IBM Software
Thought Leadership White Paper

May 2011

The evolution of quality management: security, privacy and accessibility

Brian Moran, Rational Application Security offering manager, IBM Software Group;
Susann Ulrich, Rational North America security practice leader, IBM Sales and Distribution

IBM

Introduction

Forward-thinking development organizations recognize that quality management does not begin and end with quality assurance (QA) testing. Requirements-driven quality has a proven return on investment (ROI) with reduced cycle times and reduced development costs. For the enterprises that view quality management as more than just functional and load testing, the next evolution is to extend this approach to security, privacy and accessibility. Quite simply, the evolution of quality management is to make your applications *secure by design*.

IBM® Secure by Design™ is the IBM philosophy that security and privacy must be fully considered and prioritized throughout the development life cycle of your applications, systems, networks and business processes. When applied to application development, Secure by Design draws upon the principles of requirements-driven quality that IBM has delivered to our clients for years. By applying requirements-driven quality to security, privacy and accessibility, enterprises can better manage—and mitigate—the business risk tied to their applications.

To implement this evolution of quality management, the first step is to create a collaborative environment that allows development, quality assurance, security and other stakeholders to agree on requirements and objectives. While collaboration between diverse teams and changes to ingrained practices cannot happen overnight, or simply with good intentions, enterprises can expand quality management to include security, privacy and compliance through the adoption of the culture, processes and best practices that measurably reduce enterprise risk.

This white paper aims to:

1. Identify the challenges and risks of today's disconnect between security and quality.
2. Explain the principles of requirements-driven quality.
3. Introduce the best practices of how security, privacy and accessibility can transform quality management with measurable results.

Application risk and compliance demands

As applications drive more business-critical processes, these applications introduce increasing security and privacy risks—and compliance demands that are designed to ensure proper security is in place.

For example, it's now commonplace for hospitals to manage their admission process for scheduled procedures with web-based applications. Rather than arriving at the hospital an hour early to complete paperwork, patients can go online before the procedure to complete the necessary forms. The web application enables the hospital to streamline its admission process with direct inputs into the various hospital systems for billing and patient records—in addition, the patient experience improves by reducing the time the patient spends in the waiting room.

However, these benefits don't come without business risk. The business process and data collected by the web application includes detailed medical history, insurance data and payment information, such as credit card numbers or check routing information. A profit-driven hacker might consider this the mother load of Personally Identifiable Information (PII).

Imagine the consequences of an attack that compromises this hospital admission application and exposes the most sensitive personal and financial information about its patients. Potential consequences include:

- Law suits
- Regulatory fines
- Identity protection services for every patient
- Lost business from patients who no longer trust the hospital

With so much at stake, security and privacy must be a requirement and a critical element of application requirements, design and quality. According to the web hacking incident database (WHID) maintained by the Web Application Security

Consortium,[1] there have been relatively few reports of healthcare web application attacks (seven attacks) as compared with web application attacks against governments (89), Web 2.0 attacks (67), attacks on retailers (54) and financial services (46).

The high volume of total successful attacks against web applications (660 attacks since 2000) can be traced to two critical factors that create the opportunity for the attacks to take place:

1. The disconnect between security and development organizations that pushes security and privacy to the back burner to be addressed by specialized testing teams at the end of the development process.
2. The profit-driven motivation for hackers to continually work to find new security holes and attacks that allow them to steal information and compromise enterprise applications and systems.

Vulnerabilities and data loss

Two of the most critical variables in measuring the business risk of using web applications are the potential for attack and the potential results of an actual attack. The first of these variables—potential for attack—is best measured by the presence of security vulnerabilities that would allow a hacker to exploit unintended access to the application or the underlying database.

Unfortunately, web applications are subject to a growing number of critical vulnerabilities. According to the IBM X-Force 2010 Trend and Risk Report, 49 percent of the more than 8,000 security vulnerabilities disclosed in 2010 were related to commercial web applications and their plug-ins.[2]

The rise in web application vulnerabilities creates the opportunity for hackers to exploit unpatched vulnerabilities. Unfortunately, many enterprises have seen the effects of these attacks and can measure the damage in data loss. According to the *2010 Verizon Data Breach Investigations Report*, 92 percent of compromised records were lost through attacks against web applications.[3]

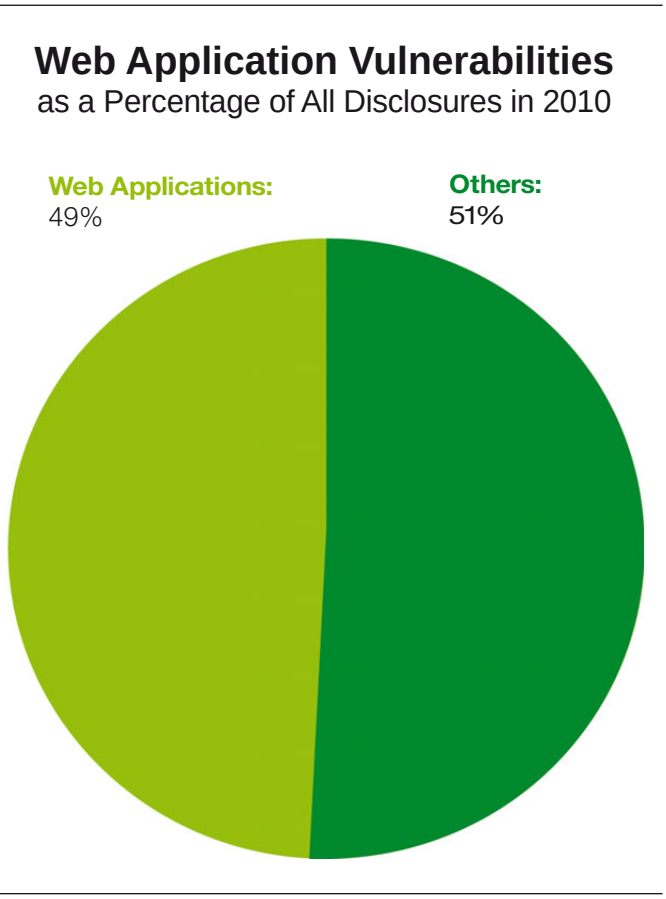

Figure 1: Web application vulnerabilities as a percentage of all disclosures in 2010
Source: IBM X-FORCE™

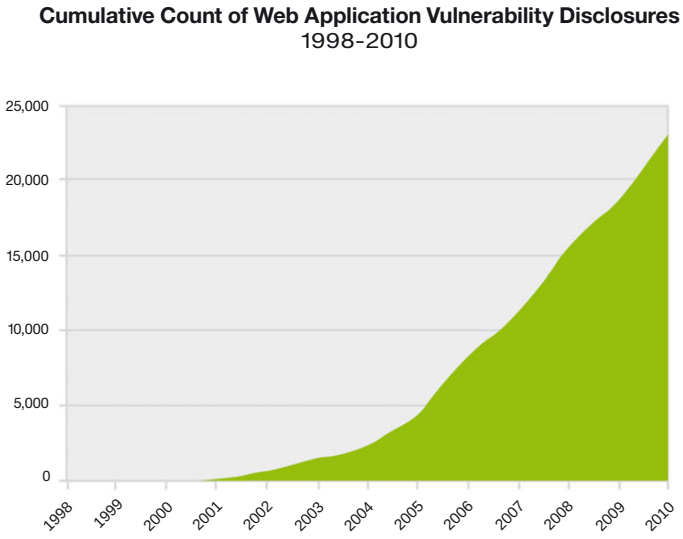

Figure 2: Cumulative count of web application vulnerability disclosures from 1998 to 2010
Source: IBM X-FORCE

Privacy and accessibility

While the greater part of this white paper focuses on the increasing risks of application attacks and security threats, applications also face privacy and accessibility demands that introduce their own specific business risks and compliance demands. Privacy efforts often run parallel to security, but with stakeholders who tend to be much less technically knowledgeable. The three key challenges to privacy as it relates to web applications are to:

1. Identify and document where in your application you collect data from users.
2. Catalog the types of data your application collects.
3. Ensure that this data is not exposed.

Accessibility of a web application refers to an application's ability to work with assistive technology, such as screen readers, brail readers and other tools that can help people with disabilities to access the application or content of a website. Accessibility of web applications is a requirement for enterprises that do business with the US government under Section 508 amendment to the Rehabilitation Act of 1973. Enterprises that do not do business with the US government are affected by accessibility requirements that are needed to serve their diverse customers. Enterprises that *do not* make their applications accessible to people with disabilities can be subject to lawsuits that allege the enterprise doesn't grant equal access to retail facilities and other private businesses.

Privacy, accessibility and security stakeholders may come from three disparate groups within an enterprise, but their inclusion in quality management can follow the same path to ensure these demands are considered throughout the development process.

Regulatory demands

Enterprises can—and should—tackle application security, privacy and accessibility to reduce their business risk. However, regulatory compliance is a common motivator driving enterprises to address these challenges.

If your organization is legally bound—and most are—to protect the privacy and security of PII and hackers gain access to this sensitive information, you can run the risk of being noncompliant with a host of mandated legislation and requirements, including the Payment Card Industry Data Security Standard (PCI DSS), the Children's Online Privacy Protection Act (COPPA), the Health Insurance Portability and Accountability Act (HIPAA), the Gramm-Leach-Bliley Act (GLBA) and the Sarbanes-Oxley Act. The PCI DSS, for example, was designed to protect cardholder information by maintaining secure electronic commerce. More recent updates to the PCI standard include additional requirements for merchants, mandating that they protect web-facing applications against known attacks or face noncompliance.

The disconnect between development and security

Too often, enterprises don't fully consider the security, privacy or accessibility of an application until the final stages of the software development life cycle. In many cases, security or compliance teams conduct preproduction tests just before launch and identify security vulnerabilities or compliance issues that introduce significant risk. At this point in the development process, code revisions are extremely expensive and threaten to derail project schedules.

When enterprises don't consider security requirements until late in the development process, they face two options:

1. Add development cycles that may delay the launch and increase project costs.
2. Accept the risk of data loss from targeted attacks, application downtime or compliance penalties by launching the application with the security vulnerabilities and compliance issues.

When the results of security testing present these two unattractive options, adversarial relationships often arise between security and development organizations. Development teams are measured on their ability to deliver applications on time and on budget, so they may be more inclined to accept the business risk. However, headlines of SQL injection attacks, compliance fines and data loss demonstrate that these risks can be much higher than originally anticipated.

Security teams dedicated to application testing are often composed of just a few headcount. They devote their time to finding security vulnerabilities and lack the resources to actually help mitigate the problems they uncover. However, the security team is not helping to mitigate the risk of discovered vulnerabilities by simply presenting development with a list of security defects.

Without proper prioritization, detailed explanations and recommended corrections, this list of security vulnerabilities becomes just another point of input for development managers to consider for last-minute code revisions or—more likely—potential inclusion in future application updates. By simply identifying the vulnerabilities and passing them on to another team to deal with, the security team may feel like they've done their job, without having achieved any measurable results that decrease the actual business risk.

This disconnect between development and security teams leads to the proliferation of vulnerabilities and increases the risk of data loss, application downtime and regulatory fines.

Security vulnerabilities are defects, too

Development teams value the input and testing of traditional QA, because this testing has been a long-standing phase of the development process, with proven ability to identify defects that detract from the user experience. For many organizations, security vulnerabilities don't receive the same consideration as defects identified in traditional functional testing. But security vulnerabilities are defects, too, because they often include unintended functions that don't meet the intended design of the application.

However, many QA teams struggle to execute security testing, because security testing often requires testers to approach the application from a new point of view. Functional testing evaluates an application to ensure it meets the defined functional requirements. Security testing approaches the application from the point of view of identifying the unintended functionality and bugs that allow malicious activity.

For example, if a search field in an application does not have input validation, an attacker could insert malicious script to capture a cookie. This subverts the intended functionality of the search field. While testing for unintended functions represents a sharp contrast to traditional QA testing, this example demonstrates that security vulnerabilities are directly linked to code quality and shows why security should become part of greater quality management.

Top reasons security testing does not reach full potential

Lack of collaboration	Although development and QA teams are used to interacting and working together to identify and fix functional defects that are identified by the QA analysts, security testing is often treated as something that is done solely by the security team. Many times a confrontational atmosphere can result when a list of vulnerabilities that need to be remediated is presented to the development team late in the schedule.
Lack of security education	Security can mean different things to members of the development and QA teams, because security has not been a traditional part of the education process for these individuals. A security team that simply hands over security results or tests, or both, to developers or QA teams—without proper explanation—can cause friction and greater time delays.
Security team testing alone is not scalable	Relying solely on the security team to provide security testing is not scalable for today's organizations that have many complex, interrelated applications.
Security is not part of the software development life cycle (SDLC) process	Most organizations have evolved a structured approach to developing applications (that is, iterative, agile, waterfall). However, security is typically not formalized in these processes and is an activity completed at the end of the process right before the application is deployed.
Lack of management support	Development and QA managers are measured on their ability to deliver quality software on time and on budget. These time pressures often cause security to be an activity delayed to the end of the schedule.

Requirements-driven quality

For over 20 years, IBM Rational® has advocated and enabled software quality management that drives business transformation while balancing business risk. Quite simply, this can only happen when enterprises prioritize quality in each phase of their development process—beginning with requirements. Requirements-driven quality is in direct contrast to the traditional view of quality assurance, which holds that quality can only be improved by adding more quality testing.

Enterprises that attempt to improve quality by simply conducting more and more testing are on an economically unsustainable path. At some point, resources devoted to application development and value creation must take priority over the resources that focus on mitigating risk. Quality testing will always remain a core step in the software life cycle, but the evolution of quality management demands that quality become a priority at every step in the process.

Many enterprises are well on their way to adopting requirements-driven quality. The next step in this quality management evolution is to include security, privacy and accessibility as elements of quality. By taking this next step, enterprises put the Secure by Design principles into practice.

Quality at the center of the iron triangle

Software development projects are often defined—and constrained—by the "iron triangle" of scope, cost and time.[4] The application's intended purpose and functional requirements should be designed to drive business transformation with measurable business benefits. Cost and time must be carefully divided between creating value with application functionality and minimizing business risk that come from poor quality and security vulnerabilities.

Quality resides in the middle of this iron triangle and, under traditional approaches, only improves with increased resources of time and cost. This is precisely why enterprises must adopt new approaches to quality management that view quality as more

than just testing for functionality and scalability. By simply adding security testing to the QA testing, the iron triangle often dictates that additional time and cost be required for the project. However, to remain competitive, businesses need to find ways to improve quality while shortening time to market for business-critical software.

While development teams must respect the constraints of the iron triangle, quality management can enable business transformation with requirements and application design that address quality at the beginning of the development process. For example, if requirements demand input validation, developers can write code that meets this requirement. By writing secure code, security tests won't find dozens of cross-site scripting and SQL injection vulnerabilities. Security testing is still required to ensure the application meets the requirement, but project time and cost can actually decrease, because resources don't have to spend dozens of hours managing defects, rewriting code and retesting.

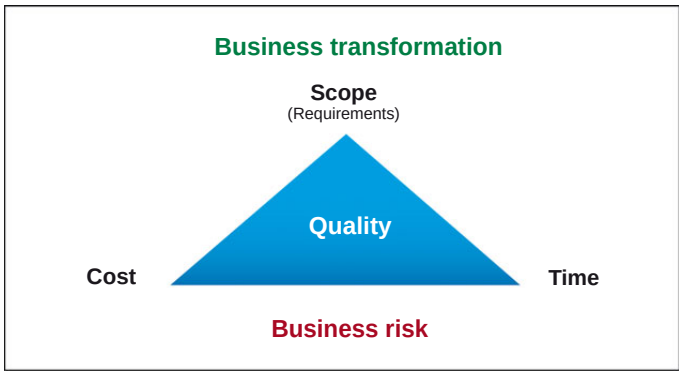

Figure 3: The "iron triangle"—managing the balance between transformation and risk

Applying best practices of quality management to security, privacy

Enterprises that view quality management as more than just functional and load testing have already established best practices and processes for integrating quality throughout their development organization. The next step in this quality management evolution is to include security, privacy and accessibility in these practices and processes. These best practices can be grouped into a few core areas, including:

- Collaboration in requirements and throughout the SDLC
- Reuse and standards
- Risk and prioritization
- Education and empowerment of developers and QA
- Dynamic test plan maintenance
- Testing automation

Collaboration in requirements and throughout the SDLC

Today more than ever, quality is a team sport that unites an organization with common objectives of a stable, fast and secure experience. To include security, privacy and accessibility in quality management, the first step is to engage security, privacy and compliance stakeholders throughout the development process—starting with requirements and design.

For example, security teams should develop threat models for the planned application and validate these threat models with system architects. Security teams can then contribute requirements and design specifications based on the actual business risk identified in the threat models.

And collaboration should be extended throughout the software development life cycle. Each of the subsequent best practices, from the creation of software standards to risk prioritization and automated testing, requires diverse stakeholders to share information, educate the other groups and form a consensus on key decisions that effectively reduces business risk.

Reuse and standards

Project costs and timelines can grow uncontrollably if security requirements and test cases must be reinvented for every project. While every project will face some unique challenges, organizations such as Open Web Application Security Project (OWASP) and the SANS Institute publish lists of the top threats that span all web applications. To build a common set of requirements that can be applied to multiple applications, enterprises can refer to these industry standards and the accepted practices of mitigating risk for specific threats to their applications.

With these common requirements, security teams can then set about creating reusable test cases and test scripts—both of which are essential to enabling nonexperts to execute security testing. Forcing QA testers to become security experts can be expensive and extremely difficult. However, security teams can train the QA team to execute basic security test scripts that scale among multiple applications and cover the common security requirements.

Reuse should not just apply to requirements and testing. Once enterprises spend the resources to write and validate secure code, the next step is to create standards and secure code libraries that can be applied to multiple applications. Secure code libraries should be based on secure code principles and include common components that don't have to be rewritten every time, such as:

- Code snippets
- APIs
- Plug-ins

Creating reusable components to validate data and to interface with non-IBM components will allow code to be quickly updated. For example, reusable components may include modules for data sanitization when outputting data to the user interface or when interacting with the file system or database. Enterprises that use secure code libraries benefit from allowing their developers to focus on creating the application's core functions in less time with fewer security vulnerabilities.

Reuse of requirements, test cases and code libraries is critical to managing the scope, cost and time. However, enterprises must be aware of evolving threats that may force them to evolve those secure code standards. Profit-driven hackers are always looking for new vulnerabilities to exploit and to discover new ways to attack old applications. For this reason, enterprises should maintain a catalog of their security requirements, code libraries and the applications that use these standards. New vulnerabilities can pose threats to these security standards, so it is critical that enterprises know which applications are vulnerable and need to be patched.

Risk and prioritization

Although reuse of requirements and test cases breed consistency and common understanding of key objectives, applications—and their potential threats—face varying levels of risk, based on how and where the application is deployed and who are the potential users.

Not all vulnerabilities and bugs pose the same level of risk. Security, privacy and compliance stakeholders should assign a risk rating for each requirement and test case, specific to the application. For example, user authentication should receive a higher priority and risk rating for online banking applications than for internal applications. Internal applications should prioritize test cases that focus on insiders abusing their privileges in the application.

Risk and prioritization start with requirements and test cases, but should also be extended to defect tracking and resolution. Once vulnerabilities are identified in the application build phase of the development life cycle, security and development organizations should collaborate to:

- Review the complete list of vulnerabilities.
- Group vulnerabilities into common categories of threats and application components.
- Map vulnerability categories to the preassigned risk rating from threat models.
- Assign a priority to each category.
- Agree on the recommended remediation.

Vulnerabilities should then be managed like other software bugs in defect tracking systems that assign ownership, communicate the recommended fix and track the status of the vulnerability through correction and retest.

Some enterprises have gone so far as to create "security heat maps" by cross-referencing identified application vulnerabilities with application portfolio governance. Heat maps help security and development teams agree on the key issues that threaten the business, by visualizing application security vulnerabilities in the context of enterprise risk. Security teams get beyond "possible threats" and provide business managers with quantifiable facts that drive better decisions—and more secure applications.

Heat maps are just one way to execute overall risk management practices in your application development process. By tracking all security vulnerabilities and their risk ratings over time, enterprises now have the necessary metrics to monitor key indicators, such as:

- Total number of high-risk, medium-risk and low-risk vulnerabilities over time and between projects.
- Frequency of recurring vulnerabilities.
- Time between vulnerability identification and remediation.
- Time dedicated to retests.

Education and empowerment of developers and QA

Further research is needed to examine the cultural shift and best practices that influence developers who write secure code. This paper touches upon a few areas that enterprises should consider. The first step in introducing the security requirement to developers is education about why security is important and why secure code libraries should be used. To reinforce the education, development processes should execute code reviews to ensure the application follows security standards, leverages code libraries and meets security design requirements, such as including input validation.

But developers need more than just education and oversight. Development tools are available to empower developers to test their code for security vulnerabilities from within their integrated development environment (IDE). Enterprises can reduce their overall project costs and development time by identifying vulnerabilities in the coding phase of the development life cycle. Defects and vulnerabilities identified and corrected by developers in the coding phase cost just USD80 per defect as compared to USD960 per defect in the QA or testing phase and USD7,600 after release.

Secure code practices have the potential for the greatest ROI, but this represents a huge cultural shift for many development organizations. To implement secure code practices, enterprises should consult additional resources.

Security training for QA teams may be a stretch for enterprises that approach quality testing with an army of button pushers. However, QA teams that leverage automated functional testing tools have an opportunity to expand their skill and value. Security stakeholders should design basic education for QA that focuses on common security threats, such as the OWASP Top 10. With a better understanding of these threats, QA teams can then learn to execute basic security tests with automated security testing solutions.

Dynamic test plan maintenance

Just as requirements frequently change throughout a project, test plans should not be a static document or a hierarchical web page. It's a live, dynamic, always-up-to-date asset that allows for the definition and sharing of test cases and strategies, business-level reporting against quality objectives and—just as important—support for collaborative team activities, such as reviews and approvals.[5]

The dynamic test plan becomes another essential point of collaboration. As functional requirements change, security, privacy and accessibility stakeholders should analyze the impact of those new requirements. While QA still owns the test plan, security teams should provide input and test cases that are based on unintended functions or "what the application should not do."

Testing automation

This paper makes a clear case for why quality management must be much more than testing. However, testing will always remain a core element in the process, especially as quality management extends to security, privacy and accessibility. The key is to automate as much testing as possible in order to free up resources to tackle the more challenging aspect of testing that requires manual effort.

When applied appropriately, automated testing can reduce the costs and time of projects, by helping individuals throughout organizations—development, QA, security—do their jobs better and more efficiently. For developers, source code analysis can be executed in IDEs. This method of security testing is called Static Application Security Testing (SAST) or "white box" testing, because it analyzes the internal structure of the application.

To test the earliest iterations of compiled applications, build-phase managers should combine SAST with Dynamic Application Security Testing (DAST). DAST is also referred to as "black box" testing, because it focuses on the functionality of the application and interacts with the compiled application as an attacker would—without access to the source code. DAST is the best option for QA teams because they are more comfortable dealing with the compiled application.

For developers, build-phase managers and QA teams to successfully execute automated security testing, the security team must design the test scripts and test cases that can be executed by personnel who are not experts in security. The security team should be the owner of the automated testing solution, and design these test cases to focus on the handful of high-risk areas that are specific to the security-related design requirements.

With developers, build-phase managers and QA teams executing automated security testing for the 5 - 10 major security issues, security teams are available to conduct the more complex analysis required of domain experts. Automation is not a panacea, but it will enable security teams to do more of the heavy lifting and detailed testing if they're no longer required to spend time on the basic cross-site scripting (XSS) and SQL injection vulnerabilities. Manual testing can never be completely eliminated, but automation can address and resolve a majority of the issues before the application reaches the security team.

Solutions to extend quality management to security, privacy and accessibility

As a key proponent of quality management, IBM offers application life cycle management solutions that enable enterprises to:

- ***Collaborate*** among and between business, development and test teams with dynamic process- and activity-based workflows for test planning and execution.
- ***Automate*** labor-intensive life-cycle processes and catch quality issues early, reducing time to market, cutting costs and mitigating business risk.
- ***Report*** prioritized metrics tailored for individuals and teams, facilitating greater visibility and enabling decision makers to act with confidence.
- ***Deliver*** greater predictability by mapping successful deployment patterns to operational key performance indicators (KPIs).[6]

To drive the evolution of quality management to include security, privacy and accessibility, IBM delivers solutions that automate security, privacy and accessibility testing. By including these testing solutions in the IBM Jazz™ platform, clients benefit from improved governance and risk management. These integrated solutions deliver governance that operationalizes security in the software development life cycle with scalability and centralized control. Enterprises improve their risk management with the ability to measure, monitor and report on the effectiveness of their security and privacy efforts.

IBM solution	Description
IBM Rational Quality Manager	• Web-based centralized test management environment for business, system and IT decision makers and quality professionals • Collaborative and customizable solution for test planning, workflow control, tracking and metrics reporting that is capable of helping to quantify how project decisions and deliverables impact and align with business objectives
IBM Rational AppScan – Enterprise Edition	• Scalable, automated application security testing that integrates into the software development life cycle • Helps enable organizations to facilitate communication and collaboration between information security, development and management
IBM Rational AppScan Source Edition	• Scalable, automated SAST (or “white box”) analysis • Includes packages designed for the specific needs of users in security, development and remediation
IBM Rational Policy Tester	• Helps to identify and prioritize web privacy, accessibility and regulatory compliance risks • Helps minimize compliance risks from web technology consolidations by way of continuous monitoring and reporting
IBM Rational Functional Tester	• Can provide testers with automated testing capabilities for functional testing, regression testing, GUI testing and data-driven testing • Storyboard testing can simplify test visualization and editing, using natural language and rendered screenshots
IBM Rational DOORS®	• Requirements-management application that can help you reduce costs, increase efficiency and improve quality by enabling you to optimize requirements communication, collaboration and verification
IBM Rational Team Concert™	• Helps build better software by enabling and accelerating full agile and formal practice coverage • Helps streamline development processes with integrated source control, work-item and build capabilities

Summary

Business and compliance risks demand that enterprises can no longer approach security and privacy as bolt-on issues at the end of the software development process. IBM encourages enterprises to embrace the principles of Secure by Design and evolve their quality management to include security, privacy and accessibility. This evolution of quality management empowers enterprises to tackle business transformation while managing risk.

For more information

To learn more about security and quality management solutions from IBM, please contact your IBM marketing representative or IBM Business Partner, or visit the following website: **ibm.com**/software/rational/offerings/security

Additionally, financing solutions from IBM Global Financing can enable effective cash management, protection from technology obsolescence, improved total cost of ownership and return on investment. Also, our Global Asset Recovery Services help address environmental concerns with new, more energy-efficient solutions. For more information on IBM Global Financing, visit: **ibm.com**/financing

IBM Corporation
Route 100
Somers, NY 10589
U.S.A.

Produced in the United States of America
May 2011

[1] Web Application Security Consortium. Data as of April 25, 2011. http://www.projects.webappsec.org/w/page/13246995/Web-Hacking-Incident-Database

[2] "2010 Trend and Risk Report." IBM X-Force Threat Reports. **ibm.com**/services/us/iss/xforce/trendreports/

[3] 2010 Data Breach Investigations Report. Verizon. July 2010.

[4] Moshe Cohen and Michael Lundblad. IBM white paper. *Software quality optimization: balancing business transformation and risk.* http://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-ratsqo

[5] Ibid.

[6] Ibid.

Please Recycle

RAW14269-USEN-00