Test Automation Patterns Exercise

The following are possible test automation scenarios for (fictitious) organisations.

Choose one or more and explore TestAutomationPatterns.wikispaces.com to help them identify and solve their problems. Some solution ideas will be given.

Scenario 1: Nancy Naive

Nancy is new to test automation. The company bought a test execution tool and she was given the task of automating the manual regression tests. She was shown a demo of the tool and was very impressed with the capture replay idea, so she began recording the regression tests as she performed them for the next release of the system.

However, when the next release was ready, she found that many of the recorded tests didn't actually play back or didn't do what they should. She spent 2 days trying to fix the tests, and ended up re-recording some of them. She feels that this is too much time to spend on maintaining the tests! She is now concerned about what will happen the next time the system changes and wants to make sure that her test maintenance costs are kept to a reasonable level.

Using the diagnostics, what is Nancy's main issue?

What pattern(s) would be most appropriate for her to apply?

Scenario 2: Ivan Indispensible

Ivan is the test architect and test automator for the project. He designed the automation using good programming practices, since he has a good background as a developer. His scripts are well-structured, and there is little if any duplication because he knows to create a new script with the common code which is called by each test that needs to do whatever is in that common script. The automation is a great success, with more and more tests being automated.

But the problem is that Ivan is getting more and more busy working on more and more tests that people now want him to automate, and he has less and less time for script maintenance. There are some scripts that could do with being re-factored, some common areas that should now be made into common scripts, and some tests that are failing because the applications being tested have changed in ways that the scripts didn't plan to cope with. Ivan is getting stressed! No one else has the technical expertise to do the maintenance, and Ivan is too busy writing new automated tests.

Using the diagnostics, what is Ivan's main issue?

What pattern(s) would be most appropriate for him to apply?

Scenario 3: Polly Pressured

Polly is under a great deal of pressure. The automation is well structured and working fairly well, but there are always a few things that need to be seen to just to make sure that it will continue to work when each new release. Polly has many other things to do, but knows that this maintenance of the automation, if not done on a regular basis, will soon build up into a mountain of technical debt, so she is keen to keep on top of it.

The problem is that her manager can't see the value of putting any time into the automation, since he has already been told that it is working well. He doesn't understand that maintenance is needed – surely the automation was designed to be maintenance-free from the start, wasn't it? Anyway, a few minor problems aren't important, Polly – you need to get on with your other work which is much more important and urgent.

Using the diagnostics, what is Polly's main issue?

What pattern(s) would be most appropriate for her to apply?

Scenario 4: Sam Starting

Sam is just starting out on automation. He has done some investigation of commercial and open source tools and is eager to get started. He is confident that automation will be a great thing for his company because he and the other testers currently spend a lot of their time doing tedious and repetitious regression tests.

Using the diagnostics, what is Sam's main issue?

What pattern(s) would be most appropriate for him to apply?

Scenario 5: Ted Toolswitcher

Ted's automation was going just great – lots of tests automated, regression suites run every night, scripts were well-structured, and Ted was really happy with the tool they have build their tests in. It was a well-established tool and they were confident if would be around for a long time – that's why they went for it.

But now there is a problem. The tool development is headed in one direction, and the applications Ted is testing are going in a completely different direction. Ted has tried to convince the tool vendors to support their company, but they won't do it. They will no longer support the version Ted is using, which is several releases out of date already, since the new features in the tool were not compatible with their tests. What should Ted do? Does he need to start again from scratch with a new tool? How can he make sure things will be better next time?

Using the diagnostics, what is Ted's main issue?

What pattern(s) would be most appropriate for him to apply?

Test Automation Patterns: Solution Ideas

Scenario 1: Nancy Naive

Nancy is new to test automation. The company bought a test execution tool and she was given the task of automating the manual regression tests. She was shown a demo of the tool and was very impressed with the capture replay idea, so she began recording the regression tests as she performed them for the next release of the system.

However, when the next release was ready, she found that many of the recorded tests didn't actually play back or didn't do what they should. She spent 2 days trying to fix the tests, and ended up re-recording some of them. She feels that this is too much time to spend on maintaining the tests! She is now concerned about what will happen the next time the system changes and wants to make sure that her test maintenance costs are kept to a reasonable level.

Using the diagnostics, what is Nancy's main issue?

There are a number of aspects in Nancy's case.

- *LIMITED EXPERIENCE* is one possible starting issue, but she may not even realise this at this time.
- She definitely does want to improve test automation, so would probably choose this option.
- On the "Improve or Revive" page, the most relevant selection would be "Unsatisfactory quality of automation"
- On the "Unsatisfactory Quality" page, the cost of maintenance is her main concern, so *HIGH MAINTENANCE COSTS* is the issue that she wants to deal with.
- On the *HIGH MAINTENANC COSTS* page, the issue is *COSTLY SCRIPT ADAPTATIONS*, and on that page, *BRITTLE SCRIPTS* is the most appropriate issue.

What pattern(s) would be most appropriate for her to apply?

- the main thing Nancy wants to do is to get more maintainable automation, so the pattern MAINTAINABLE TESTWARE would be a good choice for the first pattern to look at.
- there is probably a lot of duplication as the scripts have not been designed well ("spaghetti scripts"), so using GOOD PROGRAMMING PRACTICES is also important. The scripts need to be designed according to what developers have learned over the years. This will lead on to other patterns such as DESIGN FOR REUSE, etc.
- it would also be useful to consider DATA-DRIVEN TESTING and KEYWORD-DRIVEN TESTING as ways to build MAINTAINABLE TESTWARE.

Scenario 2: Ivan Indispensible

Ivan is the test architect and test automator for the project. He designed the automation using good programming practices, since he has a good background as a developer. His scripts are well-structured, and there is little if any duplication because he knows to create a new script with the common code which is called by each test that needs to do whatever is in that common script. The automation is a great success, with more and more tests being automated.

But the problem is that Ivan is getting more and more busy working on more and more tests that people now want him to automate, and he has less and less time for script maintenance. There are some scripts that could do with being re-factored, some common areas that should now be made into common scripts, and some tests that are failing because the applications being tested have changed in ways that the scripts didn't plan to cope with. Ivan is getting stressed! No one else has the technical expertise to do the maintenance, and Ivan is too busy writing new automated tests.

Using the diagnostics, what is Ivan's main issue?

- Ivan does want to improve the test automation, so this would be his choice from the first Diagnostic page
- On the "Improve" page, there are a number of choices that could be applicable. Probably the best one is "Lack of support" because Ivan is having to do everything himself.
- On the "Lack of support" page, perhaps Ivan feels that the testers aren't providing sufficient support for him, leaving him to do all of the automation work on the tests that should be automated.
- On the "testers don't help" page, the issue *NON-TECHNICAL-TESTERS* is probably the most relevant. The testers may be willing to help with the automation, but it requires technical skills that they don't have.

What pattern(s) would be most appropriate for him to apply?

- Looking at the issue *NON-TECHNICAL-TESTERS,* the first one is DOMAIN-DRIVEN TESTING. This is where the automation is designed so that testers without programming/scripting skills are able to write (and run) automated tests. If the testers could write the new automated tests, this would free Ivan to work on maintenance of the tests. Note that Ivan would have to design a framework for the automation (a "front end") so that the testers were able to write the tests, so TEST AUTOMATION FRAMEWORK would also be needed.

Scenario 3: Polly Pressured

Polly is under a great deal of pressure. The automation is well structured and working fairly well, but there are always a few things that need to be seen to just to make sure that it will continue to work when each new release. Polly has many other things to do, but knows that this maintenance of the automation, if not done on a regular basis, will soon build up into a mountain of technical debt, so she is keen to keep on top of it.

The problem is that her manager can't see the value of putting any time into the automation, since he has already been told that it is working well. He doesn't understand that maintenance is needed – surely the automation was designed to be maintenance-free from the start, wasn't it? Anyway, a few minor problems aren't important, Polly – you need to get on with your other work which is much more important and urgent.

Using the diagnostics, what is Polly's main issue?

- Polly does want to improve her automation, so this is the first option for her
- On the "Improve" page, "Lack of support" could be Polly's main issue
- On the "Lack of support" page, the first option may be the nearest – managers don't see the value of automation. In this case, managers don't see the value of maintenance of the automation. This could lead to automation isn't what they were expecting, and hence to *UNREALISTIC EXPECTATIONS*
- An alternative from the "Improve" page, would be "Expectations for automation not met"
- On the "expectations not met" page, "Maintenance of the automation is too expensive" leading to either "Updating the automation scripts" or "Refactoring the automation scripts" (possibly both of these).
- From "Updating the automation scripts", the most likely issue is *BRITTLE SCRIPS*.
- From "Refactoring the automation scripts", the most likely issue is *UNREALISTIC EXPECTATIONS* (which we saw before)

What pattern(s) would be most appropriate for her to apply?

- from *UNREALISTIC EXPECTATIONS*, the pattern SHARE INFORMATION would be useful, applied specifically to the building up of technical debt.
- from *BRITTLE SCRIPTS,* the pattern MANAGEMENT SUPPORT is the most relevant. In this pattern, SELL THE BENEFITS might be helpful, focused on the benefits of keeping up with maintenance and avoiding technical debt.

Scenario 4: Sam Starting

Sam is just starting out on automation. He has done some investigation of commercial and open source tools and is eager to get started. He is confident that automation will be a great thing for his company because he and the other testers currently spend a lot of their time doing tedious and repetitious regression tests.

Using the diagnostics, what is Sam's main issue?

- this scenario is fairly easy to identify: Sam has not done automation before, so the issue is *NO PREVIOUS TEST AUTOMATION*

What pattern(s) would be most appropriate for him to apply?

- in this case, all of the patterns listed in the issue are relevant for Sam. This is the only issue that recommends the patterns to be done in a particular order, but here it is important to know what the goals are for automation before your start, for example.

Scenario 5: Ted Toolswitcher

Ted's automation was going just great – lots of tests automated, regression suites run every night, scripts were well-structured, and Ted was really happy with the tool they have build their tests in. It was a well-established tool and they were confident if would be around for a long time – that's why they went for it.

But now there is a problem. The tool development is headed in one direction, and the applications Ted is testing are going in a completely different direction. Ted has tried to convince the tool vendors to support their company, but they won't do it. They will no longer support the version Ted is using, which is several releases out of date already, since the new features in the tool were not compatible with their tests. What should Ted do? Does he need to start again from scratch with a new tool? How can he make sure things will be better next time?

Using the diagnostics, what is Ted's main issue?

- Ted wants to improve his automation
- From the "improve" page, the best option for Ted is "Unsatisfactory quality of automation.
- On the "Unsatisfactory quality of automation" page, the issue *TOOL DEPENDENCY* is the most appropriate. The issue INADEQUATE TOOLS may also be useful.

What pattern(s) would be most appropriate for him to apply?

- TOOL INDEPENDENCE looks like the best pattern to apply.
- Having a TEST AUTOMATION FRAMEWORK that supports KEYWORD-DRIVEN TESTING or DOMAIN-DRIVEN TESTING will take the tests away from the implementation and any specific tool.
- The TEST AUTOMATION FRAMEWORK pattern talks about designing your TESTWARE ARCHITECTURE using ABSTRACTION LEVELS.
- The issue *INADEQUATE TOOLS* mentions some other useful patterns: RIGHT TOOLS will help in choosing their next tool and implementing it in a less tool-specific way.. The pattern TEST AUTOMATION FRAMEWORK is also mentioned here. Another very useful pattern is LEARN FROM MISTAKES.

Although this will take a fair amount of effort now, effort is essential in any case to change to a new tool. If the testware is re-designed as it is converted, this will prevent such problems in the future (or at least make them not as serious).