due to software bugs
fast-growing organizations
firewalls
FIRST
function size
functional design
functional testing

--G--
good code

--H--
hidden functionality
HTML validators
HTML validators

--I--
IEEE
IEEE
IEEE Standards
implementing QA
incentive
incremental integration testing
independent verification and validation
informal evaluation
inspection

software project management books
software QA
software QA books
software QA engineer
software QA manager
software quality
software quality assurance
software quality assurance books
software requirements engineering books
software risk management books
software test manager
software testing
software testing
software testing books
spaghetti code
specifications
SPICE
SPR (Software Problem Report)
SSQ
standards, coding
standards, IEEE
standards, ISO
stress testing

--T--
table of contents
teamwork
test case
test engineer
test manager
test plan
test planning
test planning
test team management
test tools
test tools
test tools (web testing)
testable requirements
testable requirements
testing
testing

testing completion
testing small projects
test-to-break attitude
TickIT
Trillium

--U--
unit testing
usability testing
usability testing

--V--
validation
validators
validators
verification
version control
version control

--W--
walkthrough
web functional test tools
web load test tools
web management tools
web optimization tools
web performance testing
web publishing control tools
web regression test tools
web revision control tools
web scalability
web security FAQ
web security issues
web security test tools
web site management tools
web site management tools
web site monitoring tools
web site QA
web site QA
web site testing
web site testing
web site testing
web site usability

|Home/TOC |FAQ 1 |FAQ 2 |Other Resources |Tools |Web Tools |Bookstore |Index |About |

Send any comments/suggestions/ideas to: Rick Hower

Software QA/Test Resource Center

About the Software QA/Test Resource Center and the author:

This web site is oriented to:

- experienced software QA and testing engineers who need some quick information from the convenience of their desktop
- those new to testing and software QA
- anyone who has an interest in software develoment processes
- those hired as either testers or QA personnel, but who end up performing both testing and QA roles (a common situation)

The author has no particular justification for his authorship of this web site other than:

- real-world software testing, QA, and development experience, primarily in medium-sized (10-100 technical personnel) and large (100-1000 technical personnel) software development organizations
- contract and consulting experience with a wide variety of companies
- a willingness to do the work of developing the site
- a willingness to express his (admittedly biased) ideas
- a need to collect a variety of useful QA and testing information in one organized and convenient place, since his collection of yellow sticky notes has gotten out of hand
- an interest in having an FAQ with a related but different focus than the comp.software.testing FAQ (e.g., more on QA)
- a willingness to consider additions/corrections/suggestions

Date of first web publication of this site: 11/22/96.

This site has not been tested on all versions of all browsers - if you have any problems viewing it, send info re the problem, platform, browser, and browser version to the e-mail link at bottom of this page.

Send any comments/suggestions/ideas to: Rick Hower