FY16 Test Standard

Monday, August 31, 2015 10:57 AM

Who, What, Why - draft 1

Sunday, February 07, 2016 7:04 AM

QA Test Standards and the Vision for the Future (draft - need feedback, then drive)

Every Functionality Test describes the "Who, What, Why, and How" of a given feature's business objective.

QA's tests today provide detailed and explicit functionality instructions that are directly and tightly coupled with application implementation – we'll call this the "How". This approach has historically been necessary in order to provide Manual Testers with enough information in order to navigate each application, and in order to exercise very specific logic or data flows. This was required for two key and valid reasons:

- Application implementations had historically been very inconsistent and often unintuitive. It is noteworthy that this not only hinders QA Testers from effectively executing tests, but it also frustrated Client end users equally.
- Application UI was the only way to exercise certain "under the hood" logic or flows, as the architecture and tools provided no easy way to test integrations between components of a given system, nor were these integration points well known to QA.

While this approach has served QA well for years, it requires very frequent updates to Test Inventory as UI implementation changes regularly even. Further, the "under the hood" technologies are not fully exercised, increasing the risk of missed integration-related defects.

Over the past few years, QA has introduced and evolved Test Inventory Standards, which has begun to improve on test reliability and consistency. In addition, FactSet's migration toward standardized Web technologies, distributed services and common UI libraries (e.g. Thief) is making our applications significantly more intuitive and consistently implemented across the entire spectrum. This is of course great news for our Clients, but it also finally makes it possible to elevate the perspective of QA's Tests to focus on "Business" – the actual work flow that Client end users are interested in. These new technologies also demand greater speed from QA – both in Test Documentation and in its Execution – because the technology standards increase the speed at which Engineering introduces changes that must be tested.

So – QA must continue its evolution toward agility: speed, efficiency, and effectiveness. The current Test Inventory reliance on explicit instructions coupled tightly with UI implementation must be eliminated and replaced with the philosophy of validating the business user's work flow. We'll call this flow the "Who, What, and Why", where the "Who" is the user of the application, the "What" is the user's task, and the "Why" is the value the user perceives when the task is accomplished. Any Manual Tester should be able intuitively navigate the actual UI implementation (the "How") in order to achieve the "Who, What, Why" even when the "How" changes! If a Tester cannot, then one of 3 things has occurred:

- An actual software Bug in the implementation
- A "usability" defect, in that the implementation is no longer intuitive
- A "training opportunity" – where the Tester or documentation must get up to speed. Ideally, this only need be done once!

This is most obvious to apply to an automated test, where the "How" is managed in the automated script, and its failure can be easily associated with a specific implementation defect. Using the Cucumber toolset for Behavior Driven Development (BDD), QA Analysts manage the "Who, What, Why" flows in one file (called the Feature File), where each of its instructions then executes the corresponding automated script tied to each instruction (the "How"). In the case of a test not yet automated, the Manual Tester should easily be able to execute the flow defined in the Feature File. If the feature file calls for a SIT test then it must exist or has to be written.

The beauty of this is how neatly it is tied to Agile processes, where a "User Story" is essentially the feature specs describing "Who, What, Why". So when QA participates in these feature discussions, and Acceptance Tests are discussed for the User Story, the actual BDD-based tests are DONE during the meeting. Very little Analyst test creation is needed afterwards, and the next step is accomplished by the Test Engineer (to automate the "How" in script) and the App Engineer (to build the feature itself and pass the automated test). Even if the test isn't being automated yet, the Analyst is

DONE writing it.

Does this happen all at once to the entire Test Inventory for all applications across the entire spectrum? No. Will this be easy? No. **But we should let it be FUN AND REWARDING.**

BDD is being rolled out on some projects via automation first, and learning will occur as this progresses. In the meantime, we will need to provide Feature File guidance for both Analysts and Manual Testers alike to foster a common interpretation. Once a test is automated, we do not expect manual execution anymore, nor do we expect Manual Testers to review automation failures and re-execute steps manually, as that is a waste of their time. We do however expect that Tests written via Cucumber format in Feature Files will be executed by Manual Testers until automation can be established. This means that going forward Analysts should endeavor to write tests only in Cucumber format.

To make this viable for Manual Testers, expansion of our Test Inventory Standards needs to occur with more documentation on common "How" execution steps needed. For example, in 2014 the Standard added a common rule for launching an application (via the @ shortcut keyword), which eliminated the need for any Test Case to include that instruction explicitly! The list of these standard actions must expand – perhaps the best resource for implementing these standards is the living Thief library itself, e.g. http://thief.factset.com/versions/3.4.0/docs/widgets/. "Who, What, Why" instructions in a test case should be able to easily refer to a given Widget, leveraging its documented behaviors. If a Manual Tester questions the behavior a simple lookup to the documentation can answer the question.

"Under the hood" testing is now a viable reality as well with the advent of the System Integration Test (SIT) automation tool provided by Information Systems. This tool allows web-service calls at the integration layers to be exercised for schema validity, data validity, response time, etc. No longer will UI be the only way to test whether integrated systems are connected properly, responding correctly, etc. This will reduce the reliance on UI tests.

More documentation on Cucumber, BDD, and its syntax will be needed. QAI and Tellus will need to be enhanced to support this.

These are the next steps to make the above Vision turn into reality, and it will certainly be a journey rather than a cutover.

Draft Ken

Monday, February 08, 2016 1:49 PM

QA Test Standards and the Vision for the Future (draft - need feedback, then drive)

Every Functionality Test describes the "Who, What, Why, and How" of a given feature's business objective.

QA's tests today provide detailed and explicit functionality instructions that are directly and tightly coupled with application implementation – we'll call this the "How". This approach has historically been necessary in order to provide Manual Testers with enough information in order to navigate each application, and in order to exercise very specific logic or data flows. This was required for two key and valid reasons:

- Application implementations had historically been very inconsistent and often unintuitive. It is noteworthy that this not only hinders QA Testers from effectively executing tests, but it also frustrated Client end users equally.
- Application UI was the only way to exercise certain "under the hood" logic or flows, as the architecture and tools provided no easy way to test integrations between components of a given system, nor were these integration points well known to QA.

While this approach has served QA well for years, it requires very frequent updates to Test Inventory as UI implementation changes regularly even. Further, the "under the hood" technologies are not fully exercised, increasing the risk of missed integration-related defects.

Over the past few years, QA has introduced and evolved Test Inventory Standards, which has begun to improve on test reliability and consistency. In addition, FactSet's migration toward standardized Web technologies, distributed services and common UI libraries (e.g. Thief) is making our applications significantly more intuitive and consistently implemented across the entire spectrum. This is of course great news for our Clients, but it also finally makes it possible to elevate the perspective of QA's Tests to focus on "Business" – the actual work flow that Client end users are interested in. These new technologies also demand greater speed from QA – both in Test Documentation and in its Execution – because the technology standards increase the speed at which Engineering introduces changes that must be tested.

So – QA must continue its evolution toward agility: speed, efficiency, and effectiveness. The current Test Inventory reliance on explicit instructions coupled tightly with UI implementation must be eliminated and replaced with the philosophy of validating the business user's work flow. We'll call this flow the "Who, What, and Why", where the "Who" is the user of the application, the "What" is the user's task, and the "Why" is the value the user perceives when the task is accomplished. Any Manual Tester should be able intuitively navigate the actual UI implementation (the "How") in order to achieve the "Who, What, Why" even when the "How" changes! If a Tester cannot, then one of 3 things has occurred:

- An actual software Bug in the implementation
- A "usability" defect, in that the implementation is no longer intuitive
- A "training opportunity" – where the Tester or documentation must get up to speed. Ideally, this only need be done once!

This is most obvious to apply to an automated test, where the "How" is managed in the automated script, and its failure can be easily associated with a specific implementation defect. Using the Cucumber toolset for Behavior Driven Development (BDD), QA Analysts manage the "Who, What, Why" flows in one file (called the Feature File), where each of its instructions then executes the corresponding automated script tied to each instruction (the "How"). In the case of a test not yet automated, the Manual Tester should easily be able to execute the flow defined in the Feature File. If the feature file calls for a SIT test then it must exist or has to be written.

The beauty of this is how neatly it is tied to Agile processes, where a "User Story" is essentially the feature specs describing "Who, What, Why". So when QA participates in these feature discussions, and Acceptance Tests are discussed for the User Story, the actual BDD-based tests are DONE during the meeting. Very little Analyst test creation is needed afterwards, and the next step is accomplished by the Test Engineer (to automate the "How" in script) and the App Engineer (to build the feature itself and pass the automated test). Even if the test isn't being automated yet, the Analyst is DONE writing it.

Does this happen all at once to the entire Test Inventory for all applications across the entire spectrum? No, this will be a phased in process based on Products. Will this be easy? No. **But we should let it be FUN AND REWARDING.** We expect an initial learning curve, but as folks gain experience they can mentor less experienced ones.

BDD is being rolled out on some projects via automation first, and learning will occur as this progresses. In the meantime, we will need to provide Feature File guidance for both Analysts and Manual Testers alike to foster a common interpretation. Once a test is automated, we do not expect manual execution anymore, nor do we expect Manual Testers to review automation failures and re-execute steps manually, as that is a waste of their time. We do however expect that Tests written via Cucumber format in Feature Files will be executed by Manual Testers until automation can be established. This means that going forward Analysts should endeavor to write tests only in Cucumber format.

To make this viable for Manual Testers, expansion of our Test Inventory Standards needs to occur with more documentation on common "How" execution steps needed. For example, in 2014 the Standard added a common rule for launching an application (via the @ shortcut keyword), which eliminated the need for any Test Case to include that instruction explicitly! The list of these standard actions must expand – perhaps the best resource for implementing these standards is the living Thief library itself, e.g. http://thief.factset.com/versions/3.4.0/docs/widgets/. "Who, What, Why" instructions in a test case should be able to easily refer to a given Widget, leveraging its documented behaviors. If a Manual Tester questions the behavior a simple lookup to the documentation can answer the question.

“Under the hood” testing is now a viable reality as well with the advent of the System Integration Test (SIT) automation tool provided by Information Systems. This tool allows web-service calls at the integration layers to be exercised for schema validity, data validity, response time, etc. No longer will UI be the only way to test whether integrated systems are connected properly, responding correctly, etc. This will reduce the reliance on UI tests to determine if backend services are available and reliable and need not be included in the functional test.

More documentation on Cucumber, BDD, and its syntax will be needed. QAI and Tellus will need to be enhanced to support this. QAI should be more feature file heavy, while Tellus would add more information abut the underlying code generation and debug.

These are the next steps to make the above Vision turn into reality, and it will certainly be a journey rather than a cutover.

Draft Kristin (from Ken's)

Thursday, February 11, 2016 2:16 PM

QA Test Standards and the Vision for the Future (draft - need feedback, then drive)

Every Functionality Test describes the "Who, What, Why, and How" of a given feature's business objective.

QA's tests today provide detailed and explicit functionality instructions that are directly and tightly coupled with application implementation – we'll call this the "How". This approach has historically been necessary in order to provide Manual Testers with enough information in order to navigate each application, and in order to exercise very specific logic or data flows. This was required for two key and valid reasons:

- Application implementations had historically been very inconsistent and often unintuitive. It is noteworthy that this not only hinders QA Testers from effectively executing tests, but it also frustrated Client end users equally.
- Application UI was the only way to exercise certain "under the hood" logic or flows, as the architecture and tools provided no easy way to test integrations between components of a given system, nor were these integration points well known to QA.

While this approach has served QA well for years, it requires very frequent updates to Test Inventory as UI implementation changes regularly even. Further, the "under the hood" technologies are not fully exercised, increasing the risk of missed integration-related defects.

Over the past few years, QA has introduced and evolved Test Inventory Standards, which has begun to improve on test reliability and consistency. In addition, FactSet's migration toward standardized Web technologies, distributed services and common UI libraries (e.g. Thief) is making our applications significantly more intuitive and consistently implemented across the entire spectrum. This is of course great news for our Clients, but it also finally makes it possible to elevate the perspective of QA's Tests to focus on "Business" – the actual work flow that Client end users are interested in. These new technologies also demand greater speed from QA – both in Test Documentation and in its Execution – because the technology standards increase the speed at which Engineering introduces changes that must be tested.

So – QA must continue its evolution toward agility: speed, efficiency, and effectiveness. The current Test Inventory reliance on explicit instructions coupled tightly with UI implementation must be eliminated and replaced with the philosophy of validating the business user's work flow. We'll call this flow the "Who, What, and Why", where the "Who" is the user of the application, the "What" is the user's task, and the "Why" is the value the user perceives when the task is accomplished. Any Manual Tester should be able intuitively navigate the actual UI implementation (the "How") in order to achieve the "Who, What, Why" even when the "How" changes! If a Tester cannot, then one of 3 things has occurred:

- An actual software Bug in the implementation
- A "usability" defect, in that the implementation is no longer intuitive
- A "training opportunity" – where the Tester or documentation must get up to speed. Ideally, this only need be done once!

This is most obvious to apply to an automated test, where the "How" is managed in the automated script, and its failure can be easily associated with a specific implementation defect.

Using the Cucumber toolset for Behavior Driven Development (BDD), QA Analysts manage the "Who, What, Why" flows in one file (called the Feature File), where each of its instructions then executes the corresponding automated script tied to each instruction (the "How"). In the case of a test not yet automated, the Manual Tester should easily be able to execute the flow defined in the Feature File. If the feature file calls for a SIT test then it must exist or has to be written.

The beauty of this is how neatly it is tied to Agile processes, where a "User Story" is essentially the feature specs describing "Who, What, Why". So when QA participates in these feature discussions, and Acceptance Tests are discussed for the User Story, the actual BDD-based tests are DONE during the meeting. Very little Analyst test creation is needed afterwards, and the next step is accomplished by the Test Engineer (to automate the "How" in script) and the App Engineer (to build the feature itself and pass the automated test). Even if the test isn't being automated yet, the Analyst is DONE writing it.

Does this happen all at once to the entire Test Inventory for all applications across the entire spectrum? No, this will be a phased in process based on Products. Will this be easy? No. **But we should let it be FUN AND REWARDING.** We expect an initial learning curve, but as folks gain experience they can mentor less experienced ones.

BDD is being rolled out on some projects via automation first, and learning will occur as this progresses. In the meantime, we will need to provide Feature File guidance for both Analysts and Manual Testers alike to foster a common interpretation. Once a test is automated, we do not expect manual execution anymore, nor do we expect Manual Testers to review automation failures and re-execute steps manually, as that is a waste of their time. We do however expect that Tests written via Cucumber format in Feature Files will be executed by Manual Testers until automation can be established. This means that going forward Analysts should endeavor to write tests only in Cucumber format.

To make this viable for Manual Testers, expansion of our Test Inventory Standards needs to occur with more documentation on common "How" execution steps needed. For example, in 2014 the Standard added a common rule for launching an application (via the @ shortcut keyword), which eliminated the need for any Test Case to include that instruction explicitly! The list of these standard actions must expand – perhaps the best resource for implementing these standards is the living Thief library itself, e.g. http://thief.factset.com/versions/3.4.0/docs/widgets/. "Who, What, Why" instructions in a test case should be able to easily refer to a given Widget, leveraging its documented behaviors. If a Manual Tester questions the behavior a simple lookup to the documentation can answer the question.

Question: In the meantime, as we move forward with robust Test Inventory Standard documentation, implement the appropriate changes based on those standards and get manual testers up to speed on self addressing questions by using resources available such as the thief documentation, what can we do to move BDD forward? One way this is being done right now is by adding a Feature file QAI in prepartion for automation while the the legacy way of having cases and steps still exists for the manual run.

We also should consider here changes that need to be made to QA tools such as QAI. The testing application is set up to run test steps and cucumber files are associated at the test case level. Changes will need to be made to allow a plan to be executed in cucumber mode as fare as the testing application display as well as sending results to a SWP from tellus.

"Under the hood" testing is now a viable reality as well with the advent of the System Integration Test (SIT) automation tool provided by Information Systems. This tool allows web-service calls at the integration layers to be exercised for schema validity, data validity, response time, etc. No longer will UI be the only way to test whether integrated systems are connected properly,

responding correctly, etc. This will reduce the reliance on UI tests to determine if backend services are available and reliable and need not be included in the functional test.
Question: I'm not clear on why we get into SIT in this document? This seems like just details on the **How** for automated tests. My feeling that there is enough to cover/figure out with BDD that it is not necessary to get into SIT.

More documentation on Cucumber, BDD, and its syntax will be needed. QAI and Tellus will need to be enhanced to support this. QAI should be more feature file heavy, while Tellus would add more information abut the underlying code generation and debug.

These are the next steps to make the above Vision turn into reality, and it will certainly be a journey rather than a cutover.

AB- Draft (from Kristin's)

Wednesday, February 17, 2016 1:58 PM

QA Test Standards and the Vision for the Future (draft - need feedback, then drive)

Every Functionality Test describes the "Who, What, Why, and How" of a given feature's business objective.

QA's tests today provide detailed and explicit functionality instructions that are directly and tightly coupled with application implementation – we'll call this the "How". This approach has historically been necessary in order to provide Manual Testers with enough information in order to navigate each application, and in order to exercise very specific logic or data flows. This was required for two key and valid reasons:

- Application implementations had historically been very inconsistent and often unintuitive. It is noteworthy that this not only hinders QA Testers from effectively executing tests, but it also frustrated Client end users equally.
- Application UI was the only way to exercise certain "under the hood" logic or flows, as the architecture and tools provided no easy way to test integrations between components of a given system, nor were these integration points well known to QA.

While this approach has served QA well for years, it requires very frequent updates to Test Inventory as UI implementation changes regularly even. Further, the "under the hood" technologies are not fully exercised, increasing the risk of missed integration-related defects.

Over the past few years, QA has introduced and evolved Test Inventory Standards, which has begun to improve on test reliability and consistency. In addition, FactSet's migration toward standardized Web technologies, distributed services and common UI libraries (e.g. Thief) is making our applications significantly more intuitive and consistently implemented across the entire spectrum. This is of course great news for our Clients, but it also finally makes it possible to elevate the perspective of QA's Tests to focus on "Business" – the actual work flow that Client end users are interested in. These new technologies also demand greater speed from QA – both in Test Documentation and in its Execution – because the technology standards increase the speed at which Engineering introduces changes that must be tested.

So – QA must continue its evolution toward agility: speed, efficiency, and effectiveness. The current Test Inventory reliance on explicit instructions coupled tightly with UI implementation must be eliminated and replaced with the philosophy of validating the business user's work flow. We'll call this flow the "Who, What, and Why", where the "Who" is the user of the application, the "What" is the user's task, and the "Why" is the value the user perceives when the task is accomplished. Any Manual Tester should be able intuitively navigate , new applications that have had Design as a part of development, the actual UI implementation (the "How") in order to achieve the "Who, What, Why" even when the "How" changes! If a Tester cannot, then one of 3 things has occurred:

- An actual software Bug in the implementation
- A "usability" defect, in that the implementation is no longer intuitive
- A "training opportunity" – where the Tester or documentation must get up to speed. Ideally, this only need be done once!

This is most obvious to apply to an automated test, where the "How" is managed in the

automated script, and its failure can be easily associated with a specific implementation defect. Using the Cucumber toolset for Behavior Driven Development (BDD), QA Analysts manage the "Who, What, Why" flows in one file (called the Feature File), where each of its instructions then executes the corresponding automated script tied to each instruction (the "How"). In the case of a test not yet automated, the Manual Tester should easily be able to execute the flow defined in the Feature File. If the feature file calls for a SIT test then it must exist or has to be written.

The beauty of this is how neatly it is tied to Agile processes, where a "User Story" is essentially the feature specs describing "Who, What, Why". So when QA participates in these feature discussions, and Acceptance Tests are discussed for the User Story, the actual BDD-based tests are DONE during the meeting. Very little Analyst test creation is needed afterwards, and the next step is accomplished by the Test Engineer (to automate the "How" in script) and the App Engineer (to build the feature itself and pass the automated test). Even if the test isn't being automated yet, the Analyst is DONE writing it.

Does this happen all at once to the entire Test Inventory for all applications across the entire spectrum? No, this will be a phased in process based on Products. Will this be easy? No. **But we should let it be FUN AND REWARDING.** We expect an initial learning curve, but as folks gain experience they can mentor less experienced ones. Documentation on best practices from people's experiences should be maintained and in an organized fashion. Part of the Analyst Development program.

BDD is being rolled out on some projects via automation first, and learning will occur as this progresses. In the meantime, we will need to provide Feature File guidance for both Analysts and Manual Testers alike to foster a common interpretation. Once a test is automated, tests should be stable and reliable enough to file RPDs automatically, we do not expect manual execution anymore, nor do we expect Manual Testers to review automation failures and re-execute steps manually, as that is a waste of their time. Only tests going straight to automation should be written in Cucumber at first. We do however expect that Tests written via Cucumber format in Feature Files will be executed by Manual Testers until automation can be established. This means that going forward Analysts should endeavor to write tests only in Cucumber format.

To make this viable for Manual Testers, expansion of our Test Inventory Standards needs to occur with more documentation on common "How" execution steps needed. For example, in 2014 the Standard added a common rule for launching an application (via the @ shortcut keyword), which eliminated the need for any Test Case to include that instruction explicitly! The list of these standard actions must expand – perhaps the best resource for implementing these standards is the living Thief library itself, e.g. http://thief.factset.com/versions/3.4.0/docs/widgets/. "Who, What, Why" instructions in a test case should be able to easily refer to a given Widget, leveraging its documented behaviors. If a Manual Tester questions the behavior a simple lookup to the documentation can answer the question.

Question: In the meantime, as we move forward with robust Test Inventory Standard documentation, implement the appropriate changes based on those standards and get manual testers up to speed on self addressing questions by using resources available such as the thief documentation, what can we do to move BDD forward? One way this is being done right now is by adding a Feature file QAI in preparation for automation while the legacy way of having cases and steps still exists for the manual run.

We also should consider here changes that need to be made to QA tools such as QAI. The testing application is set up to run test steps and cucumber files are associated at the test case level. Changes will need to be made to allow a plan to be executed in cucumber mode as fare as the testing application display as well as sending results to a SWP from tellus.Suggest we start with test cases that go directly to Automation for this reason. Agile projects fit this process the best as

a first pass.

“Under the hood” testing is now a viable reality as well with the advent of the System Integration Test (SIT) automation tool provided by Information Systems. This tool allows web-service calls at the integration layers to be exercised for schema validity, data validity, response time, etc. No longer will UI be the only way to test whether integrated systems are connected properly, responding correctly, etc. This will reduce the reliance on UI tests to determine if backend services are available and reliable and need not be included in the functional test.
Question: I'm not clear on why we get into SIT in this document? This seems like just details on the **How** for automated tests. My feeling that there is enough to cover/figure out with BDD that it is not necessary to get into SIT. Question: Would we expect SIT to change how they expect Engineers and QA to define tests to conform to BDD? Provided we want to take that on in the 1st phase as Kristin mentions.

More documentation on Cucumber, BDD, and its syntax will be needed. QAI and Tellus will need to be enhanced to support this. QAI should be more feature file heavy, while Tellus would add more information abut the underlying code generation and debug. We need to organize the current Cucumber, BDD documentation and review what may need updates.

These are the next steps to make the above Vision turn into reality, and it will certainly be a journey rather than a cutover.

Kate - Cucumber

Sunday, February 21, 2016 12:08 PM

Updated the existing QA Standards below to include BDD

- We need to decide on how we are going to implement BDD in QAI before any standards can be written
  - My updates below are largely based on questions I have regarding implementation
- Side note; what happened to "Given, When, Then"? If that is the cucumber standard, we might confuse people with "Who, What, Why, and How".

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context (in the Description field QAI, In Perforce).
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary of all included Test Cases and Steps, so that anyone in PD or SOE can understand what is being tested.
         - Here is a List Builder Report to check your compliance: Test Cases Without Descriptions
   c. **BDD Improve Editor** - Type ahead
      a. Type ahead should be per application ie. Asset Allocation
      b. Within the feature file, when writing the "when" statement, it should give suggestions based on previously used statements
   d. **BDD Improve Editor** - Have a standard for creating tables in Gherkin (Ruby vs. Protractor)
      a. It should be implemented within the Feature File Editor that is in QAI
      b. Do not allow user to add format that is not correct
   e. **BDD Improve Editor** - Automatic indentation for "and" statements
   f. **BDD** Integrate Jira/User Stories? https://try.behave.pro/
   g. Use the User Story ID as the name or how we tie the test to the user story?
   h. Pull description and RPD numbers if any from Jira
      a. Improve description area in QAI so that it can handle things like wireframes and formatting
         1) OR should we use Jira to house all of this?
         2) Does Jira keep user stories in a database? How long do we keep information in Jira?

   i. Startup and shutdown: Does this become an automated standard with Cucumber?
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Ap plications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Start Up/ Shut Down Instructions
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
            - Here is a List Builder Report to check your compliance:
               - Test Plan - Startup/Shut Down Instructions
               - Test Case - Startup/Shut Down Instructions

   j. Screenshots in Test Steps must be avoided unless absolutely necessary Only necessary for charting and error logging
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are representing.
         - *Hint 1:* Screenshots should isolate focus on particular area NOT an entire desktop.
         - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.

   k. Test Cases must cover the features/functionality - nothing more, nothing less.
      i. Use the statements below to help clarify differences and separate the testing of values versus the testing of functionality
      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   l. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
   m. **BDD standard: Using variables**
      a. When Enter "SPN:SP50.N" as the first Proxy ID
         1) Make "SPN:SP50.N" a variable so that we can do mass updates easily
         2) Also makes it reusable for other tests

      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"
         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.

   n. All Test Steps must have clear written instructions and clear, predictable written results.

   o. ~~Test Plans must NOT require judgment calls to be made by the tester.~~
      i. ~~A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.~~
      ii. ~~Avoid Steps like "pick any component" - be specific.~~

   p. Ensure Test Cases continue to flow after altering them due to new enhancements. **BDD standard:** Confirm that automated tests still run after updates.

   q. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Applica tion.
      i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
      ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained so it can be run entirely on its own. **BDD standard:** This would still be true
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
   b. When taking into account the execution of your Test Case realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
   c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
4) **BDD standard:**
   a. Naming convention
      i. Would the integration of Jira change how we name our test cases?
      ii. Ex. Would we use user story numbers along with a short name?

   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from The FactSet Product Tree).
      i. If Usage Product doesn't exist to map to, coordinate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that).
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

5) Newly developed Test Plans and Test Cases: **BDD standard:** This would still be true
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date.
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

6) Peer Reviews
   a. Analysts reviewers are expected to provide:
      i. Ensure that the test case exercises the functionality that it was written for, as noted in the Test Plan description.
      ii. Verify that tests are focusing on application specific functionality

   b. Regression reviewers should focus on section 1 rules

   c. What Triggers a Test Plan/Case Review:
      i. An application or functionality that requires a spec or a meeting to create a test plan/case
      ii. 'QA Regression Results' RPDs have increased by a noticeable amount compared to other test plans
      iii. If an Analyst requests a review of their test plan/case, a Regression team review is also required

   d. When a Test Plan isn't reviewed but changed, notification should be sent via "Regression Coverage Change Control" RPD product
      i. Example
         1) Reducing number of steps in a test plan: RPD:15237065
         2) Multiple notifications can be sent in one RPD for a specific product

7) Once the Test Case is written, you must document related RPD(s) with appropriate details.
   a. QAI Test Cases must include link(s) to the RPD where you were asked to create the Test Case.
   b. Comment in the RPD indicating status of your work, through completion, and reference the link to the QAI Test Case.

8) Test Plan Limits:
9) **BDD standard:**
   a. Test case length
      i. Each "When" is essentially one test step
      ii. Allow at least 25 "When's" per test case but we might need 35+

   b. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
      - Here is a List Builder Report to check your compliance: Test Plans w/ greater than 100 steps

   c. Each Test Case must have **no more than 25 Test Steps**.
      - Here is a List Builder Report to check your compliance: Test Cases w/ greater than 25 steps

   d. Test Plan Expected Duration must be under **60 minutes**.
      i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
         - Here is a List Builder Report to check your compliance: Test plan run time is greater than 60 minutes

   e. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
         1) Clearly this situation would necessitate a Test Case peer review
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
      iv. DO NOT try to game these limits. Test Steps must be concisely written.
      v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.

   f. Exceptions to Limits:

i. **Require QA Manager Approval before implementation**, provide supporting rationale.
ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

10) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

11) ~~The Feature Matrix must be updated with the new Test Case name and respective features.~~

    a. ~~All Feature Matrix additions need to be peer reviewed until QA Manager deems review no longer necessary (same as #11).~~
        i. ~~If the test plan is New or Active, it must have Feature Matrix mappings~~

12) Automation Script "Quality" standards: **BDD** - How does Tellus currently run BDD tests? Or what is the plan to run BDD?
    Analyst/Scripter Workflow Guidelines

    a. All Automated tests must be in QAI
        i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
        ii. These test cases must also follow the standards above.
        iii. Build dependent test cases should be noted in QAI. ? (Build Dependencies)

    b. Test Scripts must include
        i. QAI test step ID's
        ii. Readable and informative comments

    c. Naming conventions and code standards
        i. Folder structure that is scalable and understandable to you and your consumers
        ii. Consistency in naming page objects or utilities
            1) Follow the conventions of the code base being automated, come to agreement with the software engineer(s)
        iii. Test Case names should match QAI names
        iv. Use engineering code standards if they are available

    d. Reuse of code
        i. Startup /Shutdown have their own utilities which can be shared by all scripts.
        ii. Use existing Page objects/ Utilities where available

    e. Page Objects/ Utilities
        i. For repetitive functions create Page Objects and Utilities where they apply
        ii. Functionality that should be in a Page Object should not be in a script
            1) Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPAThH identification unless completely necessary.
        iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what functionality broke
        iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
            1) If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

    f. Promoting Test Cases to Production
        i. Test on your laptop and then a QA Single use VM.
        ii. Test Cases require a code review
        iii. Test must be run in the Development environment
        iv. Link to Scheduling Document

    g. Failures in the Tellus Production Environment
        i. Analysts are responsible for monitoring scripts that are assigned to them
        ii. If a test case fails 3 times as a result of script errors or environment issues the script should be removed from Production back to the Development environment.
            1) Additional manual testing time needs to be accounted for and reviewed with Regression TL prior to the start of the regression

Draft Dan - From (Alisa's)

Tuesday, February 23, 2016 8:22 AM

QA Test Standards and the Vision for the Future (draft - need feedback, then drive)

Every Functionality Test describes the "Who, What, Why, and How" of a given feature's business objective.

QA's tests today provide detailed and explicit functionality instructions that are directly and tightly coupled with application implementation – we'll call this the "How". This approach has historically been necessary in order to provide Manual Testers with enough information in order to navigate each application, and in order to exercise very specific logic or data flows. This was required for two key and valid reasons:

- Application implementations had historically been very inconsistent and often unintuitive. It is noteworthy that this not only hinders QA Testers from effectively executing tests, but it also frustrated Client end users equally.
- Application UI was the only way to exercise certain "under the hood" logic or flows, as the architecture and tools provided no easy way to test integrations between components of a given system, nor were these integration points well known to QA.

While this approach has served QA well for years, it requires very frequent updates to Test Inventory as UI implementation changes regularly even. Further, the "under the hood" technologies are not fully exercised, increasing the risk of missed integration-related defects.

Over the past few years, QA has introduced and evolved Test Inventory Standards, which has begun to improve on test reliability and consistency. In addition, FactSet's migration toward standardized Web technologies, distributed services and common UI libraries (e.g. Thief) is making our applications significantly more intuitive and consistently implemented across the entire spectrum. This is of course great news for our Clients, but it also finally makes it possible to elevate the perspective of QA's Tests to focus on "Business" – the actual work flow that Client end users are interested in. These new technologies also demand greater speed from QA – both in Test Documentation and in its Execution – because the technology standards increase the speed at which Engineering introduces changes that must be tested.

So – QA must continue its evolution toward agility: speed, efficiency, and effectiveness. The current Test Inventory reliance on explicit instructions coupled tightly with UI implementation must be eliminated and replaced with the philosophy of validating the business user's work flow. We'll call this flow the "Who, What, and Why", where the "Who" is the user of the application, the "What" is the user's task, and the "Why" is the value the user perceives when the task is accomplished. Any Manual Tester should be able intuitively navigate , new applications that have had Design as a part of development, **Dan**: I don't agree with design needing to be involved for it to be intuitive. Even for legacy apps we should be able to drive the who, what and why the actual UI implementation (the "How") in order to achieve the "Who, What, Why" even when the "How" changes! If a Tester cannot, then one of 3 things has occurred:

- An actual software Bug in the implementation
- A "usability" defect, in that the implementation is no longer intuitive
- A "training opportunity" – where the Tester or documentation must get up to speed. Ideally, this only need be done once!

This is most obvious to apply to an automated test, where the "How" is managed in the automated script,

and its failure can be easily associated with a specific implementation defect. Using the Cucumber toolset for Behavior Driven Development (BDD), QA Analysts manage the "Who, What, Why" flows in one file (called the Feature File), where each of its instructions then executes the corresponding automated script tied to each instruction (the "How"). In the case of a test not yet automated, the Manual Tester should easily be able to execute the flow defined in the Feature File. If the feature file calls for a SIT test then it must exist or has to be written.

The beauty of this is how neatly it is tied to Agile processes, where a "User Story" is essentially the feature specs describing "Who, What, Why". So when QA participates in these feature discussions, and Acceptance Tests are discussed for the User Story, the actual BDD-based tests are DONE during the meeting. Very little Analyst test creation is needed afterwards, and the next step is accomplished by the Test Engineer (to automate the "How" in script) and the App Engineer (to build the feature itself and pass the automated test). Even if the test isn't being automated yet, the Analyst is DONE writing it.

Does this happen all at once to the entire Test Inventory for all applications across the entire spectrum? No, this will be a phased in process based on Products. Will this be easy? No. **But we should let it be FUN AND REWARDING.** We expect an initial learning curve, but as folks gain experience they can mentor less experienced ones. Documentation on best practices from people's experiences should be maintained and in an organized fashion. Part of the Analyst Development program.

BDD is being rolled out on some projects via automation first, and learning will occur as this progresses. In the meantime, we will need to provide Feature File guidance for both Analysts and Manual Testers alike to foster a common interpretation. Once a test is automated, tests should be stable and reliable enough to file RPDs automatically, we do not expect manual execution anymore, nor do we expect Manual Testers to review automation failures and re-execute steps manually, as that is a waste of their time. Only tests going straight to automation should be written in Cucumber at first. **Dan**: This seems to contradict with the next sentence. Why can't we write in Cucumber format and execute manually? We do however expect that Tests written via Cucumber format in Feature Files will be executed by Manual Testers until automation can be established. This means that going forward Analysts should endeavor to write tests only in Cucumber format.

To make this viable for Manual Testers, expansion of our Test Inventory Standards needs to occur with more documentation on common "How" execution steps needed. For example, in 2014 the Standard added a common rule for launching an application (via the @ shortcut keyword), which eliminated the need for any Test Case to include that instruction explicitly! The list of these standard actions must expand – perhaps the best resource for implementing these standards is the living Thief library itself, e.g. http://thief.factset.com/versions/3.4.0/docs/widgets/. "Who, What, Why" instructions in a test case should be able to easily refer to a given Widget, leveraging its documented behaviors. If a Manual Tester questions the behavior a simple lookup to the documentation can answer the question.
Question: In the meantime, as we move forward with robust Test Inventory Standard documentation, implement the appropriate changes based on those standards and get manual testers up to speed on self addressing questions by using resources available such as the thief documentation, what can we do to move BDD forward? One way this is being done right now is by adding a Feature file QAI in preparation for automation while the legacy way of having cases and steps still exists for the manual run.

We also should consider here changes that need to be made to QA tools such as QAI. The testing application is set up to run test steps and cucumber files are associated at the test case level. Changes will need to be made to allow a plan to be executed in cucumber mode as fare as the testing application display as well as sending results to a SWP from tellus.Suggest we start with test cases that go directly to Automation for this reason. Agile projects fit this process the best as a first pass.

"Under the hood" testing is now a viable reality as well with the advent of the System Integration Test (SIT) automation tool provided by Information Systems. This tool allows web-service calls at the

integration layers to be exercised for schema validity, data validity, response time, etc. No longer will UI be the only way to test whether integrated systems are connected properly, responding correctly, etc. This will reduce the reliance on UI tests to determine if backend services are available and reliable and need not be included in the functional test.
Question: I'm not clear on why we get into SIT in this document? This seems like just details on the **How** for automated tests. My feeling that there is enough to cover/figure out with BDD that it is not necessary to get into SIT. Question: Would we expect SIT to change how they expect Engineers and QA to define tests to conform to BDD? Provided we want to take that on in the 1st phase as Kristin mentions.

More documentation on Cucumber, BDD, and its syntax will be needed. QAI and Tellus will need to be enhanced to support this. QAI should be more feature file heavy, while Tellus would add more information abut the underlying code generation and debug. We need to organize the current Cucumber, BDD documentation and review what may need updates.

Question: As part of the journey, why do we need Tellus anymore? Can this be merged with QAI and enhanced in one place rather than two and not have to build bridges between two huge systems? Seems slow, and for the end user not intuitive.

These are the next steps to make the above Vision turn into reality, and it will certainly be a journey rather than a cutover.

FY15 Test Standard

Friday, September 19, 2014 3:11 PM

QA Glossary

Friday, November 21, 2014 9:12 AM

Need definitions for QA keywords - examples:

- Test Plan
- Test Case
- Test Step
- Utility
- Page Object
- Startup/Shutdown?
- Automation Job Scheduling (CRON)
- SWP
- Automation

  Job - We use the term loosely but it can mean executing a test case or a series of test cases, or it can mean to perform a task.

  Test Complete - Is the name of the software testing IDE that we use. However, there are two versions on it. The other version is called Test Execute and it's a command line version of Test Complete the GUI version. Test Execute resides on all of the QAAutomation testing Virtual Machines and is used to execute Jscript functions.

  Jscript - Is Microsoft's version of JavaScript(trademark owned by SUN). Jscript supports conditional compilation and is the language of choice for writing automated tests for Test Complete.

  Ruby - Is the scripting language of choice for writing automated test for Selenium2 (WebDriver etc.) It is a cross-platform interpreted language which has many features in common with Perl and Python. First released in 1995 by Yukihiro Matsumoto.

  Rails - Is a framework for Ruby web development commonly called "Ruby On Rails" . It is difficult to program using Ruby On Rails unless you know Ruby first.

  Ruby In Steel - Is a licensed version of Ruby On Rails for Microsoft Studio which we don't use.

Quality Assurance Test Plan Standards

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context (in the Description field QAI, In Perforce).
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary of all included Test Cases and Steps, so that anyone in PD or SOE can understand what is being tested.
         - Here is a List Builder Report to check your compliance: Test Cases Without Descriptions

   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Start Up/ Shut Down Instructions
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
            - Here is a List Builder Report to check your compliance:
               - Test Plan - Startup/Shut Down Instructions
               - Test Case - Startup/Shut Down Instructions

   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are representing.
         - *Hint 1:* Screenshots should isolate focus on particular area NOT an entire desktop.
         - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.

   e. Test Cases must cover the features/functionality - nothing more, nothing less.
      i. Use the statements below to help clarify differences and separate the testing of values versus the testing of functionality
      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"
         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.

   g. All Test Steps must have clear written instructions and clear, predictable written results.

   h. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related

to "skip step" below.
   ii. Avoid Steps like "pick any component"- be specific.

i. Ensure Test Cases continue to flow after altering them due to new enhancements.

j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
   i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
   ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained so it can be run entirely on its own.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
   b. When taking into account the execution of your Test Case realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
   c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from The FactSet Product Tree).
      i. If Usage Product doesn't exist to map to, coordinate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that).
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date.
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Peer Reviews
   a. Analysts reviewers are expected to provide:
      i. Ensure that the test case exercises the functionality that it was written for, as noted in the Test Plan description.
      ii. Verify that tests are focusing on application specific functionality

   b. Regression reviewers should focus on section 1 rules

   c. What Triggers a Test Plan/Case Review:
      i. An application or functionality that requires a spec or a meeting to create a test plan/case
      ii. 'QA Regression Results' RPDs have increased by a noticeable amount compared to other test plans
      iii. If an Analyst requests a review of their test plan/case, a Regression team review is also required

   d. When a Test Plan isn't reviewed but changed, notification should be sent via "Regression Coverage Change Control" RPD product

i. Example
    1) Reducing number of steps in a test plan: RPD:15237065
    2) Multiple notifications can be sent in one RPD for a specific product

6) Once the Test Case is written, you must document related RPD(s) with appropriate details.
    a. QAI Test Cases must include link(s) to the RPD where you were asked to create the Test Case.
    b. Comment in the RPD indicating status of your work, through completion, and reference the link to the QAI Test Case.

7) Test Plan Limits:
    a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
        - Here is a List Builder Report to check your compliance: Test Plans w/ greater than 100 steps

    b. Each Test Case must have **no more than 25 Test Steps**.
        - Here is a List Builder Report to check your compliance: Test Cases w/ greater than 25 steps

    c. Test Plan Expected Duration must be under **60 minutes**.
        i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
            - Here is a List Builder Report to check your compliance: Test plan run time is greater than 60 minutes

    d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
        i. Extreme scrutiny must be applied.
        ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
            1) Clearly this situation would necessitate a Test Case peer review
        iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
        iv. DO NOT try to game these limits. Test Steps must be concisely written.
        v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.

    e. Exceptions to Limits:
        i. **Require QA Manager Approval before implementation**, provide supporting rationale.
        ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

8) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

9) The Feature Matrix must be updated with the new Test Case name and respective features.

    a. All Feature Matrix additions need to be peer reviewed until QA Manager deems review no longer necessary (same as #11).
        i. If the test plan is New or Active, it must have Feature Matrix mappings

10) Automation Script "Quality" standards:
    Analyst/Scripter Workflow Guidelines

    a. All Automated tests must be in QAI
        i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
        ii. These test cases must also follow the standards above.
        iii. Build dependent test cases should be noted in QAI. ? (Build Dependencies)

    b. Test Scripts must include
        i. QAI test step ID's
        ii. Readable and informative comments

    c. Naming conventions and code standards
        i. Folder structure that is scalable and understandable to you and your consumers
        ii. Consistency in naming page objects or utilities
            1) Follow the conventions of the code base being automated, come to agreement with the software engineer(s)
        iii. Test Case names should match QAI names
        iv. Use engineering code standards if they are available

    d. Reuse of code

i. Startup /Shutdown have their own utilities which can be shared by all scripts.
ii. Use existing Page objects/ Utilities where available

e. Page Objects/ Utilities
   i. For repetitive functions create Page Objects and Utilities where they apply
   ii. Functionality that should be in a Page Object should not be in a script
      1) Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPAThH identification unless completely necessary.
   iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what functionality broke
   iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
      1) If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

f. Promoting Test Cases to Production
   i. Test on your laptop and then a QA Single use VM.
   ii. Test Cases require a code review
   iii. Test must be run in the Development environment
   iv. Link to Scheduling Document

g. Failures in the Tellus Production Environment
   i. Analysts are responsible for monitoring scripts that are assigned to them
   ii. If a test case fails 3 times as a result of script errors or environment issues the script should be removed from Production back to the Development environment.
      1) Additional manual testing time needs to be accounted for and reviewed with Regression TL prior to the start of the regression

Final v4

Friday, November 21, 2014

FY15 Final v2:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context (in the Description field QAI, In Perforce).
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
      iv. Here is a List Builder Report to check your compliance Test Cases Without Descriptions
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Start Up/ Shut Down Instructions
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      - *Hint 1:* Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more, nothing less.
      i. Use the statements below to help clarify differences and separate the testing of values versus the testing of functionality

      In cases where: Values may change and/or not the focus of the test:

      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:

      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"
         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
   g. All Test Steps must have clear written instructions and clear, predictable written results.
   h. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component"- be specific.
   i. Ensure Test Cases continue to flow after altering them due to new enhancements .
   j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
      i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
      ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
    a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
    b. When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
    c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
    a. *Libraries are being phased out in FY15 - this rule will change during the year*
    b. Library Name must match the Application name (from The FactSet Product Tree).
        i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
    c. Test Plan and Case Names should be concise:
        i. Names are merely labels for the feature being tested.
            1) A Name is not a sentence (use the Description field for that)..
            2) A Name is not an RPD#.
    d. Automation Naming Conventions
        i. Follow the folder names for the Test Plan you are automating in QAI
        ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases**:**
    a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
    b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
    c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
    d. Upon review completion, set Status to Active.
    e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
        i. **New** – New application not ready for testing, but will be shortly.
        ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
            1) This is the *only* Status that enables the Test plan on a Status Webpage.
        iii. **Inactive** – When you are rewriting or consolidating a test plan.
            1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
            2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test P lans on the Status Webpage that were being tested.)
        iv. **Retired** – Used when the application or its test coverage is being retired.
            1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Peer Reviews
        a. Analysts reviewers are expected to provide:
            1) Ensure that the test case exercises the functionality that it was written for, as noted in the Test Plan description.
            2) Verify that tests are focusing on application specific functionality
        b. Regression reviewers should focus on section 1 rules

    a. What Triggers a Test Plan/Case Review:
        a. An application or functionality that requires a spec or a meeting to create a test plan/case
        b. 'QA Regression Results' RPDs have increased by a noticeable amount compared to other test plans
        c. If an Analyst requests a review of their test plan/case, a Regression team review is also required
    b. When a Test Plan isn't reviewed but changed, notification should be sent via "Regression Coverage Change Control" RPD product
        a. Example
            1) Reducing number of steps in a test plan: http://is.factset.com/rpd/summary.aspx?messageid=15237065
            2) Multiple notifications can be sent in one RPD for a specific product

6) Once the Test Case is written, you must document related RPD(s) with appropriate details.
    a. QAI Test Cases must include link(s) to the RPD where you were asked to create the Test Case.
    b. Comment in the RPD indicating status of your work, through completion, and reference the link to the QAI Test Case.

7) Test Plan Limits:
    a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
        - http://is.factset.com/core/listbuilderportal/?viewId=539468
    b. Each Test Case must have **no more than 25 Test Steps**.
        - http://is.factset.com/core/listbuilderportal/?viewId=539469
    c. Test Plan Expected Duration must be under **60 minutes**.
        i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.

- http://is.factset.com/core/listbuilderportal/?viewId=539464

d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
   i. Extreme scrutiny must be applied.
   ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
      1) Clearly this situation would necessitate a Test Case peer review
   iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
   iv. DO NOT try to game these limits. Test Steps must be concisely written.
   v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.

e. Exceptions to Limits:
   i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
   ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

8) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

9) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

Automation Guidelines:

10) Test "Quality" standards

Analyst/Scripter Workflow Guidelines

   a. All Automated tests must be in QAI
      i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
      ii. These test cases must also follow the standards above.
      iii. Build dependent test cases should be noted in QAI. ? (Build Dependencies)

   b. Test Scripts must include
      i. QAI test step ID's
      ii. Readable and informative comments

   c. Naming conventions and code standards
      i. Folder structure that is scalable and understandable to you and your consumers
      ii. Consistency in naming page objects or utilities
         i. Follow the conventions of the code base being automated, come to agreement with the software engineer(s)
      iii. Test Case names should match QAI names
      iv. Use engineering code standards if they are available

   d. Reuse of code
      i. Startup /Shutdown have their own utilities which can be shared by all scripts.
      ii. Use existing Page objects/ Utilities where available

   e. Page Objects/ Utilities
      i. For repetitive functions create Page Objects and Utilities where they apply
      ii. Functionality that should be in a Page Object should not be in a script
         i. Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPA ThH identification unless completely necessary.
      iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what functionality broke
      iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
         i. If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

   f. Promoting Test Cases to Production
      i. Test on your laptop and then a QA Single use VM.
      ii. Test Cases require a code review
      iii. Test must be run in the Development environment
      iv. Link to Scheduling Document

   g. Failures in the Tellus Production Environment
      i. Analysts are responsible for monitoring scripts that are assigned to them

ii. If a test case fails 3 times as a result of script errors or environment issues the script should be removed from Production back to the Development environment. ??? (is it taken off SWP only or removed from Tellus Production as well)
    i. Additional manual testing time needs to be accounted for and reviewed with Regression TL prior to the start of the regression

Final v3

Friday, November 21, 2014

FY15 Final v2:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context (in the Description field QAI, In Perforce).
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
      iv. Here is a List Builder Report to check your compliance Test Cases Without Descriptions
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Instructions: H:\Department\QualityAssurance\Managers\Startup and Shutdown Instructions version 2.docx
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      - *Hint 1:* Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more, nothing less.
      i. Use the statements below to help clarify differences and separate the testing of values versus the testing of functionality

      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"

         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
   g. All Test Steps must have clear written instructions and clear, predictable written results.
   h. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component"- be specific.
   i. Ensure Test Cases continue to flow after altering them due to new enhancements .
   j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
      i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
      ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
    a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
    b. When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
    c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
    a. *Libraries are being phased out in FY15 - this rule will change during the year*
    b. Library Name must match the Application name (from The FactSet Product Tree).
        i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
    c. Test Plan and Case Names should be concise:
        i. Names are merely labels for the feature being tested.
            1) A Name is not a sentence (use the Description field for that)..
            2) A Name is not an RPD#.
    d. Automation Naming Conventions
        i. Follow the folder names for the Test Plan you are automating in QAI
        ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases**:**
    a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
    b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
    c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
    d. Upon review completion, set Status to Active.
    e. All Test Plans/Cases written by QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems review is no longer mandatory. This is typically within 6 months of hire. However, this does NOT mean an Analyst's tests are never reviewed after that period.
        a. Analysts reviewers are expected to provide:
            1) Ensure that the test case exercises the functionality that it was written for, as noted in the Test Plan description.
            2) Verify that tests are focusing on application specific functionality
        b. Regression Team reviewers are expected to provide feedback in these areas:
            1) The description for the test plan/case is clear and understood
            2) The continuity of the plan/case is clear and understood
            3) The plan/case is grammatically clear and correct
            4) Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.
    f. What Triggers a Test Plan/Case Review:
        a. An application or functionality that requires a spec or a meeting to create a test plan/case
        b. 'QA Regression Results' RPDs have increased by a noticeable amount compared to other test plans
        c. If an Analyst requests a review of their test plan/case, a Regression team review is also required
    g. When a Test Plan isn't reviewed but changed, notification should be sent via "Regression Coverage Change Control" RPD product
        a. Example
            1) Reducing number of steps in a test plan: http://is.factset.com/rpd/summary.aspx?messageid=15237065
            2) **Update QAI to make this easier for everyone.**

    h. Use the following Justifier Test Plan & Test Case Status definition appropriately:
        i. **New** – New application not ready for testing, but will be shortly.
        ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
            1) This is the *only* Status that enables the Test plan on a Status Webpage.
        iii. **Inactive** – When you are rewriting or consolidating a test plan.
            1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
            2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test P lans on the Status Webpage that were being tested.)
        iv. **Retired** – Used when the application or its test coverage is being retired.
            1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must document related RPD(s) with appropriate details.
    a. QAI Test Cases must include link(s) to the RPD where you were asked to create the Test Case.
    b. Comment in the RPD indicating status of your work, through completion, and reference the link to the QAI Test Case.

6) Test Plan Limits:
    a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.

- http://is.factset.com/core/listbuilderportal/?viewId=539468

b. Each Test Case must have **no more than 25 Test Steps**.
   - http://is.factset.com/core/listbuilderportal/?viewId=539469

c. Test Plan Expected Duration must be under **60 minutes**.
   i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
   - http://is.factset.com/core/listbuilderportal/?viewId=539464

d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
   i. Extreme scrutiny must be applied.
   ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
      1) Clearly this situation would necessitate a Test Case peer review
   iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
   iv. DO NOT try to game these limits. Test Steps must be concisely written.
   v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.

e. Exceptions to Limits:
   i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
   ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

Automation Guidelines:

9) Test "Quality" standards

Scripter/Analyst Workflows: H:\Department\QualityAssurance\Managers\AutomationWorkFlowGuildlines.docx

   a. All Automated tests must be in QAI
      i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
      ii. These test cases must also follow the standards above.
      iii. Build dependent test cases should be noted in QAI. ? (Build Dependencies)

   b. Test Scripts must include
      i. QAI test step ID's
      ii. Readable and informative comments

   c. Naming Conventions and code standards
      i. Folder structure that is scalable and understandable to you and your consumers
      ii. Consistency in naming page objects or utilities
         i. Follow the conventions of the code base being automated, come to agreement with the software engineer(s)
      iii. Test Case names should match QAI names
      iv. Use engineering code standards

   d. Reuse of code
      i. Startup /Shutdown have their own utilities which can be shared by all scripts.
      ii. Use existing Page objects/ Utilities where available

   e. Page Objects/ Utilities
      i. For repetitive functions create Page Objects and Utilities where they apply
      ii. Functionality that should be in a Page Object should not be in a script
         i. Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPA ThH identification unless completely necessary.
      iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what they do without contacting the original scripter
      iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
         i. If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

   f. Promoting Test Cases to Production
      i. Test on your laptop and then a QA Single use VM.

ii. Test Cases require a code review
iii. Test must be run in the Development environment
iv. Link to Scheduling Document

g. Failures in the Tellus Production Environment
    i. Analysts are responsible for monitoring scripts that are assigned to them
    ii. If a test case fails 3 times as a result of script errors or environment issues the script should be removed from Production back to the Development environment. ??? (is it taken off SWP only or removed from Tellus Production as well)
        i. Additional manual testing time needs to be accounted for and reviewed with Regression TL prior to the start of the regression

Final v2

Friday, November 21, 2014

FY15 Final v2:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Instructions: H:\Department\QualityAssurance\Managers\Startup and Shutdown Instructions version 2.docx
            i. **Alisa please get Werner to put that into a web page FAQ on QAI site, we'll insert THAT link here.**
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      - *Hint 1:* Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more, nothing less.
      i. These statements to help clarify differences when values are in the scope of the test, rather than purely functionality:
         1) Kate to make that make sense above.

      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"

         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
   g. All Test Steps must have clear written instructions and clear, predictable written results.
   h. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component"- be specific.
   i. Ensure Test Cases continue to flow after altering them due to new enhancements .
   j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
      i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.

ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
   b. When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
   c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. ~~**This isn't done yet - needs a round two with Alisa/Kate/Dan.**~~
      i. ~~**Spell out the threshold that triggers the review (1 step? 10?)**~~
      ii. ~~**This process below isn't yet efficient, but it is more thorough and has some good guidance in there. Let's get to both.**~~
      iii. ~~**Also it's wordy.**~~
   f. All Test Plans/Cases written by QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems review is no longer mandatory. This is typically within 6 months of hire. However, this does NOT mean an Analyst's tests are never reviewed after that period.
      a. Analysts reviewers are expected to provide:
         1) Ensure that the test case exercises the functionality that it was written for, as noted in the Test Plan description.
         2) Verify that tests are focusing on application specific functionality
      b. Regression Team reviewers are expected to provide feedback in these areas:
         1) The description for the test plan/case is clear and understood
         2) The continuity of the plan/case is clear and understood
         3) The plan/case grammatically is clear and correct
      c. Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.
   g. What Triggers a Test Plan/Case Review:
      a. An application or functionality that requires a spec or a meeting to create a test plan/case
      b. 'QA Regression Results' RPDs have increased by a noticeable amount compared to other test plans
      c. If an Analyst requests a review of their test plan/case, a Regression team review is also required
   h. When a Test Plan isn't reviewed but changed, notification should be sent via "Regression Coverage Change Control" RPD product
      a. Example
         1) Reducing number of steps in a test plan: http://is.factset.com/rpd/summary.aspx?messageid=15237065

   i. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must document related RPD(s) with appropriate details.
   a. QAI Test Cases must include link(s) to the RPD where you were asked to create the Test Case.
   b. Comment in the RPD indicating status of your work, through completion, and reference the link to the QAI Test Case.

6) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Test Plan Expected Duration must be under **60 minutes**.
      i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
   d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
         1) Clearly this situation would necessitate a Test Case peer review
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
      iv. DO NOT try to game these limits. Test Steps must be concisely written.
      v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   e. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

Automation Guidelines:

This policy should be:
- **Expectations on how automated test scripts should be constructed and inventoried, Including htings like**
  - "test script must have comments that refer to QAI test step ID's"
  - Test script must be linked to qai test case via tellus test mgr
  - Test case must be promoted to production in order to be used on swp's - not sure I like that here, but there you go

This policy should NOT be
- A how to/instruction guide for analysts or scripters on how to use their tools or interact with each other
- We need and like those guides, but they don't belong in this doc - just link to them from here

9) Test "Quality" standards
   a. What should be expected when receiving a test case to automate
      i. Test Case is in the Justifier and follows QA standards
      ii. Scripter reviews the test plan or recording manually
      iii. Scripter determines what is automatable reports back to analyst
         i. If it is determined that parts of the test are not automatable test case should be sent to the analyst
         ii. Only exception is for pre agreed upon workarounds
            a. Ex: Typing in US:DJII in place of opening a VMS dialog
      iv. Partial automation is not allowed

   b. Scripter reviews what can be automated
      i. Use Inspect.exe to verify the test case can be automated. Automation person determines that the objects are available for automation
      ii. If the scripter requires the engineers to expose some elements/ objects which are not accessible via automation the Scripter raises RPD under Product: Object Request Component and adds the Application product

   c. How do we define workarounds - what do we do with them?
      i. Workarounds for Automated Test Steps in a Test Case
         i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
         ii. The defective Test Step can either be commented out or jumped over.
      ii. Automated Test Cases execution times
         i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
      iii. Address goals for solving the problem. (so we do not forget that there is a problem)
         i. Application is the way it is and we cannot automate any other way
         ii. Open file dialog is not HTML right now but someday that will change so this workaround is temporary

d. All Automated tests must be in QAI
    i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
    ii. These test cases must also follow the standards above.
    iii. The QAI Test Case must have enough description and detail to correlate with the feature under test.

e. Naming Conventions
    i. Folder structure that is understandable to you and your consumers
    ii. Consistency in naming page objects or utilities
        i. Follow the conventions of the code base being automated

f. Startup/Shutdown
    i. Startup /Shutdown have their own utilities which can be shared by all scripts.
    ii. All scripters should use the existing startup/shutdown scripts.

g. Page objects/utilities
    i. For repetitive functions create Page Objects and Utilities where they apply
    ii. Functionality that should be in a Page Object should not be in a script
        i. Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPATH identification unless completely necessary. If XPATHS are only used in the Page Object code/Utilities then we don't need to look for XPATH errors in the script.
    iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what they do without contacting the original scripter
    iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
        i. If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

h. Devel/Production testing
    i. Prior to submitting to the Tellus Development Environment
        i. Test on your laptop and then a QA Single use VM.
        ii. Test Cases will undergo a code review
            i. All code must be reviewed by assigned reviewer prior to being introduced to the Development environment in Tellus.
            ii. A large edit to the script will require an additional code review
        iii. QA lead schedules a couple of runs using System Adhoc(No SWP) from the Tellus Job Entry page and verifies if the test is good. (In-case their name is not present in the email recipient list on Tellus Job Entry page, file an RPD to get it added. Product: App Qa Automation, Category: **Automation - Tellus UI** eg. RPD:15000005)

    ii. Testing in the Tellus Production Environment
        i. QA Analyst and the QA Automation Analyst agree that the test case is suitable it can be moved to the Tellus Production Environment (see Peer Review standards above)
        ii. Scripter maps the test case from Tellus Test Manager and tests on http://tellusdev at least twice to verify that it is stable.
        iii. Automation scripter will merge the script to Production branch in Perforce.
        iv. Scripter **Promotes to Production** from Tellus Test Manager and tests it on http://tellus which will automatically check the Automated flag in QAI.
        v. Scripter must update the tool type of the test case in QAI until bug is fixed.

    iii. Scheduling

        i. Release CRON jobs - Tellus admin to modify and maintain.
            i. Scripter files an RPD to have the script added to CRON schedule.
            ii. Scripter works with Analyst to decide the release the schedule.
                i. See the scheduling configuration document. Ken to provide.
            iii. The test case is no longer run manually
        ii. Rerun CRON - for SWP

    iv. Failures in the Tellus Production Environment
        i. Analysts are responsible for monitoring scripts that are assigned to them
        ii. It a test case fails 3 times in a row as a result of script errors or environment issues the script should be removed by a scripter from Production back to the Development environment.
        iii. Once the multi-failure Test Case issues are resolved in the Development environment it can be moved back to Production again

v. Build dependencies
    i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in QAI.

Final v1 Automation updated

Tuesday, November 11, 2014 2:04 PM

FY15 Final v1:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available - guidelines for use:
      iii. Used for setting up all required configuration parameters for the application to function properly (e.g. Database order, "time of day" rules for test execution, etc).
      iv. Used for special end of test instructions such as "do not save any changes made to workspace".
      v. Do NOT include these instructions in actual test case steps.
      vi. Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
      vii. **Not Done until Dan inserts links to the QAI FAQ doc that has examples of start/shut cases**
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      - Hint 1: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      - Hint 2: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more nothing less.
      i. These statements to help clarify differences when values are in the scope of the test, rather than purely functionality:
         1) Kate to make that make sense above.

      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"

         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.

g. All Test Steps must have clear written instructions and clear, predictable written results.
h. Test Plans must NOT require judgment calls to be made by the tester.
    i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
    ii. Avoid Steps like "pick any component"- be specific.
i. Ensure Test Cases continue to flow after altering them due to new enhancements .
j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
    i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
    ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
    a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
    b. When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
    c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
    d. Suggested approaches for the defined start and end to a Test Case:
        i. Dan Hoffkins to do the Start/Shut FAQ thing and link to it (same as above)

3) Naming Conventions for Libraries/Plans/Cases/Step:
    a. *Libraries are being phased out in FY15 - this rule will change during the year*
    b. Library Name must match the Application name (from Usage list of Applications).
        i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
    c. Test Plan and Case Names should be concise:
        i. Names are merely labels for the feature being tested.
            1) A Name is not a sentence (use the Description field for that)..
            2) A Name is not an RPD#.
    d. Automation Naming Conventions
        i. Follow the folder names for the Test Plan you are automating in QAI
        ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
    a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
    b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
    c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
    d. Upon review completion, set Status to Active.
    e. This isn't done yet - needs a round two with Alisa/Kate/Dan.
        i. Spell out the threshold that triggers the review (1 step? 10?)
        ii. This process below isn't yet efficient, but it is more thorough and has some good guidance in there. Let's get to both.
        iii. Also it's wordy.
    f. All Test Plans/Cases written by QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems the Analyst no longer needs review.
        i. Analysts reviewers are expected to provide critical and crucial feedback on the functionality being tested within the plan/case. The primary goal is to ensure that the test case is exercising the functionality that it was written for, as noted in the Test Plan description. Verify that tests are written with the focus being on application specific functionality, ie. Not multiple Office versions or negative testing
            1) Functionality, ie. "does it test what it's supposed to"
                i. Please refer to section 1.B (Test Plan Description)
                ii. It should be clearly stated in the Review RPD that this test plan matches and follows the description of the test
            1. Adhere to all test plan standards
            2. Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.
        a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
            1. Adherence to all test plan standards
            2. This is the primary opportunity to ensure the test case is ready for production before it counts
                i. Ask questions

a. Be specific
b. Request of demo's
    i. Or eLearning/OA pages/Spec?

g. Use the following Justifier Test Plan & Test Case Status definition appropriately:
    i. **New** – New application not ready for testing, but will be shortly.
    ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
        1) This is the *only* Status that enables the Test plan on a Status Webpage.
    iii. **Inactive** – When you are rewriting or consolidating a test plan.
        1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
        2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
    iv. **Retired** – Used when the application or its test coverage is being retired.
        1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
    a. Jay to word this better it sucks.
    b. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
    c. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
    d. In the QA Section of the RPD Properties (top right corner) under New Tests section, enter the Case/Step ID.

6) Test Plan Limits:
    a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
    b. Each Test Case must have **no more than 25 Test Steps**.
    c. Test Plan Expected Duration must be under **60 minutes**.
        i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
    d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
        i. Extreme scrutiny must be applied.
        ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
            1) Clearly this situation would necessitate a Test Case peer review
        iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
        iv. DO NOT try to game these limits. Test Steps must be concisely written.
        v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
    e. Exceptions to Limits:
        i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
        ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

    a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
        i. If the test plan is New or Active, it must have Feature Matrix mappings

Automation Guidelines:

1) Test "Quality" standards
    a. What should be expected when receiving a test case to automate
        i. Test Case is in the Justifier and follows QA standards
        ii. Scripter reviews the test plan or recording manually
        iii. Scripter determines what is automatable reports back to analyst
            i. If it is determined that parts of the test are not automatable test case should be sent to the analyst
            ii. Only exception is for pre agreed upon workarounds
                a. Ex: Typing in US:DJII in place of opening a VMS dialog

iv. Partial automation is not allowed

b. Scripter reviews what can be automated
   i. Use Inspect.exe to verify the test case can be automated. Automation person determines that the objects are available for automation
   ii. If the scripter requires the engineers to expose some elements/ objects which are not accessible via automation the Scripter raises RPD under Product: Object Request Component and adds the Application product

c. How do we define workarounds - what do we do with them?
   i. Workarounds for Automated Test Steps in a Test Case
      i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
      ii. The defective Test Step can either be commented out or jumped over.
   ii. Automated Test Cases execution times
      i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
   iii. Address goals for solving the problem. (so we do not forget that there is a problem)
      i. Application is the way it is and we cannot automate any other way
      ii. Open file dialog is not HTML right now but someday that will change so this workaround is temporary

d. All Automated tests must be in QAI
   i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
   ii. These test cases must also follow the standards above.
   iii. The QAI Test Case must have enough description and detail to correlate with the feature under test.

e. Naming Conventions
   i. Folder structure that is understandable to you and your consumers
   ii. Consistency in naming page objects or utilities
      i. Follow the conventions of the code base being automated

f. Startup/Shutdown
   i. Startup /Shutdown have their own utilities which can be shared by all scripts.
   ii. All scripters should use the existing startup/shutdown scripts.

g. Page objects/utilities
   i. For repetitive functions create Page Objects and Utilities where they apply
   ii. Functionality that should be in a Page Object should not be in a script
      i. Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPATH identification unless completely necessary. If XPATHS are only used in the Page Object code/Utilities then we don't need to look for XPATH errors in the script.
   iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what they do without contacting the original scripter
   iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
      i. If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

h. Devel/Production testing
   i. Prior to submitting to the Tellus Development Environment
      i. Test on your laptop and then a QA Single use VM.
      ii. Test Cases will undergo a code review
         i. All code must be reviewed by assigned reviewer prior to being introduced to the Development environment in Tellus.
         ii. A large edit to the script will require an additional code review
      iii. QA lead schedules a couple of runs using System Adhoc(No SWP) from the Tellus Job Entry page and verifies if the test is good. (In-case their name is not present in the email recipient list on Tellus Job Entry page, file an RPD to get it added. Product: App Qa Automation, Category: **Automation - Tellus UI** eg. RPD:15000005)

ii. Testing in the Tellus Production Environment
    i. QA Analyst and the QA Automation Analyst agree that the test case is suitable it can be moved to the Tellus Production Environment (see Peer Review standards above)
    ii. Scripter maps the test case from Tellus Test Manager and tests on http://tellusdev at least twice to verify that it is stable.
    iii. Automation scripter will merge the script to Production branch in Perforce.
    iv. Scripter **Promotes to Production** from Tellus Test Manager and tests it on http://tellus which will automatically check the Automated flag in QAI.
    v. Scripter must update the tool type of the test case in QAI until bug is fixed.

iii. Scheduling

    i. Release CRON jobs - Tellus admin to modify and maintain.
        i. Scripter files an RPD to have the script added to CRON schedule.
        ii. Scripter works with Analyst to decide the release the schedule.
            i. See the scheduling configuration document. Ken to provide.
        iii. The test case is no longer run manually
    ii. Rerun CRON - for SWP

iv. Failures in the Tellus Production Environment
    i. Analysts are responsible for monitoring scripts that are assigned to them
    ii. It a test case fails 3 times in a row as a result of script errors or environment issues the script should be removed by a scripter from Production back to the Development environment.
    iii. Once the multi-failure Test Case issues are resolved in the Development environment it can be moved back to Production again

v. Build dependencies
    i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in QAI.

Final v1

Tuesday, November 11, 2014 2:04 PM

FY15 Final v1:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available.
         1) Instructions: H:\Department\QualityAssurance\Managers\Startup and Shutdown Instructions version 2.docx
            i. **Alisa please get Werner to put that into a web page FAQ on QAI site, we'll insert THAT link here.**
         2) Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      i. Words are sufficient to describe instructions and results in almost all situations.
      ii. In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      - *Hint 1:* Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      - *Hint 2:* Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more nothing less.
      i. These statements to help clarify differences when values are in the scope of the test, rather than purely functionality:
         1) Kate to make that make sense above.

      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

   f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"

         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
   g. All Test Steps must have clear written instructions and clear, predictable written results.
   h. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component"- be specific.
   i. Ensure Test Cases continue to flow after altering them due to new enhancements .
   j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application.
      i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.

ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.
   b. When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
   c. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.

3) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. This isn't done yet - needs a round two with Alisa/Kate/Dan.
      i. Spell out the threshold that triggers the review (1 step? 10?)
      ii. This process below isn't yet efficient, but it is more thorough and has some good guidance in there. Let's get to both.
      iii. Also it's wordy.
   f. All Test Plans/Cases written by QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems the Analyst no longer needs review.
      i. Analysts reviewers are expected to provide critical and crucial feedback on the functionality being tested within the plan/case. The primary goal is to ensure that the test case is exercising the functionality that it was written for, as noted in the Test Plan description. Verify that tests are written with the focus being on application specific functionality, ie. Not multiple Office versions or negative testing
         1) Functionality, ie. "does it test what it's supposed to"
            i. Please refer to section 1.B (Test Plan Description)
            ii. It should be clearly stated in the Review RPD that this test plan matches and follows the description of the test
         1. Adhere to all test plan standards
         2. Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.
      a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
         1. Adherence to all test plan standards
         2. This is the primary opportunity to ensure the test case is ready for production before it counts
            i. Ask questions
            a. Be specific
            b. Request of demo's
               i. Or eLearning/OA pages/Spec?

   g. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
   a. Jay to word this better it sucks.
   b. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!

c. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
d. In the QA Section of the RPD Properties (top right corner) under New Tests section, enter the Case/Step ID.

6) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Test Plan Expected Duration must be under **60 minutes**.
      i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
   d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:13366861
         1) Clearly this situation would necessitate a Test Case peer review
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:14486774
      iv. DO NOT try to game these limits. Test Steps must be concisely written.
      v. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   e. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

Automation Guidelines:

1) Test "Quality" standards
   a. What should be expected when receiving a test case to automate
      i. Scripter reviews the test plan manually and meets with the QA lead to discuss questions if any and gets the test plan updated if there are any specifics required for automation
         i. There could be some cases where a test step cannot be automated.
         ii. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one
         iii. This could be tricky so we need to be careful. I do not want functionality removed from testing just because it was not automatable.

   b. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.
      i. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.
      ii. The tool location for Inspect.exe that they will use to identify objects or the lack of objects will be at: H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe

   c. What to do if you have to write a test cases? (ie. Page object)
      i. All Automated Test Cases must have a corresponding QAI Test Case to map to, prior to being run in Production.
      ii. These test cases must also follow the standards above.
      iii. The QAI Test Case must have enough description and detail to correlate with the feature under test.

   d. Naming Conventions
      i. For example, PA3 protractor folder names use the same names as engineering
      ii. Consistency in naming page objects or utilities (depending on the software being used)

   e. Startup/Shutdown
      i. Startup /Shutdown have their own utilities which can be shared by all scripts.
      ii. All scripters should use the existing startup/shutdown scripts.

   f. Page objects/utilities
      i. Rebecca volunteered to write a blurb
      ii. Utilities/Page Objects

i. For repetitive functions create Page Objects and Utilities where they apply
ii. Functionality that should be in a Page Object should not be in a script
    i. Example: Protractor scripts use Page Objects to access elements on the page. A Page Object normally consists of an XPATH to identify the element on the page and any utility code needed. A test script generally should not access elements through XPATH identification unless completely necessary. If XPATHS are only used in the Page Object code/Utilities then we don't need to look for XPATH errors in the script.
iii. Page Objects/Utilities should be designed and commented in such a way that other scripters can easily identify what they do without contacting the original scripter
iv. When creating Page Objects/Utilities the scripter should make sure that they are not recreating functionality that another object already defines.
    i. If two different Page Objects/Utilities are designed to access the same element then there will be two different failures if the element is changed, which is a waste of time to investigate

g. Devel/Production testing
    i. Testing in the Tellus Development Environment
        i. Prior to moving your code to the Tellus Development Environment it first must be tested on your laptop and then a QA Single use VM.
        ii. Test Cases will undergo a code review via the code view board before being posted to the Tellus Development Environment.
        iii. Scripter maps the test case from Tellus Test Manager and tests on http://tellusdev at least twice.
    ii. Code Review
        i. All code will be reviewed with an assigned reviewer prior to being introduced to the Development environment in Tellus.
        ii. If a script is edited to include a large change, code should be submitted for review again

    iii. Testing in the Tellus Production Environment
        i. Once the QA Analyst and the QA Automation Analyst agree that the test case is suitable (no script errors) it can be moved to the Tellus Production Environment by the scripter using the Tellus Test Manager

    iv. Failures in the Tellus Production Environment
        i. Analysts are responsible for monitoring scripts that are assigned to them
        ii. It a test case fails 3 times in a row as a result of script errors or environment issues the script should be removed by a scripter from Production back to the Development environment.
        iii. Once the multi-failure Test Case issues are resolved in the Development environment it can be moved back to Production again

h. How do we define workarounds - what do we do with them?
    i. Workarounds for Automated Test Steps in a Test Case
        i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
        ii. The defective Test Step can either be commented out or jumped over.
    ii. Automated Test Cases execution times
        i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
    iii. Address goals for solving the problem. (so we do not forget that there is a problem)
        i. Application is the way it is and we cannot automate any other way
        ii. Open file dialog is not HTML right now but someday that will change so this workaround is temporary
    iv. Scripter raises RPD under Product: Object Request Component and adds the Application product if he/she requires the engineers to expose some elements/ objects which are not accessible via automation

i. Scheduling
    i. CRON Scheduling
        i. Release CRON jobs - Vasu
        ii. Rerun CRON - for SWP

        i. QA lead schedules a couple of runs using System Adhoc(No SWP) from the Tellus Job Entry page and verifies if the test is good. (In-case their name is not present in the email recipient list on Tellus Job Entry page, file an RPD to get it added. Product: App Qa Automation, Category: **Automation - Tellus UI** eg. RPD:15000005)
        ii. Files an RPD to have the script added to CRON schedule with the following details(Script names, Configuration required, Preferred Day and Time) and removes the test plan from the Regression team's plate. Product: App Qa Automation, Category: **Automation - CRON** eg. RPD:13860495

    ii. Build dependencies
        i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in QAI.

Draft Alisa/Dan/Kate

Wednesday, November 05, 2014 10:37 AM

FY15 v3:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
      iv. Include relevant links to Online Assistant Help Files and Application Specification & Requirements documentation, which will often include up to date Screenshots and Use Cases surrounding functionality. - should we make a separate field for these links? Is this a good way to coax improvement on SDLC or is it too early or are there better ways?
      v. **KK:** This might cause a big mess that we need to update and clean up later on. Maybe this is better managed by a spec/doc management system. If analysts want to learn more about functionality, they can go there.
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available - guidelines for use:
      iii. Used for setting up all required configuration parameters for the application to function properly (e.g. Database order, "time of day" rules for test execution, etc).
      iv. Used for special end of test instructions such as "do not save any changes made to workspace".
      v. Do NOT include these instructions in actual test case steps.
      vi. Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      Words are sufficient to describe instructions and results in almost all situations.
      In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      Hint 1: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      Hint 2: Use of colored circles/boxes/arrows to draw attention to area of test is also

recommended.

e. Test Cases must cover the features/functionality - nothing more nothing less.
f. Statements to help clarify value differences. These statements can probably be used for tests with or without screenshots:

   In cases where: Values may change and/or not the focus of the test:
   "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

   In cases where: Verifying values is crucial to the expected result:
   "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
   If differences exists between DEVEL/QA and Live – report a Bug RPD.
   If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

g. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
   i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
   ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
      1) Good: "Change the *foo* parameter to value *bar*"

      2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
   iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
h. All Test Steps must have clear written instructions and clear, predictable written results.
i. Test Plans must NOT require judgment calls to be made by the tester.
   i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
   ii. Avoid Steps like "pick any component"- be specific!
j. Ensure Test Cases continue to flow after altering them due to new enhancements .
k. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application!
   i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
   ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.

      When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.

   b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
   c. Suggested approaches for the defined start and end to a Test Case:

i. The decision here was to write up a thorough scenario-based Justifier help document, make that document available in Justifier help section, and then link to that document here in the policy. Who will own this?

Scenario 1 (One Startup at beginning and end of Case):

Startup Instruction: (i.e. "start"):

- Type @US in the Search box
- The Universal Screening application will open
- Make sure all other previous applications are closed

Cases and Steps etc. are written to test real Universal Screening functionality

Shutdown Instruction: (i.e. "end"):

- Close the application
- Select NO if you are asked to SAVE the screen, workspace or any other data

Scenario 2 (Startup at EACH Case within a Plan):

Scenario 3 (exceptions in some cases within a plan that has a main Startup)

3) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. All Test Plans/Cases written by QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems the Analyst no longer needs review.
      i. Analysts reviewers are expected to provide critical and crucial feedback on the functionality being tested within the plan/case. The primary goal is to ensure that the test case is exercising the functionality that it was written for, as noted in the Test Plan description. Verify that tests are written with the focus being on application specific functionality, ie. Not multiple Office versions or negative testing
         1) Functionality, ie. "does it test what it's supposed to"
            i. Please refer to section 1.B (Test Plan Description)

ii. It should be clearly stated in the Review RPD that this test plan matches and follows the description of the test
1. Adhere to all test plan standards
2. Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.

a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
1. Adherence to all test plan standards
2. This is also the "learning" phase
a. Ask questions
b. Be specific
c. Request of demo's
i. Or eLearning/OA pages/Spec?

e. Upon review completion, set Status to Active.
f. Use the following Justifier Test Plan & Test Case Status definition appropriately:
i. **New** – New application not ready for testing, but will be shortly.
ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
1) This is the *only* Status that enables the Test plan on a Status Webpage.
iii. **Inactive** – When you are rewriting or consolidating a test plan.
1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
iv. **Retired** – Used when the application or its test coverage is being retired.
1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
c. In the QA Section of the RPD Properties (top right corner) under New Tests section, enter the Case/Step ID.

6) Test Plan Limits:
a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
b. Each Test Case must have **no more than 25 Test Steps**.
c. Test Plan Expected Duration must be under **60 minutes**.
i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
i. Extreme scrutiny must be applied. Are there better/more modern examples below?
ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
e. DO NOT try to game these limits. Test Steps must be concisely written.
i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
f. Exceptions to Limits:
i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly

constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

    a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).
        i. If the test plan is New or Active, it must have Feature Matrix mappings

9) Automation Guidelines
    a. Roles and Responsibilities - Functional Managers, Internal Applications, QA Leads, QA Automation developers
        i. One or more of the QA Automation folks in Hyderabad will be assigned to a functional manager and Internal Apps group.
        ii. The Functional Manager will prioritize automation projects for the Assigned automated test case developer
        iii. The Goal is to give the regression team the ability to execute an automated test case and review the results
            i. Before that can happen the Team Lead must approve that test case as automated
            ii. If there are test steps that can not be automated then the test case will not be moved to the regression team
        iv. The QA Automation person will familiarize themselves with the project
            i. The QA Automation person will need an engineering SPOC (Single Point of Contact) among the respective application developers/engineers to work on Object definitions and other issues.
            ii. For e.g. Bernard C. Peng worked with Patty and Bhupender for Sidebar and he has an idea of the automation we have done. Due to this, he has taken care of keeping the object references as consistent as possible and if there is some change coming up, he informs us in advance. This really helped us with the automation to be prepared for the changes coming. An instance of how it really helped, adding the RPD for reference: RPD:7362324 - Eyme
        v. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.
            i. The tool that they will use it identify objects or the lack of objects will be: H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe
            i. They may also select additional tools for this effort.
        vi. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.
            i. There could be some cases where a test step cannot be automated.
                i. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one
                ii. This could be tricky so we need to be careful. I do not want functionality removed from testing just because it was not automatable.
        vii. The Automation person automates the Test Plan
            i. It will be determined that there should be a manual test case for those test steps that could not be automated
            ii. I disagree with the statement above (i). We test workflows. When you start removing

steps, you are limiting the workflows. Analysts designed test cases to target workflows around a piece of functionality so we have to be very careful with removing steps.

b. Automating Test Cases
    i. Utilities/Page Objects
        i. For repetitive functions create Page Objects and Utilities where they apply
            i. We do current for example use utilities for Startup and Shutdown
    ii. Testing in the Tellus Development Environment
        i. Prior to post you code for code view if first must be tested on your laptop and then a QA Single use VM
        ii. Test Cases will undergo a code review via the code view board before being posted to the Tellus Development Environment
    iii. Testing in the Tellus Production Environment
        i. Once the QA Analyst and the QA Automation Analyst agree that the test case is suitable (no script errors) it can be moved to the Tellus Production Environment using the Tellus Test Manager
    iv. Failures in the Tellus Production Environment
        i. Once in the Production Environment it a test cases fails more than three times due too a script error or an environment issue it needs to be moved back to the Tellus Development environment.
    v. Workarounds for Automated Test Steps in a Test Case
        i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
    vi. Automated Test Cases execution times
        i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
    vii. CRON Scheduling
        i. The Analyst can use the Tellus Job Page to schedule CRON jobs.
        ii. Currently when CRON jobs are dependent on QAI builds due to build promotions those will have to be modified by a Tellus admin.
            i. This process will be automated in the near future.
    viii. Build dependencies
        i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in QAI.

KK: We need automation sections on:

i. Naming conventions (need to add to automation)
    1) For example, PA3 protractor folder names use the same names as engineering
    2) Consistency in naming page objects or utilities (depending on the software being used)
ii. Startup/Shutdown? Are we automating that every time?
iii. Testing in devel
iv. Scheduling jobs to run in production(cron)
v. Removing jobs from production if they are garbage
vi. Limit to how long a test case can be
vii. Code Review?
viii. How do we define workarounds - what do we do with them?

Draft 3

Monday, October 13, 2014 2:26 PM

To do:

- Peer Review: **KATE/DAN/ALISA** will gather to write up HOW to do peer reviews under the 3 scenarios that need peer review. Today, sometimes it's just a rubber stamp approval, for example, so we need to spell out what's expected during/after review.
- **Ken** to refactor the Automation section, take Kate's suggestions into account, and move appropriate points out of the automation section into "main" sections (e.g. naming conventions shouldn't be separate, just use the main section).
- **Need Volunteers:**
  - Startup/Shutdown scenarios: The decision here was to write up a thorough scenario-based Justifier help document, make that document available in Justifier help section, and then link to that document here in the policy. Who will own this?
  - Test Plan Limit examples - the current ones are old, please provide newer ones.

DH: I can do the startup/shutdown doc

FY15 v3:

QA Associate Directors consider adherence to the Test Plan Standard when evaluating employee performance in their teams. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
      iv. Include relevant links to Online Assistant Help Files and Application Specification & Requirements documentation, which will often include up to date Screenshots and Use Cases surrounding functionality. - should we make a separate field for these links? Is this a good way to coax improvement on SDLC or is it too early or are there better ways?
      v. **KK:** This might cause a big mess that we need to update and clean up later on. Maybe this is better managed by a spec/doc management system. If analysts want to learn more about functionality, they can go there.

DH: I like a separate field for these links, as it is a way to start improvement on SDLC. Context behind the plan and further documentation helps, such as OA Page or Spec, the RPD field is vague and confusing; does it lead to an enhancement, known issue, something different?

   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***.
            i. Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available - guidelines for use:
      iii. Used for setting up all required configuration parameters for the application to function properly (e.g. Database order, "time of day" rules for test execution, etc).
      iv. Used for special end of test instructions such as "do not save any changes made to workspace".
      v. Do NOT include these instructions in actual test case steps.
      vi. Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      Words are sufficient to describe instructions and results in almost all situations.
      In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      Hint 1: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      Hint 2: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Cases must cover the features/functionality - nothing more nothing less.
   f. Statements to help clarify value differences. These statements can probably be used for tests with or without screenshots:

      In cases where: Values may change and/or not the focus of the test:
      "Note: The expected result should be used as a reference only. Do not compare actual values as they may fluctuate."

      In cases where: Verifying values is crucial to the expected result:
      "Note: Values must match the expected results. If they do not match, please compare devel/qa to live.
      If differences exists between DEVEL/QA and Live – report a Bug RPD.
      If DEVEL/QA and Live match – follow the normal procedure for filing Live bug RPDs.

DH: The wording on what this is trying to state needs to be clearer.

   g. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
      i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
      ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases:
         1) Good: "Change the *foo* parameter to value *bar*"
         2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
      iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
   h. All Test Steps must have clear written instructions and clear, predictable written results.
   i. Test Plans must NOT require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" 1choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component"- be specific!
   j. Ensure Test Cases continue to flow after altering them due to new enhancements .
   k. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application!

1(h):
ii: **Sometimes we have to say pick any item if the item constantly changes. For example: 1. date dropdown 2. Use of Dynamic Ticker**

i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

2) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.

      When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.

   b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
   c. Suggested approaches for the defined start and end to a Test Case:
      i. The decision here was to write up a thorough scenario-based Justifier help document, make that document available in Justifier help section, and then link to that document here in the policy. Who will own this?

      Scenario 1 (One Startup at beginning and end of Case):
      Startup Instruction: (i.e. "start"):
      - Type @US in the Search box
      - The Universal Screening application will open
      - Make sure all other previous applications are closed

      Cases and Steps etc. are written to test real Universal Screening functionality

      Shutdown Instruction: (i.e. "end"):
      - Close the application
      - Select NO if you are asked to SAVE the screen, workspace or any other data

      Scenario 2 (Startup at EACH Case within a Plan):

      Scenario 3 (exceptions in some cases within a plan that has a main Startup)

3) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
      i. *Do we have an eta on this - when will it get phased out.*
   b. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.
   d. Automation Naming Conventions
      i. Follow the folder names for the Test Plan you are automating in QAI
      ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)

4) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

5) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
   a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
   b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
   c. In the QA Section of the RPD Properties (top right corner) under New Tests section, enter the Case/Step ID.

6) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Test Plan Expected Duration must be under **60 minutes**.
      i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
   d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied. Are there better/more modern examples below?
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
   e. DO NOT try to game these limits. Test Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   f. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as # 11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

9) All Test Plans/Cases written by *new* QA Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems the Analyst no longer needs review.
   a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
   b. Inclusion of PD/SOE review is optional but highly encouraged to foster familiarity with the test coverage.

10) Automation Guidelines
   a. Roles and Responsibilities - Functional Managers, Internal Applications, QA Leads, QA Automation developers
      i. One or more of the QA Automation folks in Hyderabad will be assigned to a functional manager and Internal Apps group.
      ii. The Functional Manager will prioritize automation projects for the Assigned automated test case developer
      iii. The Goal is to give the regression team the ability to execute an automated test case and review the results
         i. Before that can happen the Team Lead must approve that test case as automated
         ii. If there are test steps that can not be automated then the test case will not be moved to the regression team

DH: I disagree with this, if there are test steps that can't be automated then the functional lead should take this into account and restructure the plan so that all steps are automated in the case/plan, and therefore can be moved to the regression team.

      iv. The QA Automation person will familiarize themselves with the project
         i. The QA Automation person will need an engineering SPOC (Single Point of Contact) among the respective application developers/engineers to work on Object definitions and other issues.
         ii. For e.g. Bernard C. Peng worked with Patty and Bhupender for Sidebar and he has an idea of the automation we have done. Due to this, he has taken care of keeping the object references as consistent as possible and if there is some change coming up, he informs us in advance. This really helped us with the automation to be prepared for the changes coming. An instance of how it really helped, adding the RPD for reference: RPD:7362324 - Eyme
      v. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.
         i. The tool that they will use it identify objects or the lack of objects will be: H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe
         i. They may also select additional tools for this effort.
      vi. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.
         i. There could be some cases where a test step cannot be automated.
            i. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one
            ii. This could be tricky so we need to be careful. I do not want functionality removed from testing just because it was not automatable.
      vii. The Automation person automates the Test Plan
         i. It will be determined that there should be a manual test case for those test steps that could not be automated
         ii. I disagree with the statement above (i). We test workflows. When you start removing steps, you are limiting the workflows. Analysts designed test cases to target workflows around a piece of functionality so we have to be very careful with removing steps.

DH: Agreed, but I guess it depends on how much you're sacrificing. If it's substantial, maybe its not a good candidate for automation.

   b. Automating Test Cases
      i. Utilities/Page Objects
         i. For repetitive functions create Page Objects and Utilities where they apply
            i. We do current for example use utilities for Startup and Shutdown
      ii. Testing in the Tellus Development Environment
         i. Prior to post you code for code view if first must be tested on your laptop and then a QA Single use VM
         ii. Test Cases will undergo a code review via the code view board before being posted to the Tellus Development Environment
      iii. Testing in the Tellus Production Environment
         i. Once the QA Analyst and the QA Automation Analyst agree that the test case is suitable (no script errors) it can be moved to the Tellus Production Environment using the Tellus Test Manager
      iv. Failures in the Tellus Production Environment
         i. Once in the Production Environment it a test cases fails more than three times due too a script error or an environment issue it needs to be moved back to the Tellus Development environment.
      v. Workarounds for Automated Test Steps in a Test Case
         i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
      vi. Automated Test Cases execution times
         i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
      vii. CRON Scheduling
         i. The Analyst can use the Tellus Job Page to schedule CRON jobs.
         ii. Currently when CRON jobs are dependent on QAI builds due to build promotions those will have to be modified by a Tellus admin.
            i. This process will be automated in the near future.
      viii. Build dependencies
         i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in QAI.

DH: Lots of typos and grammatical problems in section B.

KK: We need automation sections on:
   i. Naming conventions (need to add to automation)
      1) For example, PA3 protractor folder names use the same names as engineering
      2) Consistency in naming page objects or utilities (depending on the software being used)
   ii. Startup/Shutdown? Are we automating that every time?
   iii. Testing in devel
   iv. Scheduling jobs to run in production(cron)
   v. Removing jobs from production if they are garbage
   vi. Limit to how long a test case can be
   vii. Code Review?

viii. How do we define workarounds - what do we do with them?

Draft 2

Monday, October 13, 2014 2:26 PM

Feedback from your teams?

For FY15, start from the currently published v1 from FY14 (below). Initial needs:

- ~~Mandate use of "Startup/Shutdown" instructions and "Instances"~~
- ~~Make tests maintenance-free~~
  - ~~Define tight rules on when it's allowable to use screenshots~~
  - ~~Define rules around explicit versioning info in steps~~
- Include rules around Automation
- ~~Time to complete a test plan should be under 1 hour (include startup/shutdown)~~
- ~~Clean up grammar issues~~
- ~~One test plan that covers automation and manual~~
- ~~#2 c when filing the RPD about test cases added include the Test Plan name.~~
- ~~Have a set way to open applications~~

FY15 v2:

QA Associate Directors include application of the Test Plan Standard in employee evaluations for Analysts. Associate Directors maintain reports to identify compliance with the rules outlined below.

1) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: Test Plan REVAMP: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired –** Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

2) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Test Plan Expected Duration must be under **60 minutes**.
      i. This time applies to Active Test Plans, and is derived by averaging the previous 5 Test Run times.
   d. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied. Are there better/more modern examples below?
      ii. **KK:** I have a security exposures example that I can provide but I am not sure how long you want this to be? We can make something up and add it to the Justifier and provide a link
      iii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
      iv. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
   e. ~~Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**~~
      i. ~~DH: Is this necessary since QAI won't allow this structure?~~
   f. DO NOT try to game these limits. Test Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   g. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

3) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
      iv. Include relevant links to Online Assistant Help Files and Application Specification & Requirements documentation, which will often include up to date Screenshots and Use Cases surrounding functionality. - should we make a separate field for these links? Is this a good way to coax improvement on SDLC or is it too early or are there better ways?
      v. **KK:** This might cause a big mess that we need to update and clean up later on. Maybe this is better managed by a spec/doc management system. If analysts want to learn more about functionality, they can go there.
   c. Startup and shutdown:
      i. Initial launch of all Applications should be done in a consistent manner, department-wide, for functionality tests within each given Application.
         1) Therefore, it isn't necessary to "waste" extra Test Steps or Startup instructions on the known common method for launching Applications.
         2) The proper method for launching a Workstation-based Application is via its ***@shortcut***. **Everyone OK with this yes?**
            a) **KK: - Yes - okay**
            1) Specific Workstation Application Test Cases that exercise alternate launching methods may be used, but only for this purpose and not for exercising functionality within the Application.
         3) The proper method for launching Web applications is via Chrome using an appropriate ***URL*** provided by Engineering.
      ii. As of the start of FY15, new Startup and Shutdown features are available - guidelines for use:
      iii. Used for setting up all required configuration parameters for the application to function properly (e.g. Database order, test execution at a certain time).
      iv. Used for special end of test instructions such as "do not save any changes made to workspace".
      v. Do NOT include these instructions in actual test case steps.
      vi. Expected Duration time includes Startup and Shutdown instruction run time, and counts toward the 60 minute max.
   d. Screenshots in Test Steps must be avoided unless absolutely necessary
      Words are sufficient to describe instructions and results in almost all situations.
      In the rare event that screenshots are needed, they must be accompanied by instructions on what they are meant for.
      Hint 1: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      Hint 2: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.

DH: general thought, are we happy with the ordering of the bullets? Discuss more in meeting but I think they should go 3, 4, 6, 2, 1, 7, 9, 8
DH: Document should have more of a logical "from conception to completion" flow, this seems out of order.

AB: Filed: RPD:14721878 what is the priority? Requirements & Examples please review. Kristin will review with managers as well.
DH: to me this is a high priority that my team would utilize often

AB: Real examples help. Possibly each team supply an RPD example for each.
DH: I like this but maybe we pick an example we all like and develop it into a "generic" that will Work for all teams?

AB: We would want to ensure there is a link to the SPEC that we could add to QAI Test cases as needed. The tieback to the spec. (similar to RPD tieback) Sort of have this with the New Tests section in RPD. RPD:12507665
DH: would be best managed by a spec/document system, but an OA field in QAI might not be a bad idea since these page ID's rarely change, and is customer facing documentation. I actually like that.

AB: YES, Platform would need to test other methods to ensure they work.
DH: yup

e. Test Cases must cover the features/functionality - nothing more nothing less.

Statements to help clarify value diffs. These statements can probably be used for tests with or without screen shots

In cases where: Values may change and/or not the focus of the test:
"Note: The attached screenshot should be used as a reference only. Do not compare values as they may fluctuate."

In cases where: Verifying values is crucial to the expected result:
"Note: Values should match the attached screenshot. If they do not match the screenshot, compare devel/qa to live.
If differences exists between DEVEL/QA and Live – report a Bug RPD
If DEVEL/QA and Live are the same – follow the normal procedure for filing Live bug RPDs.

f. Application Functionality changes - this is reality. Test Cases often require accompanying updates, but not always.
   i. Test Cases that follow the above Screenshot and Startup/Shutdown rules will be more resilient.
   ii. Appropriate use of action phrases in instructions can avoid brittle Test Cases: **Everyone OK with this?**
      a) **KK: Yes - okay**
      1) Good: "Change the *foo* parameter to value *bar*"

      2) Bad: "Change the *foo* parameter by clicking the cog icon (see screenshot) and set the value to *bar*", since the cog icon may get changed to a wrench icon instead, for example.
   iii. Test Cases receiving frequent "feedback" from the Regression team (e.g. "update step or screenshot") or from PD and Engineering (via "QA Miss" RPDs) will be flagged for review.
g. All Test Steps must have clear written instructions and clear, predictable written results.
h. Test Plans must NOT require judgment calls to be made by the tester.
   i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
   ii. Avoid Steps like "pick any component"- be specific!
i. Ensure Test Cases continue to flow after altering them due to new enhancements
j. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test or the Application!
   i. If the Application has known Bugs preventing Steps from being properly executed, then the test should be rendered Inactive or Retired, and removed from Status Pages for future Test Runs, until the Application has been repaired sufficiently for Test Runs to pass.
   ii. Guidance: 3 recent failures due to "known issue" should trigger this remedial action.

4) Naming Conventions for Libraries/Plans/Cases/Step:
   a. *Libraries are being phased out in FY15 - this rule will change during the year*
   b. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   c. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.

AB: just added the link to the entity mapper for Application names. Also could be used for Test plan names if based of reports.

5) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

DH: Does linking need it's own bullet? Can this be added under another?

6) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.

      When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.

   b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
   c. Suggested approaches for the defined start and end to a Test Case:
      PLEASE WRITE SCENARIOS/EXAMPLES OF HOW TO USE STARTUP/SHUTDOWN, DOCUMENT THAT IN THE JUSTIFIER HELP SECTION AND THIS POLICY WILL MERELY LINK TO THAT RATHER THAN SPELLING OUT ALL THE SCENARIOS HERE.
      Test Case 1
         Startup Instruction: (i.e. "start"):
            - ~~Click on the F button~~ In the edit box type @US
            - ~~Click on~~ the application will open
            - make sure all other previous applications are closed
         Step 1, 2, 3, 4, etc.
         Shutdown Instruction: (i.e. "end"):
            - Close the application
            - Select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate

      The use of Linking Test Cases/Steps can be very powerful if the above Test Case example were to be repeated within a Test Plan. See #7 above for Linking guidance.

AB: updated the example. We can review together.

7) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
   a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
   b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
   c. In the QA Section of the RPD Properties (top right corner) under New Tests section, enter the Case/Step ID.

AB: new section to document why a test case or step was created or revised. New workflow may enhance this workflow. Move to one RPD product.
DH: Agreed

8) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as # 11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

9) All ~~new~~ Test Plans/Cases written by *new* QA ~~Leads~~ Analysts must be peer reviewed before implementation until their QA Manager *and* Regression Team Lead deems the ~~Lead~~ Analyst no longer needs review. **Since it's up to YOU to decide when an Analyst no longer requires review, I'm not sure we need to change this rule… you've complained about reviews taking a long time, but they're up to you whether they're necessary for your Analysts.**
   a. **KK:** What do we do about reviews done by HYD? They are also taking up time and yet we still receive tons of QA Regression results RPDs daily.
   b. DH: To have a newly written or heavily revised test plan/case reviewed by the regression team, file an RPD with the product 'Regression Coverage Change Control' and assign to Atlas TLs, Online TLs or both, depending on which cycles your test plans are executed
      i. The Test Plan will be reviewed during prebuild and burn-down time periods when active regression testing is not taking place
      ii. A new test plan may also be reviewed by executing a prebuild run of your product with the new test plan

DH: I think we need more policy around how peer reviews happen, particularly between analysts and the regression team. Sometimes we file separate RPD for a review and this done during burn-down time, other times the review is done while running a prebuild. Both are considered OK but it's not really documented. I tried to do that in my bullet points left.

JLM: KATE/DAN/ALISA will gather to write up HOW to do peer reviews under the 3 scenarios that need peer review. Today, sometimes it's just a rubber stamp approval, for example, so we need to spell out what's expected during/after review.

i. For this all policies and procedures must be followed for prebuild submission, such as request for testing, status page creation, and calendar/Twiki documentation.

c. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.

d. ~~Optional~~ Inclusion of PD/SOE review is optional but highly encouraged to allow all product team members to become familiar with the test coverage.

10) Automation Guidelines

a. Roles and Responsibilities - Functional Managers, Internal Applications, QA Leads, QA Automation developers

i. One or more of the QA Automation folks in Hyderabad will be assigned to a functional manager and Internal Apps group.

ii. The Functional Manager will prioritize automation projects for the Assigned automated test case developer

iii. The Goal is to give the regression team the ability to execute an automated test case and review the results

i. Before that can happen the Team Lead must approve that test case as automated

ii. If there are test steps that can not be automated then the test case will not be moved to the regression team

iv. The QA Automation person will familiarize themselves with the project

i. The QA Automation person will need an engineering SPOC (Single Point of Contact) among the respective application developers/engineers to work on Object definitions and other issues.

ii. For e.g. Bernard C. Peng worked with Patty and Bhupender for Sidebar and he has an idea of the automation we have done. Due to this, he has taken care of keeping the object references as consistent as possible and if there is some change coming up, he informs us in advance. This really helped us with the automation to be prepared for the changes coming. An instance of how it really helped, adding the RPD for reference: RPD:7362324 - Eyme

v. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.

i. The tool that they will use it identify objects or the lack of objects will be: H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe

i. They may also select additional tools for this effort.

vi. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.

i. There could be some cases where a test step cannot be automated.

i. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one

ii. This could be tricky so we need to be careful. I do not want functionality removed from testing just because it was not automatable.

vii. The Automation person automates the Test Plan

i. It will be determined that there should be a manual test case for those test steps that could not be automated

ii. I disagree with the statement above (i). We test workflows. When you start removing steps, you are limiting the workflows. Analysts designed test cases to target workflows around a piece of functionality so we have to be very careful with removing steps.

b. Automation in General - this is not a standard

i. The whole idea here is to free up a persons time spent doing manual test cases so they can look in other high impact areas for failures within FactSet products. The long and arduous testing of manual process that rarely find errors and those that are long running and repetitive are good candidates. Another area that is prime for automation are multi-browser and multi-platform configuration changes. Also we like to keep them relatively short about and hour and a half as opposed to ten hour jobs. These jobs can be divided and execute across multiple Virtual Machines.

c. Workflow in General - this is not a standard

i. Our workflow is based on our software testing environment. At the front of our whole environment are user interfaces. Where the user selects Virtual Machines that they want their tests to execute on and the tests themselves that they want executed. There are two main management sections. The first is configuration management. This is after the person has selected the Virtual Machines they want and our software testing automation framework handles that for them. The second is Test Case Management where we retrieve and distribute the test case from Perforce to the Virtual Machines , then gather and post the results.

KK: We need a section on:

i. Naming conventions (need to add to automation)

i. For example, PA3 protractor folder names use the same names as engineering

ii. Consistency in naming page objects or utilities (depending on the software being used)

ii. Startup/Shutdown? Are we automating that every time?

iii. Testing in devel

iv. Scheduling jobs to run in production(cron)

v. Removing jobs from production if they are garbage

vi. Limit to how long a test case can be

vii. Code Review?

viii. How do we define workarounds - what do we do with them?

Draft 1-Automation

Friday, October 24, 2014 9:59 AM

1) Automation Guidelines
    a. Roles and Responsibilities - Functional Managers, Internal Applications, QA Leads, QA Automation developers
        i. One or more of the QA Automation folks in Hyderabad will be assigned to a functional manager and Internal Apps group.
        ii. The Functional Manager will prioritize automation projects for the Assigned automated test case developer
        iii. The Goal is to give the regression team the ability to execute an automated test case and review the results
            i. Before that can happen the Team Lead must approve that test case as automated
            ii. If there are test steps that can not be automated then the test case will not be moved to the regression team
        iv. The QA Automation person will familiarize themselves with the project
            i. The QA Automation person will need an engineering SPOC (Single Point of Contact) among the respective application developers/engineers to work on Object definitions and other issues.
            ii. For e.g. Bernard C. Peng worked with Patty and Bhupender for Sidebar and he has an idea of the automation we have done. Due to this, he has taken care of keeping the object references as consistent as possible and if there is some change coming up, he informs us in advance. This really helped us with the automation to be prepared for the changes coming. An instance of how it really helped, adding the RPD for reference: RPD:7362324 - Eyme
        v. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.
            i. The tool that they will use it identify objects or the lack of objects will be: H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe
            i. They may also select additional tools for this effort.
        vi. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.
            i. There could be some cases where a test step cannot be automated.
                i. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one
                ii. This could be tricky so we need to be careful. I do not want functionality removed from testing just because it was not automatable.
        vii. The Automation person automates the Test Plan
            i. It will be determined that there should be a manual test case for those test steps that could not be automated
            ii. I disagree with the statement above (i). We test workflows. When you start removing steps, you are limiting the workflows. Analysts designed test cases to target workflows around a piece of functionality so we have to be very careful with removing steps.
    b. Automating Test Cases
        i. Naming Conventions
            i. Follow the folder names for the Test Plan you are automating in QAI
            ii. Where possible use the same naming conventions that Engineering is using (Page objects, utilities)
        ii. Utilities/Page Objects
            i. For repetitive functions create Page Objects and Utilities where they apply
                i. We do current for example use utilities for Startup and Shutdown
        iii. Testing in the Tellus Development Environment
            i. Prior to post you code for code view if first must be tested on your laptop and then a QA Single use VM
            ii. Test Cases will undergo a code review via the code view board before being posted to the Tellus Development Environment
        iv. Testing in the Tellus Production Environment
            i. Once the QA Analyst and the QA Automation Analyst agree that the test case is suitable (no script errors) it can be moved to the Tellus Production Environment using the Tellus Test Manager
        v. Failures in the Tellus Production Environment
            i. Once in the Production Environment it a test cases fails more than three times due too a script error or an environment issue it needs to be moved back to the Tellus Development environment.
        vi. Workarounds for Automated Test Steps in a Test Case
            i. If a Test Step is in error but can be run manually then the defective Test Step can be bypassed in the Automation.
        vii. Automated Test Cases execution times
            i. In order to fit in our CRON schedule time slots which are a hour long automated test cases should be limited to 60 minutes execution time.
        viii. CRON Scheduling
            i. The Analyst can use the Tellus Job Page to schedule CRON jobs.
            ii. Currently when CRON jobs are dependent on QAI builds due to build promotions those will have to be modified by a Tellus admin.
                i. This process will be automated in the near future.
        ix. Build dependencies
            i. If a QAI Test Case becomes build dependent (do this for this build and that for another build) it should be stated in that test case in

QAI.

KK: We need a section on:

i. Naming conventions (need to add to automation)
    i. For example, PA3 protractor folder names use the same names as engineering
    ii. Consistency in naming page objects or utilities (depending on the software being used)
ii. Startup/Shutdown? Are we automating that every time?
iii. Testing in devel
iv. Scheduling jobs to run in production(cron)
v. Removing jobs from production if they are garbage
vi. Limit to how long a test case can be
vii. Code Review?
viii. How do we define workarounds - what do we do with them?

Draft 1

Friday, September 19, 2014 3:37 PM

For FY15, start from the currently published v1 from FY14 (below). Initial needs:

- Mandate use of "Startup/Shutdown" instructions and "Instances"
- Make tests maintenance-free
  - Define tight rules on when it's allowable to use screenshots
  - Define rules around explicit versioning info in steps
- Include rules around Automation
- Time to complete a test plan should be under 1 hour (include startup/shutdown)
- Clean up grammar issues
- One test plan that covers automation and manual
- #2 c when filing the RPD about test cases added include the Test Plan name.
- Have a set way to open applications
-

FY15 v1:

1) FactSet Workstation-based applications must not have Test Plans run under the DIRECTIONS environment without approval from your manager.

2) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of October 1, 2014. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

3) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
   d. Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**
   e. DO NOT try to game these limits. Test Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   f. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

4) All Test Plans must have up to date Standard Execution Time.
   a. As of May 1, 2013, this is available only at the Test Plan level, but this will apply to Case level in the future.

5) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Use startup and shutdown instructions for getting in and out of the application that needs testing, and setting up all required special parameters
      i. Do not include these instructions in the test case steps
   d. Test Case must cover the features/functionality - nothing more nothing less.
   e. All Test Steps must have clear and predictable results.
      i. A test step should not use screenshots unless absolutely necessary to highlight a visual element that's required for the test
      ii. All screenshots must be accompanied by instructions on what they are meant for.
      iii. Hint 1: Use screenshots to depict said result.
      iv. Hint 2: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      v. Hint 3: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   f. Test Plans must not require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc"

Do we still need point # 1?

When will case level status be available?

- ~~Enhancement request: add test step count at the test case and test plan level in Test Manager~~
  - ~~If greater than allowed number, requires manager approval, so have a field indicating who signed off and when on the allowed overage. Do not allow 'Active' status until approved~~

choices. This is related to "skip step" below.
ii. Avoid Steps like "pick any component"- be specific!

g. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test!

6) Naming Conventions for Libraries/Plans/Cases/Step:
   a. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   b. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.

      When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.

   b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
   c. Suggested approaches for the defined start and end to a Test Case:

      Point 'C' should probably be in the startup/shutdown portion

      Test Case 1:
         Step 1 (i.e. "start"):
            - Click on the F button
            - Click on the application
            - make sure all other previous applications are closed
         Step 2, 3, 4, etc.
         Step (i.e. "end"):
            - Close the application
            - Select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate

      The use of Linking Test Cases/Steps can be very powerful if the above Test Case example were to be repeated within a Test Plan. See #7 above for Linking guidance.

9) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
   a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
   b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
   c. In the Resolution Section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.

10) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as # 11).
      i. If the test plan is New or Active, it must have Feature Matrix mappings

      Note: if there's a spec the feature matrix is built off that

11) All new Test Plans/Cases written by *new* QA Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead no longer needs review.
   a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
   b. Optional inclusion of PD/SOE review.

Kate:

Manual:
Statements to help clarify value diffs. These statements can probably be used for tests with or without screen shots

In cases where: Values may change and/or not the focus of the test:
"Note: The attached screenshot should be used as a reference only. Do not compare values as they may fluctuate."

In cases where: Verifying values is crucial to the expected result:
"Note: Values should match the attached screenshot. If they do not match the screenshot, compare devel/qa to live.
If differences exists between DEVEL/QA and Live – report a Bug RPD
If DEVEL/QA and Live are the same – follow the normal procedure for filing Live bug RPDs.

Peer review:
- Maybe we need to look at which tests are generating the most observations, updates etc and do only peer reviews for those tests
- Regression tests use peer review in the Justifier - the third biggest category from the data Alisa provided is in "Observation/Question" why doesn't that come up in peer review?

Looking at Platform Question RPDs many turn out to be Update Step/Screenshot

Top Analytics RPDs
1. Update screenshot
   a. Maybe the rule should be to write test cases without screenshots and only add them if there is confusion

in the peer review?

2. Update Step
    a. We have an excessive amount of these.
    b. Same as above - maybe we peer review only the apps that produce the most of these RPDs
3. Observation
    a. See Peer review above

Automation:

- For angular/protractor (maybe for other tools as well but I am speaking from what I know)
    - Naming convention for page objects - align with engineering?
    - Need to separate out libraries for different applications
    - Naming convention and structure for all analytics (and other) apps should be consistent and as similar as possible while working within one tool such as Test Complete etc.
- Workarounds
    - Example - we cannot automate the vms port/bench lookup however as a workaround we can type in US:DJII into the portfolio lookup.
        - If the analysts know of the workaround, they can add an "automation" instruction to the manual test
- Peer review
    - Will the regression team need to review automation tests?

- Automation status
    - We have a checkbox to show a test has been automated
    - How do we set a status/checkbox or whatever to show that a test is *ready* for automation?
        - I personally do not want to start filing RPDs to automate a test but how else to we generate a queue for scripters?
- Limits
    - The limits should be the same as the manual limits.

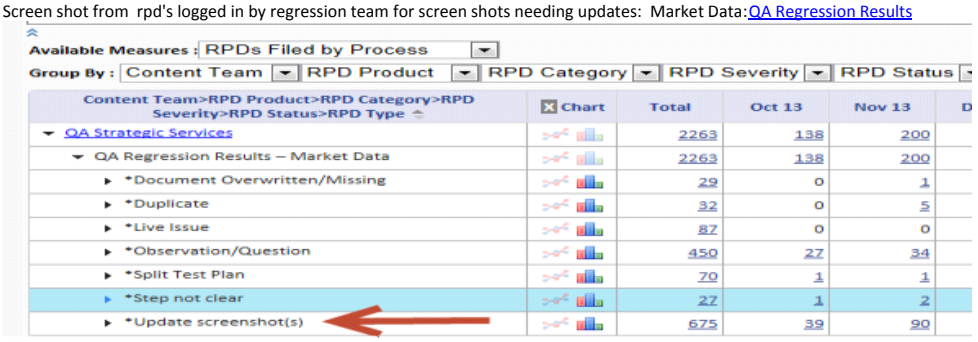
Robert:

Manual:

- Mandate use of "Startup/Shutdown" instructions and "Instances" - Agreed (Started to Implement with the team)

- Make tests maintenance-free
    - Define tight rules on when it's allowable to use screenshots
    - Remove all screen shots from Instructions and Expected Results for Navigation Steps.
        - Only use a screen shot when visual validation based on formatting (font types - size / colors / bullet points) & rendering (html tags) - When a screen needs to look exactly when it shows (verification point - end result).

Screen shot from rpd's logged in by regression team for screen shots needing updates: Market Data: QA Regression Results

- 

Content Team>RPD Product>RPD Category>RPD Severity>RPD Status>RPD Type	Chart	Total	Oct 13	Nov 13
QA Strategic Services		2263	138	200
QA Regression Results – Market Data		2263	138	200
*Document Overwritten/Missing		29	0	1
*Duplicate		32	0	5
*Live Issue		87	0	0
*Observation/Question		450	27	34
*Split Test Plan		70	1	1
*Step not clear		27	1	2
*Update screenshot(s)		675	39	90

- Feedback from my team - Per Pavan each step takes an average of 1 minute but our requirement's state that a test plan can contain up to 100 steps? Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps. Each Test Case must have **no more than 25 Test Steps**.

Automation:

- Create a standard test case template for the team to use - important when we are going to start on FDSWeb project

Ken:

Automation:

- Test Cases
    - Test cases should remain independent of each other, while Test Steps can be dependent on each other in an automated test case.

- Test Steps
    - If we send a Test Step failure over to QAI the individual probably won't be able to execute that test step by itself. They will have to execute those steps leading up to that one. New technologies such as Selenium the steps should line up with the corresponding test step in QAI, However, for older OI type tests using Test Complete test steps have been combined together.

- Test Case automated updates.
  - We are looking at automated test case updates and those updates should be rolled up to the Test Case in QAI. We are dependent on the QA Lead for documenting updates which would make this process backwards.

- Naming conventions
  - Naming conventions - we should follow what's in QAI. Fy15 Automation Standards will contain coding naming conventions.

- Screen shots for automated test cases
  - Screen shots should be eliminated but there are situations where the automation should grab a screen shot on a failure if it applies. Screen shot is more helpful than a log.

- Can it be automated
  - In QAI there should be a check box that indicates if a test case can be automated.

- Metrics
  - Tellus needs to drive more metrics to QAI with actual automation duration times which we could do. SWP link to the Progress Page http://tellusdev/progress.php?groupid=49761&tp=ActiveFieldsWord_Automated

- RPD's
  - This FY Tellus should be generating an RPD on a test case failure, it should follow the same formatting as the QAI RPDs and inserting the same configuration information.
  - Need process for Tellus generated RPDs

- Build versioning in test steps
  - If a test step is dependent on a specific build version (higher or lower) it should be indicated in the QAI test step.

- Problem Automated Test Cases need to be removed from the Production Automation environment
  - If the Test Case consistently fails for 3 times in a row for a specific environment or script issue

- Skip rules for automation if there are known issues at the test step level.

Dan:

Manual:
- Test steps should not be greater than 25 steps per test case and 100 steps per test plan
  - Enhancement request: add test step count at the test case and test plan level
  - If greater than allowed number, requires manager approval, so have a field indicating who and when signed off on the allowed overage
- Peer review
  - Currently takes a long time to be reviewed by another analyst and then the regression team
  - How can we speed this process up?
    - Can we skip analyst review and go right to the regression team for questions?
    - Can the process of getting the test into the regression team's hands, and providing feedback on that test, be automated from within the test manager?

Automation:
- Test case naming conventions
  - Maintain the same naming convention as the test case in the Justifier for easy tracing
- Logging and screenshots
  - Add as much logging around failures as you possibly can and screenshot captures where appropriate to "see" what occurred during the failure
  - This has really helped us when narrowing down a failure
- Justifier Test Step Linking
  - If it's automated it also must exist in Justifier, and have test step ID failure linkage
- Avoid GUI Interaction unless testing the GUI
  - Actions like installing, uninstalling, going into control panel, start menu, etc should all be done via command line

Sheikh
- Mandate use of "Startup/Shutdown" instructions and "Instances"
  - Like the idea of mandating "startup/Shutdown"
  - How do we handle that for automation?
- Define tight rules on when it's allowable to use screenshots
  - We should eliminate screenshots (Except special cases)
- Include rules around Automation
  - Automation test plans should follow the same rules as Manual
  - Like the idea of having a button on test case level to indicate automation ready
  - Making failure logs more intuitive for regression testers
- Time to complete a test plan should be under 1 hour
  - Up to 60 minutes is fine
- Regression Team reviewing test plans
  - We should set some rules on this as it's being slowing down
  - We did not see too many issues found during review but end up finding issue during actual run
- One Test Plan with Priority/Importance on test case level
  - We don't have 100% test coverage on each application
    - We try to cover most important and features with high usage
    - Leave out low usage features as we don't want to overwhelmed our regression team
  - Could write extensive test plans with all possible coverage and execute only those test cases that need to be for each stage/iteration.
  - We could have a dropdown on test case level with priority/importance of that particular test case and option to filer test cases based on importance:
    - High/Medium/Low (Could add Atlas to it as well)
    - One test plan will be used everywhere
    - Should be able to prioritize testing when important functionality need to be tested
    - Will not have much value for Automation but will be able to prioritize manual testing
    - Reduce maintenance time significantly

Draft-GuidelinesWritingAutomationTestCases

Thursday, October 09, 2014 8:28 AM

Guidelines for writing Test Cases for Automation

1) **Roles and Responsibilities**

Functional Managers, Internal Applications, QA Leads, QA Automation developers

   a. One or more of the QA Automation folks in Hyderabad will be assigned to a functional manager and Internal Apps group.
   b. The Functional Manager will prioritize automation projects for the Assigned automated test case developer
   c. The Goal is to give the regression team the ability to execute an automated test case and review the results
      i. Before that can happen the Team Lead must approve that test case as automated
      ii. If there are test steps that can not be automated then the test case will not be moved to the regression team
   d. The QA Automation person will familiarize themselves with the project
      i. The QA Automation person will need an engineering SPOC (Single Point of Contact) among the respective application developers/engineers to work on Object definitions and other issues.
      ii. For e.g. Bernard C. Peng worked with Patty and Bhupender for Sidebar and he has an idea of the automation we have done. Due to this, he has taken care of keeping the object references as consistent as possible and if there is some change coming up, he informs us in advance. This really helped us with the automation to be prepared for the changes coming. An instance of how it really helped, adding the RPD for reference: RPD:7362324 - Eyme
   e. Before automating use Inspect.exe to verify the test case can be automated. Automation person can access a build with the feature code in it then they will use a tool to determine with the objects for that feature are and if there are any initial issues. Resolving them at this point will save a lot of time downstream.
      i. The tool that they will use it identify objects or the lack of objects will be:
      H:\Department\QualityAssurance\QAAutomation\Automation\Inspect.exe
      i. They may also select additional tools for this effort.
   f. The automation person will work with the QA Lead to review the manual test case/steps as they move forward to make sure that the automation will produce the desired results.
      i. There could be some cases where a test step cannot be automated.
         i. It will have to be determined at this point If the process should be continued or a more automatable test case should replace the current one
   g. The Automation person automates the Test Plan
      i. It will be determined that there should be a manual test case for those test steps that could not be automated

2) **Naming Conventions**
   a. The Library and Test Plan name should be unique from other Library and Test Plan Names
   b. Test Case should not start with the word "Test"
   c. The test case name should be as short as Possible
   d. Each test case name starts with a Capital Letter and each unique word also starts with a capital letter, each non-word should be all capital letter, example ExcelLinkFDSCodeFDSPlusOverwriteYesAndNo
   e. The automated script name should be the same as the test case name in Justifier with the Application or Library name as a prefix eg. UploadWizard.sj .
3) **Directions**
   a. We will not be automating Directions. Talk to Jay if you feel you need to automate Directions.

4) **QA Guidelines**: H:\Department\QualityAssurance\Managers\QA Mgr Notebook\QA guidelines.one

Draft-AutomatedTestCaseDevelopment

Thursday, October 09, 2014 8:30 AM

QAAutomation Automated Test Case Development

This is a working document. Why? The process for writing automated test cases is evolving as we find ways to make improvements. In addition to Test Complete, we are currently adding in Selenium2 as a major second automated testing tool and that whole process is still being discovered.

Overview
This document is geared toward writing automated testing scripts as it applies to the QA department and our Software Testing Automation Framework. We want this to be a work flow for Test Complete and Selenium. First we'll talk about automation in general then we'll present two paths one for Test Complete and the other for Selenium.

Automation in General
The whole idea here is to free up a persons time spent doing manual test cases so they can look in other high impact areas for failures within FactSet products. The long and arduous testing of manual process that rarely find errors and those that are long running and repetitive are good candidates. Another area that is prime for automation are multi-browser and multi-platform configuration changes. Also we like to keep them relatively short about and hour and a half as opposed to ten hour jobs. These jobs can be divided and execute across multiple Virtual Machines.

Workflow in General
Our workflow is based on our software testing environment. At the front of our whole environment are user interfaces. Where the user selects Virtual Machines that they want their tests to execute on and the tests themselves that they want executed. There are two main management sections. The first is configuration management. This is after the person has selected the Virtual Machines the want and our software testing automation framework handles that for them. The second is Test Case Management where we retrieve and distribute the test case from Perforce to the Virtual Machines , then gather and post the results.

Some Terminology we think is important
Job - We use the term loosely but it can mean executing a test case or a series of test cases, or it can mean to perform a task.
Test Complete - Is the name of the software testing IDE that we use. However, there are two versions on it. The other version is called Test Execute and it's a command line version of Test Complete the GUI version. Test Execute resides on all of the QAAutomation testing Virtual Machines and is used to execute Jscript functions.
Jscript - Is Microsoft's version of JavaScript(trademark owned by SUN). Jscript supports conditional compilation and is the language of choice for writing automated tests for Test Complete.
Ruby - Is the scripting language of choice for writing automated test for Selenium2 (WebDriver etc.) It is a cross-platform interpreted language which has many features in common with Perl and Python. First released in 1995 by Yukihiro Matsumoto.
Rails - Is a framework for Ruby web development commonly called "Ruby On Rails" . It is difficult to program using Ruby On Rails unless you know Ruby first.
Ruby In Steel - Is a licensed version of Ruby On Rails for Microsoft Studio which we don't use.

Document Flow
If you are going to do Test Complete scripting(Jscript) then start with step 1 through 3. If you are going to do Selenium(Ruby) then start with step 1 but then move to step 4.

1. **QA Automation**
a. Main Twiki
i. http://infonet/view/QualityAssurance/AppQA-AutomationAndFrameworks

1. **Test Complete Overview**
i. Test Complete 9 Documentation
1) http://support.smartbear.com/viewarticle/32760/
a) General Information
b) For Existing Users
c) Testing with Test Complete
i. Getting Started
1) http://support.smartbear.com/viewarticle/30932/
i. Video Lessons
1) http://support.smartbear.com/screencasts/testcomplete/
i. Data-Driven Testing Tutorial
1) http://support.smartbear.com/viewarticle/31901/
i. Creating, Recording and Running Tests
1) http://support.smartbear.com/viewarticle/32203/Tutorials
a. **Test Complete Tutorials**
i. http://support.smartbear.com/viewarticle/30670/
a. **Jscript - Microsof**t
i. Getting started with Jscript
1) http://msdn.microsoft.com/en-us/library/vstudio/3bf5fs13(v=vs.100).aspx
i. Writing, Compiling, and Debugging Jscript Code
1) http://msdn.microsoft.com/en-us/library/vstudio/fw8ace91(v=vs.100).aspx
i. Displaying information with Jscript
1) http://msdn.microsoft.com/en-us/library/vstudio/3y58ttzw(v=vs.100).aspx
i. Introduction to Regular Expressions
1) http://msdn.microsoft.com/en-us/library/vstudio/28hw3sce(v=vs.100).aspx
i. Jscript Reference
1) http://msdn.microsoft.com/en-us/library/vstudio/x85xxsf4(v=vs.100).aspx
a. **Jscript in Test Complete**
i. Sample Jscript code
1) http://testingbuzzz.blogspot.com/2011/04/sample-script-jscript-in-test-complete.html
a. **Automated Test Case source code control**
i. All of our automated source code for test complete is under source code revision control
ii. They reside in our repository application Perforce P4V
iii. For Test Complete they reside under
1) Development //depot/smqa/Tellus/TC8
2) Production //depot/smqa/Tellus/TCMain
i. Test Cases are group together by function
1) Example: all the Data Central test cases are in //depot/smqa/Tellus/TCMain/DataCentral

2) When a Jscript function is or could be used across multiple test cases it is turned in to a utility functions
    a) Those Utility functions are also group together in a common file.
    b) The location of those in Perforce is //depot/smqa/Tellus/TCMain/Utils/
    c) Example: Data Central utilities are in UtilsDataCentral.sj

1. **Test Complete Scripting**
a. **Writing scripts overview**
i. http://support.smartbear.com/viewarticle/31252/
1) Writing Scripts - Overview
2) Writing Scripts - Quick Start
3) Naming Objects
4) Object Tree Models
a. How to write scripts
i. http://support.smartbear.com/viewarticle/28679/
1) Writing Scripts
2) Working With Tested Applications
3) Common Questions
a. **Process document for creating sample code**
i. Start by looking at existing spread sheets
ii. Test complete functions in Perforce(P4)
iii. Talk about utilities in Perforce(P4)
1) http://infonet/view/QualityAssurance/TCUtilityFunctions
i. Test Complete Script Template
1) \\pc.factset.com\dfs\Department\QualityAssurance\QAAutomation\Automation\TestTemplate.sj
a. **What you need to know**
i. Converting existing older versions of Test Complete test cases TC3/7
1) http://infonet/view/QualityAssurance/RegressionTestCaseAutomation
a. **MSAA developers guidelines**
i. http://infonet/view/QualityAssurance/MSAADeveloperGuideLines

1. **Selenium (this is a new effort for us and this section of the document we will updated as we more forward)**
i. Selenium Internal Overview OneNote
1) H:\Department\QualityAssurance\QAAutomation\Selenium\Selenium
i. The version of Selenium that we use is called the WebDriver version or Selenium2.
1) Docs can be found at http://docs.seleniumhq.org/projects/webdriver/
i. Our Twiki page is located at
1) http://infonet/view/QualityAssurance/AppQA-AutomationAndFrameworks#SELENIUMCHROMIUM
i. Additional Selenium notes and Applications OneNote

a. **Ruby scripting for Selenium**
i. Ruby org can be found at
1) http://www.ruby-lang.org/en/
i. Ruby bindings for Selenium can be found at
1) https://code.google.com/p/selenium/wiki/RubyBindings
a. **Automated Test Case source code - Selenium**
i. Proposed Selenium Framework and scripts:
1) Production: //depot/smqa/Tellus/Selenium/
2) Development: //depot/smqa/helios/Selenium/ or //depot/smqa/**Tellus**/SeleniumDev/ (As things stand now, the helios branch informally means development, so we can use the first location, or we can stop storing things in the helios branch and use the Tellus branch, anything should be fine)
i. For xml or python files that are used
1) Only for Test Complete have to be suffixed by TC E.g:- CreateProjectSuite**TC**.xml
2) Only for Selenium have to be suffixed by Sel/Seln E.g:- GetBrowserName**Sel**.py
3) Used by both Test Complete and Selenium shouldn't have any suffix. E.g:- LabMgrParser.py

By following the naming conventions for the files we can easily differentiate between the files that are used by both or ones that are used specifically.
a. **Testing using Selenium**
i. There are several areas that need to be tested by Selenium those include
1) Cross browser testing of html/5 content
2) Headless browser testing of html and Java script
3) Html/5 within the workstation with Chromium
4) Test of Thief elements

1. **Other**
a. eLearning - TBD
b. QAAutomation Training videos - TBD
c. PPTs - TBD

Data

Monday, October 06, 2014 1:39 PM

These are CMIS reports on QA Regression Results data.

All categories per team:

- Analytics: QA Regression Results
- Market Data: QA Regression Results
- Platform: QA Regression Results
- CoMa: QA Regression Results

A Category across teams:

- Step not clear
- Split Test Plan
- Update Screenshot
- Update Step
- Document Overwritten
- Live
- Duplicate
- Observation/Question

Test cases with more then 25 steps:
http://is.factset.com/core/listbuilderportal/?ihv2zrrphb

QA Team (Department)	
⊞ Analytics QA	(69)
⊞ Company & Markets	(38)
⊞ Internal QA	(43)
⊞ Market Data QA	(23)
⊞ Platform Engineerinç	(3)
⊞ Platform QA	(59)
⊞ Platform Regression	(5)

Test plans that contain Directions in instructions:
http://is.factset.com/core/listbuilderportal/?ynwuwpljea

We got good at shortening test plans based on the standard last year, but it apparently became ignored at some point during the year:

http://is.factset.com/core/listbuilderportal/?viewId=127266

Client Tes

Run | Clear | **848** results | Client Test

QA Team (Department)	QA Lead	Library Name	Test Plan Name	Test Plan Expected Duration (mins)
Market Data QA	Daniel Entic...	Research Manage...	RMSC - NIRN : Creat...	802.063657421867
Platform QA	Yasaswi Tali...	Office Integration ...	^= Refresh Perform...	651.033768535058
Platform QA	Yasaswi Tali...	Office Integration ...	=FDS Templates Per...	470.834495201111
Analytics QA	Mamta Kank...	Portfolio Analysis ...	CL4_2	318.75923239001
Analytics QA	Mamta Kank...	Portfolio Analysis ...	CL4_1	318.221201567849
Analytics QA	Shen Liu	Portfolio Simulatio...	Reports General - 5	277.625521444778
Platform QA	Anthony C. ...	FactSet Mobile - iP...	FactSet Drive - iPad	250.140536941886
Company & Markets QA	Asha Veerap...	Company Analysis	Supply Chain	243.834198853655
Analytics QA	Nallini Chan...	Axioma Portfolio R...	AXR_Tellus_Automat...	237.141719448566
Platform QA	Terrence Tse...	Atlas - Interactive...	Right Click Options -...	237.010649547378
Platform QA	Yasaswi Tali...	Office Integration ...	Sidebar(Excel) Perfo...	236.947437638839
Analytics QA	Mounika Ko...	PA3	charts-1	229.839143117666
Analytics QA	Chandrika D...	PA3	reports	228.054860432318
Platform QA	Yasaswi Tali...	Office Integration ...	DM - SDM Performa...	226.538983033101
Platform QA	Terrence Tse...	Atlas - Interactive...	Drawing Tools - Edit ...	216.834746759137
Market Data QA	Karen Hodgk...	Research Manage...	NIRN - RCP	215.413765917636
Platform QA	Terrence Tse...	Atlas - Interactive...	Intraday - Miscellane...	213.052264149006
Market Data QA	Remya Visw...	Factset Instant Me...	FIM Chatrooms	212.292592139062
Market Data QA	Sampath Pu...	StreetAccount	Other Calendars cat...	209.27107539773
Analytics QA	Hema Latha ...	Portfolio Dashboard	Jumping to Other Ap...	199.194167749087
Market Data QA	Madhuri Tan...	Portfolio View	Intraday Contributio...	198.187230149905
Platform QA	David V. Car...	Atlas - Interactive...	Advanced Labels 1	193.219693304896
Platform QA	David V. Car...	Atlas - Interactive...	Price Change Monito...	192.220236635076
Market Data QA	Joseph Valia...	Full Quote	Full Quote - Online	191.569904501438
Analytics QA	Mamta Kank...	Portfolio Analysis ...	IOS_6_General_4	190.070547403296
Platform QA	Priya Suryad...	Atlas - FDSChart (...	Intraday Scale-2	189.75267157147
Platform QA	Priya Suryad...	Atlas - FDSChart (...	Crosshairs	189.464448060241
Market Data QA	Jim Butler	News	Doc Display - New Vi...	181.71963731885
Platform QA	Yasaswi Tali...	Office Integration ...	DM - PresLink export...	181.656099995499
Analytics QA	Sean Kehoe	PA3	accounts	181.339746898611
Platform QA	Yasaswi Tali...	Sidebar - Detail Tab	Detail Tab - Working...	179.882804595232

Peer Reviews

Saturday, November 01, 2014 7:58 PM

Background:
Peer reviews presently can sometimes be a rubber stamp process, that raise more questions once the test plans/cases are in production. In addition, they can take a lot of time to review, process and be put in production, creating a bottleneck to boot. It takes time for new people (analysts and regression folks) to understand our workflow. This whole process naturally takes time and rushing things had lead to cut-corners and missed issues.

Mission:
Create a process that is efficient and allows for timely review of a test plan/case by the regression team. Once it is in production we should not see many requests asking us to clear up confusion in the plan/case because it had been presumably reviewed and given the seal of approval.

How:
Make a specific amount of prebuild time (call it 50%) dedicated solely for test plan review
Different focuses for the two groups: analysts and regression

- Want analysts to review for "is it a good test" or "does it test what it's supposed to"
- Want regression to review for "grammatical problems", understanding the test step, does it fit in standard for time and steps etc.

Let's split the responsibilities!

Analysts:

1. Standards
2. Functional, ie. "does it test what it's supposed to"

Regression:

1. Standards, ie. Grammar, step count, time count.
2. This is the "learning" phase
    a. Ask questions
    b. Be specific
    c. Request of demo's
        i. Or eLearning/OA pages/Spec?

Meta:

- Peer reviews are done by analysts first until the manager feels their creation skills are suitable to skip the analyst review
- Regression folks then receive and do these reviews either via direct request (Regression Change Control) or during prebuild phase
    - This second option cause some mehhh feelings amongst folks and needs to be sorted out
- The reason the regression team wants to see the plans sooner is because they can get familiar with the plan/functionality, ask their questions, keep to standard, etc.
- When you do the review there should be a test run, link THAT into the Regression Change Control RPD request, then you can see the results by looking at the test run

AutomationWorkFlowGuildlines

Wednesday, December 03, 2014 2:39 PM

First what are the RPD Products and the Categories needed?

There are two main Automation RPD Products; **Test Case Request - Automation** and **App QA Automation**

- Test Case Request - Automation
  - This product will be use to request a automated test case
- App QA Automation
  - This product will be used internally by the QA Team for automation RPD's for other requests and errors

What categories for each Product should you use.?

- For **Test Case Request - Automation** you should use the following categories below. There are others but just concentration on these for now.
  - Analytics
  - CompanyMarkets
  - InternalApps
  - MarketData
  - Platform
  - Platform API
  - Platform - Mobile
- For **App QA Automation** you should use the following categories below. There are others but just concentrate on these for now.
  - Automation - CRON
  - Automation - STAF
  - Automation - Tellus
  - Automation - Tellus Calendar
  - Automation - Tellus UI
  - Automation - Test Case Issue - Analytics
  - Automation - Test Case Issue - CompanyMarkets
  - Automation - Test Case Issue - InternalApps
  - Automation - Test Case Issue - MarketData
  - Automation - Test Case Issue - Platform
  - Automation - Test Case Issue - Platform API
  - Automation - Script Error - Analytics
  - Automation - Script Error - CompanyMarkets
  - Automation - Script Error - InternalApps
  - Automation - Script Error - MarketData
  - Automation - Script Error - Platform
  - Automation - Script Error - Platform API
  - Bug found by Automation - Analytics
  - Bug found by Automation - CompanyMarkets
  - Bug found by Automation - InternalApps
  - Bug found by Automation - MarketData
  - Bug found by Automation - Platform
  - Bug found by Automation - Platform API

Some examples

- Product: App QA Automation , Category: **Automation – Script Error – Analytics** -> If it's an issue with the script itself
- Product: App QA Automation , Category: **Automation – Test Case Issue – Analytics** -> If it's an issue with the QAI test case and automated script needs to be updated or an enhancement
- Product: App QA Automation , Category: **Bug Found by automation – Analytics** and the Application Product-> If it's a genuine bug in the application

Workflow - what should you do to request a automated test case to be written? What should the QA Scripter do. We'll look at this from the views of the Q Analyst and the QA Scripter

- It is assumed that the functional manager will prioritize automation projects for the Assigned automated test case scripter.
- **QA Analyst -** Insures that the manual version of the test case resides in QA Inventory under the proper Product and Test Plan
- **QA Analyst** - Opens an RPD under the Product: Test Case Request - Automation, Category : Functional Group with QAI test plan link and any other additional info if required eg. RPD:13860153
- **QA Scripter** - Reviews the test plan manually and meets with the QA Analyst to discuss questions if any and gets the test plan updated if there are any specifics required for automation
- **QA Scripter** -Share the documentation of what can and cannot be automated in the Automation request RPD

- **QA Scripter** - Opens an RPD under Product: Object Request Component and adds the Application product if he/she requires the engineers to expose some elements/ objects which are not accessible via automation
- **QA Scripter** - Automates the test case on the laptop, tests it on the VMs to accommodate any environment latencies
- **QA Scripter -** Has a code review with the assigned reviewer and submits it to the Perforce development branch
- **QA Scripter -** Maps the test case from Tellus Test Manager and tests in on http://tellusdev a couple of times
- **QA Scripter -** Once the script is stable, request the assigned person from the automation team to merge the script to Production branch in Perforce.
- **QA Scripter -** Selects the **Promotes to** Production Option from Tellus Test Manager and tests it on http://tellus which will automatically check the Automated flag in QAI.
- **QA Scripter -** Updates the Automation request RPD that the test case is available in production and provides the documentation if something else was or was not automated as opposed to the initial agreement
- **QA Analyst -** Schedules a couple of runs using System Adhoc(No SWP) from the Tellus Job Entry page and verifies if the test is good. (In-case their name is not present in the email recipient list on Tellus Job Entry page, file an RPD to get it added. Product: App Qa Automation, Category: **Automation - Tellus UI** eg. RPD:15000005)
- **QA Analyst -** Files an RPD to have the script added to CRON schedule with the following details(Script names, Configuration required, Preferred Day and Time) and removes the test plan from the Regression team's plate. Product: App Qa Automation, Category: **Automation - CRON** eg. RPD:13860495
- **QA Analyst -** Set the Automation request RPD is resolve

Automation scheduling

Friday, December 05, 2014 11:33 AM

HTML scheduling for Online applications

- Use LARGESETCACCESS
- Your serial #'s should match between your lima extension and what you are logging into the workstation as
- What FactSet version does PA3 url pick up?
  - 2013.5l (whatever the regression team is currently using)
- Example RPD: RPD:14471304

Format below is: Day - vms, html

Build chooser HTML language/translation is:
Production - sat_to_live
Staging (QA) - sat_to_qa
Devel - sat_to_devel

Lima extension language:
Select custom and type overrides in the empty field
Environment is always staging

Release candidate testing schedule:
Wednesday - devel, sat_to_devel
Thursday - devel, sat_to_devel
Friday - qa, sat_to_qa
Monday - qa, sat_to_qa
Tuesday - qa, sat_to_qa

Backwards compatibility schedule:
Wednesday - devel, sat_to_qa
Thursday - devel, sat_to_qa
Friday - qa, sat_to_devel
Monday - qa, sat_to_live
Tuesday - qa, sat_to_live

FY14 Test Standard

Friday, September 19, 2014 3:09 PM

QA Analyst notebook

Tuesday, March 26, 2013 9:43 AM

QA Analyst

Brainstorm Session 1 - 4/10/13

Wednesday, April 10, 2013 11:14 AM

☐ **Don't change what's on here, make a Brainstorm 2 page with your next idea!**

Test Plan Guidelines

- The purpose of this project is to build "standards" for writing and maintaining test plans
- Test plans need to be effective, so how they're written will impact that
- They need to be clear for the regression team and clear for future review
- The goal obviously to is have great, effective test plans that test the target features , nothing more nothing less
- Managers can use this standard to grade their employees on how well they follow the standard
  - Whether it's Meet Expectations, Needs Improvement or Exceeds (e.g. employee helps create new standards)

Guidelines for Writing Test Cases

1) Test cases need to be written for **workstation ONLY** unless specified otherwise.
2) **Use the Staging folder** within the application library to work on your test cases while they are in progress (aka - NEW).
   a. When you have completed the test case, move it into the appropriate test plan that was indicated in the test case request.
   b. Ex. You *can* use a Staging Copy of Plan to replace the need for "skip step" in cases when "skip" is meant to prepare for a future feature change. Need to do this now until Justifier ER #1 gets implemented sometime in FY14 maybe sorta kinda.
3) Test Plan Limits
   a. Each **test plan should have no more than 100 steps**. Create a new test plan if the addition of the new test case would make that test plan > 100 steps.
   b. Each test case should have **no more than 25 steps**.
   c. DO NOT try to game these limits. Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
   d. Exceptions:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale
      ii. Ex. "Workflows" that could have hundreds of cases. They need to be sensibly constructed with Mgr oversight.
4) **We need to add guidelines for qualitative instruction for good test plans, for starters:**
   a. Write in english and watch your grammar
   b. All screenshots must be accompanied by instructions on what their meant for
   c. Again, test case must cover the features/functionality - nothing more nothing less
   d. Test plans must not require judgement calls to be made by the tester
      i. A good indicator that you're out of bounds is if your test plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
      ii. Avoid steps like "pick any component"- be specific!
5) **Skip Step usage** needs review, see Justifier #3
6) **Do NOT** link steps. See Justifier #3
7) **Do NOT** nest steps. (this may be irrelevant if Justifier disables that feature, AB to verify)
8) Each test case must have a **defined start and end**.
   a. The first step of the test case should be login to Factset, go to workstation, select the F icon >…
      i. However, Jay thinks Justifier ER #3 should get done in order to eliminate the need for this repetitive step documentation
   b. The last step of the test case should be to close the application and **must state:** select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate.
9) Once the test case is written, you must **document the RPD with the details**.
   a. This means that every new test plan/case must be linked to the ER # that inspired the test case in the first place, which should also include the PD SPEC!!!
   b. See Justifier ER #2 for potential support.
   c. Leave a comment in the test case request RPD indicating that you have completed the test case and indicate the test case folder id#.
   d. In the resolution section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.
   e. Every test plan must have a 'standard execution' time documented in Justifier.
10) The feature matrix must also be updated with the new test case name and the features. All feature matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).
    a. We will create and enforce QA training on Feature Matrix so that we can remove the peer review check on this item.
11) All new test plans/cases written by new Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead doesn't need review. See Justifier ER #5

What does Justifier need to do to support the Standard?

1. Status is currently only at the Plan level, we need it to also apply to Case (and maybe Step?)
   a. One cool use of that means that if the Case status is "new", it won't show up on SWP until Lead sets status to "Active".
2. Linked RPD field at the Plan/Case/Step level should allow for inheritability
   a. If you set an RPD at Plan, then it is assumed that all Cases/Steps "below" the Plan are associated with that RPD.
   b. Lead can set a different RPD link at Case/Step levels of course when new/changed features are introduced over time.
3. To reduce need for "skip", we need to somehow make it clear to the tester what "environment" to run in.
   a. For instance, many cases describe what to install and run, and tell the tester skip based on environment changes like what Atlas version to use.
   b. Merely presenting environment variables might make this more efficient.
4. Consider tracking standard execution field at the Case level.
5. Test peer review enforcer: see RPD:3937252

Questions for Session 2

Friday, April 12, 2013 8:40 AM

Feedback from Platform Team

Use the Staging folder

1. Using Staging helps manage work
    a. Useful for Test Cases that are "in-progress"
2. Seen as most advantageous for Test Steps
3. This may just be an extra step, why not just use the "New" status?**

Test Plan Limits

1. Suggested limit vs. a hard stop
    - Dependent on complexity, it could be a challenge to stick to 100 steps.
2. Time limit rather than steps?**
3. Can automation handle 100+ test steps?

Linking Steps

1. Reuse steps where functionality is the same, especially in Office world
2. Something that's useful and can help when it works right

Defined start and end.

1. What level of definition?
    a. Application should be defined (FactSet Workstation, Workspace, Office, etc)

Test Plan Peer Review

1. Could be a time constraint
2. In addition to peer review do we do PD review as well as requirement?**

Feedback from the Strategic Team

Linked Steps/Cases: I don't think that "Don't use Linked Steps" should be a requirement. It would be helpful to receive feedback from managers if there is a better way to use linking**.** **Right now we have people using skip for test plan variability, which should not be a test plan best practice/standard.** (Think this is mentioned in #4 part d) Using skip in this way wastes tester time and can potentially cause a missed bug. **My feeling is that a test manager should have the ability to share test cases across test plans so that it is not required to have duplicate cases when test plan variability is needed**. This is what "Linking" was made for, but perhaps there is a better way to do this?

Standards on how much should be contained in a test step Some test managers set up test steps like test cases, and therefore even though we cannot setup nested test cases through the Justifier, you can get around that by making a test step look like a test case. This makes it the size of test plans (based on execution time) vary significantly. For example 100 steps might take 60 minutes or 120 minutes depending upon how much information is in a test step. **I propose one instruction and one expected result per test step.**

Be able to have an average time for a step based off these guidelines.
What makes up a test step… how many instructions?

Don't embed multiple instructions in a single step – this is prevalent in market data test plans. We can change it going forward to standardize test plans structure across the board.

Skip: This items is addressed in this document for environmental setup scenarios. I have also seen situations where the step asks the tester to skip if something (data/page/etc.) does not appear. In my opinion, skip should not be used in this way. **The best practice should be create test cases that have predictable results.**

Test Plan Peer Review :
Author of test plan reviews first before deploying to Hyderabad team for review – this will address most of the sequencing/continuity issues of test steps in the test plan, that usually get reported during review by regression testers.

Market Data Feedback : Click Here

3. Test Plan Limit (3c)
   *Do NOT add **steps** to existing **cases** for new functionality - create new test **case** instead.

Need further clarifications, do we need to create new test case if the functionality change is with in existing feature?

Reminder:

- Relating Test Plan writing skills to Career Progression Plan
- Author's ability to explain the application to test plan reviewer from business perspective.

Brainstorm Session 2 - 4/22/13

Wednesday, April 10, 2013 11:14 AM

- [ ] **Don't change what's on here, make a Brainstorm 3 page with your next idea!**
- [ ] **All - review #2, 4, 6-8 before session #3**
- [ ] **Alisa to file all the justifier ER's on the right with timing expectations managed (some may take awhile to get to)**

Test Plan Guidelines

- The purpose of this project is to build "standards" for writing and maintaining test plans
- Test plans need to be effective, so how they're written will impact that
- They need to be clear for the regression team and clear for future review
- The goal obviously to is have great, effective test plans that test the target features , nothing more nothing less
- Managers can use this standard to grade their employees on how well they follow the standard
  - Whether it's Meet Expectations, Needs Improvement or Exceeds (e.g. employee helps create new standards)

Guidelines for Writing Test Cases

1) Test cases need to be written for **workstation ONLY** unless specified otherwise.
2) **Use the Staging folder** within the application library to work on your test cases while they are in progress (aka - NEW).
   a. ** the same could be accomplished with just Status=New, but the Staging folder offers ease of isolation of in-progress test plan development… perhaps we just make the STAGING bullet optional but mandate Status=New
   b. When you have completed the test case, move it into the appropriate test plan that was indicated in the test case request.
   c. Ex. You *can* use a Staging Copy of Plan to replace the need for "skip step" in cases when "skip" is meant to prepare for a future feature change. Need to do this now until Justifier ER #1 gets implemented sometime in FY14 maybe sorta kinda.
   d. Justifier ER #6 - when Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
3) Test Plan Limits
   a. Each **test plan should have no more than 100 steps**. Create a new test plan if the addition of the new test case would make that test plan > 100 steps.
   b. Each test case should have **no more than 25 steps**.
   c. Do NOT add **steps** to existing **cases** for new functionality - create new test **case** instead.
   d. Note that all **Plans** must have **Cases** - no **Steps** can be written directly under a **Plan**
   e. DO NOT try to game these limits. Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
      ii. Needs Justifier ER #8
   f. Exceptions:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale
      ii. Ex. "Workflows" that could have hundreds of cases. They need to be sensibly constructed with Mgr oversight.
4) All Plans must have up to date Standard Execution Time
   a. See Justifier ER #7 to do so at Case/Step level
   b. Lead/Regression partners analyze the operational difference between the expected Standard time, versus the actual run time, red flagging big diffs.
5) Test "Quality" standards
   a. Write in English and watch your grammar
   b. Again, test case must cover the features/functionality - nothing more nothing less
   c. All steps must have clear and predictable results
      i. e.g. use screenshots to depict said result
      ii. All screenshots must be accompanied by instructions on what they are meant for
   d. Test plans must not require judgment calls to be made by the tester
      i. A good indicator that you're out of bounds is if your test plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
      ii. Avoid steps like "pick any component"- be specific!
6) **Naming Conventions for Libraries/Plans/Cases/Step**
   a. **We need these, let's come up with that**
   b. **See** Test Case naming conventions **For ideas**
7) **Skip Step usage policy (still needs review)**
   a. **see Justifier #3 - if that's done can we kill Skip functionality?**
   b. **Maybe - if a bug occurs in one step and further steps can't be run as a result, what do we do?**
   c. If there are repeated Go Acceptable Bbugs in a given Case, then:
      i. Advise SOE/PD that you are removing the Case until Bug is fixed.
      ii. Do NOT use "skip step", rather move Case to Test Plan = "Go Acceptable" until further notice
      iii. When ER #1 is done, also change Case status to Inactive
   d. Do NOT use "skip" for version testing - create a new separate **Case** instead
   e. Do NOT use "skip" for unpredictable behavior
8) **Do NOT link steps. See Justifier #3**
9) Each test case must be independently self-contained with a **defined start and end**.
   a. The first step of the test case should be login to Factset, go to workstation, select the F icon >…
      i. However, Jay thinks Justifier ER #3 should get done in order to eliminate the need for this repetitive step documentation
   b. The last step of the test case should be to close the application and **must state:** select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate.
10) Once the test case is written, you must **document the RPD with the details**.
    a. This means that every new test plan/case must be linked to the ER # that inspired the test case in the first place, which should also include the PD SPEC!!!
    b. See Justifier ER #2 for potential support.
    c. Leave a comment in the test case request RPD indicating that you have completed the test case and indicate the test case folder id#.

What does Justifier need to do to support the Standard?

1. Status is currently only at the Plan level, we need it to also apply to Case (and maybe Step?)
   a. One cool use of that means that if the Case status is "new", it won't show up on SWP until Lead sets status to "Active".
   b. Also accommodates "go acceptable" scenarios.
2. Linked RPD field at the Plan/Case/Step level should allow for inheritability
   a. If you set an RPD at Plan, then it is assumed that all Cases/Steps "below" the Plan are associated with that RPD.
   b. Lead can set a different RPD link at Case/Step levels of course when new/changed features are introduced over time.
3. To reduce need for "skip", we need to somehow make it clear to the tester what "environment" to run in.
   a. For instance, many cases describe what to install and run, and tell the tester skip based on environment changes like what Atlas version to use.
   b. Merely presenting environment variables might make this more efficient.
4. Consider tracking standard execution field at the Case level.
5. Test peer review enforcer: see RPD:3937252
6. When Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
7. Consider requiring Standard Execution time for the Case and Step level. Then the Plan execution time can be automatically summed up.
8. New pending justifier web testing application must show description field to the tester

d. In the resolution section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.
e. Every test plan must have a 'standard execution' time documented in Justifier.

11) The feature matrix must also be updated with the new test case name and the features. All feature matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).

    a. We will create and enforce QA training on Feature Matrix so that we can remove the peer review check on this item.

12) All new test plans/cases written by *new* Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead doesn't need review.
    a. Optional inclusion of PD/SOE review.
    b. See Justifier ER #5

Brainstorm Session 3 - 5/1/13

Wednesday, April 10, 2013 11:14 AM

Test Plan Guidelines

- The purpose of this project is to build "standards" for writing and maintaining test plans
- Test plans need to be effective, so how they're written will impact that
- They need to be clear for the regression team and clear for future review
- The goal obviously to is have great, effective test plans that test the target features , nothing more nothing less
- Managers can use this standard to grade their employees on how well they follow the standard
  - Whether it's Meet Expectations, Needs Improvement or Exceeds (e.g. employee helps create new standards)

Guidelines for Writing Test Cases

1) Test cases need to be written for **workstation ONLY** unless specified otherwise.
2) **Use the Staging folder** within the application library to work on your test cases while they are in progress (aka - NEW).
   a. ** the same could be accomplished with just Status=New, but the Staging folder offers ease of isolation of in-progress test plan development… perhaps we just make the STAGING bullet optional but mandate Status=New
   b. All New test plans should have the status of New once the Regression Review (RPD should be filed to notify the regression team of the testing review and potential release date) is completed then the status is changed to Active
   c. Ex. You *can* use a Staging Copy of Plan to replace the need for "skip step" in cases when "skip" is meant to prepare for a future feature change. Need to do this now until Justifier ER #1 gets implemented sometime in FY14 maybe sorta kinda.
   d. Justifier ER #6 - when Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
3) Test Plan Limits
   a. Each **test plan should have no more than 100 steps**. Create a new test plan if the addition of the new test case would make that test plan > 100 steps.
   b. Each test case should have **no more than 25 steps**.
   c. Do NOT blindly add **steps** to existing **cases** for new functionality - Extreme scrutiny must be applied.
      i. Significant amount of new functionality requires new test **case**. Example RPD:
      ii. However, some new functionality is minor and does not require new cases. Example RPD:
   d. Note that all **Plans** must have **Cases** - no **Steps** can be written directly under a **Plan**
   e. DO NOT try to game these limits. Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
      ii. Needs Justifier ER #8
   f. Exceptions:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale
      ii. Ex. "Workflows" that could have hundreds of cases. They need to be sensibly constructed with Mgr oversight.
4) All Plans must have up to date Standard Execution Time
   a. See Justifier ER #7 to do so at Case level
   b. See Justifier ER #9 for the reporting that will follow this behavior
5) Test "Quality" standards
   a. Write in English and watch your grammar
   b. Again, test case must cover the features/functionality - nothing more nothing less
   c. All steps must have clear and predictable results
      i. e.g. use screenshots to depict said result
      ii. All screenshots must be accompanied by instructions on what they are meant for
   d. Test plans must not require judgment calls to be made by the tester
      i. A good indicator that you're out of bounds is if your test plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
      ii. Avoid steps like "pick any component"- be specific!
   e. No Case/Step should instruct the tester to skip steps. If you do, this indicates something is deficient in your test as written!
6) Naming Conventions for Libraries/Plans/Cases/Step
   a. Library Name must match the Application name (from Usage list of Applications)
      i. If Usage product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
      ii. See Justifier ER #10 to make this easier
   b. Others? **See** Test Case naming conventions **For ideas**
      i. Consider Steps won't have NAME, just referred to by it's Case->Step # nomenclature. This requires justifier web rollout.
   c. Plan and Case Names should be concise
      i. Just a label for the feature being tested
      ii. Not a sentence (use the Description field for that)
      iii. Not an RPD#
7) If your Test Case / Step uses the Linking feature
   a. Understand the linking relationship - the Original source step that Subsequent steps link to is authoritative. So deleting that step will delete linked Subsequent steps!
   b. Defer to your Manager in the use of Linking.
8) Each test case must be independently self-contained with a **defined start and end**.
   a. The first step of the test case should be login to Factset, go to workstation, select the F icon >…
      i. However, Jay thinks Justifier ER #3 should get done in order to eliminate the need for this repetitive step documentation
   b. The last step of the test case should be to close the application and **must state:** select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate.

What does Justifier need to do to support the Standard?

1. Status is currently only at the Plan level, we need it to also apply to Case (and maybe Step?) RPD:8634680
   a. One cool use of that means that if the Case status is "new", it won't show up on SWP until Lead sets status to "Active".
   b. Also accommodates "go acceptable" scenarios.
2. Linked RPD field at the Plan/Case/Step level should allow for inheritability
   a. If you set an RPD at Plan, then it is assumed that all Cases/Steps "below" the Plan are associated with that RPD.
      i. Is that a one time event or is that RPD added for new test steps and Test Cases?
      ii. I'd think it is a one time event so we can follow what was created when the enhancement went out. If we add more enhancements those will have new RPDs
   b. Lead can set a different RPD link at Case/Step levels of course when new/changed features are introduced over time.
3. To reduce need for "skip", we need to somehow make it clear to the tester what "environment" to run in.
   a. For instance, many cases describe what to install and run, and tell the tester skip based on environment changes like what Atlas version to use.
   b. Merely presenting environment variables might make this more efficient.
4. Consider tracking standard execution field at the Case level.
5. Test peer review enforcer: see RPD:3937252
6. When Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
7. Consider requiring Standard Execution time for the Case level. Then the Plan execution time can be automatically summed up.
8. New pending justifier web testing application must show description field to the tester
9. Create regular report sent to each Product's Lead/Regression partners to analyze the operational difference between the expected Standard time, versus the actual run time, red flagging big diffs. Have this done based on AVG run time, not ABS run time.
10. **Mapper for Library Name <-> Application Name**
    a. **Use the same product picker that's in RPD**
    b. **This will file a new Usage Product request (that jay/bilal approve)**
    c. **This assumes you have agreement with PD/SOE on what that product is**

9) Once the test case is written, you must **document the RPD with the details**.
    a. This means that every new test plan/case must be linked to the ER # that inspired the test case in the first place, which should also include the PD SPEC!!!
    b. See Justifier ER #2 for potential support.
    c. Leave a comment in the test case request RPD indicating that you have completed the test case and indicate the test case folder id#.
    d. In the resolution section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.
    e. Every test plan must have a 'standard execution' time documented in Justifier.
10) The feature matrix must also be updated with the new test case name and the features. All feature matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).

    a. We will create and enforce QA training on Feature Matrix so that we can remove the peer review check on this item.
11) All new test plans/cases written by *new* Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead doesn't need review.
    a. Optional inclusion of PD/SOE review.
    b. See Justifier ER #5
12) Automated test writing standards for the Justifier
    a. See proposed Spec and discuss how this should work before we write Policy that supports it
        i. Need link from Alisa/Ken:
    b. Right now we have 2 or 3 test plans for the same coverage (1. Manual cases, 2. Automated cases and sometimes 3. Original full test plan)
    c. There should be the possibility of one test plan and we use flags at the test case level to indicate if automated (suppress that step when test plan is run manually) **(test case independence)
    d. If automation breaks we could have the option to show all test cases in manual run.

Brainstorm Session 4 - Automation

Thursday, May 02, 2013 1:38 PM

Test Plan Guidelines

- The purpose of this project is to build "standards" for writing and maintaining test plans
- Test plans need to be effective, so how they're written will impact that
- They need to be clear for the regression team and clear for future review
- The goal obviously to is have great, effective test plans that test the target features , nothing more nothing less
- Managers can use this standard to grade their employees on how well they follow the standard
  - Whether it's Meet Expectations, Needs Improvement or Exceeds (e.g. employee helps create new standards)

Wednesday, April 10, 2013
11:14 AM

Guidelines for Writing Test Cases

1) Test cases need to be written for **workstation ONLY** unless specified otherwise.
2) **Use the Staging folder** within the application library to work on your test cases while they are in progress (aka - NEW).
   a. ** the same could be accomplished with just Status=New, but the Staging folder offers ease of isolation of in-progress test plan development… perhaps we just make the STAGING bullet optional but mandate Status=New
   b. All New test plans should have the status of New once the Regression Review (RPD should be filed to notify the regression team of the testing review and potential release date) is completed then the status is changed to Active
   c. Ex. You *can* use a Staging Copy of Plan to replace the need for "skip step" in cases when "skip" is meant to prepare for a future feature change. Need to do this now until Justifier ER #1 gets implemented sometime in FY14 maybe sorta kinda.
   d. Justifier ER #6 - when Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
3) Test Plan Limits
   a. Each **test plan should have no more than 100 steps**. Create a new test plan if the addition of the new test case would make that test plan > 100 steps.
   b. Each test case should have **no more than 25 steps**.
   c. Do NOT blindly add **steps** to existing **cases** for new functionality - Extreme scrutiny must be applied.
      i. Significant amount of new functionality requires new test **case**. Example RPD:
      ii. However, some new functionality is minor and does not require new cases. Example RPD:
   d. Note that all **Plans** must have **Cases** - no **Steps** can be written directly under a **Plan**
   e. DO NOT try to game these limits. Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
      ii. Needs Justifier ER #8
   f. Exceptions:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale
      ii. Ex. "Workflows" that could have hundreds of cases. They need to be sensibly constructed with Mgr oversight.
4) All Plans must have up to date Standard Execution Time
   a. See Justifier ER #7 to do so at Case level
   b. See Justifier ER #9 for the reporting that will follow this behavior
5) Test "Quality" standards
   a. Write in English and watch your grammar
   b. Again, test case must cover the features/functionality - nothing more nothing less
   c. All steps must have clear and predictable results
      i. e.g. use screenshots to depict said result
      ii. All screenshots must be accompanied by instructions on what they are meant for
   d. Test plans must not require judgment calls to be made by the tester
      i. A good indicator that you're out of bounds is if your test plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
      ii. Avoid steps like "pick any component"- be specific!
   e. No Case/Step should instruct the tester to skip steps. If you do, this indicates something is deficient in your test as written!
6) Naming Conventions for Libraries/Plans/Cases/Step
   a. Library Name must match the Application name (from Usage list of Applications)
      i. If Usage product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
      ii. See Justifier ER #10 to make this easier
   b. Others? **See** Test Case naming conventions **For ideas**
      i. Consider Steps won't have NAME, just referred to by it's Case->Step # nomenclature. This requires justifier web rollout.
   c. Plan and Case Names should be concise
      i. Just a label for the feature being tested
      ii. Not a sentence (use the Description field for that)
      iii. Not an RPD#
7) If your Test Case / Step uses the Linking feature
   a. Understand the linking relationship - the Original source step that Subsequent steps link to is authoritative. So deleting that step will delete linked Subsequent steps!
   b. Defer to your Manager in the use of Linking.
8) Each test case must be independently self-contained with a **defined start and end**.
   a. The first step of the test case should be login to Factset, go to workstation, select the F icon >…
      i. However, Jay thinks Justifier ER #3 should get done in order to eliminate the need for this repetitive step documentation
   b. The last step of the test case should be to close the application and **must state:** select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate.
9) Once the test case is written, you must **document the RPD with the details**.
   a. This means that every new test plan/case must be linked to the ER # that inspired the test case in the first place, which should also include the PD SPEC!!!
   b. See Justifier ER #2 for potential support.
   c. Leave a comment in the test case request RPD indicating that you have completed the test case and indicate the test case folder id#.
   d. In the resolution section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.
   e. Every test plan must have a 'standard execution' time documented in Justifier.
10) The feature matrix must also be updated with the new test case name and the features. All feature matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).
    a. We will create and enforce QA training on Feature Matrix so that we can remove the peer review check on this item.
11) All new test plans/cases written by *new* Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead doesn't need review.
    a. Optional inclusion of PD/SOE review.
    b. See Justifier ER #5
12) Automated test writing standards for the Justifier
    a. See proposed Spec and discuss how this should work before we write Policy that supports it
       i. Need link from Alisa/Ken:
    b. Right now we have 2 or 3 test plans for the same coverage (1. Manual cases, 2. Automated cases and sometimes 3. Original full test plan)
    c. There should be the possibility of one test plan and we use flags at the test case level to indicate if automated (suppress that step when test plan is run manually) **(test case independence)

Automation:

- Need a Naming convention that will allow the Justifier and TC (other automation in Perforce) to match up.

Example:
has:
P4: Directory Name: MSCI
Tellus: name: MSCI
P4Test Case

P4	Tellus	Justifier	Example P4	Example Tellus	Example Justifier
Directory Name	Test Plan	Test Plan	MSCI	MSCI	MSCI Automation
Test Case	Test Case	Test	TestMSCICapitalMarketsTab.sj	TestMSCICapitalMarketsTab| MSCI/	NA

Status:
Can we discuss standards or definitions for the status?

Example: If I am cleaning up a test plan and no longer require the old one, is that considered a retired test plan or inactive?

d. If automation breaks we could have the option to show all test cases in manual run.

What does Justifier need to do to support the Standard?

1. Status is currently only at the Plan level, we need it to also apply to Case (and maybe Step?) RPD:8634680
   a. One cool use of that means that if the Case status is "new", it won't show up on SWP until Lead sets status to "Active".
   b. Also accommodates "go acceptable" scenarios.
2. Linked RPD field at the Plan/Case/Step level should allow for inheritability
   a. If you set an RPD at Plan, then it is assumed that all Cases/Steps "below" the Plan are associated with that RPD.
      i. Is that a one time event or is that RPD added for new test steps and Test Cases?
      ii. I'd think it is a one time event so we can follow what was created when the enhancement went out. If we add more enhancements those will have new RPDs
   b. Lead can set a different RPD link at Case/Step levels of course when new/changed features are introduced over time.
3. To reduce need for "skip", we need to somehow make it clear to the tester what "environment" to run in.
   a. For instance, many cases describe what to install and run, and tell the tester skip based on environment changes like what Atlas version to use.
   b. Merely presenting environment variables might make this more efficient.
4. Consider tracking standard execution field at the Case level.
5. Test peer review enforcer: see RPD:3937252
6. When Case status changes from New to Active, stop Lead if there's no Standard Execution time entered, since that's required.
7. Consider requiring Standard Execution time for the Case level. Then the Plan execution time can be automatically summed up.
8. New pending justifier web testing application must show description field to the tester
9. Create regular report sent to each Product's Lead/Regression partners to analyze the operational difference between the expected Standard time, versus the actual run time, red flagging big diffs. Have this done based on AVG run time, not ABS run time.
10. **Mapper for Library Name <-> Application Name**
    a. **Use the same product picker that's in RPD**
    b. **This will file a new Usage Product request (that jay/bilal approve)**
    c. **This assumes you have agreement with PD/SOE on what that product is**

Questions/Feedback for Session 3

Tuesday, April 23, 2013 2:46 PM

MDQA:

- We should have definitions for Test Plan, Test Case and Test Step
  - Test Plan: A test plan documents the strategy that will be used to verify and ensure that a product or system meets its design specifications and other requirements.
  - Test Case : A specific executable test that examines all aspects including inputs and outputs of a system and then provides a detailed description of the steps that should be taken, the results that should be achieved, and other elements that should be identified.
  - Test Step : Test Step are individual steps required to execute a Test case. Steps explained in a test case include all details even if it they are assumed to be common knowledge.

- It's understood when the policy is approved by the team all "new" test plans follow this best practice. Are old test plans grandfathered until we have time to go back and modify them?

- Team would like the to see adding the new functionality of a status at the test case level for the next iteration of the Justifier. RPD:8634680

- Feedback from Team : Click Here

Feedback from Platform QA:

- We want to ensure we have a consistent format for test steps instructions and expected results, i.e not have too many instructions/expected results crammed into test steps.

- Once the new policy is in effect, team members would need a reasonable amount of time to update existing test plans to compile with the new standards.

Quality Assurance Test Plan Standards

1) FactSet Workstation-based applications must not have Test Plans run under the DIRECTIONS environment without approval from your manager.
2) Newly developed Test Plans and Test Cases:
    a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
    b. Note, Status on Case level is not yet available as of May 1, 2013. This policy will apply to Case upon availability.
    c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
    d. Upon review completion, set Status to Active.
    e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
        i. **New** – New application not ready for testing, but will be shortly.
        ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
            1) This is the only Status that enables the Test plan on a Status Webpage.
        iii. **Inactive** – When you are rewriting or consolidating a test plan.
            1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
            2) e.g.: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
        iv. **Retired** – Used when the application or its test coverage is being retired.
            1) e.g.: Directions version of the Test Plan was retired after discussions with PD.
3) Test Plan Limits:
    a. Each **Test Plan must have no more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
    b. Each Test Case must have **no more than 25 Test Steps**.
    c. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
        i. Extreme scrutiny must be applied.
        ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
        iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
    d. Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**
    e. DO NOT try to game these limits. Test Steps must be concisely written.
        i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
    f. Exceptions to Limits:
        i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
        ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

4) All Test Plans must have up to date Standard Execution Time.
    a. As of May 1, 2013, this is available only at the Test Plan level, but this will apply to Case level in the future.

5) Test "Quality" standards:
    a. Write in English and use proper grammar.
    b. Test Case must cover the features/functionality - nothing more nothing less.
    c. All Test Steps must have clear and predictable results.
        i. e.g. use screenshots to depict said result.
        ii. All screenshots must be accompanied by instructions on what they are meant for.
    d. Test Plans must not require judgment calls to be made by the tester.
        i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
        ii. Avoid Steps like "pick any component"- be specific!
    e. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test!

6) Naming Conventions for Libraries/Plans/Cases/Step:
    a. Library Name must match the Application name (from Usage list of Applications).
        i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!

b. Test Plan and Case Names should be concise:
    i. Names are merely labels for the feature being tested.
        1) A Name is not a sentence (use the Description field for that)..
        2) A Name is not an RPD#.

7) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

8) Each Test Case must be independent and self-contained with a **defined start and end**.
    a. The first Step of the Test Case should be "login to Factset, go to workstation, select the F icon >…"
    b. The last Step of the Test Case should be to close the application and **must state:** "select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate."

9) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
    a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
    b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
    c. In the Resolution Section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.

10) The Feature Matrix must be updated with the new Test Case name and respective features.

    a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).

11) All new Test Plans/Cases written by *new* QA Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead no longer needs review.
    a. Optional inclusion of PD/SOE review.

Revised May 1, 2013

Quality Assurance Test Plan Standards v.2

Tuesday, June 18, 2013 8:17 PM

1) FactSet Workstation-based applications must not have Test Plans run under the DIRECTIONS environment without approval from your manager.
2) Newly developed Test Plans and Test Cases:
    a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
    b. Note, Status on Case level is not yet available as of May 1, 2013. This policy will apply to Case upon availability.
    c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
    d. Upon review completion, set Status to Active.
    e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
        i. **New** – New application not ready for testing, but will be shortly.
        ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
            1) This is the only Status that enables the Test plan on a Status Webpage.
        iii. **Inactive** – When you are rewriting or consolidating a test plan.
            1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
            2) e.g.: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
        iv. **Retired** – Used when the application or its test coverage is being retired.
            1) e.g.: Directions version of the Test Plan was retired after discussions with PD.
3) Test Plan Limits:
    a. Each **Test Plan must have no more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
    b. Each Test Case must have **no more than 25 Test Steps**.
    c. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
        i. Extreme scrutiny must be applied.
        ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
        iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
    d. Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**
    e. DO NOT try to game these limits. Test Steps must be concisely written.
        i. But you SHOULD offer comprehensive information in the description field to provide context to the tester.
    f. Exceptions to Limits:
        i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
        ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

4) All Test Plans must have up to date Standard Execution Time.
    a. As of May 1, 2013, this is available only at the Test Plan level, but this will apply to Case level in the future.

5) Test "Quality" standards:
    a. Write in English and use proper grammar.
    b. Test Case must cover the features/functionality - nothing more nothing less.
    c. All Test Steps must have clear and predictable results.
        i. e.g. use screenshots to depict said result.
        ii. All screenshots must be accompanied by instructions on what they are meant for.
    d. Test Plans must not require judgment calls to be made by the tester.
        i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
        ii. Avoid Steps like "pick any component"- be specific!
    e. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test!

6) Naming Conventions for Libraries/Plans/Cases/Step:
    a. Library Name must match the Application name (from Usage list of Applications).
        i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
    b. Test Plan and Case Names should be concise:
        i. Names are merely labels for the feature being tested.
            1) A Name is not a sentence (use the Description field for that)..
            2) A Name is not an RPD#.
7) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

8) Each Test Case must be independent and self-contained with a **defined start and end, so it can be run entirely on its own**.
    a. The first Step of the Test Case must not rely on the previous Test Case
    b. The last Step of the Test Case must not be needed for the following test case and **must state:** "select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate."

9) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
    a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!

We need to require the Description Field be filled in at the Test Case Level. It should describe the intent/reason for the test case. Should full out what is required in a description.

b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
c. In the Resolution Section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.

10) The Feature Matrix must be updated with the new Test Case name and respective features.

a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #10).

11) All new Test Plans/Cases written by *new* QA Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead no longer needs review.
a. Optional inclusion of PD/SOE review.

Quality Assurance Test Plan Standards v.3 DRAFT

Tuesday, June 18, 2013 8:17 PM

1) FactSet Workstation-based applications must not have Test Plans run under the DIRECTIONS environment without approval from your manager.

2) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of May 1, 2013. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the only Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

3) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
   d. Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**
   e. DO NOT try to game these limits. Test Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   f. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

4) All Test Plans must have up to date Standard Execution Time.
   a. As of May 1, 2013, this is available only at the Test Plan level, but this will apply to Case level in the future.

5) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Test Case must cover the features/functionality - nothing more nothing less.
   d. All Test Steps must have clear and predictable results.
      i. All screenshots must be accompanied by instructions on what they are meant for.
      ii. Hint 1: Use screenshots to depict said result.
      iii. Hint 2: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      iv. Hint 3: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Plans must not require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This

is related to "skip step" below.
   ii. Avoid Steps like "pick any component"- be specific!
f. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test!

6) Naming Conventions for Libraries/Plans/Cases/Step:
   a. Library Name must match the Application name (from Usage list of Applications).
      i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
   b. Test Plan and Case Names should be concise:
      i. Names are merely labels for the feature being tested.
         1) A Name is not a sentence (use the Description field for that)..
         2) A Name is not an RPD#.

7) If your Test Case / Step uses the Linking feature:
   a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
   b. Defer to your Manager in the use of Linking.

8) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
   a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a specific sequence in order to run. Each Case is able to stand on its own.

      When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.
   b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
   c. Suggested approaches for the defined start and end to a Test Case:

      Test Case 1:

      Step 1 (i.e. "start"):
      - Click on the F button
      - Click on the application
      - make sure all other previous applications are closed

      Step 2, 3, 4, etc.

      Step (i.e. "end"):
      - Close the application
      - Select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate

      The use of Linking Test Cases/Steps can be very powerful when the above Test Case is to be repeated within a Test Plan. See #7 above for Linking guidance.

9) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
   a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
   b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
   c. In the Resolution Section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.

10) The Feature Matrix must be updated with the new Test Case name and respective features.

   a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).

11) All new Test Plans/Cases written by *new* QA Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead no longer needs review.
   a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
   b. Optional inclusion of PD/SOE review.

Quality Assurance Test Plan Standards v.3

Friday, November 01, 2013

1) FactSet Workstation-based applications must not have Test Plans run under the DIRECTIONS environment without approval from your manager.

2) Newly developed Test Plans and Test Cases:
   a. Use the Staging folder within the Application Library to work on your Test Cases before implementing them in production.
   b. Note, Status on Case level is not yet available as of May 1, 2013. This policy will apply to Case upon availability.
   c. RPD (**Product: Regression Coverage Change Control**) must be filed to notify the Regression Team of the testing review and potential release date).
   d. Upon review completion, set Status to Active.
   e. Use the following Justifier Test Plan & Test Case Status definition appropriately:
      i. **New** – New application not ready for testing, but will be shortly.
      ii. **Active** – Test Plans that are ready to run during regression or prebuilds.
         1) This is the *only* Status that enables the Test plan on a Status Webpage.
      iii. **Inactive** – When you are rewriting or consolidating a test plan.
         1) e.g.: You have the same Test Plan in the same Library and you want to consolidate the Test Plan.
         2) e.g.: The application has a new look and feel, but you are not double testing (you did not have two Test Plans on the Status Webpage that were being tested.)
      iv. **Retired** – Used when the application or its test coverage is being retired.
         1) e.g.: Directions version of the Test Plan was retired after discussions with PD.

3) Test Plan Limits:
   a. Each **Test Plan should not contain more than 100 Test Steps**. Create a new Test Plan if additional Test Cases would cause Test Plan to exceed 100 Test Steps.
   b. Each Test Case must have **no more than 25 Test Steps**.
   c. Do NOT blindly add **Test Steps** to existing **Test Cases** for new functionality.
      i. Extreme scrutiny must be applied.
      ii. Significant amount of new functionality requires new **Test Case**. e.g.: RPD:5108429
      iii. However, some new functionality is minor and does not require new **Test Cases**. e.g.: RPD:5713279
   d. Note that all **Test Plans** must have **Test Cases** - no **Test Steps** can be written directly under a **Test Plan**
   e. DO NOT try to game these limits. Test Steps must be concisely written.
      i. But you SHOULD offer comprehensive information in the description field (see #5) to provide context to the tester.
   f. Exceptions to Limits:
      i. **Require QA Mgr Approval before implementation**, provide supporting rationale.
      ii. Ex. "Workflows" that could have hundreds of Test Cases. They need to be sensibly constructed with Manager oversight.

4) All Test Plans must have up to date Standard Execution Time.
   a. As of May 1, 2013, this is available only at the Test Plan level, but this will apply to Case level in the future.

5) Test "Quality" standards:
   a. Write in English and use proper grammar.
   b. All Test Plans and Cases must include informative context in the Description field.
      i. Describe what you are testing and why, i.e. what is the goal of this test? What will be accomplished by executing this Test?
      ii. Do not merely reiterate the Test Plan 'Name' in the Description field, for example.
      iii. Consider the Description to be the authoritative summary for the purpose of all included Test Cases and Steps, adequate to anyone in PD or SOE to understand what is being tested.
   c. Test Case must cover the features/functionality - nothing more nothing less.
   d. All Test Steps must have clear and predictable results.
      i. All screenshots must be accompanied by instructions on what they are meant for.
      ii. Hint 1: Use screenshots to depict said result.
      iii. Hint 2: Screenshots should isolate focus on particular area of test, and NOT an entire desktop.
      iv. Hint 3: Use of colored circles/boxes/arrows to draw attention to area of test is also recommended.
   e. Test Plans must not require judgment calls to be made by the tester.
      i. A good indicator that you're out of bounds is if your Test Plan has a lot of "if… then… or… if… then… etc" choices. This is related to "skip step" below.
      ii. Avoid Steps like "pick any component" - be specific!
   f. No Case/Step should instruct the tester to skip steps. Doing so indicates something is deficient in your test!

6) Naming Conventions for Libraries/Plans/Cases/Step:

a. Library Name must match the Application name (from Usage list of Applications).
    i. If Usage Product doesn't exist to map to, cooperate with the SOE/PD lead to create a new Usage Product!
b. Test Plan and Case Names should be concise:
    i. Names are merely labels for the feature being tested.
        1) A Name is not a sentence (use the Description field for that)..
        2) A Name is not an RPD#.

7) If your Test Case / Step uses the Linking feature:
    a. Understand the Linking relationship - the *original source* Test Step that *subsequent* Steps link to is authoritative. So deleting *original source* Step will delete linked *subsequent* Steps.
    b. Defer to your Manager in the use of Linking.

8) Each Test Case must be independent and self-contained **so it can be run entirely on its own**.
    a. It is understood that a Test Plan often tests a significant piece of functionality, which often requires multiple Test Cases to accomplish that goal. The spirit of this standard is to ensure that Cases do not have to be run in a lengthy specific sequence in order to run. Each Case is able to stand on its own.

       When taking into account the execution of your Test Case however, realize that each Test Step must be "passed" or "failed", and in the context of a full Test Plan, those Test Steps can become numerous and unnecessarily repetitive if you aren't careful.

    b. The first Step of a Test Case must not rely on Steps in the previous Test Case, and the final Step must not dictate activity in the subsequent Test Case.
    c. Suggested approaches for the defined start and end to a Test Case:

        Test Case 1:

            Step 1 (i.e. "start"):
            - Click on the F button
            - Click on the application
            - make sure all other previous applications are closed

            Step 2, 3, 4, etc.

            Step (i.e. "end"):
            - Close the application
            - Select NO if you are asked to SAVE the workspace or data, or screens, or anything else appropriate

        The use of Linking Test Cases/Steps can be very powerful if the above Test Case example were to be repeated within a Test Plan. See #7 above for Linking guidance.

9) Once the Test Case is written, you must **document related RPD(s) with appropriate details**.
    a. This means that every new Test Plan/Case must be linked to the ER # that inspired the Test Case in the first place, which should also include the PD SPEC!
    b. Leave a comment in the Test Case's requesting RPD indicating that you have completed the test case and indicate the Test Case folder id#.
    c. In the Resolution Section of the RPD (top right corner), mark the relevant fields such as Case/Step ID.

10) The Feature Matrix must be updated with the new Test Case name and respective features.

    a. All Feature Matrix additions need to be peer reviewed until QA Mgr deems review no longer necessary (same as #11).

11) All new Test Plans/Cases written by *new* QA Leads must be peer reviewed before implementation until their QA Mgr *and* Regression Team Lead deems the Lead no longer needs review.
    a. Regression Team reviewers are expected to provide critical and crucial feedback on the quality of all Test Plans.
    b. Optional inclusion of PD/SOE review.