AUTOMATION TEST TOOLS

Date Created: 1st March 2001
Last Updated: 11th Sept 2001

Anyone who would like to contribute to this document, please contact me on the email below. A reference to your name and contact info will be included if your contribution is used.

Also I hope to include some other tools like TestPartner from Compuware and other tools from companies like Empirix (formerly RSW), etc.

Contact info ray.robinson@cableinet.co.uk

Purpose

This document has been created as a guide to help testing professionals select, understand and implement functional test tools in their organisation.
Due to the large number of test automation tools available I have followed the 80/20 rule and selected the most popular as a base for this document. However where necessary other products including, Test Management, performance test and defect tracking suites will be mentioned.
This base includes the following:

Rational Robot (as included in Rational TestStudio) 2001
Rational Visual Test 6.5
Mercury WinRunner 7
Segue SilkTest 5.?
Compuware QA Run 4.7.?(as included in QACenter)

Introduction

Just as a brief background for my self I have over 15 years QA, Test and development experience. I have personally used all the above tools in live customer environments at least twice except Visual Test, which I have only used on one engagement. The engagements have ranged from 3 to 12 months.
I have worked freelance for many years and have held posts which have included Test Consultant, Test Manager, Project Manager, Business Systems Manager, Group QA Manager and similar throughout my working life. As well as working permanently for one of the vendors (no it's not Rational)
I am a trained engineer (mechanical) with qualifications up to postgraduate in Computing. I have a good knowledge of VB, C, Java, SQL and Pascal as well as networking, Unix and NT.
I have worked in the following industries – Pensions, banking, Utilities, Manufacturing, Chemical processing, Telecom, E-commerce, ASP and Insurance. This has included working on PC's, Unix and Mainframe. Using the above tools as well as Performance and unit Test tools.

Why have I mentioned all this? This is to help provide testing professionals with the confidence that I have sufficient knowledge (as well as using other peoples knowledge ☺) to make this document useful and reliable.

VENDORS

Compuware

Of all the vendors mentioned in this document Compuware is by far the largest. It has a turnover in excess of $2 billion and staff of more than 15,000. 9,500 of these are professional services staff with skills covering all the development lifecycle. As such Compuware does not only supply the tools but will provide staff to initially develop your test suite and handover to internal staff as required.
Their test tool set is (in my opinion) second to Rational only on the windows platform (for coverage) but for complete coverage across platforms including mainframe and Unix they are the best. So for the larger company that requires a complete testing solution to cover these platforms it is probably best to start with Compuware as they will offer unit test, database test, mainframe, functional, load, web test, defect tracking and more in their tool set. No other vendor can offer this range.
For more information on Compuware visit their site on www.compuware.com

Rational

Rational's turnover is in excess of $I/2 billion with staff in excess of 3,300. For me they offer the most complete lifecycle toolset (including testing) of these vendors for the windows platform. When it comes to Object Oriented development they are the acknowledged leaders with most of the leading OO experts working for them. Some of their products are worldwide leaders e.g. Rational Rose, Clearcase, RequistePro, etc.
Their Unified Process is a very good development model that I have been involved with which allows mapping of requirements to use cases, test cases and a whole set of tools to support the process.
If you are developing products using an OO approach then you should include Rational in the evaluation.
For more details about Rational please see www.rational.com

Mercury Interactive

Mercury's turnover is in excess of $190 million with staff in excess of 1250. In the UK Mercury have at least 50% of the market in test tools. So as a tester you are more likely to find work as a skilled WinRunner developer and also as an employer more likely to find people with WinRunner skills.
They have a number of complimentary tools TestDirector being the most integrated one. They have a lot of third party support and test tools are usually compared first against Mercury than the others. Mercury tends to use third party companies to supply professional services support for their tools (e.g. if you require onsite development of test suites).
For more details see www.merc-int.com

Segue

Segue has a turnover of in excess of $50 million and a staff of c400.
Anyone who has used SilkTest along side any of the other test tools will agree that this is the most function rich out the box. However the learning curve (if you have no programming experience) is the steepest. In my opinion it provides the most robust facilities; an object map, test recovery facilities and object-based development language. Segue's performance test tool SilkPerformer also performs very well compared to it's rivals e.g. LoadRunner, LoadTest, etc.
SilkTest does not have a big user base in the UK compared to the other products which is a shame because for straight automation testing it has excellent facilities for creating robust test suites/frameworks
To find out more details see www.segue.com

Some of the Others

The are a host of other tools vendors but a special note will made of those I have used briefly (mainly as evaluations) that have shown very good potential both functional and cost wise.

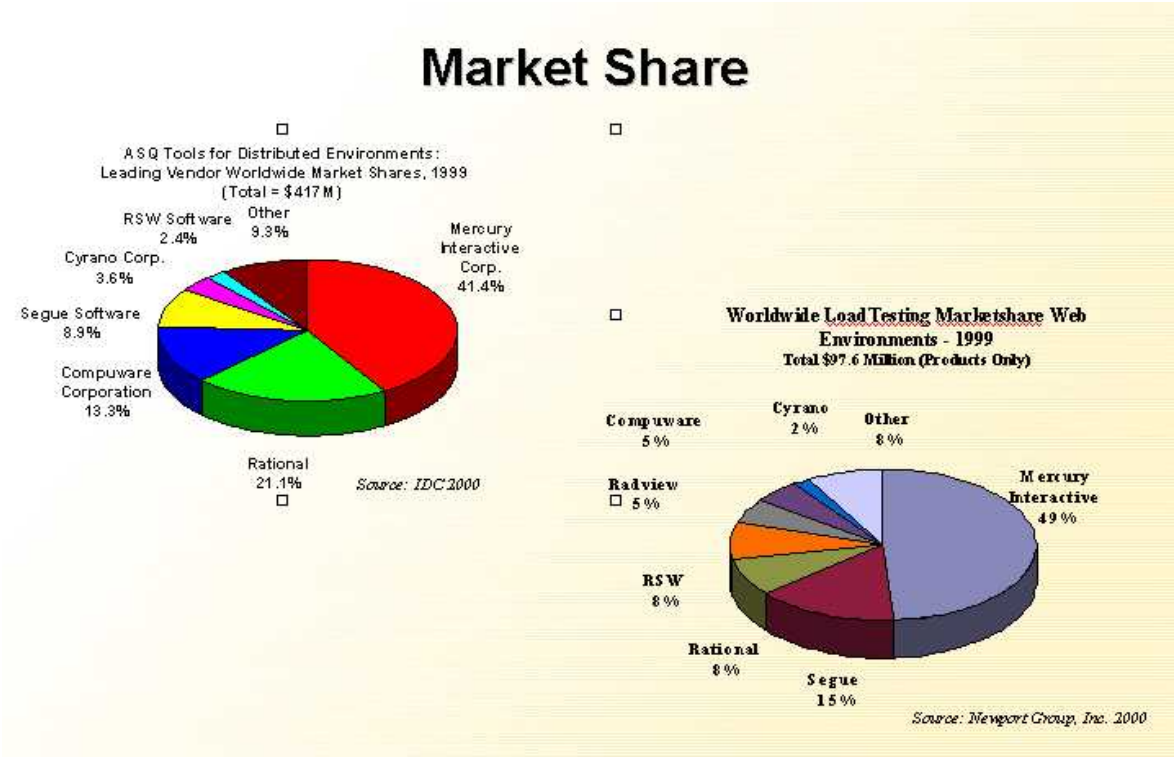
Forecast by Facilita is mainly used for performance testing but functionally it is as strong as the other performance tools and the cost saving is usually at least 50% less. You can find out more about them on www.facilita.com.

WAST from Microsoft is a free yes free, performance test tool that is very good, considering the cost ☺. It works best on ASP, Microsoft centric applications and quite often can do the job without having to buy additional performance test tools. Have a look on http://homer.rte.microsoft.com/

For ease of functional and performance testing the e-test suite from Empirix looks very good however I have only used it for less than 30 days as an evaluation and so did not do anything complex. But for more details have a look on www.empirix.com

Last but definitely not least is the product developed via Cyrano called OpenSTA. This is a very good performance test tool and once again this one is free. Yes free. It is the maturest free tool I've seen and in a few years it could rival some of the leading commercial products. It is governed by the GNU license and so you can modify the source code as you see fit. For more info see www.opensta.org

The market and automation testing



Eventually I will provide more details regarding the market as a whole but needless to say the test automation market is increasing.
Generally there is a trend towards test tools looking more like development tools. For instance Visual Test uses the same IDE as Visual C++ and TestPartner licenses VBA from Microsoft. The automation effort is very much a development exercise. Visual Test is similar to VB but also has pointers, SilkTest is Object Oriented, most of the tools allow you to run complex SQL queries, some can link to source control applications, debugging capabilities, etc, etc.
In my opinion it would be better to license existing languages like Java, C++ or VBA for the

following reasons:

- It would be cheaper to maintain as you do not require as many people developing your own niche language
- There will be a lot more books available to explain the intricacies of the language
- There will be a lot more skilled resources who have knowledge of the language
- It would be a more desirable profession as you will obtained more transferable and sort after skills
- The language will be richer
- The Test Tool Vendor can concentrate on supplying test functionality rather than providing functions like opening a file, string manipulation, etc which are already provided in existing languages

I think if this is adopted throughout the industry we will see the prices come down, more people using automation tests and the software world benefiting as a whole from more reliable software.

Most of the tools evaluated provide the same functionality but just in different ways. I would be very surprised if each Vendor did not buy the others test tools (through a holding company of course) and see what features they have. This would be used to form some of the requirements for the next release of their product. Most vendors will provide releases at least twice a year with bug fix releases in between. The market is not in the mature phase yet and as such I expect to see some more consolidation. Who will survive? Hmm lets see I expect to see Compuware and Rational but whether Mercury or Segue are brought, etc remains to be seen.
One thing I can see is these vendors will need to offer more services along side their products, as Compuware does this may be in the testing arena, performance management and related industries.

The Test Tools

Functional Test Tool Matrix

The Tool Matrix is provided for quick and easy reference to the capabilities of the test tools. Each category in the matrix is given a rating of 1 – 5. 1 = Excellent support for this functionality, 2 = Good support but lacking or another tool provides more effective support, 3 = Basic/ support only. 4 = This is only supported by use of an API call or third party add-in but not included in the general test tool/below average, 5 = No support. In general a set of criteria can be built up by using this matrix and an indicative score obtained to help in the evaluation process. Usually the lower the score the better but this is subjective and is based on the experience of the author and the test professionals opinions used to create this document.

A detailed description is given below of each of the categories used in the matrix.

1. **Record and Playback**

   This category details how easy it is to record & playback a test. Does the tool support low-level recording (mouse drags, exact screen location)? Is there object recognition when recording and playing back or does it appear to record ok but then on playback (without environment change or unique id's, etc changes) fail? How easy is it to read the recorded script.
   When automating, this is the first thing that most test professionals will do. They will record a simple script; look at the code and then playback. This is very similar to recording a macro in say Microsoft Access. Eventually record and playback becomes less and less part of the automation process as it is usually more robust to use the built-in functions to directly test objects, databases, etc. However this should be done as a minimum in the evaluation process because if the tool of choice cannot recognise the applications objects then the automation process will be a very tedious experience.

2. Web Testing

Web based functionality on most applications is now a part of everyday life. As such the test tool should provide good web based test functionality in addition to its client/server functions.
In judging the rating for this category I looked at the tools native support for HTML tables, frames, DOM, various platforms for browsers, Web site maps and links.
Web testing can be riddled with problems if various considerations are not taken into account. Here are a few examples

- Are there functions to tell me when the page has finished loading?
- Can I tell the test tool to wait until an image appears?
- Can I test whether links are valid or not?
- Can I test web based objects functions like is it enabled, does it contain data, etc.
- Are there facilities that will allow me to programmatically look for objects of a certain type on a web page or locate a specific object?
- Can I extract data from the web page itself? E.g. the title? A hidden form element?

With Client server testing the target customer is usually well defined you know what network operating system you will be using, the applications and so on but on the web it is far different. A person may be connecting from the USA or Africa, they may be disabled, they may use various browsers, and the screen resolution on their computer will be different. They will speak different languages, will have fast connections and slow connections, connect using MAC, Linux or Windows, etc, etc. So the cost to set up a test environment is usually greater than for a client server test where the environment is fairly well defined.

3. Database Tests

Most applications will provide the facility to preserve data outside of itself. This is usually achieved by holding the data in a Database. As such, checking what is in the backend database usually verifies the proper validation of tests carried out on the front end of an application. Because of the many databases available e.g. Oracle, DB2, SQLServer, Sybase, Informix, Ingres, etc all of them support a universal query language known as SQL and a protocol for communicating with these databases called ODBC (JDBC can be used on java environments). I have looked at all the tools support for SQL, ODBC and how they hold returned data e.g. is this in an array, a cursor, a variable, etc. How does the tool manipulate this returned data? Can it call stored procedures and supply required input variables? What is the range of functions supplied for this testing?

4. Data Functions

As mentioned above applications usually provide a facility for storing data off line. So to test this, we will need to create data to input into the application. I have looked at all the tools facilities for creating and manipulating data. Does the tool allow you to specify the type of data you want? Can you automatically generate data? Can you interface with files, spreadsheets, etc to create, extract data? Can you randomise the access to that data? Is the data access truly random? This functionality is normally more important than database tests as the databases will usually have their own interface for running queries. However applications (except for manual input) do not usually provide facilities for bulk data input.
The added benefit (as I have found) is this functionality can be used for a production reason e.g. for the aforementioned bulk data input sometimes carried out in data migration or application upgrades.
These functions are also very important as you move from the record/playback phase, to data-driven to framework testing. Data-driven tests are tests that replace hard coded names, address, numbers; etc with variables supplied from an external source usually a CSV (Comma Separated variable) file, spreadsheet or database. Frameworks are usually

the ultimate goal in deploying automation test tools. Frameworks provide an interface to all the applications under test by exposing a suitable list of functions, databases, etc. This allows an inexperienced tester/user to run tests by just running/providing the test framework with know commands/variables. A test framework has parallels to Software frameworks where you develop an encapsulation layer of software (framework) around the applications, databases etc and expose functions, classes, methods etc that is used to call the underlying applications, return data, input data, etc.
However to do this requires a lot of time, skilled resources and money to facilitate the first two.

5. Object Mapping

If you are in a role that can help influence the design of a product, try to get the development/design team to use standard and not custom objects. Then hopefully you will not need this functionality.
However you may find that most (hopefully) of the application has been implemented using standard objects supported by your test tool vendor but there may be a few objects that are custom ones.
Most custom objects will behave like a similar standard control here are a few standard objects that are seen in everyday applications.

- Pushbuttons
- Checkboxes
- Radio buttons
- List views
- Edit boxes
- Combo boxes

If you have a custom object that behaves like one of these are you able to map (tell the test tool that the custom control behaves like the standard) control? Does it support all the standard controls methods? Can you add the custom control to it's own class of control?

6. Image Testing

Lets hope this is not a major part of your testing effort but occasionally you may have to use this to test bit map and similar images. Also when the application has painted controls like those in the calculator app found on a lot of windows applications you may need to use this.
At least one of the tools allows you to map painted controls to standard controls but to do this you have to rely on the screen co-ordinates of the image.
Does the tool provide OCR (optical character recognition)? Can it compare one image against another? How fast does the compare take? If the compare fails how long does that take? Does the tool allow you to mask certain areas of the screen when comparing.
I have looked at these facilities in the base tool set.

7. Test/Error recovery

This can be one of the most difficult areas to automate but if it is automated, it provides the foundation to produce a truly robust test suite. Suppose the application crashes while I am testing what can I do? If a function does not receive the correct information how can I handle this? If I get an error message how do I deal with that? If I access a web site and get a warning what do I do? I cannot get a database connection how do I skip those tests?
The test tool should provide facilities to handle the above questions. I looked at built in wizards of the test tools for standard test recovery (when you finish tests or when a script fails). Error recovery caused by the application and environment. How easy is it to build this into your code?

The rating given will depend on how much errors the tool can capture, the types of errors, how it recovers from errors, etc.

8. Object Name Map

As you test your application using the test tool of your choice you will notice that it records actions against the objects that it interacts with. These objects are either identified through the co-ordinates on the screen or preferably via some unique object reference referred to as a tag, object ID, index, name, etc. Firstly the tool should provide services to uniquely identify each object it interacts with and by various means. The last and least desirable should be by co-ordinates on the screen.
Once you are well into automation and build up 10's and 100's of scripts that reference these objects you will want to have a mechanism that provides an easy update if the application being tested changes.
All tools provide a search and replace facility but the best implementations are those that provide a central repository to store these object identities. The premise is it is better to change the reference in one place rather than having to go through each of the scripts to replace it there. I found this to be true but not as big a point as some have stated because those tools that don't support the central repository scheme; can be programmed to reference windows and object names in one place (say via a variable) and that variable can be used throughout the script (where that object appears).
Does the Object Name Map allow you to alias the name or change the name given by the tool to some more meaningful name?

9. Object Identity Tool

Once you become more proficient with automation testing one of the primary means of identifying objects will be via an ID Tool. A sort of spy that looks at the internals of the object giving you details like the object name, ID and similar.
This will allow you to reference that object within a function call.
The tool should give you details of some of the object's properties, especially those associated with uniquely identifying the object or window. The tool will usually provide the tester with a point and ID service where you can use the mouse to point at the object and in some window you will see all of that objects ID's and properties.
A lot of the tools will allow you to search all the open applications in one swoop and show you the result in a tree that you can look at when required.

10. Extensible Language

Here is a question that you will here time and time again in automation forums. "How do I get {insert test tool name here} to do such and such", there will be one of four answers.

- I don't know
- It can't do it
- It can do it using the function x, y or Z
- It can't in the standard language but you can do it like this

What we are concerned with in this section is the last answer e.g. if the standard test language does not support it can I create a DLL or extend the language in some way to do it? This is usually an advanced topic and is not encountered until the trained tester has been using the tool for at least 6 – 12 months. However when this is encountered the tool should support language extension. If via DLL's then the tester must have knowledge of a traditional development language e.g. C, C++ or VB. For instance if I wanted to extend a tool that could use DLL's created by VB I would need to have Visual Basic then open say an ActiveX dll project, create a class containing various methods (similar to functions) then I would make a dll file. Register it on the machine then reference that dll from the test tool calling the methods according to their specification. This will sound a lot clearer as you go on in the tools and this document will be updated to include advanced topics like this in extending the tools capabilities.

Some tools provide extension by allowing you to create user defined functions, methods, classes, etc but these are normally a mixture of the already supported data types, functions, etc rather than extending the tool beyond it's released functionality.
Because this is an advanced topic I have not taken into account ease of use, as those people who have got to this level should have already exhausted the current capabilities of the tools. So want to use external functions like win32api functions and so on and should have a good grasp of programming.

11. Environment Support

How many environments does the tool support out the box? Does it support the latest Java release, what Oracle, Powerbuilder, WAP, etc. Most tools can interface to unsupported environments if the developers in that environment provide classes, dll's etc that expose some of the applications details but whether a developer will or has time to do this is another question.
Ultimately this is the most important part of automation. Environment support. If the tool does not support your environment/application then you are in trouble and in most cases you will need to revert to manually testing the application (more shelf ware).

12. Integration

How well does the tool integrate with other tools. This is becoming more and more important. Does the tool allow you to run it from various test management suites? Can you raise a bug directly from the tool and feed the information gathered from your test logs into it? Does it integrate with products like word, excel or requirements management tools?
When managing large test projects with an automation team greater than five and testers totalling more than ten. The management aspect and the tools integration moves further up the importance ladder. An example could be a major Bank wants to redesign its workflow management system to allow faster processing of customer queries. The anticipated requirements for the new workflow software numbers in the thousands. To test these requirements 40,000 test cases have been identified 20,000 of these can be automated. How do I manage this? This is where a test management tool comes in real handy.
Also how do I manage the bugs raised as a result of automation testing, etc? Integration becomes very important rather than having separate systems that don't share data that may require duplication of information.
The companies that will score larger on these are those that provide tools outside the testing arena as they can build in integration to their other products and so when it comes down to the wire on some projects, we have gone with the tool that integrated with the products we already had.

13. Cost

In my opinion cost is the least significant in this matrix, why? Because all the tools are similar in price except Visual Test that is at least 5 times cheaper than the rest but as you will see from the matrix there is a reason. Although very functional it does not provide the range of facilities that the other tools do.
Price typically ranges from $2,900 - $5,000 (depending on quantity brought, packages, etc) in the US and around £2,900 - £5,000 in the UK for the base tools included in this document.
So you know the tools will all cost a similar price it is usually a case of which one will do the job for me rather than which is the cheapest.
Visual Test I believe will prove to be a bigger hit as it expands its functional range it was not that long ago where it did not support web based testing.
The prices are kept this high because they can. All the tools are roughly the same price and the volumes of sales is low relative to say a fully blown programming language IDE like JBuilder or Visual C++ which are a lot more function rich and flexible than any of the test tools.

On top of the above prices you usually pay an additional maintenance fee of between 10 and 20%. There are not many applications I know that cost this much per license not even some very advanced operating systems. However it is all a matter of supply. The bigger the supply the less the price as you can spread the development costs more. However I do not anticipate a move on the prices upwards as this seems to be the price the market will tolerate.
Visual Test also provides a free runtime license.

14. Ease Of Use

This section is very subjective but I have used testers (my guinea pigs) of various levels and got them from scratch to use each of the tools. In more cases than not they have agreed on which was the easiest to use (initially). Obviously this can change as the tester becomes more experienced and the issues of say extensibility, script maintenance, integration, data-driven tests, etc are required. However this score is based on the productivity that can be gained in say the first three months when those issues are not such a big concern.
Ease of use includes out the box functions, debugging facilities, layout on screen, help files and user manuals.

15. Support

In the UK this can be a problem as most of the test tool vendors are based in the USA with satellite branches in the UK.
Just from my own experience and the testers I know in the UK. We have found Mercury to be the best for support, then Compuware, Rational and last Segue.
However having said that you can find a lot of resources for Segue on the Internet including a forum at www.betasoft.com that can provide most of the answers rather than ringing the support line.
On their website Segue and Mercury provide many useful user and vendor contributed material.
I have also included various other criteria like the availability of skilled resources, online resources, validity of responses from the helpdesk, speed of responses and similar

16. Object Tests

Now presuming the tool of choice does work with the application you wish to test what services does it provide for testing object properties?
Can it validate several properties at once? Can it validate several objects at once?
Can you set object properties to capture the application state?
This should form the bulk of your verification as far as the automation process is concerned so I have looked at the tools facilities on client/server as well as web based applications.

Matrix

What will follow after the matrix is a tool-by-tool comparison under the appropriate heading (as listed above) so that the user can get a feel for the tools functionality side by side.

	Record & Playback	Web Testing	Database tests	Data functions	Object Mapping	Image testing	Test/Error recovery	Object Name Map	Object Identity Tool	Extensible Language	Environment support	Integration	Cost	Ease of use	Support	Object Tests
WinRunner	2	1	1	2	1	1	2	1	2	2	1	1	3	2	1	1
QA Run	1	2	1	2	1	1	2	2	1	2	2	1	2	2	2	1
Silk Test	1	2	1	2	1	1	1	1	2	1	2	3	3	3	2	1
Visual Test	3	3	4	3	2	2	2	4	1	2	3	2	1	3	2	2
Robot	1	2	1	1	1	1	2	4	1	1	2	1	2	1	2	1

Matrix score

- WinRunner = 24
- QARun = 25
- SilkTest = 24
- Visual Test = 39
- Robot = 24

Test Tool Comparisons

- **Record and Playback**

  Rational Robot

  Rational Robot provides the facility to record tests from the IDE using a red GUI button; you can also record VU (virtual user) test scripts to play back in LoadTest (or Test Manager).
  While recording you can minimise the IDE and then just click on the screen objects (carry out a test) and when finished you click the stop and a script will be recorded. A small sample of a recorded script is given below:

```
Sub Main
    Dim Result(50) As Integer
    Dim i as Integer
    Dim NewResult as String

    StartBrowser
"http://pandora.ple.blahblah.co.uk/action.pega?f1000=SIGNON&profile=test3",
"WindowTag=WEBBrowser"

    Window SetContext, "WindowTag=WEBBrowser", ""
    Window WMaximize, "", ""

    delayfor 3000
    Browser SetFrame,"Type=HTMLFrame;HTMLId=__pegMainFrame",""
    Browser NewPage,"HTMLTitle=PANDORA - TEST1",""
```

```
   Result(1) = EditBoxVP (CompareProperties, "Type=EditBox;Name=f5p1", "VP=Object
Properties;Wait=2,30")
   Result(2) = EditBoxVP (CompareProperties, "Type=EditBox;Name=f6p1", "VP=Object
Properties2;Wait=2,30")
   Result(3) = EditBoxVP (CompareProperties, "Type=EditBox;Name=f8p1", "VP=Object
Properties3;Wait=2,30")
   Result(4) = EditBoxVP (CompareProperties, "Type=EditBox;Name=f9p1", "VP=Object
Properties4;Wait=2,30")
   Result(5) = PushButtonVP (CompareProperties,
"Type=PushButton;Name=@ACTION=|ENTR", "VP=Object Properties5;Wait=2,30")
   Result(6) = PushButtonVP (CompareProperties,
"Type=PushButton;Name=@ACTION=|APPR", "VP=Object Properties6;Wait=2,30")
   Result(7) = PushButtonVP (CompareProperties,
"Type=PushButton;Name=@ACTION=|BACK", "VP=Object Properties7;Wait=2,30")

   For i = 1 to 3

      Select Case i

         Case 1
            InputKeys "robir"
            PushButton Click, "Type=PushButton;Name=@ACTION=|ENTR"

         Case 2

Etc, etc

End Sub
```

As noted this is a very basic script. Robot uses a language that is very similar to Visual Basic called SQABasic it supports the following datatypes:

String, Integer, Long, variant, object, Single, Double, Currency, User-Defined.
String can be fixed or variable. Variant can be any of the other types, Object allows you to create an instance of a class stored in a dll (an ActiveX dll as an example). When an instance of the class is created you may access the methods via dot notation e.g. variablea.methoda (). This is very good for communicating with windows applications, etc and basically extending the language.
User-defined allows you to build a mixture of datatypes similar to a ‘struct’ in C or a ‘record’ in Pascal. All the other language syntax is based around VB.
Robot uses the Window SetContext syntax to know where it is in the application. All commands/functions are sent to the window or object within the current context. Most of the commands in Robot follow this template ->

Function, “recognition method”, “arg1”, “arg2”, “and so on”
The recognition method can be the object/window name, id or some other unique identifier. Functions will return a value that will be a discrete value or an error number.

Scripts can be contained within a procedure e.g. sub…end sub or function…. End function.

- You can insert new recorded tests anywhere in the script
- You can launch an application using its path via a wizard
- You can launch a browser using a wizard
- You can record low level (mouse drags, exact wait times)
- The screen is split into 4 a script design window, variables (debug) window, verification points window and an output console
- Keywords, etc can be colour co-ordinated

- You can run recorded scripts from the command line

In general Robot is a very easy product to understand out the box

Visual Test

Visual Test provides the facility to record and playback scripts, however I do not find it as slick as the others. Clicking the 'tape cassette' icon on the toolbar starts the recording session.
Each recording session is given a scenario name if objects are not recognised when you click them then a dialog will pop up asking you to associate the unknown window/object with a known type. Upon completion of your test you click the icon on the task bar and you get the option to:

- Stop and create scenario
- Continue recording events
- Quit without creating scenario

In addition to these you can

- Add a verification
- Add a comment
- Add a multi-user
- Select help

Although it is easy to follow it is not as easy to work with as the other tools even the look and feel (for me) looks inferior but it does do the job.

Here is a basic sample of a script

```
'$INCLUDE 'RECORDER.INC'

SetDefaultWaitTimeout(Timeout)

Scenario "testscenario"

'Minimize the Visual Test window.

If GetHandle(GH_HWNDCLIENT) Then WMinWnd(GetHandle(GH_HWNDCLIENT))

' Check the resolution that this script was recorded on.
CheckResolution (1280, 1024)

CurrentWindow = WFndWndC("AUTOMATION TOOLS.doc - Microsoft Word", "OpusApp", FIND_AND_MAX,
Timeout)
Play "{Click 407, 12, Left}"
Play "{Click 20, 29, Left}"
End Scenario
```

With Visual Test it is all about handles to windows and context (think of it as being similar to Robot).
Again once the handle/context is derived all commands are sent to the current window.
Visual Test uses a language called the test language that is similar to visual basic but it also has pointers. This makes it a lot easier to call WINAPI functions that require pointers but it can also cause a lot of problems.

A summary of the language and the record/playback facilities follows:

- You can turn user wait times on or off while recording
- Datatypes supported include Integer/Short, Long, Single, Double, String, Pointer and variant
- Typically each newly recorded script builds a new scenario similar to a procedure

- You can tell Visual Test to try to uniquely identify objects by selecting the ordinals only box when recording a script
- Scripts are complied to P-Code and so can be run on other machines
- The template for most functions is: Function (context, id, “arg1”, “arg2”, etc)

SilkTest

SilkTest has two main ways to start recording one is through its quickstart window and the other from the toolbar File -> New. One thing you will notice about SilkTest compared to it’s rivals is how it looks more complicated in most aspects, even in recording a script or in SilkTest’s case a Test Frame.
First you will record the applications main window and menus you then start to select all the objects in the application you may want to test later. This is called a windows declaration, it is similar to WinRunner’s rapidtest. Basically you click each item you will use in your test. In SilkTest it uses an identifier and a tag to identify each object individually. You can use multi-tags (I prefer this) to make the identification more unique. One important difference in the identification attributes available to Silk than it’s rivals is the Prior text tag type. All the others, caption, Index, ID and location are available in the other tools.
SilkTest code looks like this:

```
testcase FindTest ()

        TextEditor.File.New.Pick ()
        DocumentWindow.Document.TypeKeys ("Test Case<HOME>")
        TextEditor.Search.Find.Pick ()
        Find.FindWhat.SetText ("Case")
        Find.CaseSensitive.Check ()
        Find.Direction.Select ("Down")
        Find.FindNext.Click ()
        Find.Cancel.Click ()
        DocumentWindow.Document.VerifySelText (<text>)
        Case
        TextEditor.File.Close.Pick ()
        MessageBox.No.Click ()
```

        Blah, blah

As you can see from the code it will look a lot different to the other tools code. For instance QARun and Robot’s code is very similar. WinRunner and Visual Test is similar to these but SilkTest is different because it uses an object-oriented approach. I will explain a little. Each object on the screen is based on a class. A class is the blueprint of how you can use an object. The class will have various methods and properties that you can use to perform certain actions like set the text in an editbox. This is usually in the form object.method/property so editbox.settext() (ignore my syntax).
Anyway more on this later.

A summary of the language is as follows:

Datatypes: In excess of 20 datatypes including Boolean, long, number, string, semaphore, set, time, window, etc
Classes and methods including inheritance (single).
Collapsing and opening various parts of your script.
Indentation, replacing lines of code with above or below lines.
Usual record, pause, stop, inserts recorded code, etc.
Central repository for windows identification (change id (tag) in one place).

WinRunner

WinRunner has a record facility allowing you to record in either Context Sensitive mode or in Analogue mode. This is similar to all the other tools. When you record you can stop by a short cut key or you can click the stop yellow square on the toolbar. Winrunner has rich command line functions that allow you to run scripts on start up.

A typical Winrunner script may look like this (partially complete).

```
web_browser_invoke (IE,
"http://pandora.ple.blahblah.co.uk/action.pega?f1000=SIGNON&profile=test3"
);

       win_max ("Browser Main Window_1");

for (i=1; i<4; i++){

       set_window("__pegMainFrame",5);

       if (i<2)
              result[0]=win_check_gui("__pegMainFrame", "list4.ckl", "gui4",
1);#validate all gui objects on login screen

# __pegMainFrame - Check login with no password (loop 1), no login id (loop
2) and then with valid id and password
       switch (i) {

              case 1:
                     edit_set("f5p1","robir");
                     button_press("Enter");
                     break;
              case 2:
                     password_edit_set("f6p1","cd2da24b7bbddcb9");
                     button_press("Enter");
                     break;
              case 3:
                     edit_set("f5p1","robir");
                     password_edit_set("f6p1","cd2da24b7bbddcb9");
                     button_press("Enter");
                     win_check_bitmap("SESSION", "Img2", 32, 65, 118, 805,
427);
                     break;
              }
              if (i<3){
                     set_window("Microsoft Internet Explorer", 1);
                     result[0+i]=obj_check_gui("- Invalid signon(static)",
"list1.ckl", "gui1", 1);
                     set_window ("Microsoft Internet Explorer", 4);
                     button_press ("OK");
                     }
       }
```

As can be seen from the above I've added a bit more control to the script.
WinRunner recognises two variable types (number or a string) there is no need to declare variables up front. There are a few exceptions and these include some external functional calls, arrays in user-defined functions and similar.
The scripting language is known as TSL for short which means Test Script Language. It provides the usual array of functions and syntax but it also has specific functions for WAP, which are used through an emulator. One improvement would be a syntax checker or precompiled code rather than having to run scripts to find errors.
Variables to function can be predefined as in, out or inout (default). This obviously allows you to control the flow of data to and from functions.

Some more record/playback facilities include:

- Insert additional recording anywhere
- Colour co-ordinated script language
- The screen only has one window so it allows you to see more on the screen than some of the other tools
- Run from arrow
- Playback in Verify (standard), debug (no logging) or update (updating checkpoints) modes
- Auto indent tool bar option and comment/uncomment in blocks

- There are wizards for every function including the ones you choose to add. This makes it very easy for you to point at an object select the appropriate function and it leads you through filling in the spaces.
- Virtual Object wizard allows you to map painted icons, etc to real object types

QARUN

QA Run has a learn facility for recording and playback. This can be started by clicking the learn button on the tool bar (i.e. a red dot) The rest is similar to all tools you just click through the application under test (AUT) and to stop you just click the QA Run window. There are various ways to playback scripts e.g. command line and via the toolbar however one interesting facility that QARun has is the RunAWL. RunAWL is Compuware's script execution module that allows you to select scripts to run without having to open the script editor. You can achieve this in other tools by having one main script act as a driver for calling others and you run this from the command line. This makes it easier for those who do not use the command line.

A typical QARun script will look like this:

```
Attach "~P-NOTEPAD.EXE~Edit-Untitled-Notepad"
        Type "Testing Automation Tools"
Attach "~N-NOTEPAD.EXE~32770-Notepad"
        MenuSelect "File~Open…"
Attach "~N~NOTEPAD.EXE~32770-Notepad"
        Button "&No", 'Left SingleClick'
;attach to Notepad's open file dialog
Attach "~N~NOTEPAD.EXE~32770-Open"
        Button "Cancel", 'Left SingleClick'
```

Web Testing

Rational Robot

To enable the web based functionality in Robot you usually have to launch a browser using the StartBrowser function and by enabling the appropriate HTML box in the extension manager found under Tools -> Extension Manager. Some of the commands supported include:

- HTMLDocument for positioning the cursor in web pages
- HTMLImage for clicking an image in a web page
- HTMLLink for performing a click on a web link
- HTMLActiveX, HTMLHidden, HTMLTable and HTML follows the above pattern
- SQASetDefaultBrowser is self explanatory
- WebsiteVP allows you to validate the site using SiteCheck say to compare against a previous baseline or to check for broken links
- HTMLVP are verification points for the above.
- There are a host of Java commands too.

Once the HTML page is recognised most things follow the pattern you would use for testing client/server. Some things to note are wait times, as browsers tend to be a lot slower than client/server apps. There is cross browser support by selecting the HTML-Navigator box in the extension Manager however the default browser used for recording is IE. I have not used the Netscape playback extensively in this product but it does work without modification on standard objects like Text boxes, Combo's and similar I have not used it where browser specific rendering was an issue.
As you hover over an object the object name is displayed e.g. HTMLTable.

To select other unique id's to identify the object you would use the Inspector that displays a tree of all the objects on the screen and their properties.

Visual Test

To start a browser in Visual Test you should use the Webexplore command. This is a simple matter of writing the url in the function e.g. Webexplore("mplayer.com") and hey presto you have launched a web browser.
Visual Test 6 and 6.5 saw the release of the web testing functionality. Visual Test only supports testing of Internet Explorer pages (out the box). As this product was formerly a Microsoft product this was formerly used for testing Microsoft based applications.
There is a whole host of commands here are a few (all the rest follow a similar format).

- **WebImageClick(***context$***,** *image$***,** [*button&*], [*x&*]**,** [*y&*]**,** [*timeout&*]**)**
- **WebFrameExists(***context$, frame$, [timeout$]***)**
- **WebTableCellCount(***context$***,** *table$***,** [*timeout&*]**)**
- **WebTextAreaValue(***context$***,** *form$***,** *element$***,** [*timeout&*]**)**
- **WebActiveXHasProperty(***context$***,** *control$***,** *property$* [, *timeout&*]**)**
- **WebDisplayInfo(***context$***,** *index&***,** [*flag&* [**OR** *flag&* [**OR** *flag&*]]]**)**
- **ON WebLoadReady** [**(** [*context$* | **NULL**] **)**] **CALL** *handler$*

As you can see the functions follow a similar pattern and once you get to know one or two of them you can move ahead at speed. Context is the handle given to a particular instance of a browser usually returned from the webFndbrowser function the other arguments to the functions include the object ID(s) and an optional timeout value.

Visual Test is the most hands on of the entire test tools, as the product does not use the most appropriate recognition method. So usually you will find yourself using the window information tool to extract information about an object like it's name or for more detailed information you will browse the objects (similar to Robot's inspector) and get data that way.
The web testing capabilities are limited if you require testing on java-based web sites or Netscape. It would be better to go for one of the other tools. But if it's for an Intranet say based on a Microsoft-centric platform then you should have no problems.

SilkTest

SilkTest by the way was formerly known as QAPartner, Silk would usually enable all browser support for your test machine. However you can check and verify these by looking in the extension enabler. One thing to remember is, if you are not testing web-based applications then don't enable the web extensions.
Silk has a separate guide called testing web applications with SilkTest this is because Segue took e-commerce very seriously, so seriously that they bet their future on e-commerce and in 2000 the crash happened. Hmmm. So they now still do client/server and other environments.
Some of their web methods include the following:

- *Browser*.ClearCache()
- HTMLLInk class
- HTMLTable class
- HTMLPushbutton class
- HTMLTextField class
- WaitForReady Method this is a good function allowing you to wait for a predefined time or not until the application is ready
- Load Page method
- Invoke method

And a load more. As mentioned before it is important to remember that what you do is look for a class, say the HTMLPushbutton class and then you look at what methods are available for that class. They have an object browser that allows you to look at all the classes their methods and properties.

I have not used the latest 5.5 version yet, which has DOM support and is reported to make web testing even better. Hopefully I will have more news on this.

WinRunner

To start a browser in Winrunner you will use the web_browser_invoke function and also enable the WebTest add-in. One disadvantage that Winrunner (standard) has out the box is no support for Java. You have to buy a Java add-in. Considering Mercury tools are normally at the more expensive end of the scale without (in my opinion) adding any big advantages over the other tools, this should be included. But with the Java add-in I like Winrunner more than the other products for web based testing.

Some of the functions in its pack include:

web_password_encrypt: Obvious
web_sync: Allows the frame to appear before carrying on navigation
web_image_click:
Web_link_click
Web_url_valid
Tbl_get_cell_data: Get the data contained within an html table.
Web_refresh

Plus a lot more similar web functions.
Winrunner is normally at the cutting edge in both Java (if you have the add-in) and browser support. I have not found too much of a big problem with playback and object recognition of standard objects on Explorer upto 5.5 or Netscape.
It also has good checkpoints for capturing web text.
It also has a separate web exception handling function

QARun

QARun does support web testing but for me I found the facilities offered in WinRunner, Robot and SilkTest better. This is possibly the only section where QARun was below it's competitors but it did work on most web applications although the latest Java support was poor (Dec2000). Please note Compuware are marketing a separate product for web testing called TestPartner but I have not tested that (yet) so I can't give much comment.
QARun includes the following web functions: AnchorSelect(), BrowserToolbarCtrl(), ImageSelect(), LinkCheck(), MouseHover(), Replay.InterProfile, Replay.MouseHoverTime. QARun like Robot also provides a facility to carry out full site checks; this is particularly handy when you want to do a quick check to see what links (in the release) are broken. The checks that are available for standard objects can also be used for verifying the equivalent web based objects. Like all the tools various settings have to be tweaked to ensure that you get robust scripts. One of these is to ensure that 'learn by control ID' is set to off.

Database Tests

Rational Robot

Database tests are supported via ODBC using the following functions:
SQLOpen, SQLClose, SQLError, SQLRetrieve, SQLRetrieveToFile, SQLExecQuery, SQLGetSchema and SQLRequest.

You can carry out cursor type operations by incrementing arrays of returned datasets. All SQL queries are supplied as a string. You can execute stored procedures for instance on SQL Server you could use “Exec MyStoredProcedure” and as long as that stored procedure is registered on the SQL Server database then it should execute however you cannot interact as much as you may like by supplying say in/out variables, etc but for most instances it will cover your database test requirements

A sample database test could look like this:

```
Sub main
'  Declarations
'
   Dim connection As Long
   Dim destination(1 To 50, 1 To 125)  As Variant
   Dim retcode As long
   Dim query as String
   Dim outputStr as String
   connection = SQLOpen("DSN=SblTest",outputStr,prompt:=3)
'
'  Execute the query
   query = "select * from customer"
   retcode = SQLExecQuery(connection,query)

'  retrieve the first 50 rows with the first 6 columns of each row into
'  the array destination, omit row numbers and put column names in the
'  first row of the array

   retcode = SQLRetrieve(connection:=connection,destination:=destination,
      columnNames:=1,rowNumbers:=0,maxRows:=50, maxColumns:=6,fetchFirst:=0)

'  Get the next 50 rows of from the result set
   retcode =
      SQLRetrieve(connection:=connection,destination:=destination,columnNames:=1,row
      Numbers:=0,maxRows:=50, maxColumns:=6)
'  Close the connection
   retcode = SQLClose(connection)
End Sub
```

One problem I have found with Rational is if you get the sql syntax incorrect in some cases (SQLRetrievetofile) I have seen the production of a file that is several 100mb’s large in a few seconds. This was only this small because I killed the process --- You have been warned!

VISUAL TEST

Visual Test does not support database tests out the box. You would need to use a DLL to accomplish this.

SilkTest

As in most of these sections you will find that SilkTest has a lot more functions per section than any of the tools, this is true also in the database arena. This is not however saying that you are left at a great disadvantage as the sql queries used in the other products can be used to get some of the info. The database functions are: DB_Columnprivileges, DB_Columns, DB_Connect, DB_Disconnect, DB_ExecuteSql, DB_FetchNext, DB_FetchPrev, DB_FinishSql, DB_ForeignKeys, DB_GetConnectAttr, DB_PrimaryKeys, DB_ProcedureColumns, DB_Procedures, DB_SetConnectAttr, DB_DB_SpecialColumns, DB_Statistics, DB_TablePrivileges, DB_Tables

I have found the more useful functions (of the additional functions) is the FetchNext and FetchPrev as these allow you to cut out a lot of looping and so on that you do in the other scripts.

A sample database query could look like this:

```
STRING dat, name, ptable, stat, index, ftable, ignore,
fkcol, key-seq

hstmnt = DB_ForeignKeys (hdbc, dat, name, ptable, stat,
index, ftable)
while
       // retrieve foreign key column and key sequence; ignore the
       rest.
(DB_FetchNext (hstmt, ignore, ignore, ignore, ignore,
ignore, ignore,
ignore, fkcol, key-seq) == TRUE) {
       //print FK columns and key sequence …
}
```

WinRunner

WinRunner has good database test facilities you can either test database content via ODBC and the db_functions or via a database checkpoint. The db_functions are db_check (basically the same as the checkpoint), db_connect, db_disconnect, db_execute_query, db_get_field_value, db_get_headers, db_get_last_error, db_get_row, db_record_check, db_write_records and db_dj_convert.

There is also a neat little function that allows you to import from a database into a datatable. This allows you to use this data in your test. The hidden benefit of this is you can drive your tests from a database and use the imported data to supply variable data.

A database query may look something similar to this;

```
db_connect ("query1","DSN=Test1");
db_execute_query ("query1","SELECT * FROM Details",record_number);
record_number=record_number+1;

for (i=0; i<record_number; i++){

       Row_id="#"&i
       val = db_get_field_value ("query1",Row_id,"FName");
       }
```

db_disconnect ("query1");

QARun

The following database functions are available: dbAddNew(), dbBof(), dbClose(), dbConnect, dbDisconnect(), dbEdit(), dbEOF(), dbExecute(), dbGetField(), dbMove(), dbMoveFirst(), dbMoveLast(), dbMoveNext(), dbMovePrev(), dbRecordCount(), dbSelect(), dbSetField(), dbUpdate. The names are similar to SilkTest even though the coding is different.
You can accomplish most things either by the functions here or by creating the correct SQL statement. The dbEOF will come in useful for looping result sets.

A typical code snippet would look like this.

; connect to a data source using an ODBC driver dbConnect("DSN=Employee1")
; select employee details

dbSelect("SELECT * FROM Employees") ; move to first record in result set
dbMoveFirst( )

```
While dbEOF = 0 ; start of loop
        dbEdit( ) ; edit fields
        Age=dbGetField("Age") ; get current Age
        dbSetField( "Age", Str(Age-1) ) ; reduce Age by 1
        dbUpdate( ) ; update database
        dbMoveNext( ) ; next record
EndWhile
```

DATA FUNCTIONS

Rational Robot

Rational provides a resource for data creation called Datapools. Of the test tools this is the best facility for creating data.
The datapool table allows you to create data for feeding into an application but unlike lists, arrays and user defined records. You can create a template say of a customer with the following data:

- First name of length greater than 5 but less than 10
- Surname length greater than 3 characters
- Unique ID alphanumeric
- Age
- Date of birth (of certain range and date format)
- All the records to be unique, random, serial, etc

Then you select how many of these records you want, erm lets have 100,000 please, press the button and hey bingo they are created in a datapool of a name of your choice. It really is good and forms the backbone for creating data in Rational's big brother test tool LoadTest.
Here are some of the commands used in datapools:

- return& = SQADatapoolOpen (name$, [wrap], [sequence], [exclusive])
- return& = SQADatapoolFetch (datapool_id&) note the datapool_id is that returned from SQADatapoolOpen
- return& = SQADatapoolValue (datapool_id&, column, value$)
- return& = SQADatapoolFetch (datapool_id&) increments the datapool cursor
- return& = SQADatapoolRewind (datapool_id&)
- return& = SQADatapoolClose (datapool_id&)

A lot of the above functionality can be achieved in the other tools but with a lot more time creating lists, arrays, user-defined data types, then randomising the creation of data, making the data truly unique, then manipulating that created data.
All this is covered for you in datapools.

Visual Test

As is common with WinRunner, SilkTest and QA Run, Visual Tests provides functions for manipulating data stored in files. This data can be inputted into a spreadsheet or database and the data in these stored as CSV files. A slicker solution is to try and interface directly with the application but this requires a greater knowledge of the spreadsheet or database.

Here are some of the functions Visual Test uses; these are also used in Robot & QARun without hardly any modification:

- **OPEN** *filename$* [**FOR** *mode*] **AS** [#] *filenumber* [**LEN =** *reclength*]: **VTest**
- Open filename$ [For mode] [Access access] [lock] As [#] filenumber% [Len = reclen]: **Robot**
- **CLOSE** [[#] *filenumber1***,** [#]*filenumber2* ...]

- **EOF (** *filenumber* **)**
- FREEFILE
- **PUT** [#] *filenumber***,** [*recordnumber*] **,** *variable*

There are others but this provides the basis. These functions are very efficient and except for very large files you won't really notice any speed degradation.

These functions are heavily used when you move away from a record and playback paradigm to a data-driven approach

SilkTest

Remember this term when dealing with SilkTest "there are many ways to skin a cat". This is more the case with SilkTest than any of the tools. Here are some of the functions, facilities, etc you can use for making your tests data driven:

Associating Test Data (records) to Test cases. Records here are similar to records in Pascal and structs in 'C'. SilkTest use a script called testcase that is similar to a sub procedure in Robot and QARun. This Test case can have records associated with it for replacing hard coded data like field data, etc and also what can be associated with Test cases is a baseline state. This baseline state (that you record) allows you to drive the application to a particular application state (a particular screen) before you start to execute your tests. Believe me this is one of the selling points of Silk.

A list is a powerful and flexible data type. There is no exact equivalent in any of the other languages. Lists are like arrays in other languages but they also don't have to have initial values and like say vectors in 'Java' you can add to them. Here are some of the list functions; ListAppend, ListCount, ListDelete, ListFind, ListInsert, ListMerge, etc.

File manipulation functions are similar to the Open functions in QARun, Visual Test and Robot. Also it is similar to the ddt functions in WinRunner. You are able to read from say a CSV file and insert to an application or write to the file.

WinRunner

WinRunner uses the functions under the heading Data-Driven Test functions. Some of these functions are:

Ddt_close, ddt_get_row_count, ddt_next_row, ddt_open, ddt_val, etc. You can program these functions directly to work with an excel spreadsheet or tab separated variable file or you can use built in wizards (Winrunner has probably the best set of wizards out the tools). There is also a datatable (basically a spreadsheet) that you can populate available from the tool bar under tools. These functions allow you to do as much as you will need when it comes to selecting and inputting data. Rational shines in that It's datapool function is the best for 'creating' data. In Winrunner you can create lists of say days of the week, months, credit card numbers, etc but you have to create the list yourself at least once.

QARun

As with say Visual Test you can use the Open functions for working with files. Also QARun provides some additional data functions. These come under the category Testdata handling. It is similar to Mercury's DDT but I find QARun's functions slightly more flexible. The functions in this category include: TestData() – points to the file e.g. CSV, TestDataClose(), TestDataCurField(), TestDataCurRecord(), TestDataField(), TestDataFieldCount(), TestDataIndex(), TestDataRecordCount(), TestDataTransform(). Of these the least similar to other tools is the TestDataTransform() function. Yes it does sound like something from star trek but basically what it does is the parameters it

receives is the row and column of a CSV file and it returns what ever value is present. E.g. returnedData= TestDataTransform({R.F}). R is the row or some indicator for the row like index and F is the column or some indicator for the column. There are several you can use like the next row, an '*' to get it to randomise the return value, etc.

Object Mapping

Robot

Object mapping facilities can be accessed through the Tools -> general options menu. This allows you to include (map) a class against known classes. The classes to map against include Java, HTML, PeopleTools and standard windows controls. Usually the sign that an object is not known is when it is recorded as a generic object in Robot's script. To map an object you add the name of the class to the class of object you want to associate it with. Once mapped Robot will assume that the mapped object behaves like the object it is associated with.

Visual Test

Object mapping can be done via two ways. The first is from Tools -> Options -> Recorder Tab. What is neat is there is an ID tool that allows you to select the window/object you want to map. Then you select the object to map against e.g. pushbutton, etc and whether you want full mouse dragging to be supported.
Although the ID option is good there is a limited amount of objects you can map too.
The other way to map objects is when you are recording a scenario. If an unknown object is encountered Visual Test will bring up a dialog the same as found under the recorder tab and allow you to map the object to the predefined objects classes.
All added classes are updated in the registry of the operating system.

SilkTest

CustomWin is a word you do not want to see very often in your test script. This means that for some reason SilkTest cannot recognise the object you are clicking.
You can map the custom win to the so-called 'meta' classes i.e. BUTTON, STATIC and MDI. You do this in the windows declarations dialog it works in a similar way as the other tools in that you map an object to a class you think it behaves like. If the object is not a renamed standard object then the above will not apply and you face the prospect as in all the test tools of working with a graphical object or an unsupported object. This will not make your scripts robust. So ensure in the review that you don't get a lot of unsupported e.g. CustomWin objects that cannot be mapped to standard objects.

WinRunner

Object mapping is available usually via the rapidtest wizard and also the virtual object wizard. This allows you to map non-standard objects or painted controls to standard objects as recognised by Winrunner. The rapidtest wizard is very good and there is a similar facility available in Silktest. However rapidtest does not work when you have the webtest facility enabled. Remember setting this up can take a bit of time if your application is big and if you have a lot of non-standard objects. Also it should not normally be used if you are only going to regression test a small part of the overall application or if the application will significantly change (e.g. the recognition attributes). In this case I have found it better to let Winrunner learn the windows and objects as you go along.

QARun

QARun has good facilities for objects mapping all identified objects use a hybrid of properties to uniquely identify themselves. These properties are

- Type – whether it has a derived title or not
- Module
- Class
- Title
- Position

Also along with these physical attributes you have an associated logical name that you can change to match the description/functionality of the object. You can get QARun to automatically create name for unidentified testing objects (UTO) ☺ or you can name them when the UTO is encountered. To do this you use the Object Map configuration utility. This map allows you to do similar things to WinRunner but one thing that stands out about QARun is how it centralises everything. There is one repository you can have various versions of scripts, checks, etc. So no more woops "I've changed my code saved it and now want to roll it back but can't".

Image Testing

Robot

Robot supports image testing via its region and window image verification points. Their name explain what they do but in addition to this you can mask out parts of the screen that always changes each test so that you don't get unnecessary fails. Also a good facility is you can use the OCR facility to extract text from areas of the image when the other verification points won't do.
In reality you do not want to use this unless you have to as you can get failures for small pixel changes. You can set the threshold for OCR DPI recognition and pixels but that is more of an art than a science. If an image compare fails you will get a difference map so you can see where the failure has occurred compared to the baseline.
This is time consuming and can add significant time to tests where you have a lot of image compares.

Visual Test

Visual Test supports image testing through its screen utility and by using a window compare when recoding a scenario. The Screen utility can add masks to images and you can save the images to bitmaps, clipboard or a scn (for screen) file this is slightly more flexible than some of the other test tools. Visual test allows you to run multiple tests against screen images by doing compares of files or scn images within files. Again you do not want to use this utility unless you really have to.

SilkTest

SilkTest supports Image testing through the use of its window bitmap tool. There is no real advantage of this bitmap tool over the others. You can compare baseline against actual. You can create masks, show differences, zoom, etc.

WinRunner

WinRunner supports image testing through its Bitmap checkpoints. You can adjust the threshold of pixels for comparing images and another closely related facility (get text) could sometimes get text from these images.
As mentioned previously this functionality should only be used if the other checkpoints won't do.
Any differences between the expected and actual results can be shown after playback is completed and you may update the expected baseline with the actual image.

QARun

Image testing is handled via Bitmap checks. The usual facilities are available which are capturing baseline images, masking, comparing differences between baseline and actual. You can capture the whole screen or a region. A good facility is a closely related check known as the text check. I think this is the best text checking facility of the tools. This makes QARun suitable for environments where windows are mixed with character-based applications but more on that in the last section.

Test/Error Recovery

Robot

Robot does not have a built in recovery mechanism as SilkTest does however you can use event driven error handling by using the On Error function. This function allows you to capture certain predefined errors using an error handling procedure that processes the error as you wish. This may mean driving the script back to a previous state depending on where it is now but this is not a trivial task.
Robot can capture several errors including:
'Out of memory', 'Division by zero', 'Type mismatch', 'file not found', 'Object creation failed', 'Input buffer would be larger than 64k' and 'operating system error' to name but a few.
Traditionally coding for error trapping and test recovery are usually left to the end of the automation cycle and as such most people don't get to use them to there full extent but this is one of the most important ingredients of making a truly robust Test Suite or Framework. The ability to trap errors, log details and take the application to a state that can be tested again is what usually allows you to run suites overnight and get intelligent results when you come back in the morning.
Robot's facilities for error handling are pretty much the same as VB.

Visual Test

Error handling in Visual Test is almost the same as in robot I find Visual Tests slightly better why? Just the ability to query error numbers (in a status panel) besides that and some of the errors that they can trap they are the same.

SilkTest

Here is one of the sections where SilkTest stands out. It has all the usual facilities for handling errors by the use of the do…except facility. Basically this works like this, first you write the do statement in your code and this will look to see if any exceptions are raised in the following statements, if they are, then exception handling is passed to the lines following the except statement. E.g.

Do
    Statements to watch

Except (if an exception occurred in the '*Statements to watch*' pass to me)
    Code to deal with the exception

You can do a host of things but this facility in itself does not offer any real advantage over the other tools. But the recovery system does. The recovery system allows you to record a baseline for the application. Before your script is used the application is driven to that baseline. If any error occurs that you are not handling via the do except commands then the recovery system is activated and drives your application back to the Baseline State.

WinRunner

WinRunner does not have an error recovery system as per say SilkTest. It does however have a good exception handling facility. Exception handling can be done at the following levels:

Web Exception Handling; where you look for unexpected windows and deal with them appropriately.
Pop up exceptions; these are pop up boxes e.g. warning boxes, confirmation boxes, etc.
TSL exceptions; these are any of the Winrunner functions you want to monitor for a specific return code.
Object exceptions; these are the objects, their properties and value of the properties you want to monitor.

Again you define a function that will handle the appropriate exception; you put it in a compiled module and then turn the particular exception handler on in the test script that you want to use it.

It does work well but it can be surprising how the exception_on and exception_off functions have to be placed within your scripts to let the handlers do their jobs properly.

QARun

This works the same way as Visual Test and Rational Robot by using the On Error and associated functions.

Object Name Map

Robot

Robot does not support an object map as say WinRunner's GUI Map. If you want to minimise the amount of changes you have to make to your code if an Object/window reference name changes; it will mean replacing names with variables. You can also capture the names of windows/objects at runtime by writing code that gets all the objects of say a web page and searching through these to get the properties for each object types (including the name).

```
Sub Main

    Dim Result As long
    Dim aPropValues() as string
    Dim bPropValues() as string
    Dim iLoop as integer
    Dim Position as integer
    Dim teststring as string
    Dim recMethod as string

    recMethod =
"Type=HTMLDocument;HTMLTitle=Joe90;\;Type=HTMLFrame;Name=fraMain;\;Type=H
TMLDocument;Index=1;\;Type=HTMLFrame;HTMLId=ifraMain;\;Type=HTMLDocument;I
ndex=1"

    StartBrowser "http://somewhere/iteration1/content/defaultFrameSet.html ",
"WindowTag=WEBBrowser"

    Window SetContext, "WindowTag=WEBBrowser", ""
    Window WMaximize, "", ""

    Window SetTestContext, "WindowTag=WEBBrowser", ""
    Browser
SetFrame,"Type=HTMLFrame;HTMLId=fraMain;\;Type=HTMLFrame;HTMLId=ifraMain","
"
    Result = SQAGetChildren(recMethod, bPropValues())
```

```
    For iLoop = 0 to ubound(bPropValues)

        Position=Instr(1,bPropValues(iLoop),"EditBox",1)

        If Position <> 0 Then
            Result = SQAGetPropertyNames(Mid(bPropValues(iLoop),3),aPropValues())

        End if

    Next iLoop

End Sub
```

The above code (excuse any mistakes and poor code standards) will first get the children of a web page then look through the children for editboxes. I could then test particular things per editbox or other objects but the script example would be a lot larger

Visual Test

Visual Test does not have an Object Name map but again the names can be substituted with variables. Visual Test can also search through apps and web pages using for instance the TAB order of objects or find them programmatically.

SilkTest

The window declarations are the Facility that SilkTest uses to store details about all the objects you have selected. Like WinRunner and QARun you can change the identifier (the logical name) in one place. This is very good for script maintenance.
Once again Silk gives you more options with this facility you can add variables, functions, methods and properties to the basic declarations.
Like WinRunner selected items from the windows declarations are highlighted in the application with various bits of information displayed such as the windows name, child objects, etc.
You will use this facility very often.

WinRunner

WinRunner has an excellent GUI map called ‘GUI map’. This allows you to store object’s recognition methods in central repository. This becomes increasing useful as you develop more test scripts.
You can configure what attributes of an object you want to use to identify it. Or you can just make them optional. You can change the logical name of the object to something that better matches it’s function. E.g. if a pushbuttons name was ‘button1234’ when originally captured by Winrunner you may want to change it to ‘confirm payment’. You can use one GUI map per application or several it all depends on your in house standard

A neat facility of the GUI map allows you to select the GUI map name and it will highlight the object on the screen or you can find where the name is in the GUI map by selecting the object. This is useful.

QARun

As mentioned before QARun uses a central repository to store object references. This is possibly more similar to SilkTest and WinRunner. The Object Map looks like a grid/excel spreadsheet with details of the objects in it. You can modify the logical name to a more meaningful description. You can search through the object map to find differences between objects.

Object Identity Tool

Robot

Robot uses a utility called Inspector that is slightly different from older versions. You select inspector and it searches all the open applications and displays all related objects and properties in a tree-like view. You can select an object on the screen and Robot will show you where the object is within the tree.
As with most of the test tools you will use this utility extensively especially to find ways of identifying a control, object or window.

Visual Test

Visual Test uses the Window Information utility to capture object details. The utility is quite neat in that you first select it and point to the window/object and basic details of the window/object are displayed in the window you can also browse the object for more details. You can capture details from standard objects, ActiveX, Web and MSAA objects. MSAA is Microsoft Active Accessibility. When you browse the objects they are presented in a tree-view with properties displayed in the right pane for the selected object.

SilkTest

The object ID tool is part of the Windows Declarations facility.

WinRunner

WinRunner uses a facility called GUI Spy. This allows you to find the name and properties of windows and objects on the screen.

QARun

The object id tool is known as identify. It is the cross hair icon on the tool bar very similar to Visual Test. It does not show as many properties as you may like but it does enough when you are just finding out basic details of a window or object on the screen.

Extensible Language

Robot

Robot's language can be extended in two ways one is by declaring a function in a DLL (win32 API as an example) and then calling it (similar for the other products). You can also extend the language by creating classes in say VB and then creating an instance of the class using the CreateObject function. This allows you to use methods and set/get properties of the class. Obviously to get the full benefit of this you would need to know VB.

Visual Test

Declaring a function in a dll can extend visual Test. What is neat about Visual Test is it can use pointers so functions that require variables to be passed by reference can be used even though the majority are byVal. Before using pointers you should ensure that you have a full grasp of their use as you can crash your application or corrupt data. A good book would be Teach yourself C in 21 days and learn about pointers there.

SilkTest

You have the ability to extend the 4Test language in several ways (remember the skinning a cat).

DLL invocation allow you to call functions stored in DLL's, this is no real difference from the other tools.
I don't want to spend too much time here (because I could) but the language really is nice. It offers inheritance, abstract classes (known as virtual classes in Silk e.g. AnyWin). You can add methods to any of the classes, you can add your own verification properties to classes. – Just to step in, a method is similar to functions in the other Tools. The properties are similar to the properties you see in say the gui spy, inspector, Identity tool, etc. Ok now I'll carry on -. You can create your own classes (similar to user defined functions), you can extend your script using a more traditional language like Java, you can… I'll stop now. It really is a flexible language.

WinRunner

You can extend Winrunner's facilities by creating or using DLL's. It is rare that you will use these but some popular ones are those as found in the win32 API list. I have used the memory functions a lot when checking if an application had a potential leak.
Again to write these DLL's you have to have skills in a more traditional language.

QARun

QARun is extendable through the usual route of DLL's. As can be seen most of the tools do the same thinks but are slightly stronger in some areas.

Environment Support

Robot

Out the box Rational claims Robot has support for HTML, DHTML, Java, Visual Basic, Visual C++, Oracle Developer/2000, Delphi, SAP, PeopleSoft and Sybase Powerbuilder. I've used Robot in HTML, DHTML, Visual Basic and Visual C++ environments with no problems. However with applications using MFC I have not had it so good one application in particular Roguewave Objective views used had no recognition. Remember though the most important part of any automation exercise is that it recognises the application's objects. If it does not as stated before you will quickly stop using automation testing.

Visual Test

Visual Test supports Microsoft centric environments e.g. Internet Explorer, Windows based controls and similar. Although you may be able to get some of the functions to work against custom/non-standard controls It is better to use other tools that have specific functions for the environment. So if you are testing Java, Netscape and similar non-windows environments you should go for one of the other test tools.

SilkTest

Out the box Segue claims support for Support for HTML, XML, JavaScript, Java, ActiveX, Windows 2000, Windows 98, Windows 95 controls, and Visual Basic Multiple browser support for Netscape 4, 6, and Internet Explorer 4, 5, and 5.5, etc. I have used Silk on Client/Server and Web I have not always been impressed with their support for Java applets and HTMLTables but I have not used the latest release (as of this document 5.5) so that may have changed.

WinRunner

I have found Winrunner to work very well in web based environments using multiple browsers this is if you have WebTest enabled and Java add-in support. One thing I have found that has not always worked for me is the analogue mode. In particular if you have a

floating menu's, hint or similar this has been a problem but I have had better success with QARun in this mode.

QARun

Out the box QARun is claimed to support NS-DK, Oracle, Powerbuilder, Visual Basic, Uniface (Compuware's own 4GL), SAP, Java, Netscape, Internet Explorer and most windows applications. I have been impressed with it's character-based test facilities (Compuware have a strong mainframe test tradition) and client/server but not it's web support. On two occasions where I reviewed it for web based products once in Dec 99/ Jan 2000 (did not have good JavaScript/applet support) and Dec 2000 / Jan 2001 (did not have good Java 2 support), it did not come out well compared to the other tools. But I have used it on various client/server applications with no problem and it scored better for recording and replaying applications that had floating menu's (bit like hints). Things may have change now so check with the vendor. But Compuware does advertise TestPartner in their e-business package and QARun in their enterprise one.

Integration

Robot

Robot integrates very well with other rational products including LoadTest, ClearQuest, Purify, Quantify and Purecoverage. There is also limited integration with RequistePro and Rose through TestManager. Through TestManager there is an API that allows for further integration e.g. to run a Java test. Also a few third party products do integrate well with Robot one I have used is QADirector from Compuware to run scripts. As is common with the other tools Robot allows command line running of tests.

Visual Test

Visual Test also integrates very well with some rational products including PureCoverage, Purify, Quantify, Clearcase and ClearQuest. It also integrates with external products; it uses the Visual C++ IDE. There is not as much third party support.

SilkTest

SilkTest does offer some integration to third party products. I have successfully used QADirector the test management tool from Compuware to drive tests developed in Silk and it has worked very well. Also there is support for PVCS. It also integrates with it's own products. I have used the silk radar product (test/incident management) but was not impressed as with Test Director (Mercury), Test Manager (Rational) and QA Director.

WinRunner

WinRunner does integrate with it's own products very well, these include Loadrunner and Test Director. There is also more third party tool support popping up all the time. This I find is because Winrunner is seen as the de facto test automation tool. It is in the UK anyway. One of the latest sales pitches now includes the phase "Go and look on a job site, search for Winrunner and then search for the other tools and see which comes out the most, surely all those people can't be wrong". In the UK they are of course right. You will normally find more jobs for Winrunner than any of the rest and sometimes more than the rest put together. As we all know in daily life quantity does not always mean quality. But in general Winrunner has been a quality solution.

QARun

QARun is very strong for integration. Compuware are the largest of these vendors with a big variety of software product a lot of which integrate with QARun. For overall integration

only Rational can match them on the amount of tools across the board (unit, development, management, etc) they can offer and integrate. Rational wins on development tools but Compuware on database and mainframe. Where it may fail slightly is third party integration. WinRunner does have good third party support

Cost

Robot

There is not that much to say here all the tools come within a certain price range that will all depend on quantity, salesperson, size of business the usual stuff but look at $2,900 - £5000 (the dollar and pound signs are deliberate) per license. However I have found Robot to be on the lower side of this scale and this is not because it is a poorer product. Not in other's and my experience. You can also normally get a few more products thrown in like Test Manager and Clearquest for a modest increase.

Visual Test

This is one area that will not be under heavy debate. Visual Test is the cheapest of the test tools. Also you can distribute tests to several machines without having to have several licenses. If all the tools in you evaluation can achieve what you need then you might as well go for Visual Test you can get almost 10 for the price of one of the other test tools in some cases.

SilkTest

I have found that Segue is on the expensive side of the scale. Per release I do think that Segue generally adds more to the functionality of it's products than the other tools. I have worked with several generations of rational from SQA robot 5 or 6 (may have been Cyrano before 6) through to 2001a, from WinRunner 4 – 7, QA Run ? – 4.7, etc yet I think Silk does tend to advance its product more.

WinRunner

I have found Winrunner to be towards the more expensive side of the scale. Please don't take the "you get what you pay for" sales pitch in this case because in most cases another test tool can do the job for cheaper. I have had this talk so many times especially when trying to get a Java add-in that should be included in the first place. If they said well we employ more people in the helpdesk to service our customers, etc I would say on evidence they are probably correct as I have found their helpdesk to be excellent.

QARun

I have found QARun to be on par with Rational Robot for cost. That is less than SilkTest and WinRunner but more than Visual Test so middle of the road. Hey just catch a sales person near that much stated term "End of Quarter" and if the tool work's for you get them to throw in everything. Sometimes for the equivalent price of WinRunner you can get QARun, QADirector, TrackRecord (not too impressed but it does the job), FileAid and Sitecheck for the same price and all in all they are excellent products.

Ease of Use

Robot

For me out the box Robot is the easiest to learn. It is not as wizardly as WinRunner nor is it as robust as Silk but to get a person with no programming skills up and running I find this the easiest from the look of the development interface, the record and playback, log

files, Test Results, verification points and debugging I find very simple for the beginner. Yet powerful enough to achieve most things.

Visual Test

For a beginner I find Visual Test is fairly difficult to learn even though the language is based on VB. It has pointers, some C++ tools and the interface I don't find as slick as the others in this review. It has a lack of wizards and utilities found in the more expensive tools. Like wizard to start an app, automatic function selection, etc.

SilkTest

You will see from the matrix that Silk does not score big here. For a novice in automation or programming in general this is a big learning curve. Even for some with reasonable automation experience it does take getting use to! However in the long run you will find Silk a lot more flexible for carrying out a lot of your testing activities

WinRunner

WinRunner is middle of the road for ease (for the beginner). I would say Robot first, then QARun, Winrunner, Visual Test and then Silk. However I am just really talking about writing raw code. If you look at the wizards then hey, Winrunner has gone wizard mad. You can do almost anything via a wizard and in this case it is nearly right up there but for me not quite up there with Robot for ease

QARun

Even though the language is 50+% (my estimate) the same as Robot it just has a little bit more out the box to learn but experienced automation testers will not have any trouble.

Support

Robot

The answers to questions from the help desk have not always been good. They usually refer you to the manual even when you have told them you have read it already. I find that the forums especially www.dundee.net/sqa/ are more useful than the help desk as you can get more than one answer from people who have used the tools extensively. On this particular forum you do get the occasional answers from a rational representative that is usually quite good.

Visual Test

As Robot but a good mailing list is `http://www.vtindepth.com/mt_info.htm`.

SilkTest

Please note this is my experience (and some of my friends) in the UK. Silk's support is poor. In the states I hear this may be the case too. They were doing some good stuff round about the turn of 1999/2000 I was privileged enough to go on one of their courses 'e-confidence' for free due to the large purchase we had made. It lasted four days and was very good. But they need to improve in this area. This is a shame, as it is not the easiest product to learn.

WinRunner

I have found Mercury to have the best support, whether on their site for user contributed help or from the help desk. May be that's why they are expensive ☺. But where money is

not a big issue this can sometimes sway my decision. They will phone you periodically and ask how they/you are doing, etc and for the price you've paid they should. I have never for any of the tools brought just one license so the cost is normally tens of thousands of pounds enough to buy a house and then on top of that a maintenance fee! Of course they should be calling me to see how I am.

QARun

I have not seen very much in the way of online help (internet forum's) but the helpdesk is fairly good and if you need resources they have by far the largest professional services team of the test tool vendors.

Object Test

Robot

Robot's facilities for doing object tests are known as verification points. The following verification points are available: Alphanumeric, clipboard, Menu, Object data, Object properties, Region Image, Windows existence, windows image, file comparison, File Existence, module existence, website scan and website compare.
You can chose to verify just one object or the object and it's child objects.

To activate these checkpoints you would navigate/record through to the window, object, etc that you want to verify and then select from the tool bar (insert) the appropriate verification. When this is done you will have an array of properties that you can choose to include in the verification or not. You can select whether the verification is expected to fail or pass. You can edit the values, etc. It is very flexible. Again this is all on the premise that Robot can recognise your application objects in the first place.
You will find that this is the main evidence you will use to say whether a test has passed or not.
If verification fails during play back you can choose to ignore the fail, raise a bug or update the baseline.
A neat thing in robot is the OCR function, it doesn't always work (depending on how sensitive you make the settings) but it sometimes may be your only way of getting text from images.

Visual Test

Object tests as implemented by Visual Test are called verification similar to Robot's verification points. You usually record verifications when recording a test scenario.

SilkTest

In SilkTest you can verify the state of a given object. This can be achieved by looking at the class of object methods. Say a Pushbutton class, etc. One thing to remember is you should look at the class hierarchy (Object browser) to see if the method you want is contained in the parent class. Basically you can carry out similar checks of properties as per the other tools but you have to look for a method e.g. VerifyFocus in the control class (which is the parent of Pushbutton) to do the test you require.

WinRunner

WinRunner uses checkpoints to carryout object tests. The available checkpoints are:

GUI, Bitmap, database. You also have a get text (sort of a checkpoint) and synchronisation points.
When you use checkpoints you can capture just the default checks or you can edit the properties. A neat thing I like about WinRunner is it's ability to do multiple checks in one sitting without having to go through a cycle of selecting from the toolbar again.

If a check fails, you can choose to show the baseline, actual and the differences.

QARun

As mentioned before they have a facility know as checks, which allows you to carry out various Object tests. These tests include Bitmap, Clock, File-Aid, Form, List, Menu, Site, Table, Text and Window. Don't let the long list fool you. The others can equally do these tests except maybe text checks as you can get text from a bitmap image as a string allowing you to do more complicated verification. Also you can create patterns in your checks to look for text that follows a particular format, numbers that are less than, equal to, greater than or between two limits. So it is very flexible. Also its date ageing facility (primarily brought about to help the Y2K effort) is still handy for those accounting forward date checks and similar.

Summary

So which tool is best well the winner is VisualRoWinSilkQA. If the tools were as cheap as Visual Test, had good data creation facilities as Robot, were as good cross browser and wizardry as WinRunner, had a flexible and powerful language as SilkTest and a central repository and text facilities of QARun then hey!

But I will try to provide a limited guide.

As anyone will tell you, you will need to demo a few products on your target application. I would say demo 3 and no more or you get tool overkill.

1) Price; Ok it all depends on your criteria. If price is a big issue then I have found that Visual Test, Robot and QARun tend to be at the more economic end of the scale. This however does not mean they cannot cope. Besides Visual Test the other two are equally matched with Silk and WinRunner in most areas.
2) If price is not an issue and you want a comprehensive package that may require linking to development tools. Then Compuware and Rational win here.
3) For web testing I would tend to evaluate WinRunner, SilkTest and Robot first in that order.
4) If you will be creating huge regression packs for several projects possibly on various platforms. I would look at SilkTest, WinRunner and QARun in that order. Why? I found these tools have superior facilities for maintaining scripts in either the GUI MAP or similar facilities as well as baseline recovery systems.
5) If your going to require a lot of professional services support or helpdesk support then look at Mercury and Compuware.

As you can see I'm trying to reflect the tool's strengths and weaknesses. Again it will be down to demonstrating on your site and seeing which you feel happiest with.

Rational: Strengths: - Economical, Lots of supporting tools, good extensible language through VB, good data creation facilities, good online community.
Weaknesses: - Support, language is a little basic out the box and a bit confusing with tools breakdown e.g. visual test, TeamTest, Test studio, test factory, etc.
Mercury: Strengths: - Possibly the most popular test tool, plenty of jobs, good support, good online community, Good cross browser support.
Weaknesses: No Java support out the box, towards the more expensive solution.
Compuware: Strengths: - Strong professional services, good integration with development products, Economical.
Weaknesses: - Poor online community, Product breakdown (Should we still use QARun for web? Or buy TestPartner for web and QARun for Client/Server?).
Segue: Strengths: - Good development language, good online community, recovery system.
Weaknesses: - Helpdesk, towards the more expensive side.