Ensuring High Quality within Banking Initiatives

Ed Gundry
Senior QA Manager

WHITE PAPER

800.822.6758

Introduction

The need for high quality projects and development efforts is highly evident in banking today. Evidence comes not only from the banking industry, but also from other industries that have suffered severe consequences due to lack of stringent quality control and assurance procedures.

Software quality firm SQS asked quality industry professionals to vote on the top software failure incidents of 2014.[1] The highlights of that list included Amazon's embarrassing technical glitch during the holiday shopping season that caused prices of thousands of items to reduce to just one penny, giving eagle-eyed customers a treat. Scores of small family-owned businesses were left lamenting the error and nursing heavy losses, with some warning they could enter the New Year facing closure.

A potentially lethal error occurred in April 2014 when emergency services went dark for more than 11 million people across seven U.S. states. The incident affected 81 call dispatch centers, rendering emergency services inoperable in all of Washington state and parts of North Carolina, South Carolina, Pennsylvania, California, Minnesota and Florida. An urgent study from the Federal Communications Commission found that an entirely preventable software error caused the service to drop.

While not always life-threatening, errors in banking software development can cause severe consequences for banks, their reputations and their customers. This paper presents actions to ensure high quality in complex banking programs and initiatives.

Business Drivers for QA Management

Besides avoiding disasters and fatal errors, quality assurance management delivers benefits for banks, in the form of more effective and efficient processes that bring solutions to the marketplace that meet – and even exceed – customer expectations.

The ultimate business drivers for QA management are:

- A need to enhance customer satisfaction – The bank's products consistently have higher numbers of issues than their competition, resulting in increased customer complaints and lower customer satisfaction ratings.
- Keeping product offerings aligned with evolving customer expectations – Cost and schedule overruns due to redesign, rework and resolving defects make it challenging to bring new products to market efficiently.

[1] Continuity Central Archive, February 2015

- Improving development/maintenance process efficiency – Missing work products, poor quality, cost/time overruns, process inefficiencies, inability to detect defects and inability to cope with changes in technology are continually driving up costs.

If your institution experiences difficulty achieving high quality within its product development, there may be notable signs. Listen for indicators of quality risks and related issues, such as:

- “We have not experienced the productivity improvements expected from our recent initiatives.”
- “Too many defects are passed on for our customers to find.”
- “Our customer satisfaction ratings are down.”

A Model for Success

Using any of the various quality industry standards (such as ISO 9001 – Quality Management System Requirements, or the Software Engineering Institute’s Capability Maturity Model Integration, CMMi) is an excellent way to establish a best practice quality management framework in your organization. These standards require your institution to establish quality controls throughout its business processes. The intent of these quality controls is to ensure your customer requirements are met **before** your products and services are delivered to the customer. Quality controls are implemented at the management level and the Solution Delivery Life Cycle (SDLC) level.

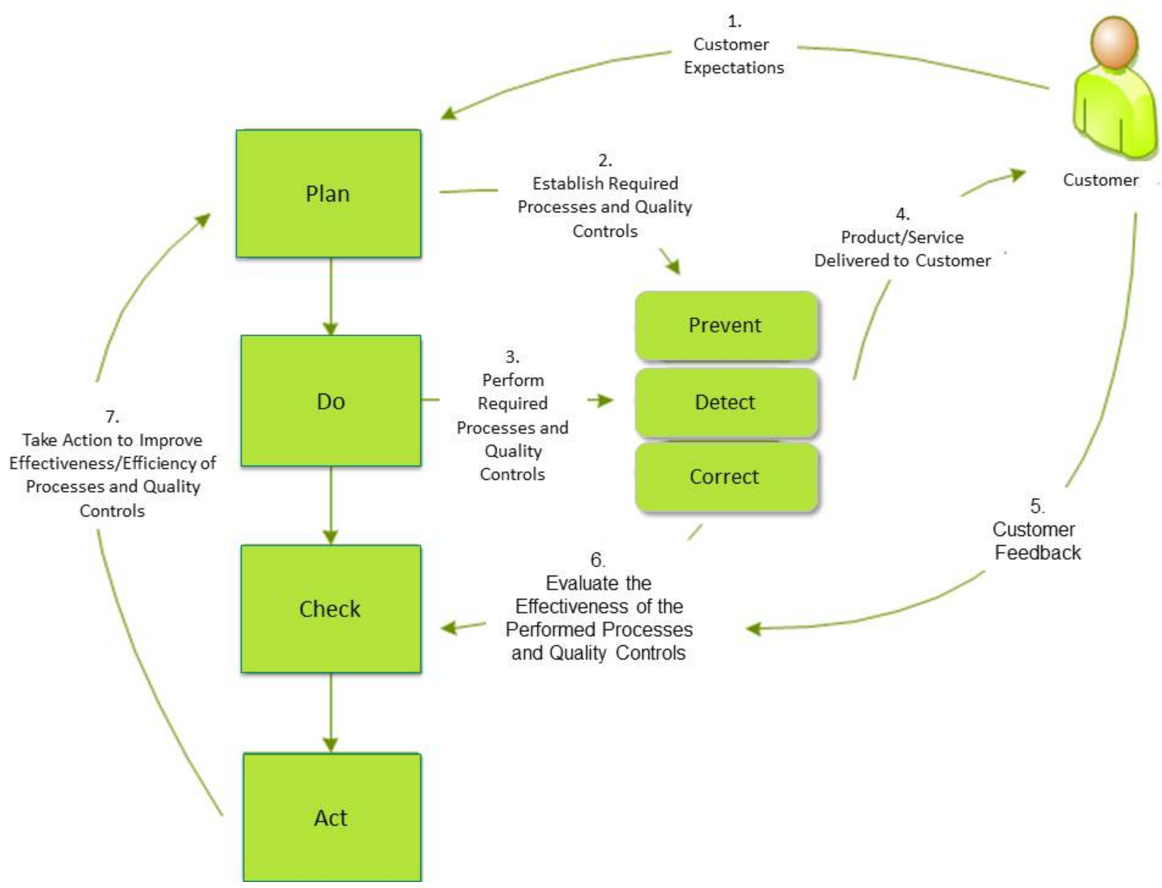
The quality controls at the management level are commonly grouped into four phases: Plan, Do, Check and Act (PDCA):

- Plan
  - The customer’s expectations and requirements are determined.
  - The processes and quality controls needed for the organization to meet the customer’s expectations are identified and documented
- Do
  - The established processes and quality controls are performed – Prevent, Detect and Correct, below – to ensure the customer’s expectations are met.
    - Prevent – The processes and quality controls are implemented to prevent the occurrence of a defect.
    - Detect – The processes and quality controls are implemented to proactively try to identify a defect.
    - Correct – The processes and quality controls are implemented to fix and resolve each identified defect.
  - The product or service is delivered to the customer.

- Check
  - Feedback from the customer is received.
  - The effectiveness of the processes and quality controls are evaluated, answering the question, “Are the customer’s expectations being met?”
- Act

Actions are taken to improve the effectiveness and efficiencies of the documented processes and quality controls to meet the customer’s expectations

The diagram below provides a high-level view of the PDCA cycle:



The remainder of this paper expands on the activities shown in the above diagram.

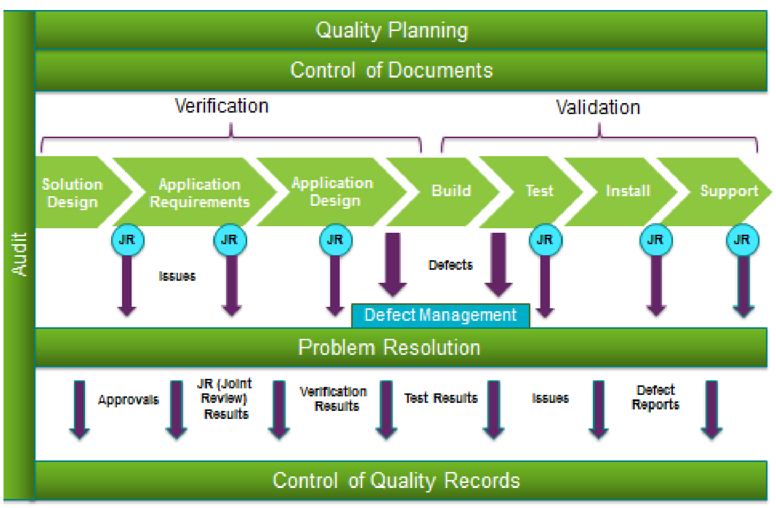
Key Activities for Ensuring Quality

Establish the QA Strategy

In developing new banking products, quality assurance covers the systematic process of checking whether the new solution in development meets all the specified requirements. A comprehensive quality assurance strategy provides the guidance and structure for those related activities. The quality strategy is created during the Plan phase of the PDCA cycle and should be fairly consistent from one project to the next. In the diagram below, a typical approach to a software development effort is depicted in the seven chevrons in the middle of the graphic. Development starts with a solution design working toward installation and then support of the completed new system.

The activities shown here include how joint reviews (JRs) serve as a forum to resolve any outstanding issues before the next development step can proceed. Verification of the application requirements and the application design is critical to establish and document mutual understandings – as is the case with high quality initiatives. The QA strategy outlines the framework for when and how these key activities occur within a program or software development initiative.

Establishing the QA Strategy



Testing activities are an important part of the QA strategy, but testing alone does not ensure high quality final solutions. The testing activities are performed during the Detect section of the Do phase of the PDCA cycle. Testing must be an integral component of the overall plan – resolving and documenting issues is just as important as conducting the actual tests. Within the overall QA strategy it becomes important to note the distinction between a developing a test strategy and executing a test plan.

Differences between a Test Plan and Test Strategy

Bankers often have a hard time distinguishing between strategies and plans, but one can't exist without the other. Plans are concrete, while strategy documents provide vision and governance. In the IT world a test strategy defines the quality assurance activities to be performed during the software life cycle. It describes the responsibilities and authorities for accomplishing software quality assurance activities and identifies the required coordination of those activities with other activities within a project.[2]

A test plan, on the other hand, describes the plans for quality testing of software applications and software systems. It describes the software test environment to be used for the testing, identifies the tests to be performed and provides schedules for test activities.[3]

The test strategy is a governance document that helps stakeholders, project and program management and other team members understand the approach to follow in meeting testing objectives. The value of the test strategy isn't in the wording or the format of the document; it is in the formalizing of the approach assuring the solution and documenting it. It brings all parties to a common understanding of the efforts and responsibilities needed to be planned.

The test plan provides a more detailed understanding of a test strategy. It describes the testing activities and workflows to be implemented in a detailed and systematic manner. The following table provides an overview of some of the key distinguishing points between a test strategy and a test plan.

	Test Strategy	**Test Plan**
Purpose	Governance and ownership assigned Defines overall approach for assuring the solution	Defines scope of testing including what functionality will be tested
Test Approach	Defines test approach for each phase within the strategy: SIT, UAT, performance testing	Covers when specific testing techniques will be employed, e.g., regression testing, data validation, test limits
Timing	Defines entry, exit and acceptance criteria for test phases	Provides testing schedule and calendar to stage specific activities
Deliverables Include	Test communications, reporting and metrics, allocation release plan, defect management and approach for change requests	Reference to test scripts, documents existing testing constraints, defines defect levels and test criteria
Other	Defines testing terminology and testing objectives	Different types of test plans, for integrated testing, application testing, systems testing, etc.

[2] International Organization for Standards, Software Lifecycle Process, August 1995
[3] ibid

Quality Assurance Activities to Implement

Beyond developing a test strategy and test plan, there are other QA activities to implement at the beginning of a program to create a foundation for success for the overall effort. Some of the most important include:

- Developing document control – A program and related supporting projects should generate multiple supporting documents. Thorough quality practices require complete document approval and sound record keeping. Program stakeholders need to know that only one version of the truth exists, and they need to know where to access that critical document. The document control process is established during the Plan phase of the PDCA cycle, and is performed as part of the Prevent section of the Do phase of the phase of the PDCA cycle.
- Ensuring receipt of quality records – Evidence that a quality activity was completed must be logged and uploaded into a readily available document repository. These records include approvals, verification of results and actual test results. The control of quality records process is also established during the Plan phase of the PDCA cycle. It is performed throughout the three sections of the Do phase of the phase of the PDCA cycle.
- Requiring joint reviews to facilitate document approvals – The joint review process is also established during the Plan phase of the PDCA cycle. It is performed during the Detect section of the Do phase of the phase of the PDCA cycle.
- Expecting a QA Summary Report – Before the client or program sponsor begins their phase of solution testing, it should review the findings in a QA summary report. This report details the defects and resolutions its partner uncovered and addressed before transitioning testing activities to the client or sponsor. The QA Summary Report is an input to the Check phase of the PDCA cycle.

Requirements Verification

Verifying the requirements of the new solution is another critical activity within the QA process. Verification is part of the Detect section of the Do phase of the PDCA cycle. The QA team must first identify deliverables to be verified and then define quality standards for each deliverable type. Next, they must review, approve and release verification standards for use in the initiative. Authors of any quality documents should produce deliverables to meet these standards.

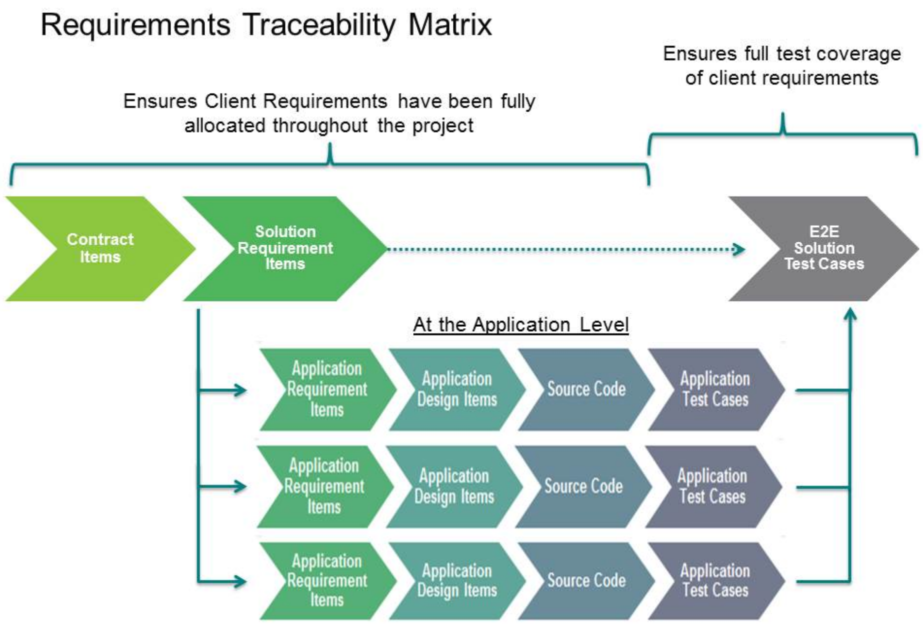
The QA team should log problems when quality standards are not met. Problems achieve resolution when a deliverable meets quality standards. After confirming the deliverable meets quality standards, the team releases it for use by the program. A partial example of the classification and logging of defect codes for one recent FIS managed QA program is shown below.

Requirements Verification

Defect Code	Category	Defect Description	Impact	Example
BR-01	Not Complete	Terms and units of measure are not defined	Multiple interpretations of solution; incorrect design is developed	The acronym CST is used requirement, but it has not been defined
BR-02	Not Traceable	A business requirement is not uniquely identifiable	Inability to trace requirements forward or backwards	Each requirement needs to be numbed. e.g. 3.1.3 Requirement Description…
BR-03	Not Traceable	A business requirement cannot be traced back to a stated business scope item. (i.e., the scope of the solution has 'crept' beyond the business need/objective)	Scope Creep – solution delivers more than stated business need/objective	Refer to a requirements traceability matrix
BR-08	Not Relevant	The requirement is a solution	Design is constrained by the requirement	Requirement is stated as: BeB will call a standard Connectware Deposit Account Balance inquire service to verify the funds in the originating account
BR-11	Not Testable	The requirement cannot be checked for correctness by a person or tool	It is not possible to confirm that the requirement has been met, or conversely, not met.	Example of untestable requirements: •System performance shall be fast. •The system shall be easy to use.

Requirements Traceability Matrix

One technique to ensure full test coverage of a client's requirements is the use of a Requirements Traceability Index. The Requirements Traceability Matrix is part of the Prevent section of the Do phase of the PDCA cycle. At FIS solution architects will create a solution design that captures client requirements. QA experts use the Requirements Traceability Matrix to ensure solution design requirements align with product definition documents created by the various FIS product teams involved in the creation of the client's new solution (product, etc.). This index ensures the allocation of each bank requirement back to the product requirements. This entire matching process helps FIS to achieve a high degree of confidence that all bank requirements receive the optimal testing coverage.



Failure Mode Effects Analysis in Banking Programs

Besides carefully tracing and testing the bank's requirements and maintaining solid QA documentation, other quality techniques can add significant value to banking initiatives. FMEA is one of them. The FMEA is part of the Prevent section of the Do phase of the PDCA cycle.

Failure mode and effects analysis (FMEA) started in the 1940s in the United States as a way to improve manufacturing for the war effort. Although it began 70 years ago, this process remains valuable to the banking industry today. Because financial institutions compete globally against more nimble nontraditional financial services providers, they need to develop products quickly and with the high quality that encourages customer loyalty.

In times of scarce quality assurance (QA) and technical resources, testing priorities must be determined in order to best mitigate project risk. FMEA lays out a clear and straightforward process for performing that critical ranking. The design elements or components that compose a complex solution are analyzed by functional lines of business in order to:

- Identify the potential failures that could impact the customer experience
- Identify the potential causes of each failure

- Assess the resulting impact on customer experience should the failure occur
- Determine the probability that the cause will occur and subsequently cause the failure
- Identify the current controls or the ability to detect the failure or cause

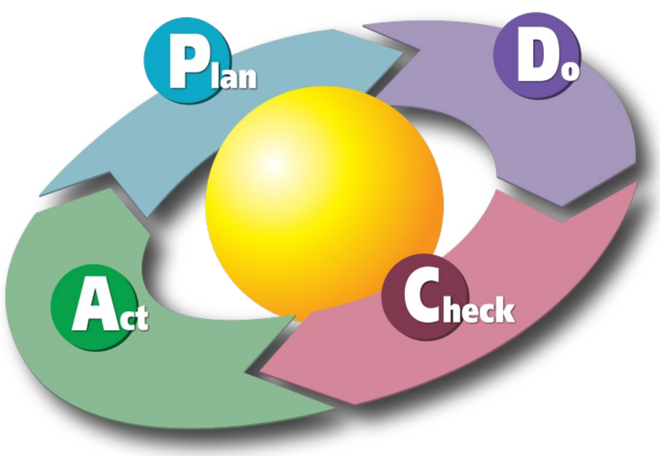
The FMEA focuses QA and testing resources in development efforts. Few banks have the luxury of unlimited time and technical resources. Prioritization in the testing can lead to fewer defects needing correcting at more expensive points in a product's life cycle. Additionally, the FMEA sessions reinforce the understanding of the solution design and functional requirements documentation. The ranking discussions quickly bring any ambiguous requirements and documentation to the surface.

FMEA participants also discuss solutions and ways to mitigate the critical and high severity risks in the project under review. As the importance of a component or business requirement becomes clear, actions to ensure it will not fail are assigned as a part of the FMEA process.

Continual QA Improvement

Over time, the long-term impact of quality assurance on a program needs to be evaluated and assessed in order to evolve and improve (i.e., the Check and Act phases of the PDCA cycle). A robust continuous improvement methodology should collect quality metrics from a wide variety of sources, analyze the data and contribute to the adjustment of QA processes. The methodology should also incorporate best practices from other complex initiatives and provide a forum for open discussion of lessons learned. These improvement activities can occur as milestones are reached in the program and during a post-mortem discussion of the initiative.

The continual improvement methodology powers a repeating cycle of Plan, Do, Check and Act that benefits and evolves from the changes spurred by the methodology, such that each repetition of the PDCA cycle results in better QA than the last.



This collection of best practices needs a place to reside for easy access of all a program's stakeholders. A well-documented Quality Management System provides a framework to inventory and maintain a record of improvements, processes and procedures. This documentation facilitates employee and stakeholder training and assists in solving issues that arise in all complex initiatives and programs.

Collecting High-quality Metrics

Metrics guide the QA process, and it is critical to select those process metrics that drive business results and customer satisfaction. Start with metrics related to the process outputs, such as product quality, total number of customer requirements and total number of defects identified. Analyze any schedule variance to determine the effect on timely milestone achievement. The same applies to any cost variance against the overall program budget. Ask if the program or project meets its cost targets for the reporting period. Expand data collection to process inputs and in-process metrics to assess requirements and design quality.

Summary

Applying sound quality practices increases customer satisfaction, decreases time to market and reduces development and maintenance expense. High quality in banking programs, then, is well worth the effort. Initiatives require proper QA planning, complete documentation and processes to verify and validate requirements to achieve results that evolve with changing customer expectations and improve development and maintenance efficiency. Adopting quality industry standards and using the PDCA cycle to continuously improve the quality assurance process is an important part of a successful, well-defined and documented QA strategy.

FIS Can Help

FIS offers proven activities and techniques, including industry-accepted practices and internal processes, to help ensure banking clients experience high quality within their programs led by FIS consultants.

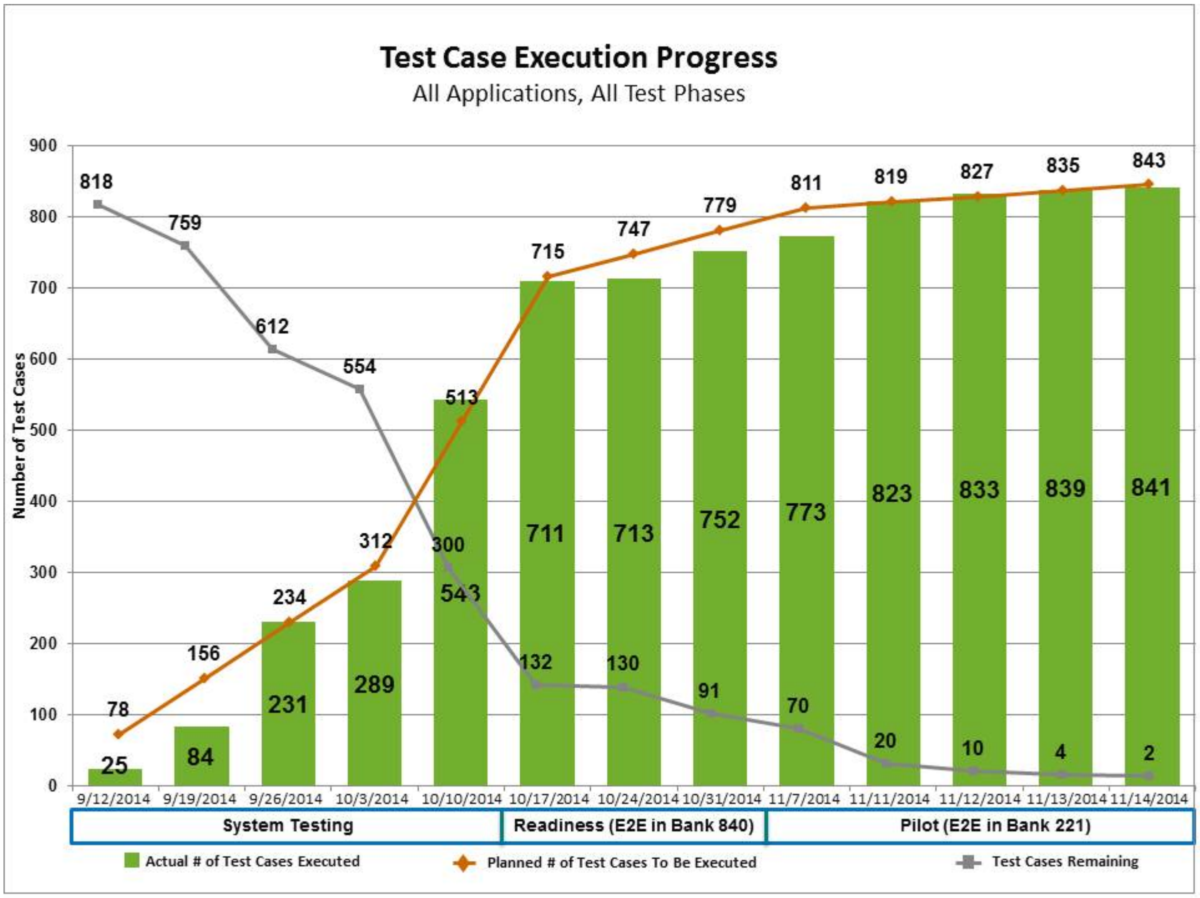
The application of these proven QA management activities improved application quality by over 23% in less than one year for a domestic bank with $15 billion in assets.

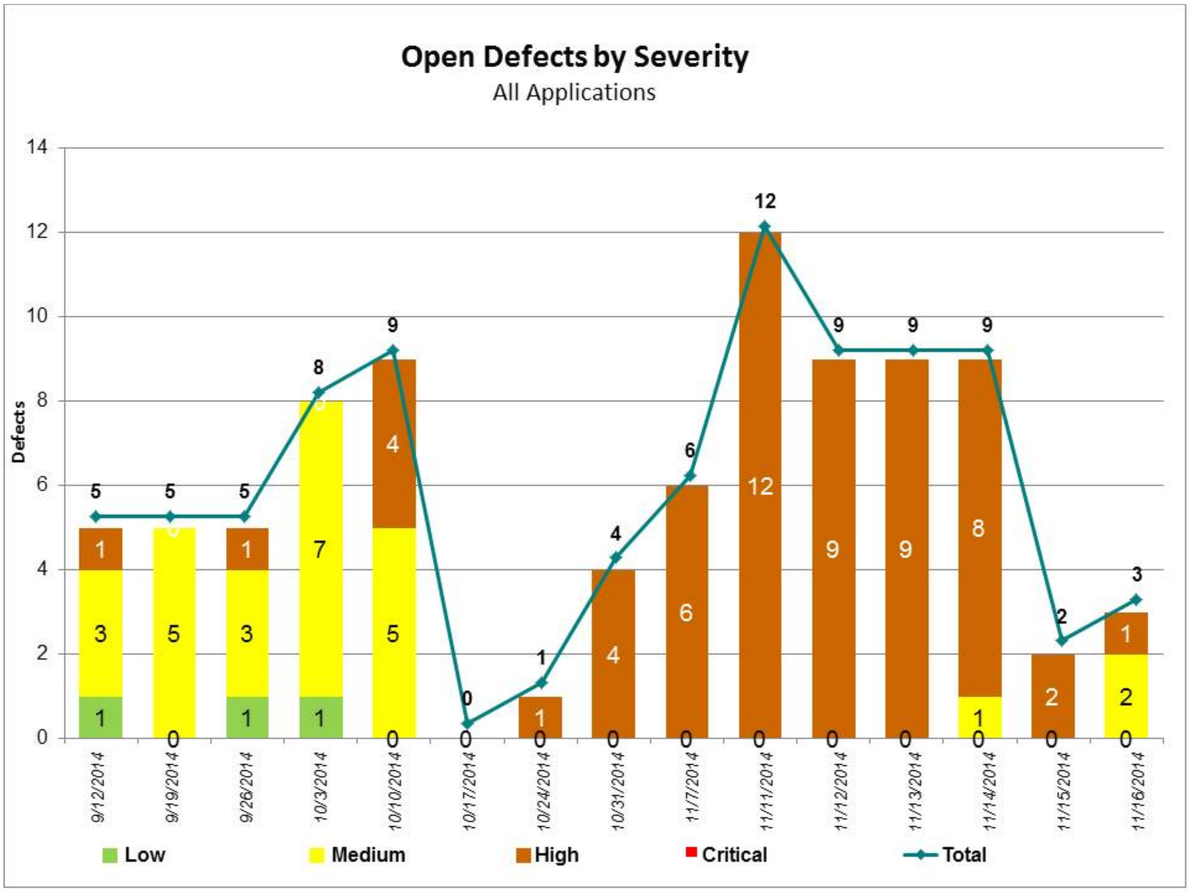
Contact Us

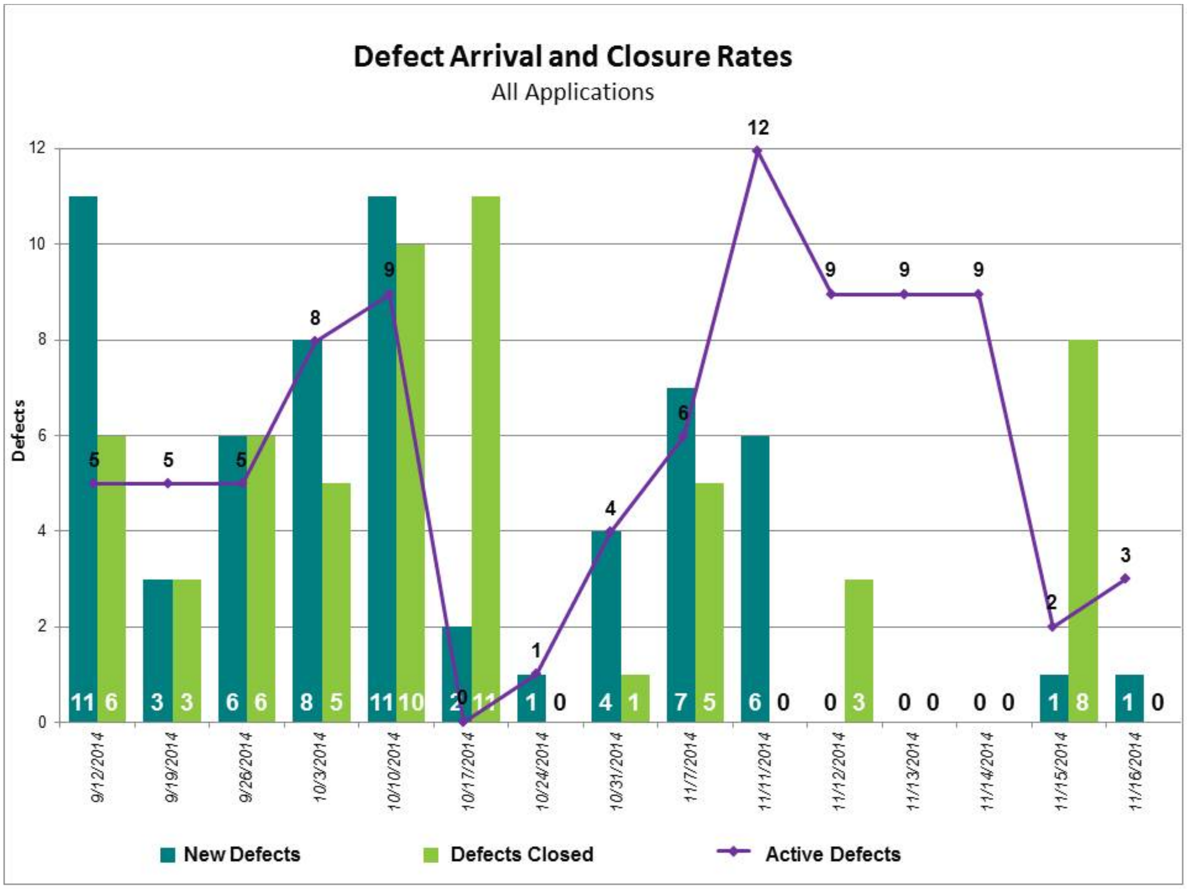
For more information about how FIS can help your institution ensure high quality, contact FIS Consulting Services at 800.822.6758 or visit www.fisglobal.com.

Appendix

The following charts offer examples of the testing progress typically displayed within the Quality Summary report.




Open Defects by Severity
All Applications
Defects
9/12/2014
9/19/2014
9/26/2014
10/3/2014
10/10/2014
10/17/2014
10/24/2014
10/31/2014
11/7/2014
11/11/2014
11/12/2014
11/13/2014
11/14/2014
11/15/2014
11/16/2014
Low
Medium
High
Critical
Total


Defect Arrival and Closure Rates
All Applications
Defects
9/12/2014
9/19/2014
9/26/2014
10/3/2014
10/10/2014
10/17/2014
10/24/2014
10/31/2014
11/7/2014
11/11/2014
11/12/2014
11/13/2014
11/14/2014
11/15/2014
11/16/2014
New Defects
Defects Closed
Active Defects