



W3QA

The Business Case for On-Demand Test Automation

The Business Case for On-Demand Test Automation

Document Title			
The Business Case for On-Demand Test Automation.			
Document Abstract			
Hrmes is the Human Readable Machine Executable Specification (pronounced Hermes) Test Framework designed and developed by Andrew Thompson.			
Document History			
Date	HRMES Version	Purpose of Revision	Author
14/02/2009	HRMES.0.1.0	This revision supports the initial release HRMES.	Andrew Thompson

Contents

1 Introduction.....4

2 Investment With Returns.....5

2.1 Higher Productivity Through Specialisation.....8

2.2 Effective Outsourcing.....9

2.3 Scalability.....10

2.4 Conclusion.....10

1 Introduction

The business case for on-demand test automation is outlined in this document. Although, founded on the general case for test automation, it challenges the validity of its underlying assumptions.

Successful test automation projects have clear design objectives and advanced frameworks that deliver them.

Test automation frameworks need to be designed, developed and implemented along the lines of any other software development project.

Automation projects conceived without thorough consideration will fail. This is especially so for automation frameworks that need to operate at the system test or UAT level, where there is little established process or methodology.

Failure can be avoided. This document sets out the benefits of on-demand test automation and the need for a well designed automation framework to deliver all of its potential.

2 Investment With Returns

Investing in test automation can deliver high returns on investment (ROI) across a number of areas.

Returns come from three sources:

1. Decreased spending on testing
2. Faster "time to market"
3. Improvements in product quality

The foundation of any business case is the model that predicts financial savings.

The standard ROI model assumes two things. Firstly, there is an initial setup cost associated with the investment in automation. This is over and above that spent on manual testing. The second assumption is that test execution cycles get compressed when you automate, i.e. it's much faster than manual testing.

An automated approach to testing will always start as the more expensive option. However, there is a point when the total cost of test automation is soon eclipsed by the higher marginal cost of manual testing. After this point automation returns greater and greater savings.

When this will happen depends on factors such as the type of system under test and the software development approach being used.

Many test automation vendors suggest a return on investment before the sixth test cycle. That may be within six months or six weeks - depending on how often you perform releases requiring system testing. This is a timescale that most organisations find acceptable. It is also why most who can invest; do so.

Earlier And Greater ROI

Test execution through Selenium is rapid. It is entirely feasible to see days of manual test execution replaced by automation cycles of less than an hour.

Automation frameworks, properly implemented will accurately identify failure and success. This leaves manual test resources simply to verify any failure and raise defects on the back of automation reports. Automation has the potential to massively compress test execution cycles and deliver very effective QA.

Test automation can achieve much, including significant financial savings.

A Closer Look At Those Assumptions

Whether faster test cycles will equate to lower costs, as the standard ROI model assumes, depends on how efficiently you are able to manage your test resources.

In practice, test resources are not as flexible as the model suggests. When you move test resources in and around different project activities and across different projects, there is usually a high degree of redundancy involved.

The model assumes you stop paying for test resources when you stop running your automated tests. This is the type of utility access that has just recently become widely available.

The main fallacy of the standard model is that without access to test resources on-demand, the full financial gain from automation will be allusive.

Hrmes is designed to provide utility access to test automation services. You only pay for the test resources you use. This enables you to capture all the potential savings from automation.

Reduced Start-up Costs

If you are able to reduce the start-up costs of your automation project you will achieve a return on investment sooner.

With the advent of Selenium and cost free, ready to use frameworks, this is now achievable.

License fees for automation tools are expensive. Despite being generally portrayed as a "one-off" they tend to be perpetual.

This is the second fallacy of the standard ROI model. Vendor license fees and involvement with vendors is continual. They are a significant share of all on-going costs, not just start-up costs.

There is another cost that usually gets packaged into start-up costs. Costs associated with converting manual data scripts into automated coded versions are significant and continual too. These are also considered as a one-off cost and part of the initial investment. They can be avoided completely.

The Hrmes test automation framework is built on Selenium. Neither have license fees. Hrmes allows test analysts to produce a singular mutable test specification. It is usable for both manual and automated testing.

The Hrmes format is generated via a free plug-in to the Firefox web browser - the Selenium IDE. The format is simple, flexible and accessible. It comes ready to load via commonly used spreadsheet packages such as MS Excel or OpenOffices' Calc.

These factors make Hrmes the most affordable means to an execution-ready test automation framework.

Using Hrmes significantly improves the ROI by radically cutting start-up costs.

Reduced Maintenance Costs

Maintaining manual test artefacts is generally regarded as less costly. It is also seen as far slower than automated artefacts. The downside for automation is that specialist automation skills are required to change code based artefacts.

The two approaches would appear to have balanced costs in this area. This probably explains why the cost of maintenance rarely gets considered in ROI models.

When considering ROI, it is critical to bear in mind maintenance costs over time and what impact different maintenance profiles have on cost.

The Hrmes framework is code light and data driven. Changes will rarely demand specialist knowledge of automation tools. Nor will they incur the higher costs normally associated with code maintenance and release management. Maintenance events will largely require changes to test data rather than test code. This will reduce maintenance costs and deliver a far greater ROI than code-driven automation frameworks.

Designing And Developing A Framework

An automation project needs well thought out objectives. Without thorough consideration to what you want automation to deliver, it will not do it. An automation project also needs a framework in which to operate and deliver these objectives.

Designing and developing a framework is generally overlooked, so there are no cost implications initially.

However, it is when you need to scale that the cost of having no design will start to put demands on your test budget.

If you accept the need to have a framework, things will work out differently. There are many choices in this area. You may be able to commit time and resources into implementing your own framework. Although, such a framework may have advantages from being bespoke, this approach will have serious implications on the cost and the inception time of your automation project. Alternatively, you could use an existing framework.

Hrmes is a framework with many advanced features and functionality. It is ready built, free and fit for purpose for all types of testing across all test levels.

Summarising The Case On Cost

The Hrmes framework radically reduces costs across a number of areas by providing:

- Utility access to automation which means you only pay for what you use
- License free software which radically cuts start-up costs
- A data-driven framework which minimises maintenance costs
- A free framework that has been designed and built to deliver advanced automation objectives

2.1 Higher Productivity Through Specialisation

Role-based Frameworks

The ability of the Hrmes format to abstract implementation from test specifications provides an opportunity to distribute test process and tasks. It allows specialists to engage in a highly efficient and productive manner.

Vendors of test automation software have over-sold the involvement of the inexperienced. Testers, regardless of experience, are often expected to become instant practitioners in test automation.

Wherever you find failing test automation projects, you will also find test automation novices working without a methodology or framework and operating way beyond their skill set. This is often a default approach to test automation at the system test level. As a general practice it is a disaster.

Test automation is a specialist function, regardless of which framework or tool is used. Testing complex requirements against complex systems in an automated way could hardly be anything else.

Many test automation projects struggle simply by under appreciating the absolute need for specialist skills and specialisation. Without doing so, neither good test analysis nor good engineering is delivered.

Role-based testing acknowledges the specialisations that naturally exist within the test automation function.

Avoiding Silos

The Hrmes approach facilitates the benefits of specialisation. It avoids the creation of silos or obfuscation of test specifications.

The clarity of the original test specification is preserved continuously through a single artefact. What gets defined - gets executed "as is".

The Hrmes model abstracts test specification from implementation and is complementary to role based testing. Its adoption will allow greater specialisation to occur where it is needed. It will enable people to contribute with what they do best.

Only a framework that recognises the need for specialisation and role based testing can deliver the highest levels of productivity. To fully realise this benefit, test automation frameworks have to fully support this - by design.

The Hrmes framework is unified across test levels. Its test artefacts are common, shared and transferable. The entire process empowers domain expertise, delivers specialisation and very high productivity.

2.2 Effective Outsourcing

Distributable and Shared Artefacts

When test specifications are merged into automation code, distributing test execution as a distinct component is difficult. The Hrmes framework supports a unified approach to testing, consisting of distinct distributable specialist components.

Hrmes is an “Executable Specification”, where business domain specialists define tests in a domain specific language (DSL) and automation specialists ensure they are executed as defined.

This distributed design presents a great opportunity for outsourcing to a specialist supplier, whilst retaining business domain expertise.

Flexible and Focused Outsourced Engagement: On-Demand Test Automation

Testing has extremes of activity. Flexible resourcing is ideally suited to the nature of testing. Outsourcing can facilitate this, but is not always possible.

Consider the following examples:

A project is using an agile methodology and going through four weekly incremental deliveries. As the project matures, the cumulative affect of regression testing is diverting test resource away from focusing on the current increment.

Another incremental project is geared to weekly scheduled releases to the live environment. There is usually a period of two to three days of system testing before a release. Here full time test resources switch between days of relative inactivity to being required to become supermen and wonderwomen when the application is ready to test.

In both cases, project managers have to decide between over resourcing their test teams or taking risks with the quality of a release. They find themselves having to fully retain highly skilled and expensive test specialists to cover small periods of test execution.

In an ideal world the “tap” of test resource would be turned on and off to match the intermittent demand. This is not always possible. Often domain expertise prevents simply moving testers around when and where they are required.

Using the Hrmes framework offers the benefit of having automation specialists involved only when you need them.

This role-based design presents an opportunity for successful outsourcing. The core project team, including all business domain and project management expertise is fully retained on-site.

Even within the Agile community, the keenest advocators of co-location are recognising its practical limitations. The acceptance of Agile Distributed Teams (ADTs) is becoming common. At the same time there is a significant retreat from poorly conceived outsourcing.

There are certainly limitations with following the extremes of any practice. What is vital is whether your test framework supports distribution and whether the people and process offsite can deliver the benefits from distribution.

The Hrmes model of test automation has been designed to facilitate on-demand access and the distributed involvement of test resources. W3QAs main expertise is its ability to deliver remote automated test services for its clients – on-demand.

2.3 Scalability

System testing without automation will not scale. Test automation without a data-driven framework will not scale effectively. A scalable design needs a scalable infrastructure.

Utility Computing and Elastic Resources

W3QA uses Amazons Web Services (AWS) - in particular AWS' Elastic Computer Cloud - EC2.

This is a cutting edge Utility Computing solution. It lends itself well to flexible hosting of complex test configurations. It is affordable and massively scalable.

The Hrmes framework is fully compatible and integrated with EC2 via Selenium Grid 1.0. W3QA can build and manage your entire automated test process remotely. Your engagement will be based on a utility model where you pay for only what you use.

This "on-demand" flexibility offers great advantages in resourcing as well as cost. It facilitates increased project planning capabilities with virtual test teams and infrastructures.

The elastic scalability offered by W3QA means that any outsourcing can be staged, controlled and used on-demand.

2.4 Conclusion

Notwithstanding the license terms Hrmes is released under, I will conclude with something of a sales pitch.

The Hrmes framework delivers a new and highly effective approach to test automation. It facilitates a successful automation project, in being:

- Affordable
- Accessible
- Productive
- Scalable

Only an on-demand solution can fully achieve the potential of test automation. Hrmes will maximise the return on investment test automation brings to your business. Hrmes is a framework that enables effective on-demand test automation.

Hrmes is fully integrated with one of the most advanced test automation platforms in the world. It is execution ready and comes with all the benefits of advanced design.

Hrmes enables a unified process for test specification, execution and reporting. It has a clear domain specific language (DSL) for test specification. Tests get executed as defined by this DSL and are not written in code.

Hrmes is the first and only framework in the world to achieve data driven automation on a fully enabled Selenium RC platform.

Hrmes is fully integrated with Selenium GRID. Out of the box you can run distributed tests on the platform configurations of your choice.

Hrmes comes with a very powerful and usable set of tools. Its framework is accessed through an industrial strength IDE - Eclipse. It facilitates a standard test process and a singular environment for all test assets across all projects.

Hrmes provides increased clarity, usability and accessibility to a Selenium RC automation project, without diluting any of its power.

You could spend years designing, implementing and fine tuning your own automation framework. Or you could build on the experience of automation specialists and implement one that is execution ready and already proven fit for purpose.

Hrmes is available free and under an open source license.