Title: SYSTEMS AND METHODS OF GENERATING A QUALITY ASSURANCE PROJECT STATUS

Abstract: A method, apparatus and a system for a quality assurance analytic technique and system are disclosed. In one embodiment, a method of a client device includes determining a forecasted completion date of a communication of a quality assurance testing data from a quality center database to a server device comprising a processor to analyze based on an algorithm the quality assurance testing data. In addition, the method includes creating a quality assurance project status of a communication of a baseline complete date to the server device to analyze the baseline complete date and the forecasted completion date.

Figure 1

Continued on next page
LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG). Published: — without international search report and to be republished upon receipt of that report (Rule 48.2(g)).
SYSTEMS AND METHODS OF GENERATING A QUALITY ASSURANCE PROJECT STATUS

CLAIM OF PRIORITY:
[0001] This application claims priority from a US Utility application number 12/582,971 titled "SYSTEMS AND METHODS OF GENERATING A QUALITY ASSURANCE PROJECT STATUS" filed on October 21, 2009.

FIELD OF TECHNOLOGY
[0002] This disclosure relates generally to the technical field of software testing and, in several embodiments, to systems and methods of generating a quality assurance project status and a forecasted completion date.

BACKGROUND
[0001] Quality assurance (QA) may be a planned and systematic production processes that provide confidence in a product's suitability for its intended purpose. It may be a set of activities intended to ensure that products (goods and/or services) satisfy customer requirements in a systematic, reliable fashion. Quality assurance for software programs/applications/products may be referred to as software testing.

[0002] Software testing may be empirical investigation conducted to provide stakeholders with information about the quality of the product or service under test, with respect to the context in which it is intended to operate. The software testing may also provides an objective, independent view of the software to allow the business to appreciate and understand the risks at implementation of the software. Test techniques include, but are not limited to, the process of executing a program or application with the intent of finding software bugs or defects. The software testing may also be stated as the process of validating and verifying that a software program/application/product that meets the business and technical requirements that guided its design and development; works as expected; or may be implemented with the same characteristics.

[0003] Providing status updates of the software testing through manual calculations may utilize a significant amount of work from quality assurance testers. Such manual
calculations are subject to errors and time delays. The manual calculations may not reflect a real-time status of the software testing. The results of the manual calculations may be difficult to interpret by individuals not familiar with quality assurance and software testing.
SUMMARY

[0004] Several methods and a system for a quality assurance analytic technique and system are disclosed. In one aspect, a method of a client device includes determining a forecasted completion date of a communication of a quality assurance testing data from a quality center database to a server device that includes a processor to analyze based on an algorithm the quality assurance testing data. In addition, the method includes creating a quality assurance project status of a communication of a baseline complete date to the server device to analyze the baseline complete date and the forecasted completion date. The method also includes determining the quality assurance project status that includes a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the quality center database and the forecasted completion date of the server device to assist a user to manage a quality assurance testing project.

[0005] In addition, the method may include creating the quality assurance project status of a communication of a baseline start date to the server device to analyze the baseline start date, the baseline complete date, and the forecasted completion date. The quality assurance testing data of the quality center database may be analyzed based on the algorithm of the server device to determine the forecasted completion date that includes a number of total tests, a number of passed tests, a first test date, and/or a last test date.

The method may also include determining a test execution trend that includes a number of passed tests, a number of failed tests, and/or a number of not complete tests based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project. The method may further include determining a group resource execution rate that includes a number of tests executed by an offshore quality assurance team and a number of tests executed by an onshore quality assurance team based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

[0006] The method may include determining an individual resource execution rate that includes a number of tests executed by a quality assurance individual based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project. In addition,
the method may include determining a defect trend that includes a linked defect, an unlinked defect, and a severity of the linked defect based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

[0007] In another aspect, a method of a server device includes generating a forecasted completion date based on an analysis performed by a processor of applying an algorithm of a quality assurance testing data that includes a number of total tests, a number of passed tests, a first test date, and a last test date of a client device. In addition, the method includes producing a quality assurance project status of a comparison of a difference in time between a baseline complete date of the client device and the forecasted completion date. The method also includes analyzing the quality assurance project status that includes a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the client device and the forecasted completion date to assist a user to manage a quality assurance testing project.

[0008] The aforementioned algorithm may include calculating a number of tests left to pass from a multiplication of a project goal percentage and the number of total tests and subtracting the number of passed tests, calculating an average passed rate by dividing the number of total tests by the difference in work days between the first test date and the last test date, calculating a number of work days needed by dividing the number of tests left to pass by the average passed rate; and generating the forecasted completion date by adding the number of work days needed to a current date.

[0009] In addition, the method may include producing a quality assurance project status of a comparison of the difference in time between a baseline start date and the baseline complete date of the client device and the forecasted completion date. The method may also include producing a quality assurance project status of an analysis of a project goal percentage of the client device to analyze a number of tests left to pass and to analyze the number of tests left to pass and the forecasted completion date to generate the quality assurance project status. The method may further include generating an actual versus expected test execution rate from a comparison of an actual execution rate from an analysis of the quality assurance testing data and an expected execution rate of the client device to assist the user to manage the quality assurance testing project.
The method may include regenerating the forecasted completion date from the quality assurance testing data that includes the number of total tests, the number of passed tests, the first test date, and/or the last test date and from a number of quality assurance testers and a number of work hours in a day of the user of the client device to adjust the forecasted completion date to assist the user to manage a quality assurance testing project. In addition, the method may include generating a number of quality assurance testers from the quality assurance testing data including the number of total tests, the number of passed tests, the first test date, and/or the last test date and from the baseline end date and a number of work hours in a day of the user of the client device assist the user to manage the quality assurance testing project.

In yet another aspect, a system includes a client device to communicate a quality assurance testing data from a quality center database and a baseline complete date from the client device to a server device to analyze the quality assurance testing data to calculate a forecasted completion date and a quality assurance project status. In addition, the system includes a server device to generate the forecasted completion date from the quality assurance testing data including a number of total tests, a number of passed tests, a first test date, and/or a last test date from the client device and the quality assurance project status by comparing a difference in time between the baseline complete date and the forecasted completion date. The system also includes a number of displays associated with a number of client devices to render the quality assurance project status to a number of users in a number of analytics dashboards rendered in each of the displays and to render the quality assurance project status including a variance in a time from the baseline complete date from an analysis of the quality assurance testing data from the client device and the forecasted completion date from the server device to the client device to assist a user of the client device to manage a quality assurance testing project.

In addition, the system may include the server device to calculate a number of tests left to pass from a multiplication of a project goal percentage and the number of total tests and subtracting the number of passed tests. In addition, the server device of the system may also calculate an average passed rate by dividing the number of total tests by the difference in work days between the first test date an/or the last test date, calculate a number of work days needed by dividing the number of tests left to pass by the average
passed rate, and generate the forecasted completion date by adding the number of work
days needed to a current date.

[0013] The system may include the server device producing a quality assurance project
status of a comparison of the difference in time between a baseline start date and the
baseline complete date of the client device and the forecasted completion date to assist a
user of the client device to manage a quality assurance testing project. In addition, the
system may include the server device to regenerate the forecasted completion date from
the quality assurance testing data including the number of total tests, the number of
passed tests, the first test date, and/or the last test date and from a number of quality
assurance testers and a number of work hours in a day communicated to the server device
from the user of the client device to adjust the forecasted completion date to assist the
user of the client device to manage a quality assurance testing project.

[0014] The server device to generate a test execution trend may include a number of
passed tests, a number of failed tests, and a number of not complete tests based on a time
from an analysis of the quality assurance testing data from an analysis of the quality
assurance testing data a quality center database to assist the user of the client device to
manage a quality assurance testing project. The system may further include the server
device to generate a actual versus expected test execution rate from a comparison of a
actual execution rate from a calculation of the quality assurance testing data from the
client device communicating with the quality center database and an expected execution
rate from the client device communicating with the user of the client device to assist the
user to manage a quality assurance testing project.

[0015] Other aspects will be apparent from the following description and the appended
claims.
BRIEF DESCRIPTION OF THE VIEWS OF DRAWINGS

[0016] Example embodiments are illustrated by way of example and not limitation in the figures of accompanying drawings, in which like references indicate similar elements and in which:

[0017] Figure 1 is a system view illustrating a process of quality assurance (QA) management, according to one or more embodiments.

[0018] Figure 2 illustrates a system view of variable function analysis service, according to one or more embodiments.

[0019] Figure 3 is a graphical user interface (GUI) illustrating a process and progress of QA, according to an example embodiment.

[0020] Figure 4 is a system view illustrating a process of generating software testing forecast, according to one or more embodiments.

[0021] Figure 5 is a system view illustrating a user using a software testing forecasting tool, according to one or more embodiments.

[0022] Figure 6 is a process view illustrating generation of quality assurance project status, according to one or more embodiments.

[0023] Figure 7A is a process flow of determining a forecast completion date, according to one embodiment.

[0024] Figure 7B is a continuation of process flow of Figure 7A, illustrating additional process, according to one embodiment.

[0025] Figure 8A is a process flow illustrating a generation of a forecasted completion date, according to one embodiment.

[0026] Figure 8B is a continuation of process flow of Figure 8A, illustrating additional process, according to one embodiment.

[0027] Figure 9 shows the configuration changes about to perform.

[0028] Figure 10 shows step by step process and builds upon knowledge gained from each step.

[0029] Figure 11 is a screenshot view of an application called the 'Configuration Editor'.

[0030] Figure 12 is a screenshot view of the project in another step.
[0031] Other features of the present embodiments will be apparent from accompanying Drawings and from the Detailed Description that follows.
DETAILED DESCRIPTION

[0032] Several methods and a system for a quality assurance analytic technique and system are disclosed. It will be appreciated that the various embodiments discussed herein need not necessarily belong to the same group of exemplary embodiments, and may be grouped into various other embodiments not explicitly disclosed herein. In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the various embodiments.

[0033] Figure 1 is a system view illustrating a process of quality assurance management, according to one or more embodiments. In particular, Figure 1 illustrates a software testing forecasting module 100, a quality center database 102, Quality Assurance (QA) testers 104i-N, QA team lead 106, a QA director 108, a Project Management Office (PMO)/project manager 110, an executive committee 120, according to one embodiment.

[0034] In one or more embodiment, the software testing forecasting module 100 may be a data processing system for managing quality assurance process. In one or more embodiments, the software testing forecasting module 100 may be automated or manually controlled. Furthermore, the software testing forecasting module 100 may be administered by the QA director 108. In one or more embodiments, the software testing forecasting module 100 may enable users (e.g., the QA testers 104i-N, the QA team lead 106, the QA director 108, the PMO/project manager 110, the executive committee 120, etc.) to track in real-time the testing progress of Quality Center (QC) projects. In addition, the software testing forecasting module 100 may enable users of the software testing forecasting module 100 to setup several unique views (e.g., graphical), including complete project views as well as sub-project views. In one or more embodiments, the administrator of the QA director 108 may enable the QA testers 104i-N to customize the views based on requirement.

[0035] In one or more embodiments, the software testing forecasting module 100 is designed to provide metrics, test plans, etc. The metrics provided by the software testing forecasting module 100 may enable the users to manage testing resources, test plans and time lines. Furthermore, the software testing forecasting module 100 may support Variable function analysis function that provides test planning analysis data. The
software testing forecasting module 100 may enable the user to plan, allot resource, and complete a testing project within a prescribed time. The quality center database 102 may manage quality assurance testing data. In one or more embodiments, the quality center database 102 may be configured by the QA director. Input to the quality center database 102 may be provided by the users and the database may be updated frequently.

[0036] The QA testers 104i_N may be professionals who process the software developed by developers for reducing faults and errors (e.g., defect rate). The QA team lead 106 may be an experienced QA tester 104i_N who manages a team of QA testers 104i_N for testing the software developed by the developers for reducing the defect rate. In one or more embodiments, the QA team lead 106 may perform QA tests on the software project from the architectural perspective. Also, the QA team lead 106 may be responsible for coordinating the project with the QA testers 104i_N. The QA director 108 may assign software QA projects to the QA team lead 106. The QA director 108 may be responsible for the total quality of the project. In one or more embodiments, the QA director 108 may handle the whole QA process for the complete software project.

[0037] The PMO/ Project manager 110 may manage complete project including, but not limited to developing, testing, QA testing, architcting and project delivery. In one or more embodiments, the PMO/project manager 110 may analyze and collect requirements, design architecture, provide timelines and provide parameters for the project. In one or more embodiments, the requirements, parameters, the timelines etc. may be input to the software testing forecasting module 100. The QA director 108 may configure the software testing forecasting module 100 based on the inputs provided by the PMO/project manager 110. In one or more embodiments, the QA director 108 may custom configure the software testing forecasting module 100 based on requirements. The QA team lead 106 and the QA testers 104i_N may test the software prepared by the developers to reduce fault rate and to make sure that the software meets the requirement in goal as well as in quality. Testing process and progress may be periodically updated in the software testing forecasting module 100. In addition, the process and/or the progress may be viewed online by the users anytime. The executive committee 120 may also be provided GUI's to enable them to understand the process and progress of projects. The executive committee 120 may include a CEO, CFO and/or CIO.
Figure 2 illustrates a system view of variable function analysis service, according to one or more embodiments. In one or more embodiments, the variable function analysis service may be provided by using history metrics. Existing metrics defined in the software testing forecasting module 100 may be gathered periodically and stored in a repository for comparison. In one or more embodiments, the variable function analysis service may compare historic metrics to the current metrics on various categories. The categories may include, but not limited to dashboard grid metrics, test activity, defects status, forecasted dates and project variance, test trends and defect trends. The categories may further include parameters that are used for comparisons. For example, the dashboard grid metrics may include parameters such as, but not limited to a first test, last test, total tests, pending defects, average tests run/day, percentage passed and status. Similarly, the test trends category may include test execution and resource trends.

The software testing forecasting module 100 may be part of a software testing forecasting application server 200 that communicates with the Quality Center (QC) server 210. The software testing forecasting application server 200 may include inter alia, a software testing forecasting configuration database 202, a variable function analysis database 204, a variable function analysis service 206, and the software testing forecasting module 100. The QC server 210 may include inter alia, QC database 212, a QC database server 214, and a QC application server 216. In one or more embodiments, the QC database 212 may be substantially similar or same as the quality center database 102.

According to one embodiment, the software testing forecasting configuration database 202 may include predefined configurations. In one or more embodiments, the software testing forecasting configuration database 202 may also include manual configurations configured by the administrators. In one or more embodiments, the configurations in the software testing forecasting configuration database 202 may be used for performing testing and quality related operations in the software testing forecasting application server 200. The variable function analysis service 206 may perform analysis of the functions to determine whether the software accomplishes for what it is prepared. In addition, the variable function analysis service 206 may also determine what the
functions in the software accomplish. The variable function analysis database 204 may store all the data of the variable function analysis service 206. The software testing forecasting module 100 may provide metrics, test plans, etc to the variable function analysis service 206 for performing variable function analysis.

[0041] In one or more embodiments the variable function analysis service 206 may execute intelligence service function to periodically gather merits from QC database 212. The QC database 212 may store all the metrics and parameters. The software testing forecasting module 100 may use this information from the QC database 212 for providing historical comparisons. Also, in one or more embodiments, a web-service interface may be used for communication between the software testing forecasting application server 200 and the QC server 210.

[0042] In one or more embodiments, the QC database 212 in the QC server 210 may manage quality assurance testing data. The QC database server 214 may manage the QC database 212. In one or more embodiments, the QC database server 214 may include programs for managing the QC database 212. The QC application server 216 may manage the applications of the QC server 210, the QC database server 214 and the QC database 212.

[0043] Figure 3 is a graphical user interface (GUI) 300 illustrating a process and progress of QA, according to an example embodiment. In particular, Figure 3 illustrates summary view of the projects in progress. Figure illustrates a table that illustrates current projects and progress of those projects. The table may illustrate fields such as project field for project name, Base Line (BL) start field indicting the planned start of testing as defined in the project's configuration, first test field indicating the date of the first test, BL complete field indicating the planned end of testing as defined in the project's configuration, last test field indicating the date of the last test executed, pending defects field indicating a number of unresolved defects of severity levels, total tests field indicating a number of distinct tests in the test sets, average tests run/day field indicating average number of tests executed per day, the average test passed/day field indicating average number of tests passed per day, percentage passed field indicating the percentage of the total tests that have passed and the status field indicating the status of the project.
The progress may be indicated in table in form of indicators such as a quality assurance project status 302A. In addition, the progress with respect to time of a project of the number of projects may also be graphically indicated such as a quality assurance project status 302B. In addition, the progress, the metrics, the progress, etc. may be shown graphically in form of animations, or in form of spread sheets, etc. Furthermore, a forecast completion date 304 may be automatically illustrated graphically in the GUI. In addition, baseline completion date 306, may indicate the testers 104I_N the baseline completion date for the testing project. In addition, the GUI may be configured to illustrate quality assurance testing data 308I_N. It should be noted that GUI may be designed based on the user. For example, the QA director 108 or the project manager 110 may be interested to view only progress of the project. In another example, the QA testers 104I_N, and the QA team lead 106 may want to see progress and process with sufficient amount of data. Therefore, the software testing forecasting module 100 may provide options to the user for generating the GUI based on requirements. However, the permissions to enable generation of the customizable GUI may be reserved with the administrator.

Figure 4 is a system view illustrating a process of generating software testing forecast, according to one or more embodiments. In particular, Figure 4 illustrates an exploded view of the software testing forecasting module 100, the quality center database 102, the QA tester 104, and the PMO/ project manager 110. According to an embodiment, the QA tester 104 may perform analysis and testing of the software developed by the developers. Progress in the testing project, progress in the process, number of errors, bugs, defects, defect status are automatically uploaded into the quality center database 102 periodically. The PMO/ project manager 110 may input data to the software testing forecasting module 100 through a data input module 404. In one or more embodiments, the PMO/ project manager 110 may input data such as time lines for the projects to be completed, intermediate update reporting dates, etc. In addition, the PMO/ project manager 110 may also input configuration data. The software testing forecasting module 100 may implement the data provided by the PMO/ project manager 110 and may store the configurations in the quality center database 102. In one or more embodiments, the software testing forecasting module 100 may include a quality assurance testing data
extraction module 402. The quality assurance testing data extraction module 402 may extract the quality data periodically from the quality center database 102 using the processor 400. In one or more embodiments, the extracted quality assurance testing data may be used to generate a console (GUI) based on the configurations. The GUI may be output to the user through a data output module 406. In alternate embodiments, the software testing forecasting module 100 may also provide a console for requesting specific data.

[0046] Figure 5 is a system view illustrating a user using the software testing forecasting tool, according to one or more embodiments. The user 510 may be a software testing professional, a developer, a project manager, etc. In one or more embodiments, the user 510 may be a software testing professional. The user 510 may test the software assigned to the user 510 in a client device 506 (e.g., a computer, a mobile device). A client device 506 may communicate a quality assurance testing data from the quality center database 102 and the baseline completion date 306 from the client device 506 to a server device 504 to analyze the quality assurance testing data to calculate a forecasted completion date and a quality assurance project status (e.g., the quality assurance project status 302A-B).

[0047] In one or more embodiments, a forecasted completion date of a communication of a quality assurance testing data (e.g., the quality assurance testing data 308I_N) from the quality center database 102 to the server device 504 that includes a processor may be determined to analyze based on an algorithm the quality assurance testing data. In one or more embodiments, the quality assurance project status (e.g., the quality assurance project status 302A-B) of a communication of the baseline completion date 306 to the server device 504 may be created to analyze the baseline complete date and the forecasted completion date.

[0048] The quality assurance project status (e.g., the quality assurance project status 302A-B) that includes a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the quality center database 102 and the forecasted completion date of the server device 504 may be determined to assist a user to manage a quality assurance testing project (e.g., as illustrated in Figure 3). In addition, the quality assurance project status 302A-B of a communication of a baseline
start date to the server device 504 to analyze the baseline start date, the baseline complete
date, and the forecasted completion date may be created. The quality assurance testing
data 308i-N of the quality center database 102 may be analyzed based on the algorithm of
the server device to determine the forecasted completion date that includes, but not
limited to a number of total tests, a number of passed tests, a first test date, and a last test
date.

[0049] In one or more embodiments, a test execution trend that includes a number of
passed tests, a number of failed tests, and a number of not complete tests based on a time
of an analysis of the server device 504 of the quality assurance testing data 308i-N of the
quality center database 102 may be determined to assist the user to manage the quality
assurance testing project. In one or more embodiments, a group resource execution rate
that includes a number of tests executed by an offshore quality assurance team and a
number of tests executed by an onshore quality assurance team based on a time of an
analysis of the server device 504 of the quality assurance testing data 308i-N of the quality
center database 102 may be determined to assist the user to manage the quality assurance
testing project. In addition, an individual resource execution rate that includes a number
of tests executed by a quality assurance individual based on a time of an analysis of the
server device 504 of the quality assurance testing data 308i-N of the quality center
database 102 may be determined to assist the user to manage the quality assurance testing project through the client device 506. Furthermore, a defect trend that includes, but not
limited to a linked defect, an unlinked defect, and a severity of the linked defect based on
a time of an analysis of the server device of the quality assurance testing data of the
quality center database to assist the user may be determined in the client device 506 to
manage the quality assurance testing project.

[0050] According to one embodiment, the process and the updates may be updated in
the quality center database 102. Software testing data from the client device 506 may be
communicated to the server device 504 and the quality center database 102. The server
device 504 may generate the forecasted completion date from the quality assurance
testing data that includes a number of total tests, a number of passed tests, a first test date,
and a last test date from the client device and the quality assurance project status by
comparing a difference in time between the baseline complete date and the forecasted completion date.

[0051] In one or more embodiments, a forecasted completion date may be generated by the server device 504 based on an analysis performed by a processor of applying an algorithm of a quality assurance testing data that includes a number of total tests, a number of passed tests, a first test date, and a last test date of a client device 506. In addition, a quality assurance project status 302A-B of a comparison of a difference in time between a baseline complete date of the client device and the forecasted completion date may be produced.

[0052] Furthermore, the quality assurance project status 302A-B that includes a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the client device 506 and the forecasted completion date may be analyzed to assist a user to manage a quality assurance testing project. The aforementioned algorithm may include steps such as calculating a number of tests left to pass from a multiplication of a project goal percentage and the number of total tests and subtracting the number of passed tests, calculating an average passed rate by dividing the number of total tests by the difference in work days between the first test date and the last test date, calculating a number of work days needed by dividing the number of tests left to pass by the average passed rate, and generating the forecasted completion date by adding the number of work days needed to a current date.

[0053] The quality assurance project status 302A-B of a comparison of the difference in time between a baseline start date and the baseline complete date of the client device 506 and the forecasted completion date may be produced. In addition, the quality assurance project status 302A-B of an analysis of a project goal percentage of the client device 506 to analyze a number of tests left to pass and to analyze the number of tests left to pass and the forecasted completion date to generate the quality assurance project status.

[0054] Furthermore, a actual versus expected test execution rate from a comparison of a actual execution rate from an analysis of the quality assurance testing data and an expected execution rate of the client device may be generated to assist the user to manage the quality assurance testing project. The forecasted completion date from the quality
assurance testing data that includes the number of total tests, the number of passed tests, the first test date, and the last test date and from a number of quality assurance testers and a number of work hours in a day of the user of the client device to adjust the forecasted completion date may be regenerated to assist the user to manage a quality assurance testing project. A number of quality assurance testers from the quality assurance testing data 308I_N that includes the number of total tests, the number of passed tests, the first test date, and the last test date and from the baseline end date and a number of work hours in a day of the user of the client device may be generated to assist the user to manage the quality assurance testing project.

[0055] In one or more embodiments, the software testing data may be communicated to the server device 504 through a network 502. The server device 504 may extract information from the quality center database 102 and may generate user interfaces to be displayed on a display 508. The display associated with the client device 506 may render the quality assurance project status 302A-B to a user in a number of analytics dashboards rendered in each of the displays and to render the quality assurance project status that includes a variance in a time from the baseline complete date from an analysis of the quality assurance testing data 308I_N from the client device 506 and the forecasted completion date from the server device 504 to the client device 506 to assist a user of the client device 506 to manage a quality assurance testing project.

[0056] In one or more embodiments, the server device 504 may calculate a number of tests left to pass from a multiplication of a project goal percentage and the number of total tests and subtracting the number of passed tests. In addition, the server device 504 may calculate an average passed rate by dividing the number of total tests by the difference in work days between the first test date and the last test date. Also, the server device 504 may calculate a number of work days needed by dividing the number of tests left to pass by the average passed rate and generates the forecasted completion date by adding the number of work days needed to a current date.

[0057] In addition, the server device 504 may regenerate the forecasted completion date from the quality assurance testing data comprising the number of total tests, the number of passed tests, the first test date, and the last test date and from a number of quality assurance testers and a number of work hours in a day communicated to the

17
server device 504 from the user of the client device 506 to adjust the forecasted completion date to assist the user of the client device 506 to manage a quality assurance testing project. Furthermore, the server device 504 may generate a test execution trend comprising a number of passed tests, a number of failed tests, and a number of not complete tests based on a time from an analysis of the quality assurance testing data from an analysis of the quality assurance testing data the quality center database 102 to assist the user of the client device 506 to manage a quality assurance testing project. Also, the server device 504 may generate an actual versus expected test execution rate from a comparison of an actual execution rate from a calculation of the quality assurance testing data from the client device 506 communicating with the quality center database 102 and an expected execution rate from the client device 506 communicating with the user of the client device 506 to assist the user to manage a quality assurance testing project.

[0058] Figure 6 is a process view illustrating generation of quality assurance project status, according to one or more embodiments. A quality assurance testing data module 604 may extract information from the quality center database 102. A forecasted completion date module 602 may utilize the information from the quality assurance testing data module 604 to forecast the completion date. A baseline completion date module 606 may be set at a time of project assignment. A quality assurance project status module 608 may intake information from the forecasted completion date module 602 and the baseline completion date module 606 to generate quality assurance project status 302A-B in a form of report or display.

[0059] Figure 7A is a process flow of determining a forecast completion date, according to one embodiment. In operation 702, a forecasted completion date of a communication of a quality assurance testing data may be determined from a quality center database 102 to the server device 504 that includes the processor 400 to analyze based on an algorithm the quality assurance testing data. In operation 704, a quality assurance project status of a communication of the baseline completion date 306 to the server device 504 may be created to analyze the baseline completion date 306 and the forecasted completion date. In operation 706, the quality assurance project status including a variance in a time may be determined from the baseline completion date 306 from an examination of the quality assurance testing data from the quality center database.
102 and the forecasted completion date of the server device 504 to assist a user to manage a quality assurance testing project. In operation 708, the quality assurance project status of a communication of a baseline start date to the server device 504 may be created to analyze the baseline start date, the baseline complete date, and/or the forecasted completion date. The quality assurance testing data of the quality center database 102 is analyzed based on the algorithm of the server device 504 to determine the forecasted completion date that includes a number of total tests, a number of passed tests, a first test date, and/or a last test date. In operation 710, a test execution trend that includes a number of passed tests, a number of failed tests, and a number of not complete tests based on a time of an analysis of the server device 504 of the quality assurance testing data of the quality center database 102 to assist the user may be determined to manage the quality assurance testing project.

[0060] Figure 7B is a continuation of process flow of Figure 7A, illustrating additional process, according to one embodiment. In operation 712, a group resource execution rate that includes a number of tests executed by an offshore quality assurance team and a number of tests executed by an onshore quality assurance team based on a time of an analysis of the server device 504 of the quality assurance testing data of the quality center database 102 may be determined to assist the user to manage the quality assurance testing project.

[0061] In operation 714, an individual resource execution rate that includes a number of tests executed by a quality assurance individual based on a time of an analysis of the server device of the quality assurance testing data of the quality center database may be determined to assist the user to manage the quality assurance testing project. In operation 716, a defect trend including a linked defect, an unlinked defect, and/or a severity of the linked defect based on a time of an analysis of the server device of the quality assurance testing data of the quality center database may be determined to assist the user to manage the quality assurance testing project.

[0062] Figure 8A is a process flow illustrating a generation of a forecasted completion date, according to one embodiment. In operation 802, a forecasted completion date may be generated based on an analysis performed by a processor of applying an algorithm of a quality assurance testing data that includes a number of total tests, a number of passed
tests, a first test date, and/or a last test date of the client device 506. In operation 804, a quality assurance project status of a comparison of a difference in time between a baseline completion date 306 of the client device 506 and the forecasted completion date may be produced. In operation 806, the quality assurance project status including a variance in a time from the baseline completion date 306 from an examination of the quality assurance testing data from the client device and the forecasted completion date may be analyzed to assist a user to manage a quality assurance testing project.

[0063] In operation 808, a quality assurance project status of a comparison of the difference in time between a baseline start date and the baseline completion date 306 of the client device 506 and the forecasted completion date may be produced. In operation 810, a quality assurance project status of an analysis of a project goal percentage of the client device 506 may be produced to analyze a number of tests left to pass and to analyze the number of tests left to pass and the forecasted completion date to generate the quality assurance project status.

[0064] Figure 8B is a continuation of process flow of Figure 8A, illustrating additional process, according to one embodiment. In operation 812, an actual versus expected test execution rate from a comparison of a actual execution rate from an analysis of the quality assurance testing data and an expected execution rate of the client device 506 may be generated to assist the user to manage the quality assurance testing project. In operation 814, the forecasted completion date from the quality assurance testing data that includes the number of total tests, the number of passed tests, the first test date, and the last test date and from a number of quality assurance testers and a number of work hours in a day of the user of the client device may be regenerated to adjust the forecasted completion date to assist the user to manage a quality assurance testing project.

[0065] In operation 816, a number of quality assurance testers from the quality assurance testing data that includes the number of total tests, the number of passed tests, the first test date, and the last test date and from the baseline end date and a number of work hours in a day of the user of the client device 506 assist the user may be generated to manage the quality assurance testing project.

[0066] Configuring the various embodiments described in Figures 1-14 Projects
[0067] After the various embodiments described in Figures 1-14 have been installed on an application server, your IT staff will provide you with the appropriate URL for your browser so that you can begin managing your development projects. The appearance of the various embodiments described in Figures 1-14 prior to the configuration changes you are about to perform is shown in Figure 9. The steps on the following pages will guide you to setting up the various embodiments described in Figures 1-14 project. Each step leads to the next one and builds upon knowledge gained from each step, as shown in Figure 10. You are now looking at an application called the 'Configuration Editor' that should look like the screenshot Figure 11. This screen is where you'll spend most of your time managing project configuration settings once you've established the database connections and other seldom-changed parameters for a project. The immediate task in this step is to establish a connection to the QC database for 'Project 1'. (You'll be able to rename this project in another step.) The screen should look like the snapshot of Figure 12.

[0068] ...and allows fine scale management of your projects such as renaming, deleting and creating new projects. These functions are described in the subsections that follow.

[0069] Once a project's database connection has been established (as shown in the previous step), the check box labeled 'Reuse selected project's settings' will be enabled. This allows a new project to be created using the currently selected project's database connection values as well as the project's list method and values (to be discussed below in another step).

[0070] Projects can be active or inactive and this state is shown here:

[0071] Changing a project's status from active to inactive is discussed in another step on the Configuration Editor screen.

[0072] Note that project names can not contain any of the following symbols: / \ ? * [ ] & . | The reason for this is that when data is exported to Excel, several worksheets within the workbook use the project name as worksheet titles and Excel doesn't allow these symbols. If you do use one of these symbols data validation will substitute a dash for the prohibited symbols so don't be overly concerned when you're typing.

[0073] The & symbol is not actually prohibited by Excel but it does affect drill-down from the Dashboard pie charts since an & is used as part of the query string to launch a
new in the various embodiments described in Figures 1-14 with the project from the previous one.

[0074] Renaming a Project

When a project is selected in the 'Current Project List' list box, its name appears in the 'Selected Project' text box as in Figures 12-14:

[0076] To rename 'Project 1' simply type a new name (30 characters max) in the 'Selected Project' text box, overtyping the previous name...

[0077] ...and click on 'Rename'. The results will look like this:

[0078] Adding a New Project

[0079] To add a new project, type the name into this text box and click on 'Add'.

[0080] The results will immediately show up in the 'Current Project List'. To reuse a previous project's settings (in this example it would be the settings from the project named 'My Project') check the checkbox labeled 'Reuse selected project's settings'. Project's that do not have a database connection, when selected, will result in the 'Reuse selected project's settings' checkbox being disabled.

[0082] One of the uses of adding projects is that you can have as many the various embodiments described in Figures 1-14 projects as you need to monitor multiple subsets of test sets in a single QC project. To make this happen, simply assign the same database connection and schema information to the newly added project. For example you might create a the various embodiments described in Figures 1-14 project called 'Regression Tests' that monitors a selected number of QC test sets and is run each time a new build is produced by your development staff. Another project might monitor test sets that conduct a 'smoke test' for new builds. All these the various embodiments described in Figures 1-14 projects would be analyzing data from the same underlying QC database, but be pulling their data from subsets of test sets. This will become more apparent in a later step where test sets are selected for each project.

[0083] Deleting a Project

[0084] Select the project you wish to delete from the 'Current Project List' and press 'Delete'. You'll be prompted with a popup window to confirm this choice.
[0085] The project will be removed from the configuration. You cannot delete a project if it is the only one remaining in the list.

[0086] Note that all actions taken by you on the screen are shown in the message box at the bottom of the screen to provide a reference for you editing session.

[0087] After a database connection has been successfully established, the functionality of the Project List Method and Values section will be enabled. Each new the various embodiments described in Figures 1-14 project that is created is by design set to use the 'QC default values' and the lower portion of the Advanced Settings screen will look like the snapshot below.

[0088] In this condition, nothing needs to be done to save this list method - note that the 'Save Method' button is disabled.

[0089] By selecting 'Blocked', you then have two other items to consider, namely determining the attributes of the custom value by using the two check boxes marked as 'E' and 'T'. This requires a bit of explanation as to the usage of these attributes: 'E' denotes a test that can be executed and 'T' denotes a test that can be reported on the Test Trends of the various embodiments described in Figures 1-14. When you move your mouse over either of the two check boxes, a tool tip will remind you of this usage. Here's the explanation...

[0090] the various embodiments described in Figures 1-14 examines tests that exist in the Test Lab of QC for two conditions, namely that the test is executable - in other words it can be 'run' - and that the test should be tracked on the Test Trends as a valid test status that shows a tester actually tried to run the test and that this tester should be credited with this test run in the statistics that keep track of a testers contribution to testing progress.

[0091] When you consider these two attributes of a test, that is the 'execute-ability' and the 'trend-ability', the various embodiments described in Figures 1-14 assumes the following for the mapped test statuses as shown in Figure 13:

[0092] There are situations where you may want to modify QC's default lists, or you may have inherited projects where the default lists have already been changed - for those lists that can be changed - as well as new items added to default lists. Below is a project
that is using custom list values and the radio button called 'Custom values' was selected resulting in a change in the edit-ability of the screen in Figures 12-14.

[0093] Note the several features on the screen as described below. First we'll examine the role of 'Mapped Test Status Values' and then we'll move on to talk about defect values.

[0094] Note that the True/False values shaded in gray are set by design in the various embodiments described in Figures 1-14 and can't be changed, but the two custom values are fully configurable, in other words you can set them to be true or false. However, you could, if desired, change the actual value mapped to 'Passed', 'Failed', 'Not Completed', 'No Run', or 'N/A' to custom values but do know that this will require extra effort in QC when tests are run to pick the correct value since the default values will always appear as choices in QC. CTE doesn't recommend that you change these mapped values.

[0095] ...the various embodiments described in Figures 1-14 will show the current status of testing as follows on the Dashboard where each mapped test status name listed will contribute to the numeric count of tests with their respective statuses:

[0096] If you were to use the 'bubble popup' on this chart by moving the mouse over each region of the pie chart, you'd discover the following statistics about the most recent status of each category of test status:

[0097] Note that the pie chart says 70% of the 10 executable tests have been run but the statistics above show that there exist a total of 12 tests. Since those marked 'N/A' are not considered executable for various reasons, then the various embodiments described in Figures 1-14 only counts 10 as candidates for 'running' and this value is the sum of the numbers in the shaded columns: Passed {2} + Failed {3} + Not Completed {2} = 7 out of 10 or 70%.

[0098] Also note the Project Metric grid for this project on the Dashboard:

[0099] Of interest here, relative to custom test statuses, is the fact that only 'Executable Tests' are examined relative to the '%Passed' which is used to forecast a completion date when testing has reached its goal (see a later section in this document on testing goals). Note from the grid above that 20% of the executable tests have passed: from the previous you can confirm this by noting that 2 out of 10 tests have passed, thus 20%.
Now on to the test trends, since 'Blocked' isn't marked as 'trend-able' then the test trends looks like this:

If 'Blocked' had been marked as 'trend-able' then you would see this display:

Also, the tester who marked the test with a status of 'Blocked' would get credit for the execution as shown in the 'Testing Resources' chart beneath the 'Test Execution Trend' chart in the various embodiments described in Figures 1-14.

Selecting Test Status Values

When you've chosen to use custom values, you'll note several states in which a custom value section of the screen may exist.

User Control State

Appearance

Initial state: screen is waiting for a selection to be made.

A selection is made and the 'E' checkbox is enabled. If 'E' remains unchecked then the various embodiments described in Figures 1-14 will count these tests for informational purposes like those that have a status of 'N/A' but the tests will not be considered executable.

Here 'E' has been checked and note that 'T' is only enabled after 'E' has been checked.

In this case the test is considered to be both executable and trend-able.

If you don't want to monitor a custom value, then make this selection prior to saving.

When selections have been made, validation code ensures that you can't save the values unless each mapped test status value is unique. For instance, you will receive a popup message that prevents saving under the conditions like those in Figures 12-14.

The last mapped value that must be selected is the one that indicates to the various embodiments described in Figures 1-14 that a defect is no longer pending, in other words it is closed. By default, we preselect the 'Bug Status' value of 'Closed' in the drop down but it does allows another value to be chosen if desired:

If you select another value that is also in the list of 'Pending Defect Statuses Monitored' (see below) a warning message will popup and not allow you to save the configuration until you de-select the new mapped value from the pending list.
Defect Status and Severity Values

the various embodiments described in Figures 1-14 reads all the values that exist for a QC project's lists for Defect Status and Defect Severity. These two lists are customized in QC under the list names of 'Bug Status' and 'Severity' respectively in the 'Customize' feature of QC.

In the Defect Status list you will not see the value of 'Closed' because it is reserved by default for the mapping shown above for the 'Defect Status Closed' value. However, all other values are available for selection.

The 'Refresh Lists' button can be used when the saved values don't include newly added custom values - this will also refresh all the drop down lists for mapped values described on previous pages.

Often these status values are given business-meaningful names. In the case where sub-lists are created, as shown in an example below, top-level sub list names are not shown nor should be used as values; note that the sub-list name 'Second Tier Cat' does not appear in the the various embodiments described in Figures 1-14 list but its sub-values do appear.

You must select at least one list value from each list in order to save the values; initially all the values are selected for convenience. Once you've made your selections, push the 'Save Method' button. If at any time the values have changed in QC since a previous configuration was established, use the 'Refresh Lists' button to obtain new values and then reselect values and press the 'Save Method' button.

To return to the Configuration Editor screen for the next step, click on the 'Return to Editor' button.

Step 6: Working with Project Management Information

When you close the 'Advanced Settings' screen, you'll return to the main configuration editor as shown in the snapshot below. Assuming the connection settings were saved as described in Step 3, you'll see data coming from QC in the 'Use Groups for Test Trend Analysis' list that you can use to complete the project configuration.

The table below describe the project management parameters this screen configures. Once you're familiar with these parameters, we'll illustrate how to adjust them for your project Setting Purpose Default Value Example Values How to Edit
[00125] A project is considered complete when the number of tests passed in the monitored test sets has reached or passed this value.

[00126] The complete variance gauge and the project status summary grid's 'traffic light' image become red in color based on this value. Enter a number that you consider a 'red' warning when the project complete date is behind by this many days. Enter a number directly into the test box or use the up and down arrows to adjust the value. The minimum value allowed is 2; there is no limit on the maximum value. The complete variance gauge and the project status summary grid's 'traffic light' image become yellow in color based on this value. Note that the gauge will still show 'green' even when the project variance is greater than zero but less than the number of days set by this numeric value.

[00127] Enables benchmarks to be established as a project progresses. This is the number of tests that are expected to be executed per hour without regard for their pass, fail or not complete status.

[00128] This value is used in the Quality Intelligence screen to determine the expected execution rate for a project and is discussed in the on-line help file.

[00129] Enter a number that estimates this value based on prior experience.

[00130] Enter a number directly into the test box or use the up and down arrows to adjust the value. The minimum value allowed is 1; there is no limit on the maximum value.

[00131] Expected Defect Rate (%)

[00132] Enables benchmarks to be established relative to how many defects are created based on the number of tests that are executed. You'll see this value on both the Dashboard and Quality Intelligence screen.

[00133] Enter a number based on prior experience with development projects.

[00134] Enter a number directly into the test box or use the up and down arrows to adjust the value. The maximum value allowed is 100 and the minimum 0.

[00135] Testers Assigned

[00136] Enables benchmarks to be established as a project progresses.

[00137] This value is used in the Quality Intelligence screen to determine the expected execution rate for a project and is discussed in the on-line help file. Enter the number of
testers assigned to the monitored test sets. You can enter fractional number of testers if needed (e.g. 1.5 testers)

[00138] Enter a number directly into the text box or use the up and down arrows to adjust the value. The minimum value allowed is 0.1; there is no limit on the maximum value.

[00139] Use Groups for Test Trend Analysis

[00140] Determines if testers have been assigned to custom group names for reporting.

[00141] All settings shown in the table above must have a non-null value. Data validation code prevents saving a project's configuration when values are missing. In addition, the validation methods check for a Baseline start date prior to the Baseline complete date; neither date value can be null and a project must complete at least one day after it starts.

[00142] Enter the values described in the table above for your project - sample values are in Figures 12-14.

[00143] Quality Center allows custom user groups to be created that allow the various embodiments described in Figures 1-14 to report on the testing activity by groups in addition to the default mode of showing test execution trends by individual testers. When custom groups are created in the various embodiments described in Figures 1-14, it is important that testers only be assigned to one of the custom groups in order to present correct trends for the group. A typical list of default QC groups is in Figures 12-14:

[00144] To use groups, first check the 'Use Groups for Test Trend Analysis' checkbox. This will enable the list of groups from which you select by clicking on the checkbox to the left of a group name.

[00145] Next, select those groups you wish to monitor. Note that several of the group names in Figures 12-14 are the 'built-in' names supplied with every QC project. In the list below, two new groups (Offshore QA Team & Onshore QA Team) have been added to the QC project explicitly for monitoring by the various embodiments described in Figures 1-14.

[00146] Each group that you select to monitor should have mutually exclusive lists of users. The arrows below illustrate the mutual exclusivity of the association of names to groups
[00147] You can, of course, create more than two groups for monitoring, just be sure that each group you create has a unique list of users in each group, the various embodiments described in Figures 1-14 tracks each user's test activity and the group to which they belong thus accumulates this test activity and the activity from other users for reporting. If a user belongs to more than one group, then the group activity will be skewed. For this reason, you should not use the default groups created for each QC project since typically the users will be assigned to several of the default groups. For example, almost all users are part of the 'Viewer' group.

[00148] Concerning the use of groups, there can be situations where a QC user was removed from the project or the site but had performed testing and their testing efforts will be shown on the Test Trend as in the example below:

[00149] Here, both 'alex_qc' and 'shelly_qc' have performed testing but have been removed from the system; when you view the Testing Resource you see this for the time interval analyzed:

[00150] Note the red label showing "Testing by deleted user(s) and unmonitored group(s) detected". This is an indication that none of the monitored groups are showing any testing activity but there was testing occurring over the time interval. In some cases this message might read "Testing by deleted users(s) detected" or "Testing by unmonitored group(s) detected" as a function of how the deletion(s) were done in QC.

[00151] However, there are cases where, if the time interval is large, that you'll see testing results as well as the warning message such as below:

[00152] Here the testers 'alice_qc', 'cecil_qc' and 'michael_qc' who are in the 'QA Tester' group, have their contributions shown but the two deleted users, 'alex_qc' and 'shelly_qc' while having done testing, are not shown in the group trend page. For reference, 'alex_qc' was in the 'Viewer' group and 'shelly_qc' in the 'Defect Reporter', 'QA Manager' and 'R & D Manager' groups; none of groups that 'shelly_qc' was in are in the monitored list.

[00153] When you drop down the 'Defined Test Sets' list control...

[00154] ...you'll see all the parent (top-level) test set folder names from the QC Test Lab for the project. In this example, the built-in QC demo project is shown.

[00155] You can select all the test sets at one time if you desire by selecting the entry in braces called '{Select All Test Sets in Test Lab}' but typically, if your the various
embodiments described in Figures 1-14 project is a subset of the overall QC project, then you'll only select one or two top level folders. Below is the result of selecting only the 'Mercury Tours Web Site' parent folder.

[00156] Note the two buttons on top of the list to the right:

[00157] The button with the checkmark will select all the test sets and the one with the minus sign will deselect all the test sets. You can of course select individual test sets manually. The screen shot below shows the results of having pressed the 'checkmark' button to select all the test sets in on top-level parent folder.

[00158] Next you add this selection to the 'Monitored Test Sets' list by pressing the 'Add to Monitored List' button...

[00159] ...and then, if you're all done selecting test sets, press the 'Save Selection' button.

[00160] You'll note that the font color on the list has now changed from red to black as an indication that the selection has been saved.

[00161] To change the selection, use the 'Clear Selection' button to the lower left of the 'Monitored Test Sets' list. If you try to add the same test set twice, data validation code will prevent this and a message in the 'Messages' text

[00162] box will appear reminding you of this fact. You can, of course, continue to add test sets from different parent folders.

[00163] If you revisit the screen after having made selections, it will look like this:

[00164] You'll also note on the main Configuration Editor screen a list of the selected test sets so you can determine if changes are needed to the test set selection.

[00165] At this point, you're done with the basic configuration for a project. You can view the settings by selecting 'Close' from the Configuration Editor screen. This is shown in the next section.

[00166] Or, if you switch to the main Dashboard that's open and select the refresh icon in the lower left hand corner of the page, you're newly configured project should pull back data from QC and populate the page.

[00167] Also, if you'd like to take advantage of the optional feature of organizing your projects into meaningful business-oriented names, see the section on Project Groups on 34.

[00168] The Configuration Settings Viewer Page
[00169] This is viewable by all users of the various embodiments described in Figures 1-14. You can also export all the configuration settings to Excel by using the small Σ symbol in the lower right hand corner of the page.

[00170] The message (here) is to remind users to notify you, the product administrator, if they are aware of any QC changes that may not be reflected in the current the various embodiments described in Figures 1-14 configuration. The most common out-of-sync conditions are when QC test sets are renamed, deleted or added. Since the various embodiments described in Figures 1-14 reads the QC test lab hierarchical test set tree, this can happen. In some cases, the various embodiments described in Figures 1-14 may continue to report accurate data but

[00171] often you'll notice a change in the project's display because test sets that have been monitored no longer exist or exist under a different name.

[00172] To rectify any out-of-sync conditions for a selected project, revisit the Monitored Test Sets screen (30) and clear and reselect the test sets to monitor.

[00173] At this point in learning how to configure a the various embodiments described in Figures 1-14 project you're probably ready to see some actual test management data. If you switch to the open browser window that contains the main Dashboard page, select the 'Refresh Source Data' icon in the lower left corner of the to force the various embodiments described in Figures 1-14 to re-read the configuration settings and it will display the current status of your project.

[00174] There is a bit more to learn about configuring the various embodiments described in Figures 1-14, for example the use of Project Groupings that is described below. For now, we've covered the basics of project configuration but read on for more ways to improve your management of testing projects.

[00175] Project Groups

[00176] When you are managing a large number of projects, it can be helpful to group related projects together into what the various embodiments described in Figures 1-14 calls a 'Project Group'. These are accessed via the drop-down control at the top of all the main application pages.

[00177] The 'built-in' group is named 'All Projects' and initially this group is set as the default, meaning that all active projects will be shown. You can add your own project
groups and adjust the default group by bringing up the 'Project Group' screen from the configuration editor.

[00178] Your Project Groups may look different from the one below as a function of how many projects you have configured and you may see only the 'All Projects' group listed, but the screenshot below shows a more fully configured list.

[00179] Note that when the system-controlled 'All Projects' list is selected that controls are disabled, indicating the fact that you can't rename or delete this group nor can you save any changes to the selected project list.

[00180] As you change the selected group in the 'Project Groups' list box on the left hand side of the page, you'll see a different set of projects selected in the 'Active Projects' list on the right.

[00181] Compare this view above with that on the previous page. Note also that the checkbox called 'Default' is now enabled. If you want to make the 'Claims' group the default view then check this checkbox and note the change in the screen:

[00182] You'll also see a message at the bottom of the screen showing the action that was taken.

[00183] Since only one 'Project Group' can be the default, then the checkbox is again disabled. Selecting another group name will re-enable the checkbox and then the newly selected group can be set to default. After a change, the reloads the data, always placing the default group as the first in the list of project groups.

[00184] You can rename, delete and add new groups using the same techniques as described for project name maintenance in step 4 that began on 10. Note that the maximum length for a project group name is 20 characters and validation code ensures that each group name is unique.

[00185] ...that reminds you to select at least one project from the 'Active Projects' list. Note that the list looks like the screenshot below after you've added a new group.

[00186] Select appropriate projects from the list and press the 'Save Group Selection' button to update the group settings.

[00187] After groups have been added, the drop-down list in the various embodiments described in Figures 1-14 will look like the following once you refresh the various
embodiments described in Figures 1-14 using the 'Refresh Source Data' icon on the lower left-hand corner of the page:

[00188] Selecting a group will reconfigure the application pages to show only those projects associated with the group name.

[00189] Now on to explaining the result of checking the 'E' and/or 'T' check boxes. An illustration will be most helpful and to do this, we'll look at the results of using the custom 'Blocked' status described above.

[00190] References to the Quality Center (QC) test management software (TMS) are made in this guide. QC is a product of the Hewlett Packard Corporation and the various embodiments described in Figures 1-14 is currently designed to interact with this TMS. Future versions of the various embodiments described in Figures 1-14 may enable metrics to be pulled from other TMS systems.

[00191] After the various embodiments described in Figures 1-14 have been installed on an application server, your IT staff will provide you with the appropriate URL for your browser so that you can begin managing your development projects. The appearance of the various embodiments described in Figures 1-14 prior to the configuration changes you are about to perform is in Figures 12-14:

[00192] The steps on the following pages will guide you to setting up a the various embodiments described in Figures 1-14 project. Each step leads to the next one and builds upon knowledge gained from each step.

[00193] Step 1: Configuration Menu

[00194] Select the 'Configuration' icon in the upper left corner of the main page. ...

[00195] ...and a new browser will open with this screen:

[00196] Enter the password provided by your IT staff and click on 'Edit Configuration'. You can also select this button by pressing 'Alt+E' from the keyboard or pressing the 'Enter' key.

[00197] Step 2: An Overview of Configuration Settings

[00198] You are now looking at an application called the 'Configuration Editor' that should look like the screenshot below.

[00199] This screen is where you'll spend most of your time managing project configuration settings once you've established the database connections and other
seldom-changed parameters for a project. We'll discuss the various settings on this in another step because we first must connect a project to its source data.

[00200] To do this select the 'Advanced Settings' button.

[00201] CTE the various embodiments described in Figures 1-14. Administrators Guide

[00202] Step 3: Establish a Database Connection

[00203] The immediate task in this step is to establish a connection to the QC database for 'Project 1'. (You'll be able to rename this project in another step.) The screen should look like the snapshot below.

[00204] Use the worksheet provided by your network staff to obtain the values for the text boxes in the upper right hand portion of the screen. For a SQL Server-based QC installation a typical worksheet will look like that shown on the next page. The steps for an Oracle-based installation are similar, so please read this section first to get a feel for the overall procedure. The differences for Oracle will be discussed in a moment.

[00205] SQL Server Configuration Worksheet Sample QC Project Name Database Name (corresponds to a QC project name) Database Schema Server Name User Name Password

[00206] Now, press the 'Save Connection' button and a successful connection will yield this message in the 'Advanced Settings Messages' text box...

[00207] ...and the label that said '(Empty)' will now read '(Saved)'.

[00208] If the user name and/or password must be changed, you can enter these new values at any time and save the new connection data.

[00209] If your QC installation uses Oracle, a typical worksheet might look like the one in Figures 12-14.

[00210] Oracle Configuration Worksheet Sample Data Source Name (Corresponds to the instance of Oracle that supports the database schemas) QC Project Name Database Schema (Corresponds to a QC project) User Name Password

[00211] Note that for Oracle, there is not a value for the 'Database Name' and the screen will look like the snapshot below:
In this case, 'Project 1' will be mapped to the QC project named 'Main Testing'. You could rename 'Project 1' to correspond to the QC project name if desired. This value is only for reference in the worksheet.

In the event that you fail to enter all the necessary information, or that the information is incorrect, a message will appear in the 'Advanced Settings Messages' text box that will reveal the problem as well as a popup window notifying you of the issue. Several examples are in Figures 12-14 that might occur when you select 'Test Connection'. Incorrect or Missing Data Message

Missing 'Data Source/Server' information. A similar message will be present for any other missing information.

Invalid user credentials

Invalid data source or server. Note that this is a SQL Server message and a similar one would be listed for Oracle.

This message may take up to a minute to appear as the application tries to establish a connection; this depends on the server's timeout value for establishing a connection.

The portion of the advanced settings screen shown on 7 that handles project names is in Figures 12-14...

…and allows fine scale management of your projects such as renaming, deleting and creating new projects. These functions are described in the subsections that follow.

Once a project's database connection has been established (as shown in the previous step), the check box labeled 'Reuse selected project's settings' will be enabled. This allows a new project to be created using the currently selected project's database connection values as well as the project's list method and values (to be discussed below in another step).

Projects can be active or inactive and this state is shown here:

Changing a project's status from active to inactive is discussed in another step on the Configuration Editor screen.

Note that project names can not contain any of the following symbols: / \ ? * [ ] &.1 The reason for this is that when data is exported to Excel, several worksheets within the workbook use the project name as worksheet titles and Excel doesn't allow these...
symbols. If you do use one of these symbols data validation will substitute a dash for the prohibited symbols so don't be overly concerned when you're typing.

[00224] The & symbol is not actually prohibited by Excel but it does affect drill-down from the Dashboard pie charts since an & is used as part of the query string to launch a new in the various embodiments described in Figures 1-14 with the project from the previous pre-selected.

[00225] Renaming a Project

[00226] When a project is selected in the 'Current Project List' list box, its name appears in the 'Selected Project' text box as in Figures 12-14:

[00227] To rename 'Project 1' simply type a new name (30 characters max) in the 'Selected Project' text box, overtyping the previous name...

[00228] ...and click on 'Rename'. The results will look like this:

[00229] Adding a New Project

[00230] To add a new project, type the name into this text box and click on 'Add'.

[00231] The results will immediately show up in the 'Current Project List'. To reuse a previous project's settings (in this example it would be the settings from the project named 'My Project') check the checkbox labeled 'Reuse

[00232] selected project's settings'. Project's that do not have a database connection, when selected, will result in the 'Reuse selected project's settings' checkbox being disabled.

[00233] One of the uses of adding projects is that you can have as many the various embodiments described in Figures 1-14 projects as you need to monitor multiple subsets of test sets in a single QC project. To make this happen, simply assign the same database connection and schema information to the newly added project. For example you might create a the various embodiments described in Figures 1-14 project called 'Regression Tests' that monitors a selected number of QC test sets and is run each time a new build is produced by your development staff. Another project might monitor test sets that conduct a 'smoke test' for new builds. All these the various embodiments described in Figures 1-14 projects would be analyzing data from the same underlying QC database, but be pulling their data from subsets of test sets. This will become more apparent in a later step where test sets are selected for each project.
Deleting a Project

Select the project you wish to delete from the 'Current Project List' and press 'Delete'. You'll be prompted with a popup window to confirm this choice.

The project will be removed from the configuration. You cannot delete a project if it is the only one remaining in the list.

Note that all actions taken by you on the screen are shown in the message box at the bottom of the screen to provide a reference for you editing session.

After a database connection has been successfully established, the functionality of the Project List Method and Values section will be enabled. Each new the various embodiments described in Figures 1-14 project that is created is by design set to use the 'QC default values' and the lower portion of the Advanced Settings screen will look like the snapshot below.

In this condition, nothing needs to be done to save this list method - note that the 'Save Method' button is disabled.

There are situations where you may want to modify QC's default lists, or you may have inherited projects where the default lists have already been changed - for those lists that can be changed - as well as new items added to default lists. Below is a project that is using custom list values and the radio button called 'Custom values' was selected resulting in a change in the edit-ability of the screen in Figures 12-14.

Note the several features on the screen as described below. First we'll examine the role of 'Mapped Test Status Values' and then we'll move on to talk about defect values.

the various embodiments described in Figures 1-14 must know the value of the test statuses you will use for the default and unchangeable values of 'Passed', 'Failed', 'Not Completed', 'No Run' and 'N/A'. You may have added to this default list, but the default values are preselected as shown on the screen. They can be changed and each drop-down control contains the full list of values from the 'Status' list that can be customized in QC.

You typically will leave these default values selected as shown. The primary reason for custom test status values is to add one or two to clarify the state of particular test when the default values aren't descriptive enough.
the various embodiments described in Figures 1-14 provides the capacity to use two additional custom test status values as shown above with the drop-down controls associated with the label's 'Custom1' and 'Custom2'.

In the case of the project shown here, only one custom value has been created, namely 'Blocked'. This might be used to indicate that a defect is blocking the execution of a particular test.

By selecting 'Blocked', you then have two other items to consider, namely determining the attributes of the custom value by using the two check boxes marked as 'E' and 'T'. This requires a bit of explanation as to the usage of these attributes: 'E' denotes a test that can be executed and 'T' denotes a test that can be reported on the Test Trends of the various embodiments described in Figures 1-14. When you move your mouse over either of the two check boxes, a tool tip will remind you of this usage. Here's the explanation...

the various embodiments described in Figures 1-14 examines tests that exist in the Test Lab of QC for two conditions, namely that the test is executable - in other words it can be 'run' - and that the test should be tracked on the Test Trends as a valid test status that shows a tester actually tried to run the test and that this tester should be credited with this test run in the statistics that keep track of a testers contribution to testing progress.

When you consider these two attributes of a test, that is the 'execute-ability' and the 'trend-ability', the various embodiments described in Figures 1-14 assumes the following for the mapped test statuses:

Note that the True/False values shaded in gray are set by design in the various embodiments described in Figures 1-14 and can't be changed, but the two custom values are fully configurable, in other words you can set them to be true or false. However, you could, if desired, change the actual value mapped to 'Passed', 'Failed', 'Not Completed', 'No Run', or 'N/A' to custom values but do know that this will require extra effort in QC when tests are run to pick the correct value since the default values will always appear as choices in QC. CTE doesn't recommend that you change these mapped values.
[00250] Now on to explaining the result of checking the 'E' and/or 'T' check boxes. An illustration will be most helpful and to do this, we'll look at the results of using the custom 'Blocked' status described above.

With a custom configuration like the following...

[00251] ...the various embodiments described in Figures 1-14 will show the current status of testing as follows on the Dashboard where each mapped test status name listed will contribute to the numeric count of tests with their respective statuses:

[00252] If you were to use the 'bubble popup' on this chart by moving the mouse over each region of the pie chart, you'd discover the following statistics about the most recent status of each category of test status:

[00253] Note that the pie chart says 70% of the 10 executable tests have been run but the statistics above show that there exist a total of 12 tests. Since those marked 'N/A' are not considered executable for various reasons, then the various embodiments described in Figures 1-14 only counts 10 as candidates for 'running' and this value is the sum of the numbers in the shaded columns: Passed {2} + Failed {3} + Not Completed {2} = 7 out of 10 or 70%.

[00254] Also note the Project Metric grid for this project on the Dashboard:

[00255] Of interest here, relative to custom test statuses, is the fact that only 'Executable Tests' are examined relative to the '%Passed' which is used to forecast a completion date when testing has reached its goal (see a later section in this document on testing goals). Note from the grid above that 20% of the executable tests have passed: from the previous you can confirm this by noting that 2 out of 10 tests have passed, thus 20%.

[00256] Now on to the test trends, since 'Blocked' isn't marked as 'trend-able' then the test trends looks like this:

[00257] If 'Blocked' had been marked as 'trend-able' then you would see this display:

[00258] Also, the tester who marked the test with a status of 'Blocked' would get credit for the execution as shown in the 'Testing Resources' chart beneath the 'Test Execution Trend' chart in the various embodiments described in Figures 1-14.

[00259] Selecting Test Status Values

[00260] When you've chosen to use custom values, you'll note several states in which a custom value section of the screen may exist.
[00261] User Control State

[00262] Appearance

[00263] Initial state: screen is waiting for a selection to be made.

[00264] A selection is made and the Ε’ checkbox is enabled. If Ε’ remains unchecked then the various embodiments described in Figures 1-14 will count these tests for informational purposes like those that have a status of 'N/A' but the tests will not be considered executable.

[00265] Here Ε’ has been checked and note that 'I' is only enabled after Ε’ has been checked.

[00266] In this case the test is considered to be both executable and trend-able.

[00267] If you don't want to monitor a custom value, then make this selection prior to saving.

[00268] When selections have been made, validation code ensures that you can't save the values unless each mapped test status value is unique. For instance, you will receive a popup message that prevents saving under the conditions like those in Figures 12-14.

[00269] The last mapped value that must be selected is the one that indicates to the various embodiments described in Figures 1-14 that a defect is no longer pending, in other words it is closed. By default, we preselect the 'Bug Status' value of 'Closed' in the drop down but it does allows another value to be chosen if desired:

[00270] If you select another value that is also in the list of 'Pending Defect Statuses Monitored' (see below) a warning message will popup and not allow you to save the configuration until you de-select the new mapped value from the pending list.

[00271] Defect Status and Severity Values

[00272] the various embodiments described in Figures 1-14 reads all the values that exist for a QC project's lists for Defect Status and Defect Severity. These two lists are customized in QC under the list names of 'Bug Status' and 'Severity' respectively in the 'Customize' feature of QC.

[00273] In the Defect Status list you will not see the value of 'Closed' because it is reserved by default for the mapping shown above for the 'Defect Status Closed' value. However, all other values are available for selection.
[00274] The 'Refresh Lists' button can be used when the saved values don't include newly added custom values - this will also refresh all the drop down lists for mapped values described on previous pages.

[00275] Often these status values are given business-meaningful names. In the case where sub-lists are created, as shown in an example below, top-level sub list names are not shown nor should be used as values; note that the sub-list name 'Second Tier Cat' does not appear in the the various embodiments described in Figures 1-14 list but its sub-values do appear.

[00276] List Values in the various embodiments described in Figures 1-14

[00277] List Values in QC

[00278] You must select at least one list value from each list in order to save the values; initially all the values are selected for convenience. Once you've made your selections, push the 'Save Method' button. If at any time the values have changed in QC since a previous configuration was established, use the 'Refresh Lists' button to obtain new values and then reselect values and press the 'Save Method' button.

[00279] To return to the Configuration Editor screen for the next step, click on the 'Return to Editor' button.

[00280] Step 6: Working with Project Management Information

[00281] When you close the 'Advanced Settings' screen, you'll return to the main configuration editor as shown in the snapshot below. Assuming the connection settings were saved as described in Step 3, you'll see data coming from QC in the 'Use Groups for Test Trend Analysis' list that you can use to complete the project configuration.

[00282] The table below describe the project management parameters this screen configures. Once you're familiar with these parameters, we'll illustrate how to adjust them for your project Setting Purpose Default Value Example Values How to Edit

[00283] Passed Test Goal %

[00284] A project is considered complete when the number of tests passed in the monitored test sets has reached or passed this value.

[00285] Enter a number directly into the test box or use the up and down arrows to adjust the value. The maximum value allowed is 100 and the minimum 0.

[00286] Red Alert After _ Day(s)
The complete variance gauge and the project status summary grid's 'traffic light' image become red in color based on this value.

Enter a number that you consider a 'red' warning when the project complete date is behind by this many days.

Enter a number directly into the test box or use the up and down arrows to adjust the value. The minimum value allowed is 2; there is no limit on the maximum value.

Yellow Alert After _ Day(s)

The complete variance gauge and the project status summary grid's 'traffic light' image become yellow in color based on this value. Note that the gauge will still show 'green' even when the project variance is greater than zero but less than the number of days set by this numeric value.

Enter a number that you consider a 'yellow' warning when the project complete date is behind by this many days.

Enter a number directly into the test box or use the up and down arrows to adjust the value. The minimum value allowed is 1; there is no limit on the maximum value.

Enter the number of hours in a typical workday that your testing group actually performs testing.

Enter a number directly into the text box or use the up and down arrows to adjust the value. The minimum value allowed is 1 and the maximum value is 24.

Exclude Sat?

Determines if rate and forecasting calculations should consider Saturday a work day.

If testing occurs regularly on Saturday then uncheck this control.

Determines if rate and forecasting calculations should consider Sunday a work day.

Checked

Enables benchmarks to be established as a project progresses. This is the number of tests that are expected to be executed per hour without regard for their pass, fail or not complete status.
[00302] This value is used in the Quality Intelligence screen to determine the expected execution rate for a project and is discussed in the on-line help file.

[00303] Enter a number that estimates this value based on prior experience.

[00304] Enter a number directly into the text box or use the up and down arrows to adjust the value. The minimum value allowed is 1; there is no limit on the maximum value.

[00305] Expected Defect Rate (%)

[00306] Enables benchmarks to be established relative to how many defects are created based on the number of tests that are executed. You'll see this value on both the Dashboard and Quality Intelligence screen.

[00307] Enter a number based on prior experience with development projects.

[00308] Enter a number directly into the test box or use the up and down arrows to adjust the value. The maximum value allowed is 100 and the minimum 0.

[00309] Enter a number directly into the test box or use the up and down arrows to adjust the value. The minimum value allowed is 0.1; there is no limit on the maximum value.

[00310] All settings shown in the table above must have a non-null value. Data validation code prevents saving a project's configuration when values are missing. In addition, the validation methods check for a BaseLine start date prior to the BaseLine complete date; neither date value can be null and a project must complete at least one day after it starts.

[00311] Enter the values described in the table above for your project - sample values are in Figures 12-14.

[00312] Quality Center allows custom user groups to be created that allow the various embodiments described in Figures 1-14 to report on the testing activity by groups in addition to the default mode of showing test execution trends by individual testers. When custom groups are created in the various embodiments described in Figures 1-14, it is important that testers only be assigned to one of the custom groups in order to present correct trends for the group. A typical list of default QC groups is in Figures 12-14:
To use groups, first check the 'Use Groups for Test Trend Analysis' checkbox. This will enable the list of groups from which you select by clicking on the checkbox to the left of a group name.

Next, select those groups you wish to monitor. Note that several of the group names in Figures 12-14 are the 'built-in' names supplied with every QC project. In the list below, two new groups (Offshore QA Team & Onshore QA Team) have been added to the QC project explicitly for monitoring by the various embodiments described in Figures 1-14.

Each group that you select to monitor should have mutually exclusive lists of users.

The management of groups in QC presents a display like that in Figures 12-14.

On the next page, note the two views that represent the two teams that are reported upon by the various embodiments described in Figures 1-14.

You can, of course, create more than two groups for monitoring, just be sure that each group you create has a unique list of users in each group. the various embodiments described in Figures 1-14 tracks each user's test activity and the group to which they belong thus accumulates this test activity and the activity from other users for reporting. If a user belongs to more than one group, then the group activity will be skewed. For this reason, you should not use the default groups created for each QC project since typically the users will be assigned to several of the default groups. For example, almost all users are part of the 'Viewer' group.

Concerning the use of groups, there can be situations where a QC user was removed from the project or the site but had performed testing and their testing efforts will be shown on the Test Trend as in the example below:

Here, both 'alex_qc' and 'shelly_qc' have performed testing but have been removed from the system; when you view the Testing Resource you see this for the time interval analyzed:

Note the red label showing "Testing by deleted user(s) and unmonitored group(s) detected". This is an indication that none of the monitored groups are showing any testing activity but there was testing occurring over the time interval. In some cases
this message might read "Testing by deleted users(s) detected" or "Testing by unmonitored group(s) detected" as a function of how the deletion(s) were done in QC.

[00322] However, there are cases where, if the time interval is large, that you'll see testing results as well as the warning message such as below:

[00323] Here the testers 'alice_qc', 'cecil_qc' and 'michael_qc' who are in the 'QA Tester' group, have their contributions shown but the two deleted users, 'alex_qc' and 'shelly_qc' while having done testing, are not shown in the group trend page. For reference, 'alex_qc' was in the 'Viewer' group and 'shelly_qc' in the 'Defect Reporter', 'QA Manager' and 'R & D Manager' groups; none of groups that 'shelly_qc' was in are in the monitored list.

[00324] Step 7: Selecting Test Sets to Monitor

[00325] When you drop down the 'Defined Test Sets' list control...

[00326] ...you'll see all the parent (top-level) test set folder names from the QC Test Lab for the project. In this example, the built-in QC demo project is shown.

[00327] You can select all the test sets at one time if you desire by selecting the entry in braces called '{Select All Test Sets in Test Lab}' but typically, if your the various embodiments described in Figures 1-14 project is a subset of the overall QC project, then you'll only select one or two top level folders. Below is the result of selecting only the 'Mercury Tours Web Site' parent folder.

[00328] Note the two buttons on top of the list to the right:

[00329] The button with the checkmark will select all the test sets and the one with the minus sign will deselect all the test sets. You can of course select individual test sets manually. The screen shot below shows the results of having pressed the 'checkmark' button to select all the test sets in on top-level parent folder.

[00330] Next you add this selection to the 'Monitored Test Sets' list by pressing the 'Add to Monitored List' button...

[00331] ...and then, if you're all done selecting test sets, press the 'Save Selection' button.

[00332] You'll note that the font color on the list has now changed from red to black as an indication that the selection has been saved.

[00333] To change the selection, use the 'Clear Selection' button to the lower left of the 'Monitored Test Sets' list. If you try to add the same test set twice, data validation code will prevent this and a message in the 'Messages' text
[00334] box will appear reminding you of this fact. You can, of course, continue to add
test sets from different parent folders.
[00335] If you revisit the screen after having made selections, it will look like this:
[00336] You'll also note on the main Configuration Editor screen a list of the selected
test sets so you can determine if changes are needed to the test set selection.
[00337] At this point, you're done with the basic configuration for a project. You can
view the settings by selecting 'Close' from the Configuration Editor screen. This is shown
in the next section.
[00338] Or, if you switch to the main Dashboard that's open and select the refresh icon
in the lower left hand corner of the page, you're newly configured project should pull
back data from QC and populate the page.
[00339] Also, if you'd like to take advantage of the optional feature of organizing your
projects into meaningful business-oriented names, see the section on Project Groups
[00340] The Configuration Settings Viewer Page
[00341] This is viewable by all users of the various embodiments described in
Figures 1-14. You can also export all the configuration settings to Excel by using the
small symbol in the lower right hand corner of the page.
[00342] The message (here) is to remind users to notify you, the product administrator,
if they are aware of any QC changes that may not be reflected in the current the various
embodiments described in Figures 1-14 configuration. The most common out-of-sync
conditions are when QC test sets are renamed, deleted or added. Since the various
embodiments described in Figures 1-14 reads the QC test lab hierarchical test set tree,
this can happen. In some cases, the various embodiments described in Figures 1-14
may continue to report accurate data but
[00343] often you'll notice a change in the project's display because test sets that have
been monitored no longer exist or exist under a different name.
[00344] To rectify any out-of-sync conditions for a selected project, revisit the
Monitored Test Sets screen (30) and clear and reselect the test sets to monitor.
[00345] At this point in learning how to configure a the various embodiments
described in Figures 1-14 project you're probably ready to see some actual test
management data. If you switch to the open browser window that contains the main
Dashboard page, select the 'Refresh Source Data' icon in the lower left corner of the page to force the various embodiments described in Figures 1-14 to re-read the configuration settings and it will display the current status of your project.

[00346] There is a bit more to learn about configuring the various embodiments described in Figures 1-14, for example the use of Project Groupings that is described below. For now, we've covered the basics of project configuration but read on for more ways to improve your management of testing projects.

[00347] Project Groups

[00348] When you are managing a large number of projects, it can be helpful to group related projects together into what the various embodiments described in Figures 1-14 calls a 'Project Group'. These are accessed via the drop-down control at the top of all the main application pages.

[00349] The 'built-in' group is named 'All Projects' and initially this group is set as the default, meaning that all active projects will be shown. You can add your own project groups and adjust the default group by bringing up the 'Project Group' screen from the configuration editor.

[00350] Your Project Groups may look different from the one below as a function of how many projects you have configured and you may see only the 'All Projects' group listed, but the screenshot below shows a more fully configured list.

[00351] Note that when the system-controlled 'All Projects' list is selected that controls are disabled, indicating the fact that you can't rename or delete this group nor can you save any changes to the selected project list.

[00352] As you change the selected group in the 'Project Groups' list box on the left hand side of the page, you'll see a different set of projects selected in the 'Active Projects' list on the right.

[00353] Compare this view above with that on the previous page. Note also that the checkbox called 'Default' is now enabled. If you want to make the 'Claims' group the default view then check this checkbox and note the change in the screen:

[00354] You'll also see a message at the bottom of the screen showing the action that was taken.
Since only one 'Project Group' can be the default, then the checkbox is again disabled. Selecting another group name will re-enable the checkbox and then the newly selected group can be set to default. After a change, the reloads the data, always placing the default group as the first in the list of project groups.

You can rename, delete and add new groups using the same techniques as described for project name maintenance in step 4 that began on 10. Note that the maximum length for a project group name is 20 characters and validation code ensures that each group name is unique.

Although the present embodiments have been described with reference to specific example embodiments, it will be evident that various modifications and changes may be made to these embodiments without departing from the broader spirit and scope of the various embodiments. Accordingly, the specification and drawings are to be regarded in an illustrative rather than a restrictive sense.
CLAIMS
What is claimed:
1. A method of a client device comprising:
   determining a forecasted completion date of a communication of a quality assurance testing data from a quality center database to a server device comprising a processor to analyze based on an algorithm the quality assurance testing data;
   creating a quality assurance project status of a communication of a baseline complete date to the server device to analyze the baseline complete date and the forecasted completion date; and
   determining the quality assurance project status comprising a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the quality center database and the forecasted completion date of the server device to assist a user to manage a quality assurance testing project.

2. The method of claim 1 further comprising:
   creating the quality assurance project status of a communication of a baseline start date to the server device to analyze the baseline start date, the baseline complete date, and the forecasted completion date.

3. The method of claim 1 wherein:
   the quality assurance testing data of the quality center database is analyzed based on the algorithm of the server device to determine the forecasted completion date comprises a number of total tests, a number of passed tests, a first test date, and a last test date.

4. The method of claim 1 further comprising:
determining a test execution trend comprising a number of passed tests, a number of failed tests, and a number of not complete tests based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

5. The method of claim 1 further comprising:

determining a group resource execution rate comprising a number of tests executed by an offshore quality assurance team and a number of tests executed by an onshore quality assurance team based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

6. The method of claim 1 further comprising:

determining an individual resource execution rate comprising a number of tests executed by a quality assurance individual based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

7. The method of claim 1 further comprising:

determining a defect trend comprising a linked defect, an unlinked defect, and a severity of the linked defect based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.

8. A method of a server device comprising:
generating a forecasted completion date based on an analysis performed by a processor of applying an algorithm of a quality assurance testing data comprising a number of total tests, a number of passed tests, a first test date, and a last test date of a client device;
producing a quality assurance project status of a comparison of a difference in time between a baseline complete date of the client device and the forecasted completion date; and
analyzing the quality assurance project status comprising a variance in a time from the baseline complete date from an examination of the quality assurance testing data from the client device and the forecasted completion date to assist a user to manage a quality assurance testing project.

9. The method of claim 8 wherein the algorithm comprises:

   calculating a number of tests left to pass from a multiplication of a project goal percentage and the number of total tests and subtracting the number of passed tests;
   calculating an average passed rate by dividing the number of total tests by the difference in work days between the first test date and the last test date;
   calculating a number of work days needed by dividing the number of tests left to pass by the average passed rate; and
   generating the forecasted completion date by adding the number of work days needed to a current date.

10. The method of claim 8 further comprising:

   producing a quality assurance project status of a comparison of the difference in time between a baseline start date and the baseline complete date of the client device and the forecasted completion date.

11. The method of claim 8 further comprising:
producing a quality assurance project status of an analysis of a project goal percentage of the client device to analyze a number of tests left to pass and to analyze the number of tests left to pass and the forecasted completion date to generate the quality assurance project status.

12. The method of claim 8 further comprising:

genrating an actual versus expected test execution rate from a comparison of an actual execution rate from an analysis of the quality assurance testing data and an expected execution rate of the client device to assist the user to manage the quality assurance testing project.

13. The method of claim 8 further comprising:

regenerating the forecasted completion date from the quality assurance testing data comprising the number of total tests, the number of passed tests, the first test date, and the last test date and from a number of quality assurance testers and a number of work hours in a day of the user of the client device to adjust the forecasted completion date to assist the user to manage a quality assurance testing project.

14. The method of claim 8 further comprising:

genrating a number of quality assurance testers from the quality assurance testing data comprising the number of total tests, the number of passed tests, the first test date, and the last test date and from the baseline end date and a number of work hours in a day of the user of the client device assist the user to manage the quality assurance testing project.

15. A system comprising:

a client device to communicate a quality assurance testing data from a quality center database and a baseline complete date from the client device to a
server device to analyze the quality assurance testing data to calculate a
forecasted completion date and a quality assurance project status;
a server device to generate the forecasted completion date from the quality
assurance testing data comprising a number of total tests, a number of
passed tests, a first test date, and a last test date from the client device and
the quality assurance project status by comparing a difference in time
between the baseline complete date and the forecasted completion date;
and
a plurality of displays associated with a plurality of client devices to render the
quality assurance project status to a plurality of users in a plurality of
analytics dashboards rendered in each of the displays and to render the
quality assurance project status comprising a variance in a time from the
baseline complete date from an analysis of the quality assurance testing
data from the client device and the forecasted completion date from the
server device to the client device to assist a user of the client device to
manage a quality assurance testing project.

16. The system of claim 15 further comprising:

the server device to:
calculate a number of tests left to pass from a multiplication of a project
goal percentage and the number of total tests and subtracting the
number of passed tests;
calculate an average passed rate by dividing the number of total tests by
the difference in work days between the first test date and the last
test date;
calculate a number of work days needed by dividing the number of tests
left to pass by the average passed rate; and
generate the forecasted completion date by adding the number of work
days needed to a current date.

17. The system of claim 15 further comprising:
the server device producing a quality assurance project status of a comparison of the difference in time between a baseline start date and the baseline complete date of the client device and the forecasted completion date to assist a user of the client device to manage a quality assurance testing project.

18. The system of claim 15 further comprising:

the server device to regenerate the forecasted completion date from the quality assurance testing data comprising the number of total tests, the number of passed tests, the first test date, and the last test date and from a number of quality assurance testers and a number of work hours in a day communicated to the server device from the user of the client device to adjust the forecasted completion date to assist the user of the client device to manage a quality assurance testing project.

19. The system of claim 15 wherein:

the server device to generate a test execution trend comprising a number of passed tests, a number of failed tests, and a number of not complete tests based on a time from an analysis of the quality assurance testing data from an analysis of the quality assurance testing data a quality center database to assist the user of the client device to manage a quality assurance testing project.

20. The system of claim 15 further comprising:

the server device to generate a actual versus expected test execution rate from a comparison of a actual execution rate from a calculation of the quality assurance testing data from the client device communicating with the quality center database and an expected execution rate from the client device communicating with the user of the client device to assist the user to manage a quality assurance testing project.
## FIGURE 3

### Table: Test Activity - Claim 2.0 All

<table>
<thead>
<tr>
<th>PROJECT</th>
<th>BASE LINE START DATE</th>
<th>FIRST TEST</th>
<th>BASE LINE COMPLETE DATE</th>
<th>LAST TEST</th>
<th>TOTAL TESTS</th>
<th>DEFECTS PENDING</th>
<th>AVG TESTING RUN/DAY</th>
<th>AVG TESTING PASSED/DAY</th>
<th>% PASSED</th>
<th>STATUS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CLAIM 2.0 UAT</td>
<td>4/7/2009</td>
<td>4/7/2009</td>
<td>6/4/2009</td>
<td>57</td>
<td>7</td>
<td>0.9554</td>
<td>0.4811</td>
<td>59.00</td>
<td>ON PROCESS</td>
<td></td>
</tr>
<tr>
<td>CLAIM 2.0 RRO</td>
<td>4/27/2009</td>
<td>4/13/2009</td>
<td>6/8/2009</td>
<td>52</td>
<td>3</td>
<td>0.7493</td>
<td>0.4340</td>
<td>85.58</td>
<td>ON PROCESS</td>
<td></td>
</tr>
</tbody>
</table>

### Pie Chart: Test Activity - Claim 2.0 All

- % PASSED: 59.00%
- % FAILED: 14.46%
- % NOT COVERED: 26.58%

### Pie Chart: Test Status - Claim 2.0 All

- % OPEN: 70.00%
- % RESOLVED: 30.00%

### Calendar: September 2009

<table>
<thead>
<tr>
<th>SUN</th>
<th>MON</th>
<th>TUE</th>
<th>WED</th>
<th>THU</th>
<th>FRI</th>
<th>SAT</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>31</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
</tr>
<tr>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
<td>1</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
</tbody>
</table>
Determine a forecasted completion date of a communication of a quality assurance testing data from a quality center database to a server device comprising a processor to analyze based on an algorithm the quality assurance testing data.

Create a quality assurance project status of a communication of a baseline complete date to a server device to analyze the baseline complete date and the forecasted completion date.

Determine the quality assurance project status comprising a variance in a time from the baseline complete date from an examination of the quality assurance testing data from a quality center database and the forecasted completion date of the server device to assist the user to manage a quality assurance testing project.

Create the quality assurance project status of a communication of a baseline start date to a server device to analyze the baseline start date, the baseline complete date, and the forecasted completion date.

Determine a test execution trend comprising a number of passed tests, a number of failed tests, and a number of not complete tests based on a time of an analysis of the server device of the quality assurance testing data of the quality center database to assist the user to manage the quality assurance testing project.
DETERMINE A GROUP RESOURCE EXECUTION RATE COMPRISING A NUMBER OF TESTS EXECUTED BY AN OFFSHORE QUALITY ASSURANCE TEAM AND A NUMBER OF TESTS EXECUTED BY AN ONSHORE QUALITY ASSURANCE TEAM BASED ON A TIME OF AN ANALYSIS OF THE SERVER DEVICE OF THE QUALITY ASSURANCE TESTING DATA OF THE QUALITY CENTER DATABASE TO ASSIST THE USER TO MANAGE THE QUALITY ASSURANCE TESTING PROJECT

DETERMINE AN INDIVIDUAL RESOURCE EXECUTION RATE COMPRISING A NUMBER OF TESTS EXECUTED BY A QUALITY ASSURANCE INDIVIDUAL BASED ON A TIME OF AN ANALYSIS OF THE SERVER DEVICE OF THE QUALITY ASSURANCE TESTING DATA OF THE QUALITY CENTER DATABASE TO ASSIST THE USER TO MANAGE THE QUALITY ASSURANCE TESTING PROJECT

DETERMINE A DEFECT TREND COMPRISING A LINKED DEFECT, AN UNLINKED DEFECT, AND A SEVERITY OF A LINKED DEFECT BASED ON A TIME OF AN ANALYSIS OF THE SERVER DEVICE OF THE QUALITY ASSURANCE TESTING DATA OF THE QUALITY CENTER DATABASE TO ASSIST THE USER TO MANAGE THE QUALITY ASSURANCE TESTING PROJECT
GENERATE A FORECASTED COMPLETION DATE BASED ON AN ANALYSIS PERFORMED BY A PROCESSOR OF APPLYING AN ALGORITHM OF THE QUALITY ASSURANCE TESTING DATA COMPRISING A NUMBER OF TOTAL TESTS, A NUMBER OF PASSED TESTS, A FIRST TEST DATE, AND A LAST TEST DATE OF THE CLIENT DEVICE

PRODUCE A QUALITY ASSURANCE PROJECT STATUS OF A COMPARISON OF THE DIFFERENCE IN TIME BETWEEN A BASELINE COMPLETE DATE OF THE CLIENT DEVICE AND THE FORECASTED COMPLETION DATE

ANALYZE THE QUALITY ASSURANCE PROJECT STATUS COMPRISING A VARIANCE IN A TIME FROM THE BASELINE COMPLETE DATE FROM AN EXAMINATION OF THE QUALITY ASSURANCE TESTING DATA FROM THE CLIENT DEVICE AND THE FORECASTED COMPLETION DATE TO ASSIST A USER TO MANAGE A QUALITY ASSURANCE TESTING PROJECT


PRODUCE A QUALITY ASSURANCE PROJECT STATUS OF AN ANALYSIS OF A PROJECT GOAL PERCENTAGE OF THE CLIENT DEVICE TO ANALYZE A NUMBER OF TESTS LEFT TO PASS AND TO ANALYZE THE NUMBER OF TESTS LEFT TO PASS AND THE FORECASTED COMPLETION DATE TO GENERATE THE QUALITY ASSURANCE PROJECT STATUS

FIGURE 8A  9/14
GENERATE A ACTUAL VERSUS EXPECTED TEST EXECUTION RATE FROM A COMPARISON OF A
ACTUAL EXECUTION RATE FROM AN ANALYSIS OF THE QUALITY ASSURANCE TESTING DATA AND
AN EXPECTED EXECUTION RATE OF THE CLIENT DEVICE TO ASSIST THE USER TO MANAGE THE
QUALITY ASSURANCE TESTING PROJECT

REGENERATE THE FORECASTED COMPLETION DATE FROM THE QUALITY ASSURANCE TESTING DATA
COMPRISING THE NUMBER OF TOTAL TESTS, THE NUMBER OF PASSED TESTS, THE FIRST TEST DATE,
AND THE LAST TEST DATE AND FROM A NUMBER OF QUALITY ASSURANCE TESTERS AND A NUMBER
OF WORK HOURS IN A DAY OF THE USER OF THE CLIENT DEVICE TO ADJUST THE FORECASTED
COMPLETION DATE TO ASSIST THE USER TO MANAGE A QUALITY ASSURANCE TESTING PROJECT

GENERATE A NUMBER OF QUALITY ASSURANCE TESTERS FROM THE QUALITY ASSURANCE TESTING DATA
COMPRISING THE NUMBER OF TOTAL TESTS, THE NUMBER OF PASSED TESTS, THE FIRST TEST
DATE, AND THE LAST TEST DATE AND FROM THE BASELINE END DATE AND A NUMBER OF WORK
HOURS IN A DAY OF THE USER OF THE CLIENT DEVICE ASSIST THE USER TO MANAGE THE QUALITY
ASSURANCE TESTING PROJECT
No Active Projects

No Active Projects

No Active Projects

No Active Projects

Figure 11
Configuration Viewer: Allows Users to View
- Project Configuration
- Test Sets Monitored
- User Groups Monitored

Configuration Editor: Allows Adjustment of...
- Project Management Variables
- Linked or Unlinked Defect Monitoring
- Use of User Groups for Test Trend Analysis
- Access to Other Configuration Pages

Advanced Settings: Allows Management of...
- Projects (renaming/deleting/creation)
- Project Database Connections
- Project List Method and Values: Default or Custom

Monitored Test Sets: Allows Management of...
- Test Sets monitored by selected Foresight projects
(These are pulled from the QC Test Lab)

Project Groups: Allows Management of...
- Groups (renaming/deleting/creation)
- Selection of Projects belonging to a Group

Figure 12
CURRENT PROJECTS

PROJECT MAINTENANCE
Selected Project: (Inactive)
Project 1

DATABASE CONNECTION
Data Source/Server:

Add a New Project:

SAVE CONNECTION

SELECTED PROJECT'S LIST METHOD AND VALUES:

--- Mapped Test Status Values ---
Passed: Passed
Failed: Failed
Not Completed: Not Completed
No Run: No Run
N/A: N/A
Custom 1:
Custom 2:

--- Mapped Defect Status Values ---
Defect closed: Closed

Advanced Settings Messages:

Pending Defect Statuses Monitored:

Pending Defect Sevities Monitored:

Figure 14