Technische Universität Ilmenau
Fakultät für Informatik und Automatisierung
Fachgebiet System- und Softwareengineering

Entwurf einer Testautomatisierung der TimeNET-GUI

vorgelegt von:
Frau Katharina Schröder
Matrikel-Nr. 47915
Studiengang Wirtschaftsinformatik

Hauptseminar Informatik

Wintersemester 2012/2013

Betreuer: Prof. Dr.-Ing. habil. Armin Zimmermann

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

GUI	Graphical User Interface
SWT	Standard Widget Toolkit
TU Ilmenau	Technische Universität Ilmenau

1 Einführung in die Thematik

1.1 Situationsanalyse und Problemstellung

Seit einigen Jahren arbeiten die Mitarbeiter des Fachgebiets System- und Softwareengineering der Technischen Universität Ilmenau in Zusammenarbeit mit Studenten an der Entwicklung eines Softwareprojekts. TimeNET ist ein interaktives Softwarewerkzeug, das die Modellierung stochastischer Petri-Netze vereinfachen soll.

In diversen studentischen Arbeiten wurden verschiedenste Funktionen der Software entwickelt und die bereits bestehenden weiter optimiert. Dies hatte zum einen zur Folge, dass TimeNET inzwischen in Version 4 mit deutlich mehr Funktionalitäten im Vergleich zum Vorgänger TimeNET 3.0 zur Verfügung gestellt werden konnte. Jedoch zog diese Vorgehensweise zum anderen auch Nachteile mit sich, da eine Vielzahl unterschiedlicher Entwickler an dem Projekt beteiligt war. Jede neu programmierte Funktion wurde vor deren produktiver Nutzung zwar für sich auf ihre Funktionsweise hin überprüft, eine Prüfung der gesamten TimeNET-Software auf Fehler wurde jedoch nach mehreren Entwicklungszyklen immer schwieriger.[1]

1.2 Motivation und Ziel der Arbeit

An dieser Stelle setzt die vorliegende Hauptseminar-Arbeit an. Es soll eine Möglichkeit zum automatisierten Testen der TimeNET-GUI geschaffen werden. Dabei ist zunächst offen, ob dies durch eine eigenständig entwickelte Test-Software oder durch ein bereits auf dem Markt erhältliches Tool realisiert werden kann. Für welche Variante sich die Verfasserin dieser Arbeit entschieden hat wird in Abschnitt 2.2.1 beschrieben.

1.3 Aufbau der Arbeit

Zunächst werden in Kapitel 2 einige Grundlagen behandelt. Dazu gehört die Klärung der zum Verständnis dieser Arbeit grundlegenden Begriffe ebenso wie die Beschreibung der Vorgehensweise bzw. des Untersuchungsaufbaus dieser Arbeit. Anschließend werden in Kapitel 3 die Ergebnisse der Untersuchung präsentiert. Im abschließenden Kapitel 4 werden die vorgestellten Testwerkzeuge miteinander verglichen und eine Handlungsempfehlung ausgesprochen. Ein kurzes Fazit sowie ein Ausblick in mögliche künftige Arbeiten vervollständigen die vorliegende Arbeit abschließend.

[1] Vgl. [TUI12]

2 Grundlagen

2.1 Grundlegende Begriffe

Zum besseren Verständnis dieser Arbeit werden zunächst zwei grundlegende Begriffe definiert. Dies sind zum einen das GUI und zum anderen die Softwaretests.

2.1.1 Graphical User Interface

Ein so genanntes „Graphical User Interface“ (kurz GUI) ermöglicht dem Anwender einer Software die Interaktion mit dieser über eine grafische Oberfläche. So wird eine Anwendung für viele Nutzer bedienbar gemacht. Je nachdem, um welche Art von Software es sich handelt, umfasst eine grafische Benutzeroberfläche mehrere Funktionsbereiche. Dazu gehören zum Beispiel die Taskleiste, eine Menüleiste oder eine Symbolleiste, die unter anderem grafische Symbole oder Text-Befehle enthalten können. Durch Mausklick auf diese Befehle können verschiedene Funktionen aufgerufen werden.[2]

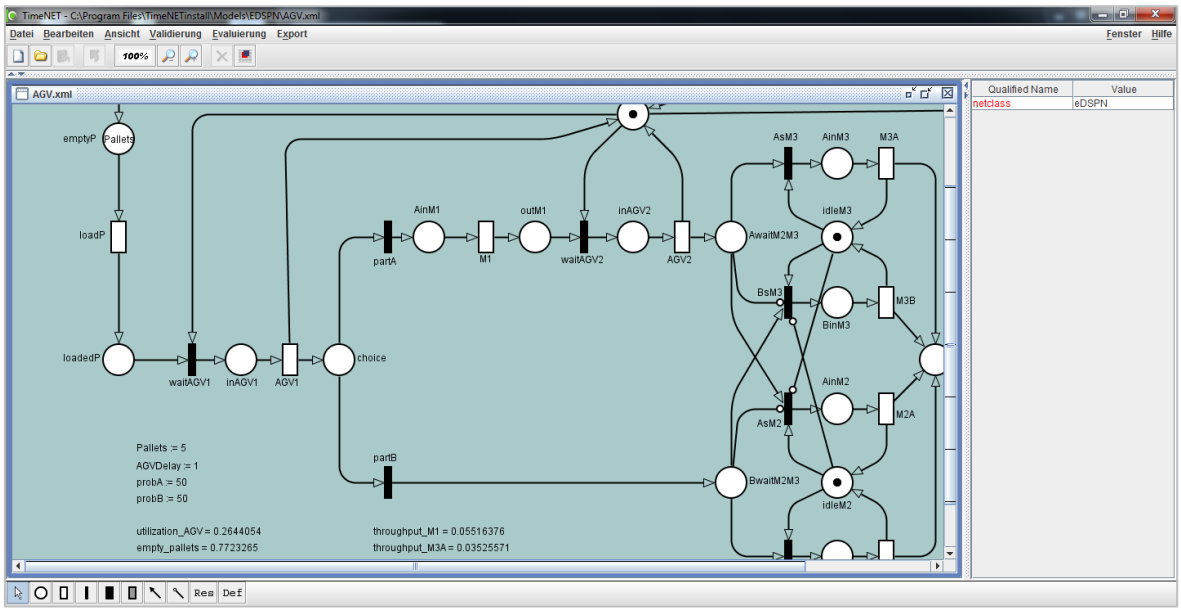

Abbildung 1: Screenshot TimeNET

Quelle: Von der Autorin erstellt

Der entscheidende Vorteil von GUIs im Vergleich zu befehlsorientierten Benutzeroberflächen (auch Command Line Interface bzw. CLI genannt), ist das selbst wenig erfahrenen Nutzern die Arbeit mit der Software erleichtert bzw. ermöglicht wird. In Abbildung 1: Screenshot TimeNET ist das GUI von TimeNET zu erkennen, welches unter anderem Standardsymbole sowie eine Menüleiste mit Befehlen wie „Datei“ → „Neu“ verwendet.

[2] Vgl. [ITW12a]

Command Line Interfaces, wie zum Beispiel das DOS-Betriebssystem mit dem entsprechenden Kommandointerpreter Shell, haben im Vergleich zum GUI deutlich höhere Anforderungen an den Nutzer, da er diverse Kommandos und Befehle erst erlernen und sie bei Bedarf abrufen muss. Aus diesem Grund wurden die befehlsorientierten Benutzeroberflächen zur grafischen Benutzeroberfläche weiterentwickelt.[3]

2.1.2 Softwaretests

Gemäß Glenford J. Myers wird der Begriff Testen folgendermaßen definiert:

> *„Testen ist der Prozess, ein Programm mit der Absicht auszuführen, Fehler zu finden."*

Auf das Testen einer Software übertragen kann man demnach von jener zielgerichteten Tätigkeit sprechen, die das Auffinden von Fehlern in einer Software ermöglichen soll.[4] Damit soll sichergestellt werden, dass die Qualität der Software auch bei deren Weiterentwicklung gleichbleibt. Dies ist insbesondere in solchen Fällen wie dem TimeNET-Projekt unverzichtbar, um die Funktionsfähigkeit der Software auch nach der Implementation einer neuen Funktion gewährleisten zu können. Damit das Testen jedoch keine oder zumindest so wenig wie möglich Ressourcen beansprucht, ist es sinnvoll, diesen Prozess zu Automatisieren.

2.2 Vorgehensweise und Methodik der Untersuchung

2.2.1 Konzeption und Untersuchungsaufbau

Das Ziel der Arbeit bestand, wie in Kapitel 1 bereits erwähnt, im Entwerfen einer Testautomatisierung für das TimeNET-GUI. Zunächst bot sich jedoch an, Nachforschungen über die bereits auf dem Markt vorhandene Test-Software zu betreiben. Dies wurde im ersten Schritt dieser Arbeit getan. Im Zuge der Forschungen wurde deutlich, dass eine aufwändige Eigenentwicklung eines GUI-Test-Tools nicht zweckmäßig ist, da bereits viele Tools entwickelt und von diversen Herstellern zum Kauf oder Download angeboten werden.

Daraus ergaben sich veränderte Ansprüche an diese Arbeit, da nun nicht mehr Anforderungen für eine neue Testsoftware entwickelt werden mussten. Vielmehr galt es nun Auswahlkriterien für die bereits vorhandene Software zu erarbeiten, um so das für die TU Ilmenau bzw. das Projekt TimeNET am besten geeignete Tool herauszufiltern. Im folgenden Abschnitt werden diese Anforderungen und Auswahlkriterien dargestellt.

---

[3] Vgl. [ITW12b]
[4] Vgl. [Mye01], S.15

2.2.2 Auswahlkriterien

Eine Grundvoraussetzung war von jeder Testsoftware zu erfüllen – es sollte damit das GUI von TimeNET, welches auf der Programmiersprache Java basiert, getestet werden können. Somit entfielen schon im ersten Ansatz sämtliche Testtools, die sich in ihrer Funktionsfähigkeit auf Last- oder Unit-Tests beschränken.

Die angebotenen Testtools sollten des Weiteren nach Möglichkeit folgende Kriterien mindestens erfüllen:

- Lauffähigkeit sowohl unter Windows, als auch unter Linux ohne doppelten Pflegeaufwand
- Möglichst geringer Pflegeaufwand der Tests
- Offene und erweiterbare Testfälle

Wünschenswert, jedoch nicht zwingend erforderlich war folgende Anforderung:

- Open Source Anwendung bzw. freie Lizenzen und damit Kostenneutralität für das TimeNET-Projekt

2.2.3 Ausgewählte Tools

Im Zuge der Forschungsarbeit wurden sehr viele Tools auf ihre Anwendbarkeit im TimeNET-Projekt überprüft. Die meisten waren jedoch nicht oder nur teilweise geeignet, da sie die oben genannten Anforderungen nicht erfüllten.

Herauskristallisiert haben sich letztlich fünf Werkzeuge, die nach der ersten Recherche zunächst als geeignet erschienen. Diese sind in nachfolgender Übersicht, welche jedem Tool seinen Anbieter sowie die zu erwartenden Kosten gegenüberstellt, zusammengefasst:

Tool	Anbieter	Anfallende Kosten
QF-Test	Quality First Software GmbH	Kostenpflichtig
Ranorex	Ranorex GmbH	Kostenpflichtig
GUIDancer	Bredex GmbH	Kostenpflichtig / Teile OpenSource
Sikuli	Forschungsprojekt	Freeware / OpenSource
Maveryx	Maveryx	Freeware / OpenSource

Tabelle 1: Übersicht der untersuchten Test-Tools

Quelle: von der Autorin erstellt

3 Analyse der ausgewählten Testautomatisierungstools

3.1 Kostenpflichtige Softwarelösungen

Zunächst wurden die ausgewählten Werkzeuge in kostenpflichtige, sowie kostenfreie Softwarelösungen untergliedert. Zu den Testtools, deren Hersteller Lizenzgebühren verlangen, zählen QF-Test und Ranorex. Die Test-Software GUIDancer ist als Mischform zu betrachten. Dessen Hersteller verlangt zwar eine Gebühr für die Nutzung; diese ist jedoch eher als Support-/bzw. Wartungsgebühr anzusehen. Alle Tools wurden auf einer Virtuellen Maschine, auf der das Betriebssystem Windows 7 installiert war, getestet.

3.1.1 QF-Test

3.1.1.1 Allgemeine Informationen

QF-Test ist die erste Lösung zum Testen einer grafischen Benutzeroberfläche, die im Rahmen der vorliegenden Seminararbeit untersucht wurde. Getestet werden können hiermit sowohl Java- als auch Web-Anwendungen. Die offiziell unterstützten Plattformen sind in diesem Fall Windows und Linux, wobei Swing Anwendungen gemäß Herstellerangaben zusätzlich auf Mac OS-X und diversen anderen Betriebssystemen lauffähig sind.

Abbildung 2: Logo QF-Test

Quelle: http://www.qfs.de/de/qftest/index.html

Die Software wird seit 1999 von der Quality First Software GmbH entwickelt und seit 2001 aktiv vertrieben. Bereits über 600 Kunden nutzen gemäß Herstellerangaben die Software. Darunter sind auch große Unternehmen wie die AUDI AG oder die RTL Television GmbH sowie Institutionen der Forschung und Lehre, wie zum Beispiel die Technische Universität Dresden vertreten.[5]

QF-Test wird in verschiedenen Varianten angeboten. Es wird je nach unterstützter GUI-Technologie in Swing, SWT und Web unterschieden. SWT steht in dem Fall für „Standard Widget Toolkit." Gemeint ist damit die Bibliothek, die bei der Erstellung grafischer Benutzeroberflächen unter Java unterstützen soll.[6]

[5] Vgl. [QFT12a]
[6] Vgl. [QFT12b]

Das Lizenzmodell ist analog der oben genannten GUI-Technologien aufgebaut. Der Preis für eine GUI-Technologie beläuft sich aktuell auf 1.995 Euro exklusive Mehrwertsteuer. Für jede weitere Technologie verlangt der Hersteller zusätzlich 500 Euro netto. Es handelt sich bei allen Modellen um so genannte Floating-Lizenzen, was bedeutet, dass jeweils eine Software Instanz gleichzeitig betrieben werden kann. Es ist dabei unerheblich von welchem Rechner dies stattfindet oder welche Person auf das Tool zugreift.[7]

Nachdem die Autorin Kontakt zur Quality First Software GmbH aufgenommen hat, zeigte sich, dass es für den Einsatz von QF-Test im akademischen Bereich die Möglichkeit einer kostengünstigeren Lizenzierung gibt. Dies stellt immer eine Einzelfallentscheidung dar. Erste Angaben zu dem TimeNet-Projekt wurden an den Hersteller übermittelt, der anschließend einen Rabatt von bis zu 50 Prozent des Lizenzpreises einräumte. Damit würden die Kosten für das TimeNET-Projekt bei circa 1.000 Euro liegen. Konkrete Vertragsverhandlungen könnten im Nachgang dieser Arbeit durchgeführt werden.

3.1.1.2 Eigene Erfahrungen

Im Gegensatz zu anderen getesteten Testtools konnte diese Software sofort und ohne Angabe persönlicher Daten heruntergeladen werden. Anschließend stand die vollständige kommerzielle Version des Tools zum Testen zur Verfügung. Die einzige Beschränkung der Testversion im Vergleich zur Software mit Lizenzerwerb bestand darin, dass keine Testsuiten gespeichert und nur solche geladen werden konnten, die bereits vorinstalliert waren. Nach einer kurzen E-Mail an den Hersteller, die direkt aus dem Tool heraus erzeugt werden konnte, wurde der Verfasserin dieser Arbeit problemfrei eine erweiterte Evaluationslizenz ausgestellt.

Der Download gestaltete sich mit rund 160 MB schnell und problemlos. Ebenso verhielt es sich mit der Installation der Software.

Besonders positiv ist die Gestaltung der Website des Unternehmens hervorzuheben. Alle wichtigen Daten und Informationen sind schnell auffindbar und gut zusammengefasst. Der Download-Button ist leicht auffindbar und die Aktualität der Homepage ist vorbildlich. Ebenso ist ein ausführliches Nutzerhandbuch verfügbar und in der Zeit der Evaluierung wird ein kostenfreier Support gewährt. Schulungen in Form von Webinaren oder Vor-Ort Trainings werden ebenso angeboten.

[7] Vgl. [QFT12c]

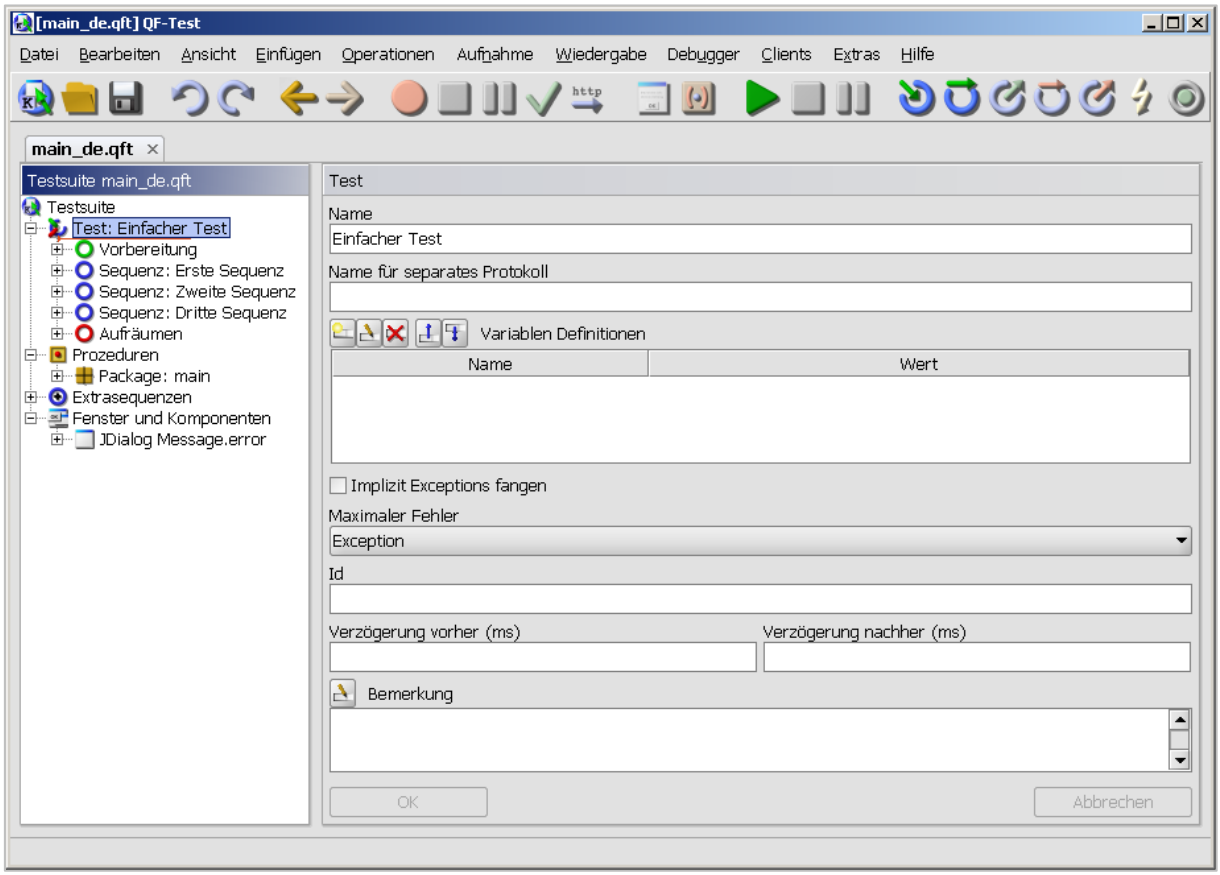

Abbildung 3: Screenshot QF-Test

Quelle: von der Autorin erstellt

Die Tests selber werden, wie in Abbildung 3 erkennbar in einer Baumstruktur abgelegt. Unter dem Knoten „Prozeduren“ können Testfälle abgelegt werden, die sich zur Wiederverwendung eignen und so Redundanzen vermeiden können. QF-Test bietet zudem eine Möglichkeit zum Aufzeichnen von Tests. Diese ist sehr intuitiv, da sie die üblichen Recorder-Funktionalitäten, wie „Play“, „Stop“ oder „Record“ nutzt.

Die Testsoftware erkennt automatisch die einzelnen Aktionen, wie z.B. Klick auf ein GUI-Element und legt diese separiert ab. Zum Überprüfen des Test-Ergebnisses können so genannte Check-Elemente in den Testfall integriert werden. Dazu bietet QF-Test bereits vorgefertigte Check-Elemente, wie zum Beispiel ob eine Grafik oder ein Text überprüft werden soll. Die Testfall-Erstellung lässt sich nach dem Aufzeichnen und dem Anlegen der Checks dann einfach über eine Drag & Drop-Funktionalität realisieren. Hat man den Test ausgeführt, wird in einer Statusleiste das Resultat angezeigt. Tritt ein Fehler auf, erscheint außerdem eine Fehlermeldung in Form eines Pop-Up-Fensters.

Zu jedem ausgeführten Test wird zudem ein Log-File angelegt, welches genauere Rückschlüsse über die einzelnen Testschritte und zum Beispiel deren Dauer zulässt. Ist ein Fehler aufgetreten, wird die betroffene Sequenz an dieser Stelle mit einem roten Rechteck gekennzeichnet. Außerdem werden automatisch Screenshots des Fensters, in dem der Fehler aufgetreten ist, erzeugt und im Log-File hinterlegt. Aus dieser Datei kann anschließend ein Protokoll in Form einer HTML oder XML Datei für weitergehende Analysen erzeugt werden.

Weitere Möglichkeiten, die QF-Test bietet sind zum einen die Erweiterung der Testskripte mithilfe von Jython oder Groovy. Zum anderen können die Tests auch mithilfe einer Batch-Datei automatisiert durchgeführt werden. Laut Hersteller gibt es noch weitere Features von QF-Test, die jedoch nicht alle im Rahmen dieser Hauptseminar-Arbeit getestet werden konnten.

3.1.2 Ranorex Automation Framework

3.1.2.1 Allgemeine Informationen

Ein weiteres Tool, mit dem sich GUI-basierende Funktionstests erstellen lassen, ist das Ranorex Automation Framework. Getestet werden können sowohl Desktop-, als auch Web-Applikationen sowie mobile Anwendungen. Ranorex unterstützt im Gegensatz zu den anderen getesteten Tools lediglich die Windows-Plattform, im Gegenzug können jedoch ganz verschiedene Anwendungen getestet werden. Ein Java-Plug-In erlaubt insbesondere das Testen von Java Swing und AWT Applikationen – und damit auch von TimeNET.[8]

Abbildung 4: Logo Ranorex

Quelle: http://www.ranorex.de/

Das Ranorex Automation Framework wird von der Ranorex GmbH, die zur EOSS Gruppe gehört, vertrieben. Zu den Kunden zählen große Unternehmen wie die Bayer AG, Roche oder Bosch, um nur einige zu nennen.[9]

Ranorex hat eine offene Architektur und basiert auf dem .NET-Framework. Zusätzlich zu einer Recorder-Funktion bietet Ranorex die Möglichkeit manuell Testfälle zu generieren oder aufgezeichnete Tests zu editieren. Die Testskriptsprachen sind C#, VB.NET und IronPython.

	Runtime	Professional	Premium
Gerätespezifische Lizenz	389 €	980 €	1480 €
Geräteunabhängige Lizenz	780 €	2080 €	3080 €

Tabelle 2: Kostenübersicht Ranorex

Bei den Lizenzen wird in Runtime, Professional und Premium unterschieden, wobei sich sowohl der Preis, als auch die Anzahl der verfügbaren Funktionen je nach Lizenz unterscheiden. Außerdem gibt es gerätespezifische und geräteunabhängige Lizenzen zum Erwerb, die ebenso im Preis variieren. In Tabelle 2: Kostenübersicht Ranorex“ sind die Kosten für die verschiedenen Lizenzen in einer Übersicht zusammengefasst. Alle Preise verstehen sich ohne Mehrwertsteuer.[10]

[8] Vgl. [Ran12a]
[9] Vgl. [Ran12b]
[10] Vgl. [Ran12e]

3.1.2.2 Eigene Erfahrungen

Für Ranorex steht eine kostenlose 30-tägige Demoversion der Ranorex Premium Lizenz zur Verfügung, die jedoch im Gegensatz zu QF-Test nicht anonymisiert bezogen werden konnte. Der Download ist mit circa 93 MB vergleichsweise klein und somit schnell durchgeführt. Die Installation von Ranorex verlief problemfrei. Der Kundenservice der Ranorex GmbH ist besonders hervorzuheben. Bereits nach dem Download erfolgte eine erste unaufdringliche Kontaktaufnahme per E-Mail, bei der Hilfe während der Evaluierung angeboten wurde. Eine Verlängerung der 30-Tage-Testlizenz war ebenso kein Problem.

Die Dokumentation von Ranorex ist vorbildlich. Darüber hinaus wird ein sehr aktives Forum gepflegt, indem spezielle Fragen geklärt werden können.

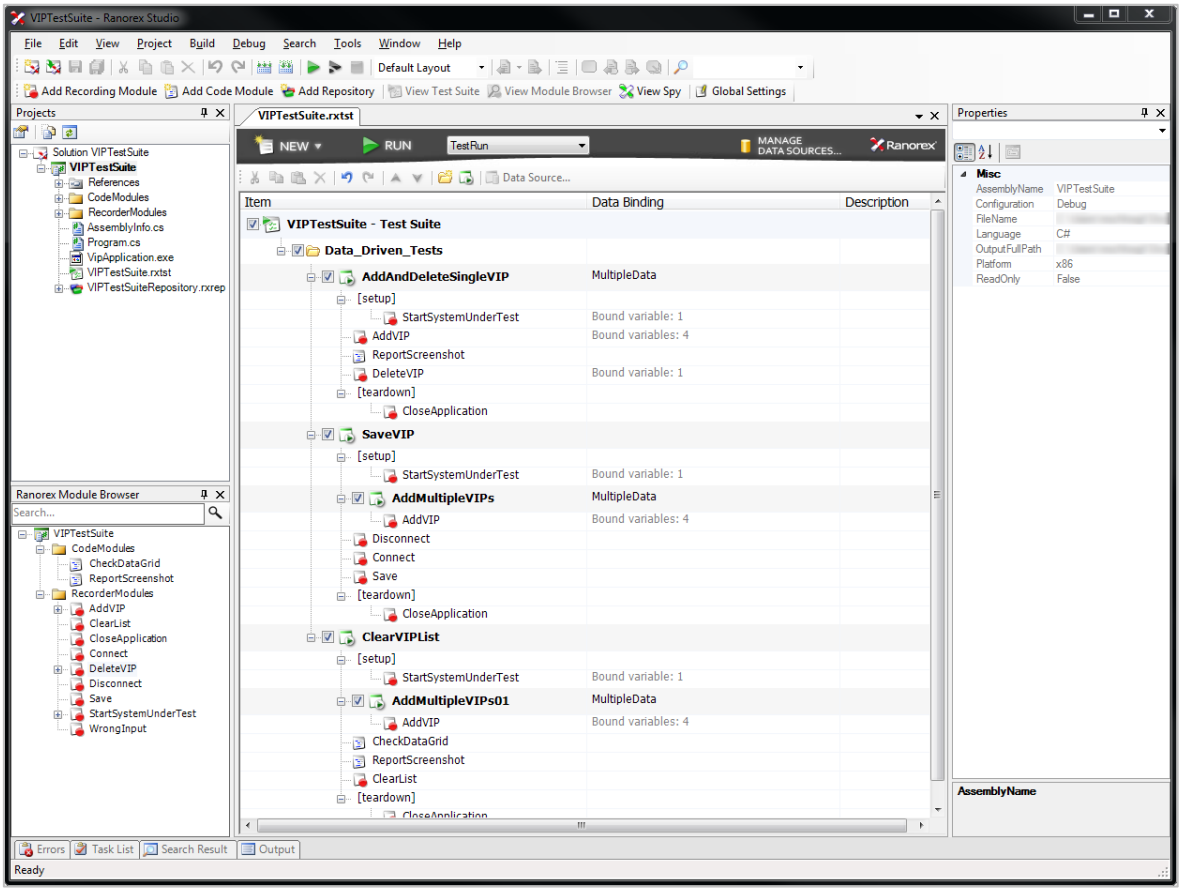

Abbildung 5: Screenshot Ranorex

Quelle: http://en.wikipedia.org/wiki/File:Ranorex_screenshot.png

Ranorex besteht aus vier Hauptfunktionen, die alle bei der Auslieferung enthalten sind: Ranorex Studio, Ranorex Spy, Ranorex Recorder und Ranorex Library API. Allgemein können mit Ranorex Studio die Tests generiert werden. Ranorex Spy sucht automatisch nach User Interface Elementen in der zu testenden Software.

Mithilfe der Aufnahmefunktion Ranorex Recorder können auch von Programmier-Unerfahrenen Testfälle aufgezeichnet bzw. generiert werden. Allen genannten Funktionen liegt die Ranorex Library API zugrunde. Durch sie können den identifizierten UI-Elementen eindeutige Bezeichner zugewiesen werden. Wird später zum Beispiel die Reihenfolge der Buttons im zu testenden GUI verändert, muss nicht jeder Testfall einzeln angepasst werden. Eine Änderung muss aufgrund der Ranorex API nur an einer Stelle vorgenommen werden, was dieses Tool sehr flexibel macht.[11]

Eine Testsequenz kann, wie in Abbildung 5 zu erkennen ist, nach der Aufnahme in verschiedene Testteile untergliedert werden. Dies ermöglicht eine Wiederverwendung bestimmter Testfälle, sodass dadurch der Testaufwand verringert werden kann. Ein weiterer Vorteil dieses Testtools ist die automatische Ausführbarkeit der Tests. Sobald man den „Run"-Knopf in Ranorex betätigt, wird eine selbstlauffähige Datei erzeugt. Ein großer Pluspunkt dabei ist, dass diese exe-Datei auch auf Rechnern gestartet werden kann, auf denen Ranorex nicht installiert ist. Des Weiteren nutzt Ranorex Standards wie XML oder XPath. So werden während der Ausführung der Tests automatisch Berichte in Form von XML-Dateien erzeugt, wodurch die Zeit zum Auffinden von Fehlern deutlich verringert werden kann.[12]

Die Einarbeitung in das Tool war im Vergleich zu QF-Test nicht so einfach. Das User Interface wirkt zunächst mit Funktionen überladen, was dazu führt, dass die Bedienung nicht intuitiv ist. Dies ist ein großer Nachteil von Ranorex. Außerdem ist dieses Tool nicht nur auf das Testen von Java-Anwendungen spezialisiert, was zur Folge hat, dass deutlich mehr Funktionen zur Verfügung stehen, als im TimeNET-Projekt benötigt werden. Es stellt sich die Frage, ob es nötig ist, die Kosten zu investieren oder andere günstigere Werkzeuge für diesen Anwendungsfall besser geeignet sind.

[11] Vgl. [Ran12c]
[12] Vgl. [Ran12d]

3.1.3 GUIDancer

3.1.3.1 Allgemeine Informationen

Das Tool GUIDancer basiert auf Eclipse und kann sowohl als eigenständige Applikation, als auch in Form eines Eclipse Plugins verwendet werden. Getestet werden können mit diesem Tool zum einen Java-, aber auch Webanwendungen. Lauffähig ist GUIDancer sowohl unter Windows, als auch unter Linux und Mac OS X.

Hergestellt und vertrieben wird GUIDancer von dem seit 1987 bestehenden und in Braunschweig ansässigen Unternehmen Bredex, das sich seit 1995 auf Java Programme spezialisiert hat. Laut Bredex unterscheidet sich dieses Tool von anderen GUI-Test-Tools vor allem dadurch, dass die Tests bereits während der Anforderungsspezifikation erstellt werden können. Aus diesem Grund eignet sich dieses Werkzeug gemäß Herstellerangaben besonders für test-driven development, der testgetriebenen Software-Entwicklung, bei der die Tests bereits vor Entwicklung der eigentlich zu testenden Softwarekomponenten erstellt werden. Es sind keine Programmierkenntnisse zum Erstellen der Tests erforderlich.[13]

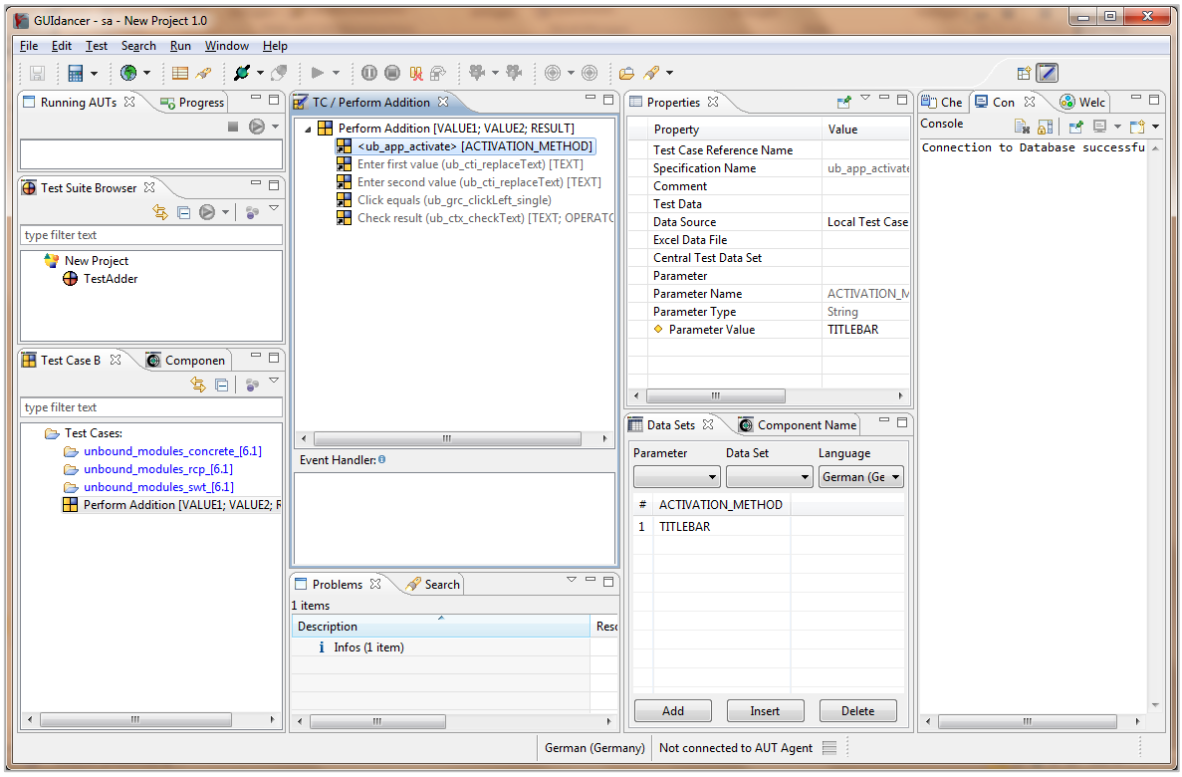

Abbildung 6: Screenshot GUIDancer

Quelle: von der Autorin erstellt

[13] Vgl. [Gui12a]

Zusätzlich zur eigentlichen Test-Funktionalität bringt GUIDancer Tools zur Durchführung von Testabdeckungsanalysen out-of-the-box mit. Dadurch können die Effektivität der Tests ebenso ermittelt werden, wie noch ungetestete Stellen in der Software. Darauf basierend können dann über verschiedene Zeiträume hinweg, bis hin zu Jahren, Testergebnisse verglichen werden. Damit ist eine besonders genaue Aussage über die Qualität der zu testenden Software möglich. Seit dem Jahr 2010 stellt Bredex große Teile seines Testautomatisierungswerkzeugs als Open-Source-Projekt zur Verfügung. [14]

3.1.3.2 Eigene Erfahrungen

GUIDancer konnte im Gegensatz zu QF-Test erst nach Angabe einiger persönlicher Daten heruntergeladen werden. Der Download ist mit 504MB im Vergleich zu den anderen getesteten Tools verhältnismäßig groß. Die Installation verlief auch hier problemlos. Beim Download werden ein Handbuch sowie einige Beispielprojekte mitgeliefert, welche zusammen mit einem sehr guten Tutorial den Einstieg in GUIDancer relativ einfach machen.

Viele vorgefertigte Testsets, die per Drag and Drop in neue Tests eingebunden werden können, erleichtern den Start ebenso. Grundsätzlich werden diese Komponenten nicht kopiert, sondern nur referenziert, was dieses Tool sehr wiederverwendbar und flexibel macht.

Tests bestehen bei GUIDancer aus drei Einzelkomponenten. Der GUI-Komponente an sich, also zum Beispiel ein Button oder ein Textfeld. Die zweite Komponente ist die Aktion, das heißt beispielsweise „Mausklick“ oder „Texteingabe“. Im letzten Schritt wird der konkrete Wert mitgegeben, also zum Beispiel „Testtext“ in einem Textfeld. Im Gegensatz zu einigen der anderen getesteten Tools wird bei GUIDancer nicht der Ansatz „Record and Replay“ verfolgt, was sicherstellen soll, dass die Tests wiederholbar sind und vor allem bereits vor Entwicklung der zu testenden Software erstellt werden können. Dennoch bleibt das Tool dabei bedienerfreundlich und es muss nicht programmiert werden.

Wie auch bei QF-Test werden im Fehlerfall automatisch Screenshots angefertigt und in einer Log-Datei gespeichert.

[14] Vgl. [Neu10]

3.2 Freeware

Im Zuge der Erstellung dieser Arbeit konnte festgestellt werden, dass einige kostenpflichtige Anbieter die in Kapitel 2.2.2 genannten Auswahlkriterien erfüllen. Da es jedoch an einer Universität wichtig ist, möglichst kostenneutral zu agieren, war die Bestrebung vorhanden, Anbieter zu finden, die ihre Test-Software kostenfrei zur Verfügung stellen. Zwei dieser Freeware-Tools, die nach der ersten Recherche zumindest teilweise die gestellten Anforderungen erfüllten, sind Sikuli und Maveryx, welche nachfolgend vorgestellt werden.

3.2.1 Sikuli

3.2.1.1 Allgemeine Informationen

Sikuli ist im Gegensatz zu den vorgenannten Softwarelösungen ein Open Source Projekt und wurde unter der MIT Lizenz veröffentlicht. Als plattformunabhängiges Tool entwickelt, ist Sikuli sowohl unter Windows, als auch unter Linux sowie Mac OS X lauffähig. Die Skriptsprache ist Jython, eine Python Implementierung in Java VM. Dadurch ist jedes Python Modul auch für die Testerstellung verwendbar.

Abbildung 7: Logo Sikuli
Quelle: http://www.sikuli.org/

Der große Vorteil im Vergleich zu den anderen getesteten Tools ist die intuitive Bedienbarkeit der Software. Weder Programmierkenntnisse noch tiefgehendes Wissen über die programmiertechnischen Hintergründe der zu testenden Software sind erforderlich. Tests können mit einfachen Befehlen wie z.B. „click“ erzeugt werden. Um festzulegen, auf welches GUI Element zum Beispiel geklickt werden soll, werden beim Erstellen des Testskripts Screenshots erzeugt. Dabei ist von Vorteil, dass das GUI Objekt nicht anhand seiner Position bestimmt wird. So werden auch Positionsänderungen der UI Elemente von Sikuli toleriert.

Außerdem gibt es im Gegensatz zu den anderen Tools keine Beschränkungen auf zum Beispiel Java- oder Web-Anwendungen, die testfähig sind. Theoretisch kann jede Anwendung, die unter den genannten Betriebssystemen läuft und über ein GUI verfügt, getestet werden. Sikuli-Tests können ebenso wie bei Ranorex in eine ausführbare Datei gewandelt werden. Dies ermöglicht eine Automatisierung der Tests.

3.2.1.2 Eigene Erfahrungen

Der Download von Sikuli verlief ohne Probleme. Im Gegensatz zu den vorgenannten Software-Testtools wurden keine persönlichen Daten abgefragt. Die Installation erfolgte schnell und problemlos. Lediglich zu beachten ist die Anforderung, dass Sikuli die Java Runtime Environment 6 voraussetzt.

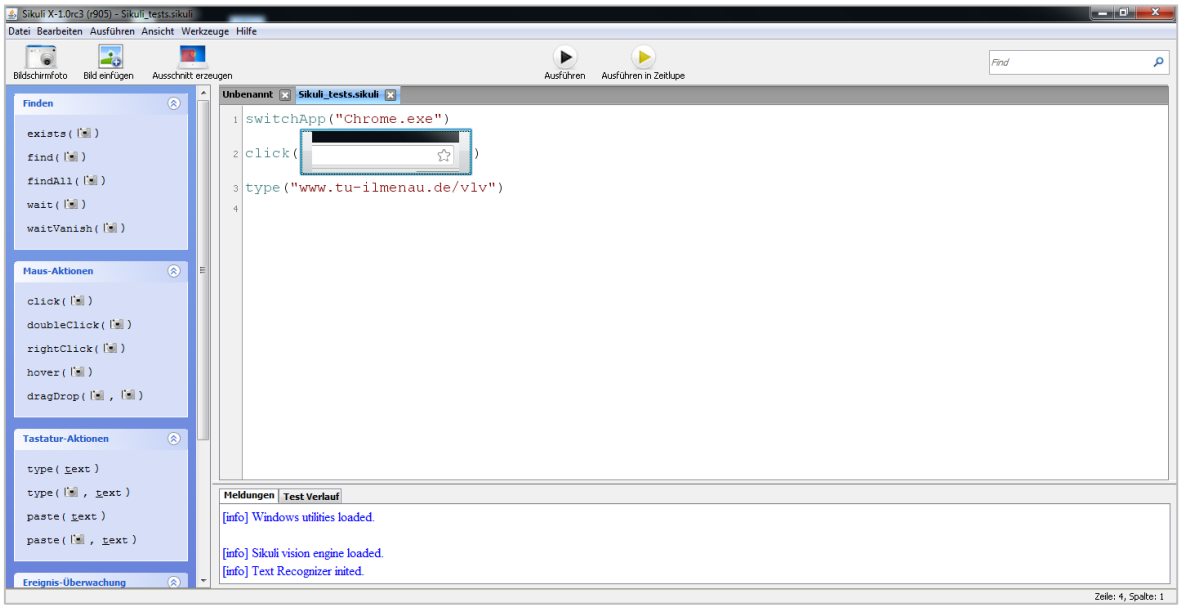

Abbildung 8: Screenshot Sikuli

Quelle: von der Autorin erstellt

Beim Erstellen von ersten Testfällen im Rahmen dieser Arbeit wurde noch einmal deutlich, dass die Bedienung der Software sehr intuitiv ist. In Abbildung 8 ist erkennbar, dass Sikuli mit einfachen Anweisungen und Screenshots arbeitet. Dennoch stellte sich im Zuge der Evaluierung heraus, dass Sikuli für den Anwendungszweck im TimeNET-Projekt eher ungeeignet erscheint. Die Software wurde nicht eigens als Testtool entwickelt, sodass es im direkten Vergleich mit den anderen getesteten Anwendungen einige Schwächen aufweist. So gibt es keine Möglichkeit, erwartete Testergebnisse festzulegen. Tritt in einem Programm bzw. einem Sikuli-Skript ein Fehler auf, so wird es lediglich an dieser Stelle abgebrochen. Eine genaue Auswertung, wie zum Beispiel bei QF-Test, ist nicht vorgesehen.

Für einen anderen Anwendungszweck, wie beispielsweise der Abarbeitung immer wiederkehrender Aufgaben, ist Sikuli jedoch sehr gut geeignet. Da dieses Tool in dieser Hinsicht unter Umständen die Arbeitsabläufe im TimeNET-Projekt auf anderer Ebene vereinfachen kann, empfiehlt die Autorin dennoch, diese Software zumindest einmal zu testen. Möglicherweise können einzelne Prozesse im Projekt auf diese Weise unterstützt werden.

3.2.2 Maveryx

3.2.2.1 Allgemeine Informationen

Maveryx ist das fünfte und letzte Werkzeug zur GUI-Testautomatisierung, welches im Rahmen der vorliegenden Hauptseminararbeit eingehender untersucht wurde. Auch dieses Tool erfüllt die meisten der in Abschnitt 2.2.2 genannten Kriterien. Es läuft zusätzlich zu den Plattformen Windows und Linux auch unter Mac OS. Im Gegensatz zu Sikuli wurde Maveryx speziell zum Testen von Java-Anwendungen entwickelt.

Abbildung 9: Logo Maveryx

Quelle: http://www.maveryx.com/

Die Software wird sowohl als Standalone Anwendung, als auch als Eclipse Plug-In angeboten. Außerdem ist Maveryx als OpenSource verfügbar. Es wurde unter der GNU GPL v2 Lizenz veröffentlicht. Die gesamte Software ist kostenfrei erhältlich. Zusätzlich bietet die Firma Maveryx verschiedene Support Pakete zum Erwerb an. Der Erwerb des Basic- oder Business-Pakets hat den Vorteil, dass anschließend eine erweiterte Dokumentation zur Verfügung steht. Außerdem werden die Service Packs bei einem Software Update geliefert und man hat die Möglichkeit den Support zu kontaktieren. Das Business-Paket unterscheidet sich lediglich in der Häufigkeit der erlaubten Kontaktaufnahme zum Kundensupport von Maveryx und ist vor allem dann zu empfehlen, wenn man zeitkritische Projekte bearbeitet. [15] Für TimeNET ist dies nach Ansicht der Autorin nicht notwendig.

In der nachfolgenden Tabelle sind die Kosten für die Supportpakete gegenübergestellt:[16]

	Free	Basic	Business
Kosten	0 €	499 €	999 €

Tabelle 3: Kostenübersicht Support-Pakete Maveryx

Die Skriptsprache der Tests ist bei Maveryx Java. Das Tool kann in jedes Java IDE, wie zum Beispiel Eclipse oder NetBeans sowie in diverse Test-Frameworks, wie etwa JUnit integriert werden. Durch die Vergabe von Schlüsselworten können Testfälle leichter wieder gefunden und damit auch wieder verwendet werden.[17]

[15] Vgl. [Mav12c]
[16] Vgl. ebenda
[17] Vgl. [Mav12b]

3.2.2.2 Eigene Erfahrungen

Der Download ist mit 15,7MB sehr klein und schnell abgeschlossen. Außerdem war keine Registrierung oder eine Angabe persönlicher Daten vonnöten. Die Installation verlief anschließend ebenso ohne Probleme und Maveryx stand vollumfänglich zur Verfügung.

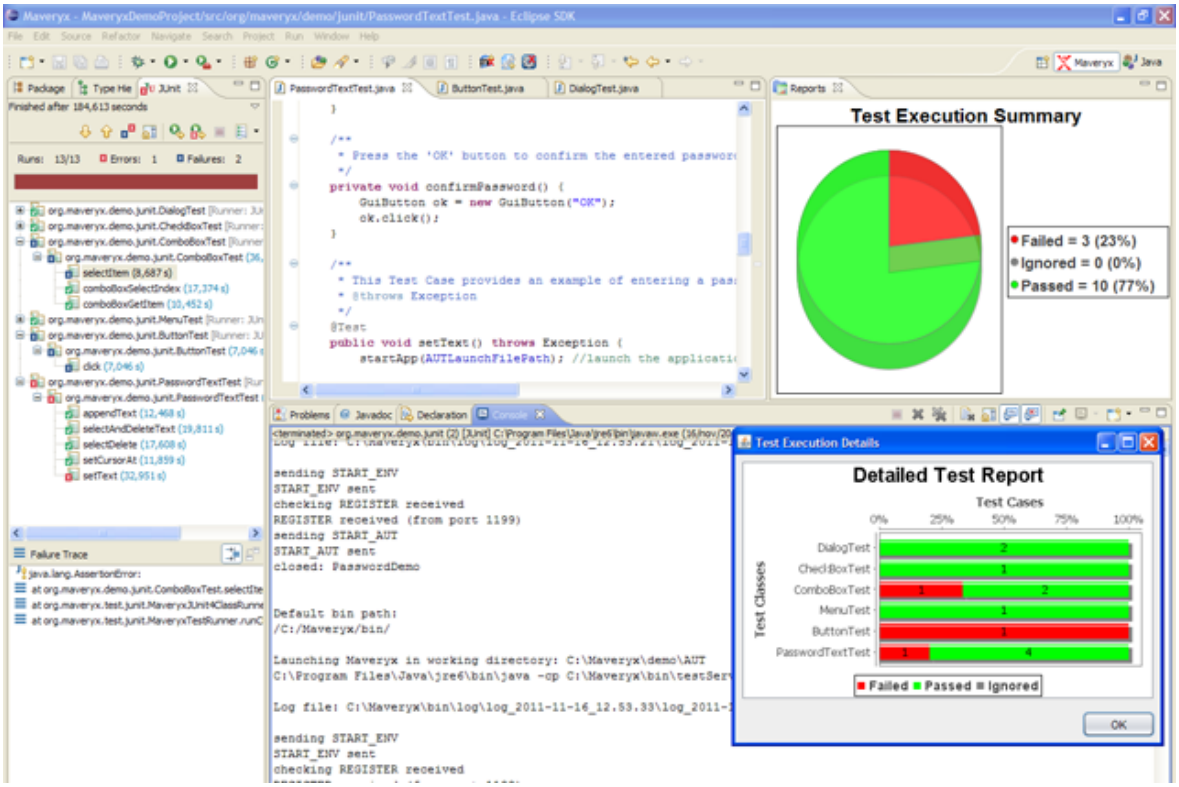

Abbildung 10: Screenshot Maveryx

Quelle: http://freecode.com/

Das Besondere an Maveryx ist, dass für die Erstellung von Tests keine GUI-Karte benötigt wird. Es ist also nicht zwingend erforderlich die genaue Lage von GUI Elementen vorher zu definieren und dem Tool zu übergeben. Maveryx scannt das zu testende GUI, macht davon Snapshots und findet durch die spezielle Funktionalität der Test-Software automatisch die verschiedenen GUI Objekte. Das hat den Vorteil, dass Tests bereits im frühen Software-Lifecycle parallel zur eigentlichen Software-Entwicklung angelegt werden können.

Die zu testende Software muss also nicht bereits vollständig programmiert sein.[18] Nach Ansicht der Autorin ist dies besonders für das TimeNET-Projekt von Vorteil, da die künftigen Entwickler noch während der Programmierung die Testfälle mit anlegen können und keine Nacharbeit erforderlich wird.

[18] Vgl, [Mav12a]

4 Fazit

4.1 Vergleich der Tools

Die Erfahrungen beim Testen der im vorangegangen Kapitel beschriebenen Testtools haben gezeigt, dass es Unterschiede in der Eignung für das TimeNET-Projekt gibt. Generell für den Anwendungszweck „Testautomatisierung der TimeNET-GUI“ ungeeignet ist das Tool Sikuli. Aus diesem Grund wird es auch in der anschließenden Gegenüberstellung der einzelnen Test-Werkzeuge aus der Betrachtung ausgeschlossen.

In nachfolgender Übersicht sind die einzelnen Vor- und Nachteile der verbliebenen Tools gegenübergestellt. Für jedes Kriterium wurden verschiedene Punkte (hier durch Emoticons dargestellt) anhand der gesammelten Erfahrungen vergeben.

Merkmal	QF-Test	Ranorex	GUIDancer	Maveryx
Kosten	☹	☹☹	☺☺	☺☺☺
Unterstützte Plattform(en)	☺☺☺	☹	☺☺☺	☺☺☺
Testfähige Anwendungen	☺☺☺	☺☺☺	☺☺☺	☺
Auf Java spezialisiert	☺☺☺	☹	☺☺☺	☺☺☺
Log-Files und Analysemöglichkeiten	☺☺☺	☺☺☺	☺☺☺	☺
Automatische Ausführbarkeit	☺☺☺	☺☺☺	☺	☺
Bedienbarkeit / Einarbeitungszeit	☺☺	☹	☺☺	☺

Tabelle 4: Gegenüberstellung der Test-Tools

Quelle: von der Autorin erstellt

Zusammengefasst sind die wichtigsten Kriterien, wie zum Beispiel die anfallenden Kosten, welche Betriebssysteme unterstützt werden oder ob das Tool speziell zum Testen von Java-Anwendungen entwickelt wurde. Im Gegensatz zu den ersten sechs Kriterien ist beim letzten Kriterium der Einarbeitungszeit zu beachten, dass diese Bewertung subjektiver Natur ist. Sie beruht auf den Erfahrungen der Autorin bei der Einarbeitung in die jeweiligen Tools. Gemäß dieser Übersicht überwiegen die Vorteile von QF-Test sowie GUIDancer.

4.2 Ausblick und Fazit

Wie in Abschnitt 4.1 deutlich wurde, gibt es zwei Tools, die für den Einsatz als GUI-Testautomatisierungs-Werkzeug geeignet erscheinen und somit zur Nutzung empfohlen werden können. Dies ist zum einen das kostenpflichtige Tool QF-Test, welches jedoch einen nicht unerheblichen Preisnachlass für akademische Projekte gewährt. Zum anderen würde sich auch das Werkzeug GUIDancer anbieten. Beide Tools laufen sowohl unter Windows, als auch unter Linux sowie zusätzlich unter Mac OS und sind auf das Testen von Java-Anwendungen spezialisiert. Die Entscheidung für eines der beiden Tools ist nach Auffassung der Autorin in erster Linie von den Kosten abhängig. QF-Test ist insgesamt professioneller und insbesondere die umfangreichen Support-Möglichkeiten sprechen für die Anwendung dieses Tools. Dem gegenüber stehen jedoch die Kosten und es ist abzuwägen, ob diese für die Universität tragbar sind. Sind die monetären Aufwände das ausschlaggebende Kriterium, sollte man sich für das kostenfreie GUIDancer entscheiden.

Die Verfasserin dieser Seminararbeit empfiehlt, dass sowohl QF-Test, als auch GUIDancer eingehender evaluiert werden sollten. Im Zuge dieser Arbeit war es nicht möglich, direkt am TimeNET-Projekt zu testen, da dies den Zeitrahmen der Bearbeitung deutlich überschritten hätte. Dies könnte in einer weiterführenden wissenschaftlichen Arbeit durchgeführt werden. Nachdem eine Entscheidung für eines der untersuchten Tools getroffen wurde, wäre es anschließend denkbar, die ersten Testfälle für das TimeNET-Projekt zu erarbeiten. Außerdem würde es Sinn machen, eine Dokumentation zur Anlage von Testfällen für TimeNET zu erstellen.

Zunächst lag die Anforderung dieser Seminararbeit bei der Neuerstellung eines Testtools, was sich jedoch als nicht nützlich erwies, da bereits eine ausreichende Zahl derartiger Werkzeuge auf dem Markt verfügbar ist. So ergaben sich neue Anforderungen mit Aufwänden, die zu Beginn der vorliegenden Arbeit nicht abzusehen waren. Dennoch konnte nach der Evaluation der diversen Testtools ein Überblick erarbeitet und damit eine Handlungsempfehlung für das TimeNET-Projekt formuliert werden. Aus Sicht der Autorin sollte unabhängig von der letztlich getroffenen Entscheidung das Thema Testautomatisierung in jedem Fall weiterhin im Fokus der Projektleitung bleiben. Softwaretests tragen deutlich zu einer besseren Softwarequalität bei und sichern diese langfristig. Wie Ludwig van Beethoven bereits feststellte:

„Sich selbst darf man nicht für so göttlich halten, dass man seine eigenen Werke nicht gelegentlich verbessern könnte.“

Ludwig van Beethoven (1770 – 1827)

Literaturverzeichnis

[Gui12a]
http://www.bredex.de/web/index.php/guidancer_jubula_de.html
Abruf: 2012-11-20

[ITW12a]
http://www.itwissen.info/definition/lexikon/graphical-user-interface-GUI-Grafische-Benutzeroberflaeche.html
Abruf: 2012-10-15

[ITW12b]
http://www.itwissen.info/definition/lexikon/command-line-interface-CLI-Befehlsorientierte-Benutzeroberflaeche.html
Abruf: 2012-10-17

[Mav12a]
http://www.maveryx.com/
Abruf: Abruf: 2012-10-15

[Mav12b]
http://www.maveryx.com/en/support/faq.html
Abruf: 2012-11-04

[Mav12c]
http://www.maveryx.com/en/services/professional-support/subscription.html
Abruf: 2012-11-04

[Mye01]
Myers, Glenford J.: „Methodisches Testen von Programmen“, 7.Auflage, Oldenbourg Verlag, 2001

[Neu10]
Neumann, Alexander: „Test-Werkzeug GUIdancer wird Eclipse-Projekt“, 29.11.2010
http://www.heise.de/developer/meldung/Test-Werkzeug-GUIdancer-wird-Eclipse-Projekt-1143640.html
Abruf: 2012-10-27

[QFT12a]
http://www.qfs.de/de/qftest/index.html
Abruf: 2012-11-05

[QFT12b]
http://www.qfs.de/de/qftest/checklist.html
Abruf: 2012-11-05

[QFT12c]
http://www.qfs.de/de/qftest/product.html
Abruf: 2012-11-05

[Ran12a]
http://www.ranorex.de/support/user-guide-20/technology-instrumentation/testing-of-java-applications.html
Abruf: 2012-11-11

[Ran12b]
http://www.ranorex.de/about-us/company.html
Abruf: 2012-11-12

[Ran12c]
http://www.ranorex.de/produkte/tools.html
Abruf: 2012-11-11

[Ran12d]
http://www.ranorex.de/support/user-guide-20/frequently-asked-questions.html
Abruf: 2012-11-13

[Ran12e]
http://www.ranorex.com/store/floating/
Abruf: 2012-11-11

[TUI12]
http://www.tu-ilmenau.de/sse/timenet/general-information/tool-description/
Abruf: 2012-10-15