Welcome. In the next few moments I would like to demonstrate to you how IBM Rational Functional Tester can help ensure the quality of SAP systems you may be deploying in your organization.

Today's business environment is putting more and more demands on testing. Organizations need to reduce cost. Manual testing simply can't scale to the levels necessary to maintain high quality service to your customers and releasing low quality applications is simply not an option. Software solutions today are complex compositions of many components composed of legacy, homegrown, SOA and packaged applications. Utilizing unique tooling for each environment wastes valuable cash and time. We can't afford to reinvent the wheel. Quality must be part of the day to day culture of every team member. This drives the need for better collaboration and the facilitation of reuse to shorten test cycles. Functional test automation is one of the components of that quality management culture we will focus on in this video.

Of the many capabilities Rational Functional Tester provides, we will explore the basic workflow of creating an automated test by simply recording what you do manually, enhancing that test through the use of easy, intuitive user interfaces, executing an enhanced test, and exploring the execution results.

Let's get started.

Here you see the user interface of IBM Rational Functional Tester built on the Eclipse framework. There is not a lot to see here until we record a test so let's go ahead and start recording a functional test script. We simply give it a name and click finish to begin the recording.

Functional Tester minimizes itself and we can begin interacting with our SAP GUI interface just as we normally would for any manual testing. Here you are going to select the "LECI – Check in Means of Transport and Visitors" transaction. As the screens appears, Functional Tester is constantly recording everything that is going on including all the interactions you are doing. We will change the Checkpoint to "Main Gate" and fill in a Check-in date using the calendar control, provide a check-in time for the vehicle, and the truck license number and drivers last name which are required fields.

It is important to consider we may want to reuse this test with different data. We don't want to create an entirely new test simply to change the truck license number or drivers last name. For that reason Rational Functional Tester includes a data driven wizard that can be operated during record time. Here you see we are placing data driven commands on the truck license number and drivers last name so that those values can be pulled from a data pool or a separate collection of data later on during playback.

We now save this transaction and finally we want to add a verification point for the status message at the bottom of the screen. We want to verify that each time we play this test back we get the correct status message at the bottom of the screen. Now the

tricky part here is that we are working with dynamic data. Each time you play back this test the sequence number will be different. Rational Functional Tester has built-in capabilities to support regular expressions. Regular expressions are pattern matching technology. You simply change the data into a regular expression and use the built-in commands to help build your regular expression. Here we are just inserting three digit character placeholders. If you are unsure you have the right combination here you can use the regular expression evaluator to make sure you have got a match and try various combinations.

This ability to create tests that are independent of specific data elements is a key feature of Rational Functional Tester enabling us to create more independent, reusable tests.

Now, back in our recording, we are going to exit the transaction and then re-enter the same transaction using the command entry area. We are going to search for the truck we recently checked in so that we can check it out. We will fill in a license plate number and name. Here again we realize we will want to data drive this information later on from a data pool.

We will now click the find button to locate our record and enable it for editing through the display change button. Now we will simply input the current date and time for Check-out time and save the record. That is it for our scenario so we will exit back out to the main screen of the SAP GUI and stop recording.

We can now see what Rational Functional Tester has captured during recording. Let's take a look at the script first. See the natural language used to depict the operations that were recorded. Operations are organized by group or by screen that was accessed. As you navigate through the test you can see that the screenshots are the visuals on the right hand side which are captured right along with the operations. These natural language test steps are editable. For example we can change the last name of the driver at this point. Notice also how the related GUI element is highlighted with a blue box in the visual on the right hand side.

Now remember our data pool elements that we inserted. Here we see those highlighted in green. They are related to the data value which has been captured in the data pool. We can effectively double the coverage of this test simply by adding a second row in our data pool. During test playback the test can iterate for both of these values.

Now let's scroll to the bottom of our test to that last screen where we confirm the exit of the truck. Let's say that you have forgotten that you would like to put in a verification point in this final step to verify that status message. Rational Functional Testers visuals are actually far more than simple screenshots. Here you can see how you can highlight these elements in the screen and insert actions or even verification points without even re-recording. Here we will insert a verification point similar to what we had done before. It now appears in our list of commands for our test.

There is one final item we want to do before we playback our test. Because we are using a data pool we no longer need the hard coded values in our test. Here you see from the context menu on each operation we can simply disable. This means we can re-enable at any time if we choose to.

Let's save the test and begin playback.

In the playback wizard we can choose the name for the log and we tell Rational Functional Tester we would like to iterate through all the rows in the data pool. Now it is just a matter of sitting back and watching Rational Functional Tester play back the automated tests hands free.

If you are quick enough you will see that the first iteration of our test used the new value of the data pool that we inserted. It will iterate twice, once for each row in the data pool. When playback is complete, Rational Functional Tester will bring the log viewer up in the browser.

Rational Functional Tester can display results in a number of different log formats. This particular format that I have chosen is an XML format using DOJO technology. The most interesting thing we want to research here in our results is this failure we seem to have found. When we click the view results button we are given a Verification Point Comparator where we can quickly highlight the exact differences. We see the reason the test failed is simply because we did not put a regular expression in that new verification point that we inserted.

I hope this brief video has given you an introduction to how you can use Rational Functional Tester to ensure the functional quality of your SAP systems. Keep in mind that everything that you saw today was done with no coding or scripting. This means your domain experts can stay domain experts without becoming programmers to test their SAP systems.

Also remember that functional verification is one part of the full line of quality management solutions offered by IBM Rational. Visit us soon on ibm.com.