3 Evaluierung

Alle evaluierten Tools wurden (sofern für das jeweilige Betriebssystem erhältlich) in Linux (Ubuntu 8.04 oder OpenSuse 10.3) und in einem Windows-System (XP oder Vista) installiert und einer Funktionsprobe unterzogen. Die eigentliche Evaluierung war (außer bei reinen Linuxtools) in Windows XP Professional. Für Tools, die sowohl Windows als auch Linux unterstützen wurde die Portierungsmöglichkeit der Testfälle getestet.

3.1 Tools für GUI-Anwendungen

3.1.1 QF-Test

Das Tool der Firma *Quality First Software GmbH* (http://www.qfs.de) ist ein sehr einfach und intuitiv zu bedienendes Werkzeug mit dem Focus auf Java und ist der Kategorie Capture / Replay zuzuordnen. Es unterstützt sowohl den analogen, als auch den objektorientierten Ansatz (d.h. es wird eigentlich auf Objektebene gearbeitet, aber eine pixelgenaue Positionsabfrage des Mousecursor o.ä. ist möglich.

Gründe für die Evaluierung waren einerseits die sehr positiven Rückmeldungen von Kunden und in Foren, der sehr intuitive Eindruck, den das Tools beim ersten Schnelldurchlauf erzielt hat und nicht zuletzt die Referenzliste (über 400 Kunden weltweit), die von Großfirmen wie AGFA, Philips, RTL über Finanzdirektionen, Krankenversicherungen bis hin zu Hochschulen reicht. Ebenfalls ein wichtiger Punkt war, dass sowohl IBM als auch Hewlett Packard in der Kundenliste aufscheinen, die selbst ein solches Produkt im Angebot haben.

Die getestete Version ist Version 2.2.2.
Bei der Lizenzierung (siehe Abschnitt 2.4.3.1) gibt es eine Entwicklerlizenz und eine Runtimelizenz. Wenn ein Entwickler auf einem Rechner Tests entwickelt und auf (einem) anderen zeitgleich nur Tests ausführt, braucht er auf den anderen Rechnern zusätzlich zu einer bestehenden Entwicklerlizenz eine Runtimelizenz, bei zeitversetzter Verwendung reicht eine Lizenz aus.

Die Installation verläuft sowohl unter Windows XP als auch beim eingesetzten Linux (Ubuntu) problemlos, eine funktionierende JDK wird vorausgesetzt. Wenngleich Windows Vista nicht offiziell unterstützt wird, so waren Installation und Ausführung des Tools ohne Fehler oder sonstige Probleme möglich. Auch das Instrumentieren der JDK hat keinerlei Probleme verursacht, es läuft (wenn man es nicht anders möchte) alles automatisch.
Da es sich um eine kommerzielle Software handelt, muss man über ein Formular eine Evaluierungslizenz beantragen. Der Wissensdurst der Firma ist vertretbar, wer ganz anonym testen möchte, kann die Software auch ohne Lizenzdatei starten, allerdings ist dann nur das Laden der mitgelieferten Testsuiten möglich, Testfälle können dann nicht gespeichert werden.
Die Lizenz wird händisch bearbeitet, was laut Webseite ein wenig dauern kann. Obwohl der Antrag an einem Samstag um 22:30 Uhr erfolgte, wurde die Lizenzdatei noch am selben Tag geliefert, was auf guten Kundenservice schließen lässt. Und dieser Eindruck hat sich im Laufe der Zeit bestätigt, die MitarbeiterInnen sind sehr freundlich und sehr bemüht und obwohl es sich in diesem Fall offensichtlich nicht um eine Kaufabsicht handelt, wird Hilfe per Mail und auch per Telefon angeboten. Die Evaluierungslizenz ist 4 Wochen gültig, in wichtigen Fällen kann direkt aus der Software heraus um eine Verlängerung angefragt werden. Auch hier eine sehr rasche, freundliche Antwort, die Verlängerung war problemlos.

Mit der Software mitgeliefert werden auch deutsch- und englischsprachige Schulungsunterlagen (ein Handbuch und ein Tutorial, jeweils im HTML- und im PDF-Format, ein Schnellstart, FAQs und Links zu den Supportseiten), die direkt unter dem Menüpunkt Hilfe aufgerufen werden können. Das sehr gut aufgebaute Tutorial zeigt eine schrittweise Anleitung für mitgelieferte Beispielanwendungen und vermittelt einen schönen Einblick in die Grundzüge der Software, die man sofort in die Tat umsetzen kann. Auch das Handbuch ist sehr gut geschrieben, übersichtlich (guter Einsatz von Screenshots), beim Tutorial ist mit ungefähren Zeitangaben für die einzelnen Kapitel auch eine Zeiteinteilung mitgeliefert, die ein versierter Tester deutlich unterbieten kann.

Erstellte Testfälle konnten direkt vom Windows- ins Linuxsystem übernommen werden, eine Bearbeitung ist nicht notwendig.

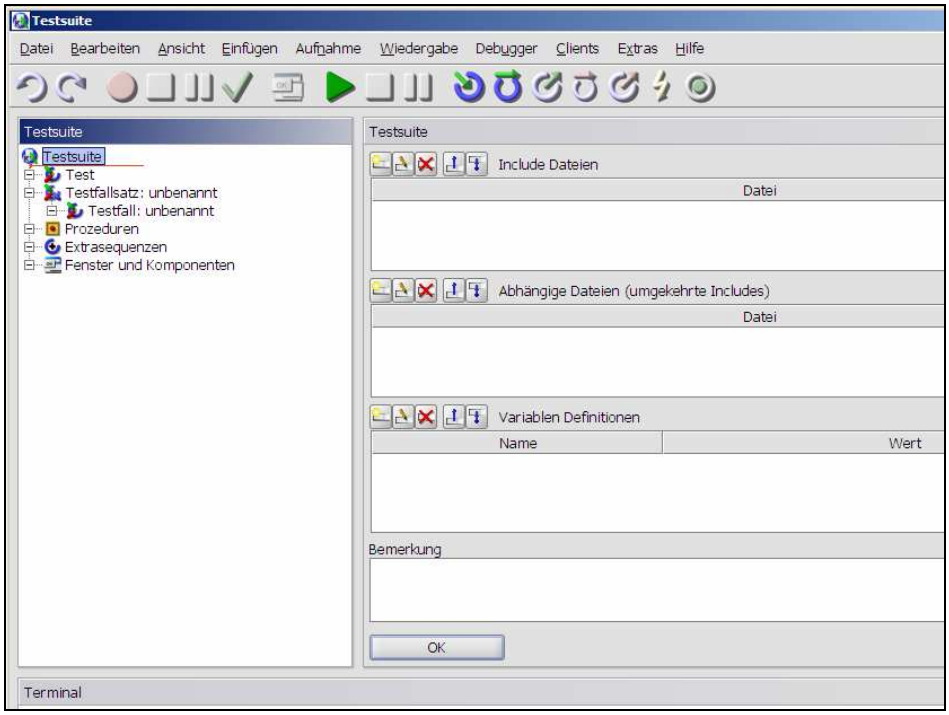

Abbildung 8: Fenster einer Testsuite, QF-Test

Das Hauptfenster (Abbildung 8) zeigt die Testsuite in einer Baumstruktur, in der die Tests, Prozeduren und vorhandenen Komponenten (Fenster, Buttons, Textfelder etc.) angezeigt werden und aus einem Zweig Extrasequenzen. Dieser Zweig ist sozusagen die „Spielwiese“ des Testers, in der man Testfälle anlegen kann, verändern kann und in der die aufgezeichneten Testfälle landen. Dies ist etwas ungewöhnlich, aber die Idee ist gut, weil man so die vorhandenen Tests laufen lassen kann, ohne durch einen noch nicht fertiggestellten Test Fehler zu bekommen. Ein zweiter Punkt der gewöhnungsbedürftig ist der, dass man einen Knoten der Baumansicht immer mit einem Klick auf das „+“ öffnen muss, um neue Elemente einzufügen (auch wenn noch keine Elemente da sind), das Aktivieren des Astes per Markierung genügt nicht.

Der Aufbau eines Tests geschieht in der üblichen Weise, die ein Programmierer aus dem Thema Unit-Test kennt:

Zuerst muss die zu testende Software (SUT) gestartet werden. Dies geschieht hier bequem direkt aus der Testsoftware unter dem Punkt „Knoten einfügen“. Auf diese Weise können auch ganze Testfälle selbst zusammengestellt werden (ohne Ausnutzen der „Capture-Funktion“) und bestehende Testfälle einfach erweitert werden, alle Useraktionen stehen als neue Knoten zur Verfügung.

Am Beginn des Testfalles sollte man ein „Setup“ machen, in dem der Client (die SUT) gestartet wird, dann sollte man zur Sicherheit auf den Client, als auch auf die richtige Komponente warten, damit die aufgezeichneten Events beim Abspielen nicht ins Leere gehen (vergleichbar mit dem Setup beim Unit-Test). Damit auch am Ende alle Anwendungsfenster geschlossen werden, ist am Schluss ein „Aufräumen“ einzufügen. In dieser Sequenz wird die SUT ordnungsgemäß geschlossen („Tear Down“ beim Unit-Test).

Zur Aufzeichnung der Testfälle hat man eine eigene Symbolleiste, die neben den zu erwartenden Record-, Pause- und Play-Tasten noch die Möglichkeit bietet, Teile der Tests (Einzelschritte, Knoten, Knoten überspringen) auszuführen oder auszulassen.

Ferner kann man während der Aufnahme so genannte Checks aufzeichnen, damit kann man z.B. Textfelder auf Inhalt, Checkboxen auf Zustand überprüfen, oder sogar Bildschirmausschnitte mit erwarteten Bildern vergleichen. Dabei ist aber zu beachten, dass bspw. das unbeabsichtigte Verschieben eines Fensters bei der Aufzeichnung (das man dann im Testfall natürlich wieder „bereinigt“) zu Fehlern im Test führt. Das ist aber selbstverständlich kein Bug des Tools, sondern zeigt die Problematik des analogen Testkonzeptes.

Die Aufzeichnung funktioniert problemlos, nach etwas Einarbeitungszeit lassen sich auf diese Weise sehr schnell neue Testfälle generieren. Die Nachbearbeitung ist auch denkbar einfach und intuitiv. Wählt man den Knoten bei den Extrasequenzen aus, so erhält man im rechten Teil des Fensters die zum Event gehörenden Eigenschaften.

Es lassen sich nahezu alle denkbaren Eigenschaften, von der Art des Mouseevents (Klick, Taste drücken, Taste loslassen, Mousebewegungen per X, Y Koordinate, Anzahl der Klicks u.v.m.) bis hin zu Verzögerungen (besonders wichtig bei Testeingaben, um die Verzögerung beim Tippen zu simulieren um eventuelle Timeouts zu testen) verändern. Die Möglichkeit Tests mit der Scriptsprache Jython zu ergänzen scheint hier für die meisten Anwendungsfälle als nicht notwendig, wenngleich ein versierter Programmierer, der mit der Sprache vertraut ist hier einen zeitlichen Nutzen ziehen kann.

Bei der Erstellung von Checkpunkten bietet QF-Test die Möglichkeit Text zu vergleichen, den Status eines RadioButtons und auch einen Bildvergleich. Beim Bildvergleich kann der Ausschnitt selbst frei gewählt werden. Bei solchen Checks kann man selbst bestimmen, ob im Falle eines nicht durchlaufenden Tests eine Nachricht, eine Warnung oder eine Fehlermeldung erfolgen soll.

Abbildung 9 zeigt eine Auswahl an möglichen Knoten, man kann ersehen, dass beinahe jedes mögliche Event im Testfall auftauchen kann.

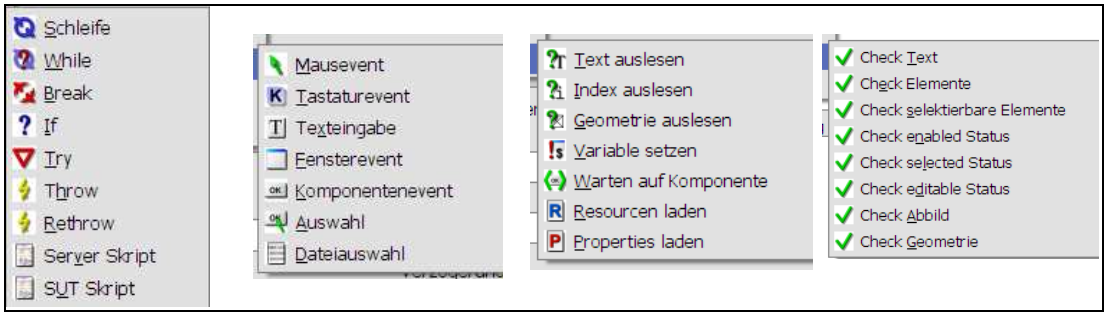

Abbildung 9: Auszug möglicher Elemente, QF-Test

Wie man erkennen kann sind Kontrollstrukturen und Schleifen möglich, wodurch auch Iterationen getestet werden können. Der Import externer Daten für die Testfälle ist möglich, es werden z.B. das CSV und das Excelformat unterstützt

Ist der Test fertig aufgezeichnet, verschiebt man die Sequenz in den Test, von wo aus man ihn ablaufen lassen kann. Dabei kann man auch selektiv

Testsequenzen hinzufügen und weglassen, oder einzelne Testschritte ausführen.

Der Test kann dann jederzeit (und beliebig oft) mit dem Play-Button gestartet werden. Als Ergebnis erhält man eine Rückmeldung, wie viele Fehler bzw. Warnungen während des Tests aufgetreten sind. Die Warnungen sind meist nur Hinweise auf „unschöne“ Programmierung die aber den Testablauf und die Funktionalität der Software nicht beeinträchtigen.

Aus dem Testlauf entsteht ein Protokoll, das die fehlerhaften Testevents kennzeichnet (Abbildung 10) und den genauen Fehler (mit Soll- und Ist-Wert) ausgibt. Die Kennzeichnung könnt hier eventuell etwas deutlicher ausfallen, bei sehr langen Tests kann man die rot umrahmte Box schon übersehen. Da bietet allerdings QF-Test noch die Möglichkeit direkt zu Fehlern zu springen. In der Abbildung nicht zu sehen ist die Protokollierung der Testzeit und der Dauer des Testdurchlaufes.

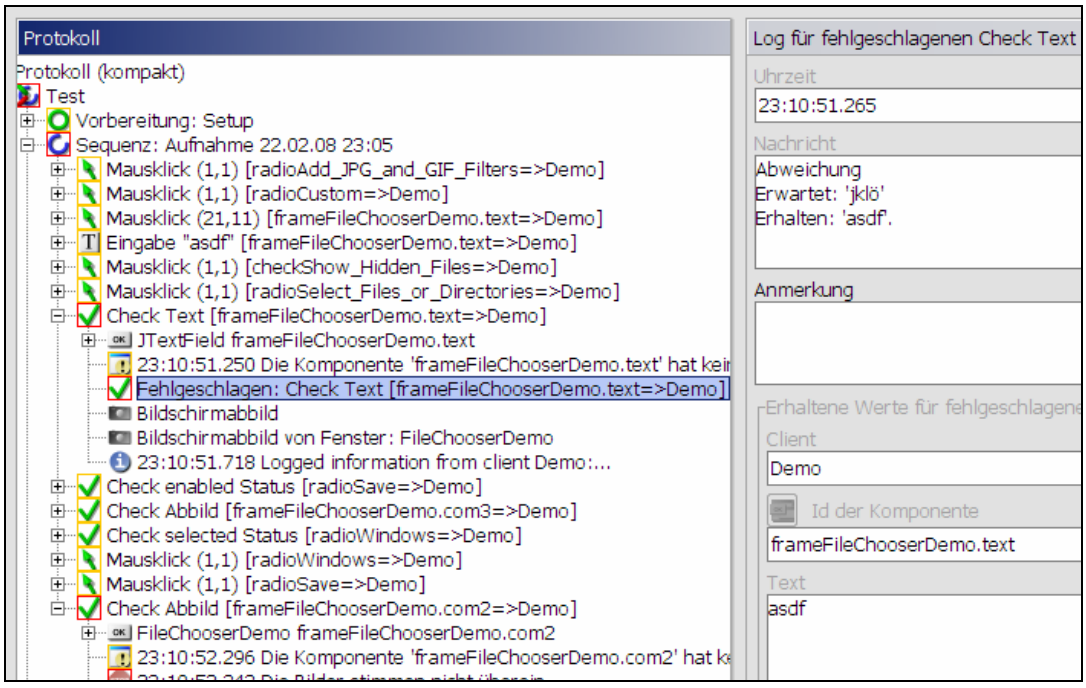

Abbildung 10: Testprotokoll, QF-Test

Bei einem Bildcheck gibt es bei QF-Test mehrere Möglichkeiten, man kann sich jedes Bild einzeln, beide zusammen (Abbildung 11) oder auch nur die

Differenz oder die XOR-Bilder anzeigen lassen (Abbildung 12) um nur die Unterschiede angezeigt zu bekommen. Hier ist besondere Vorsicht geboten (das betrifft allerdings die Bildvergleiche aller Tools), da bereits eine unterschiedliche Markierung (also wenn man bspw. beim Referenzbild eine Combobox aktiviert und danach wieder deaktiviert) zu Unterschieden zum Bild des Testdurchlaufes führen und der Test somit als fehlerhaft gekennzeichnet wird.

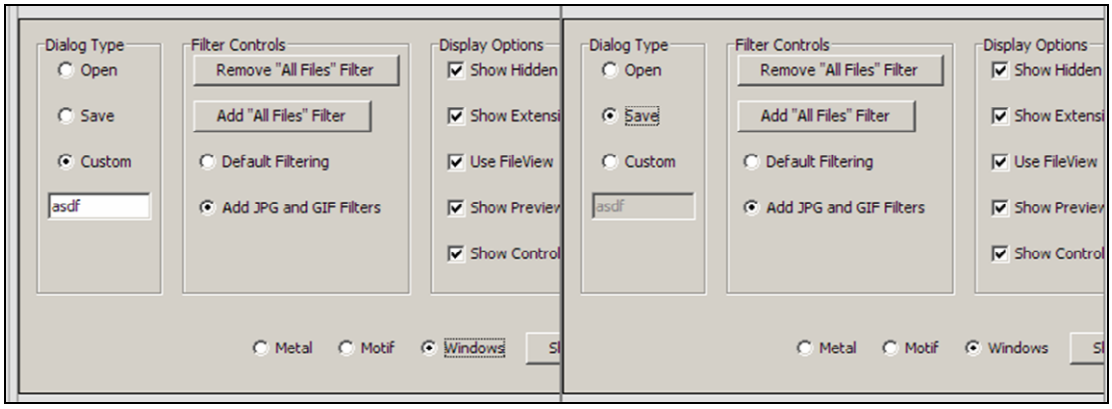

Abbildung 11: Bildvergleich, QF-Test

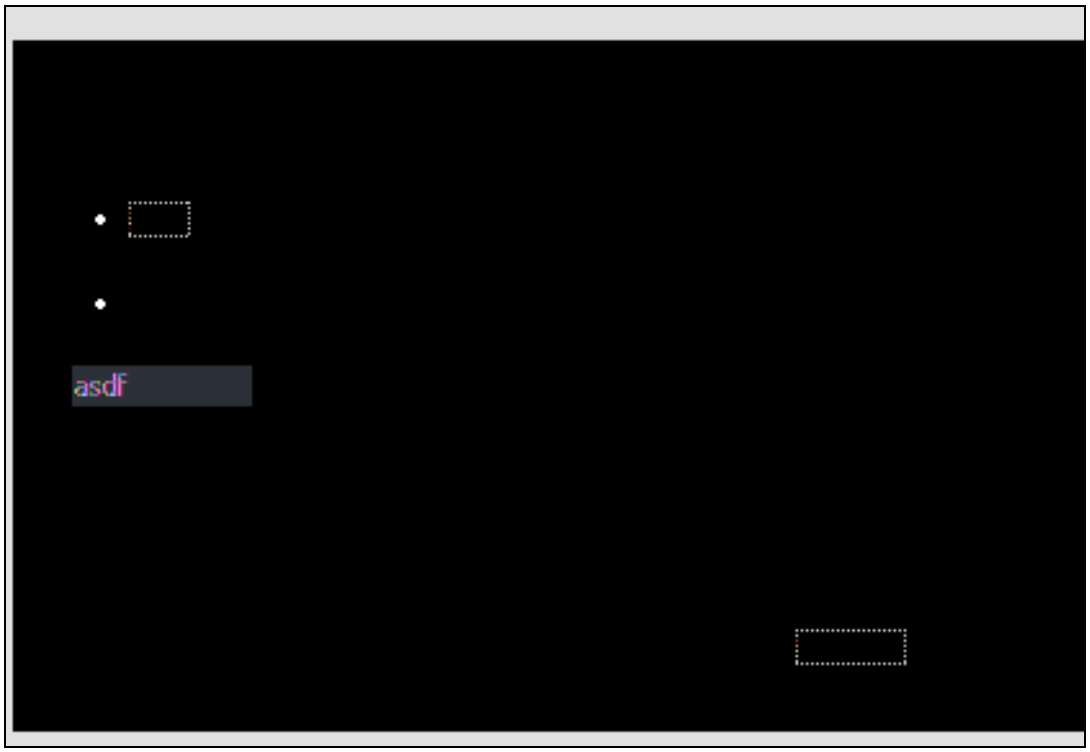

Abbildung 12: Bildvergleich XOR-Bild (Diff), QF-Test

Die Umschaltung zwischen den Darstellungen erfolgt einfach mittels Symbolleiste.

Aus dem Protokoll kann man sich einen Bericht im XML- oder HTML-Format generieren lassen und natürlich abspeichern.

Im HTML-Format (Abbildung 13) kann man sich die Fehler auch mittels Meldung und Screenshot (in beliebiger Skalierung) und die Bildvergleiche direkt darstellen lassen, der Bericht wirkt sehr sachlich und durchaus übersichtlich. Auch die Zusammenfassung und die kleine statistische Auswertung runden das Bild des Berichtes ab. Die Bilder liegen im png-Format im jeweiligen Projektordner und werden auch als Verknüpfung im XML-Bericht eingetragen, der somit eine Exportmöglichkeit zu anderen Tools ermöglicht (z.B. eine Anbindung an ein Bug-Tracking-System)

In Version 2.2.1 ist beim Testen ein Fehler bei der Testauswertung (falsche Prozentberechnung). Auf Anfrage wurde unmittelbar reagiert, das Upgrade auf Version 2.2.2 hat dieses Problem behoben.

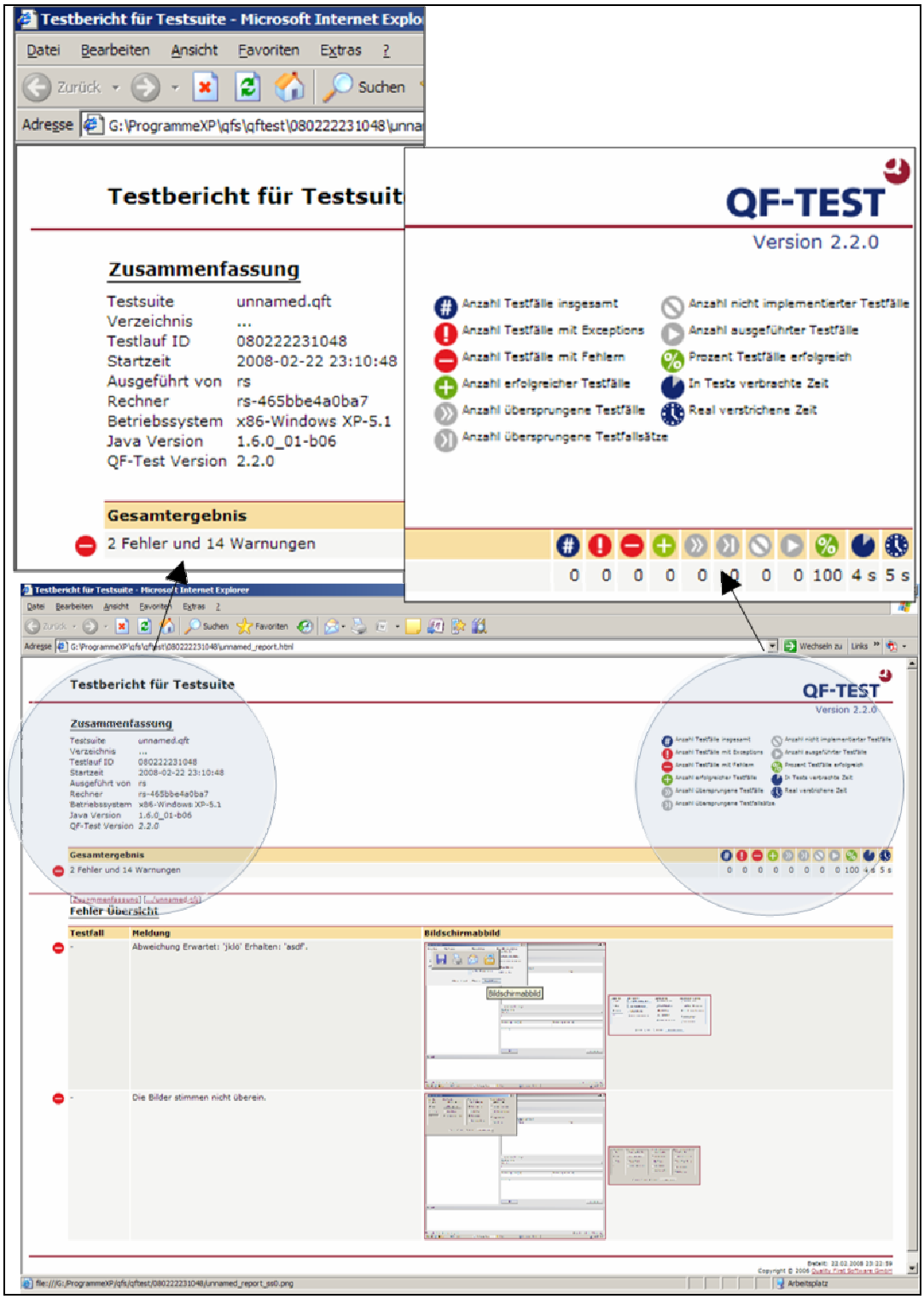

Abbildung 13: HTML-Protokoll, QF-Test

----------Seitenumbruch----------

Fazit:

Das Tool QF-Test ist ein sehr intuitiv und einfach zu bedienendes Werkzeug, das sich der Capture & Replay-Technologie bedient und eine Bearbeitung der Testfälle mittels Drag'n'Drop oder mittels Jython ermöglicht.
Die Programmoberfläche ist sehr übersichtlich gestaltet, der Start des Projektes (Einbindung der SUT) ist ohne großartige Kenntnisse möglich.

Wer Java GUI-Anwendungen testen möchte, ist bei QF-Test gut aufgehoben, tiefgreifende Programmierkenntnisse sind für gute Testergebnisse nicht erforderlich, bei der Einbindung von externen Datenquellen sind sie allerdings schon von Vorteil.