

המחלקה להנדסת תוכנה

שם הפרויקט: סוכן רשת אוטומטי

Project Name: Automatic Web Agent

ספר הפרויקט

|  |  |
| --- | --- |
| שם הסטודנט: | קובי הרשקוביץ |
| מספר תעודת זהות: | 25632654 |
| שם המנחה: | אלעד הוגן |
| תאריך ההגשה: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
| חתימת המנחה: | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |

# Contents

[1. Contents 2](#_Toc262600975)

[2. Table of Figures 6](#_Toc262600976)

[3. Introduction 6](#_Toc262600977)

[3.1. Executive Summary 6](#_Toc262600978)

[3.2. Definitions, acronyms and abbreviations 9](#_Toc262600979)

[3.3. Prologue 10](#_Toc262600980)

[3.4. Project goals and targets 11](#_Toc262600981)

[3.5. Literary Review 12](#_Toc262600982)

[3.6. Current state including dispositional alternatives analysis 14](#_Toc262600983)

[4. Software Requirements 16](#_Toc262600984)

[5. Software Specifications 17](#_Toc262600985)

[5.1. System Model 17](#_Toc262600986)

[5.2. Functional Specifications 18](#_Toc262600987)

[5.3. Major Performance Requirements 19](#_Toc262600988)

[5.4. Environmental Constraints 19](#_Toc262600989)

[5.5. Technology Alternatives Analysis 20](#_Toc262600990)

[6. Software Design 20](#_Toc262600991)

[6.1. System Architecture 20](#_Toc262600992)

[6.2. Detailed Design 21](#_Toc262600993)

[6.3. System Design Alternatives 21](#_Toc262600994)

[6.3.1. Browser Interfacing 21](#_Toc262600995)

[6.3.2. Script writing 23](#_Toc262600996)

[6.3.3. Data Base 24](#_Toc262600997)

[6.3.3.1. MS SQL Compact 3.5 24](#_Toc262600998)

[6.3.3.2. Sqlite 24](#_Toc262600999)

[6.3.4. Browser Type 24](#_Toc262601000)

[6.3.4.1. IE (Standalone) 24](#_Toc262601001)

[6.3.4.2. Firefox (Standalone) 24](#_Toc262601002)

[6.3.4.3. Embedded IE 24](#_Toc262601003)

[6.4. System Drawing 24](#_Toc262601004)

[7. Project Planning 25](#_Toc262601005)

[7.1. Risk Management 25](#_Toc262601006)

[7.2. Work Plan 26](#_Toc262601007)

[8. Software Testing and Evaluation 27](#_Toc262601008)

[8.1. Software Test Plan 27](#_Toc262601009)

[8.2. Software Test Report 27](#_Toc262601010)

[8.3. End to End operating examples 27](#_Toc262601011)

[8.3.1. Login 27](#_Toc262601012)

[8.3.2. Website Mapping 28](#_Toc262601013)

[8.3.3. Script composition 30](#_Toc262601014)

[8.3.4. Script scheduling 33](#_Toc262601015)

[8.3.5. Script activation 33](#_Toc262601016)

[8.4. Efficiency analysis 34](#_Toc262601017)

[8.5. System Performance 34](#_Toc262601018)

[8.6. System Cost 34](#_Toc262601019)

[8.7. System reliability 34](#_Toc262601020)

[8.8. System integrity 34](#_Toc262601021)

[8.9. Originality and uniqueness 34](#_Toc262601022)

[8.10. Data Security 34](#_Toc262601023)

[9. Derivatives 34](#_Toc262601024)

[10. Epilogue 34](#_Toc262601025)

[10.1. Summary and Conclusions 35](#_Toc262601026)

[10.2. Future Work and Development 35](#_Toc262601027)

[10.3. Works cited 35](#_Toc262601028)

[11. Appendix 36](#_Toc262601029)

[11.1. SRD (Software Requirements Document) 36](#_Toc262601030)

[11.2. SDD (Software Design Description) 49](#_Toc262601031)

[11.3. STD (Software Test Documentation) 65](#_Toc262601118)

[11.4. STR (Software Test Report) 67](#_Toc262601119)

[11.5. SPMP (Software Project Management Plan) 68](#_Toc262601120)

[11.6. Project Proposal 69](#_Toc262601126)

# Table of Figures

[Drawing ‎3‑1 : Context 4](#_Toc262601642)

[illustration ‎5‑1: System Model 15](file:///C:\Documents%20and%20Settings\User-1\My%20Documents\לימודים\AutoWebAgentProject2009\AutoWebAgent\docs\סוכן%20רשת%20אוטומטי%20-%20ספר%20הפרוייקט.docx#_Toc262601643)

[Drawing ‎5‑2: ERD 16](#_Toc262601644)

[Drawing ‎5‑3: DFD-0 16](#_Toc262601645)

[Drawing ‎6‑1: System Architecture 19](file:///C:\Documents%20and%20Settings\User-1\My%20Documents\לימודים\AutoWebAgentProject2009\AutoWebAgent\docs\סוכן%20רשת%20אוטומטי%20-%20ספר%20הפרוייקט.docx#_Toc262601629)

[Drawing ‎6‑2: IE Components 20](#_Toc262601630)

[Drawing ‎8‑1:Login Screen 25](#_Toc262601631)

[Drawing ‎8‑2:Navigate to target website 26](#_Toc262601632)

[Drawing ‎8‑3:mapping an element 27](#_Toc262601633)

[Drawing ‎8‑4:map first entry url 28](#_Toc262601634)

[Drawing ‎8‑5:adding new script 28](#_Toc262601635)

[Drawing ‎8‑6:adding steps to script 29](#_Toc262601636)

[Drawing ‎8‑7:step 1 config 30](#_Toc262601637)

[Drawing ‎8‑8:step 2 config 30](#_Toc262601638)

[Drawing ‎8‑9:script scheduling 31](#_Toc262601639)

[Drawing ‎8‑10:script activation 32](#_Toc262601640)

# Table of Tables

[Table ‎3‑1 7](#_Toc262602175)

[Table ‎3‑2 13](#_Toc262602176)

[Table ‎3‑3 13](#_Toc262602177)

[Table ‎5‑1 18](#_Toc262602178)

[Table ‎6‑1:Trident 20](#_Toc262602179)

[Table ‎6‑2:WatiN 21](#_Toc262602180)

[Table ‎6‑3:Standard Scripting 21](#_Toc262602181)

[Table ‎6‑4:Specialized Scripting 21](#_Toc262602182)

[Table ‎6‑5:GUI Scripting 22](#_Toc262602183)

[Table ‎6‑6:SDF 22](#_Toc262602184)

[Table ‎6‑7:Sqlite 22](#_Toc262602185)

[Table ‎7‑1:Risk Management 23](#_Toc262602186)

[Table ‎11‑1 35](#_Toc262602187)

[Table ‎11‑2 38](#_Toc262602188)

[Table ‎11‑3 48](#_Toc262602189)

[Table ‎11‑4:STR 66](#_Toc262602190)

# Introduction

## Executive Summary

WWW

Auto Web Agent application

HTTP Server

Drawing ‑ : Context

Nowadays web use has become common to practically every household. The web enables to perform many diverse tasks, ranging from acquiring entertainment data (movies, songs, gossip etc…) to financial transactions.

Most of the tasks a user might perform require the user to authenticate his identity, track data on the website and perform actions based on knowledge acquired on other websites. These actions are very similar in most of the tasks regardless of their nature, all of which require the attention of the user and might be extremely time consuming.

The Automatic Web Agent application project was devised to relieve the average/advanced user from these routine and time consuming tasks.

The Automatic Web Agent will enable the user to compose a script of actions to be taken on specific websites according to data sampled on other sites and conditions that the users will set, and determine the scheduling of the script execution.

For example, there can be a script that will sample the users favorite ski website every 4 hours and will monitor the snow depth of the sky track. Once a preset depth is detected the script will extract the prices of plane tickets to the ski track and email them to the user.

Another example can be a script that will extend the lending period of a book in the Afeka college library, which is not equipped today with a built-in agent, once every 3 days.

Alternatively, an example from the business sector could be a for a company which sells products via the web, a script can track the competitors web sites for price changes and update the prices of the products accordingly.

The uniqueness of this project is that contrary to web agent that resides in a certain website server, and is controlled by the website administrator, this project will offer an agent that will be programmed at the user side and at the server side, which will enable maximum control in achieving its designated goals.

The requirements for the project were produced by analyzing the elements of several leading websites in the academic, financial and social network fields and researching the most common action that are performed in them.

The main product of the project is a windows application which is built around an IE (Internet Explorer) browser component, through which a website is being analyzed for its elements, and its DOM (Document Object Model) is being manipulated by a specialized scripting engine to achieve automatic user interaction.

The GUI (Graphical User Interface) simplifies and abstract many complex operations of website analysis and scripting.

**תקציר מנהלים**

השימוש באינטרנט נעשה רווח כמעט בכל בית ובאמצעותו ניתן לבצע משימות רבות ומגוונות החל מקבלת מידע בידורי וכלה בפעולות בנקאיות. במרבית מן הפעולות נדרש המשתמש להזדהות, לבצע מעקב אחר נתונים באתר, ולבצע פעולות על סמך ידע שלמד מאתרים אחרים. פעולות אלו במרבית הזמן זהות/דומות בין האתרים וכמו כן ביצוען עלול לדרוש תשומת לב וזמן רב מן המשתמש.

הפרוייקט הינו סוכן רשת אוטומטי הבא להחליף מטלות שגרתיות וגוזלות זמן של משתמש דפדפן ממוצע/מתקדם. באמצעות המערכת המוצעת יוכל המשתמש לתכנן תסריט של פעולות מסויימות באתרים מסויימים בהתאם למידע הנדגם מאתרים אחרים ולקבוע את זמן ביצוע התסריט.

לדוגמא ניתן לקבוע תסריט שבו נדגם כל 4 שעות אתר סקי שהמשתמש מחבב במיוחד (לדוגמא www.champery.com) , התסריט מכוון לזהות את השדה של גובה השלג באתר וברגע שגובה השלג מגיע ל-110cm התסריט שולף מאתר ה-"דקה ה-90" את מחירי חבילות הנופש לאתר ושולח דוח מפורט עם הנתונים באי-מייל למשתמש.

דוגמא נוספת יכולה להיות תסריט שיאריך את זמן השאלת ספר באתר הספריה של מכללת אפקה ,שאינו מצוייד כיום בסוכן אוטומטי להארכת השאלות, פעם בשלושה ימים.

יחודיות הפרוייקט היא שבניגוד לסוכנים אוטומטיים הנמצאים באתר מסויים ,ונשלטים מתוך האתר ,הפרוייקט הנ"ל מציע סוכן שיתוכנת בצד הלקוח ולא בצד השרת, דבר אשר יאפשר לו נוחויות ושליטה מירבית להשגת מטרותיו.

הדרישות לפרוייקט הופקו ע"י ניתוח המרכיבים של מספר אתרים מובילים בתחום האקדמיה, הפיננסים והרשתות החברתיות ובדיקת הפעולות הנפוצות בהם.

התוצר המרכזי של הפרוייקט הוא אפליקציית Windows אשר בנויה סביב רכיב דפדפן IE (Internet Explorer) אשר באמצעותו מנתחים אתר למרכיביו וכן מבצעים מניפולציות על ה-DOM (Document Object Model) שלו באמצעות מנוע תסריטים ייעודי. הממשק הגרפי מפשט רבות את המורכבות של פעולת ניתוח האתר וכתיבת התסריטים.

## Definitions, acronyms and abbreviations

|  |  |
| --- | --- |
| DOM | Document Object Model |
| WWW | World Wide Web |
| DB | Data Base |
| Site Object | a C# Object that represent a specific element in the DOM of the website |
| HTML | Hyper Text Markup Language |
| DHTML | Dynamic HTML |
| URL | Universal Resource Link |
| HTTP | Hyper Text Transfer Protocol |
| SQL | Simple Query Language |
| API | Application Programming Interface |
| CSS | Cascading Style Sheet |
| AWA | Automatic Web Agent |
| MSVS | Microsoft Visual Studio |
| SDF | Smart device framework. DB format suitable for devices |

Table ‑

## Prologue

This Project was done as part of a B.sc. degree in software engineering, at Afeka, Tel-Aviv Academic College of Engineering.

The project is a windows application that automates monitors websites and enables scripting of various operations upon them.

The application is targeted for the average browser user, which needs to perform a lightweight, simple and no-programming knowledge required, operations automatically on a browser.

The project creator is Koby Hershkovitz, a student at Afeka College and currently a programmer at Avaya Inc.

This project advisor was Mr. Elad Hogen, Projects manager at Afeka, which accompanied this project at its various stages.

## Project goals and targets

**Goals:**

Many of the entertainment, Academic and financial operations are done using an world wide web browser for example:

* Site registrations
* Purchase and sale of stocks
* Assignment submissions.
* Course selection
* Form entry

Many of these tasks are repetitive and routine, and may tire the user, and/or cause him to make mistakes in data entry.

The project goal derived from these problems is building an application that will perform routine and deterministic tasks automatically, replacing the human user.

**Targets**:

The above goal can be divided into 3 main targets:

* Periodically monitor a website for specific pieces of data
* Automatically perform operations in a website (data entry and selection, mouse clicks etc.)
* Notification on triggered events (for example sending an email when the first google search result equals my companies name)

## Literary Review

Web agents are complex software systems that operate in the www, intranet, or LAN. Web agents are designed to perform a variety of tasks such as storing and routing data or even searching cataloging and filtering of data.

The literature describes a framework for designing and analyzing web agents, and web agent systems based on mathematical models of the environment they operate in. there are 3 central building blocks consisting a web agent:

* Stochastic models of a network graph that summarizes the distribution and connectivity of web pages and produces the basic guidelines in designing web agents, by extracting unique properties of the web agent environment.

The network can be seen as directed graphs in which its nodes are static HTML pages in its edges are links from one page to another. Article [‎18] offers and analyzes random graph models that were created from experimental web observations.

These models are different from the standard ones in the sense that edges that were chosen independently are not reflected in the observed web statistics, such that edges in the model are statistically dependant. Another difference is that nodes are added to the graph over time. This reflects the fact the web is changing over time.

* Statistical learning methods that enbles agents to learn their environment by creating stochastic models of a web page and the structure of local links. Article [‎17] describes the web as a world in which a multitude of links connect between web pages and web sites, such that a highly complex and unprecedented graph is created, which is the web graph. This article presents a new approach to study probabilistic models of the web that can be used in the process of linkage and data prediction of web pages. The method described in this article is probabilistic dimension reduction which unifies the Latent Semantic Analysis and kleinberg algorithm (Hubs-and-Authorities) in a probabilistic environment.

This is the first step in developing a probabilistic web data infrastructure. Although the article doesn't focus in a specific application, a variety of web applications can take advantage of the techniques presented, such as search engines, web crawlers and web agent systems.

* Autonomous planning and decision making algorithms in a web environment enables agents achieve their goals and adapt to a dynamic environment [‎16].

Take for example classic problems in the field of data base/data retrieval such as text search, data mining and categorization. The input to such problems is usually a collection of data/documents/ the web with its graph representation enables to improve the existing techniques with techniques from graph theory. This improvement, using graph based solutions, was demonstrated in the article for the following problems: subject search, subject numbering, categorization and crawling.

## Current state including dispositional alternatives analysis

Nowadays, when a user wants to monitor/enter data in a specific website, he can turn to the following options:

* The web site contains an integrated web agent that inform the user regarding a preset data in the website. For example ebay(<http://www.ebay.com>) offer a web agent that will bid for the user in a certain auction according to price parameters set by the user.
* Manual periodical monitoring of the user for changes in the web site (for example monitor the INTEL stock price at <http://uk.finance.yahoo.com/q?s=INTC>)
* Use of expensive automation software such as QFT or RFT:
  + QTP (Quick Test Pro) is an application created by mercury in which one can automate and test windows and web applications. For example writing an email and sending it, or scraping a web page and fill a form in it.
  + RFT (Rational Functional Tester) is an IBM software suite, which enables to perform manipulation over HTML and adobe FLEX elements, and also supports data entry over dumb terminals (for example VT102).

The price of each can reach several thousands of dollars for a single user license

* Use of open source tools:
  + Selenium Remote Control (RC):A test tool that allows you to write automated web application UI tests in any programming language against any HTTP website using any mainstream JavaScript-enabled browser.

In order to perform dispositional alternatives analysis the following criteria, with their corresponding significance weight, were selected as a comparison base:

|  |  |  |
| --- | --- | --- |
| **criteria** | **description** | **weight** |
| ease of use | 'Friendliness' of the interface, learning curve, required user skills. | 30% |
| functionality | Amount of features, supported elements and technologies (such as IFRAMES and FLEX), supported browsers, programming languages. | 15% |
| cost | how expensive is the solution in time and money | 20% |
| support | level of support given | 5% |
| security | user data security | 15% |
| reports | reporting facilities | 5% |
| allegiance to the user | how are the user interests being served | 10% |

Table ‑

The following table summarizes the dispositional alternatives analysis the ranking is 1-5 poor to best respectively:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **criteria product** | **integrated**  **agent** | **manual** | **QTP/RFT** | **Selenium** | **AWA****[[1]](#footnote-1)** |
| ease of use | 3 | 5 | 2 | 2 | 4 |
| functionality | 3 | 0 | 5 | 4 | 4 |
| cost | 5 | 3 | 2 | 5 | 5 |
| support | 3 | 1 | 5 | 3 | 2 |
| security | 2 | 1 | 5 | 2 | 4 |
| reports | 2 | 1 | 4 | 2 | 4 |
| allegiance to the user | 1 | 5 | 4 | 5 | 5 |
| Total | 3 | 2.85 | 3.35 | 3.25 | 4.2 |

Table ‑

We see from this analysis that AWA got the best score and thus it is the most cost effective.

# Software Requirements

* DOM element recording
  + allow a method for the user to populate the DB with mappings between DOM elements and Site Objects
* Site Objects management
  + allow user to change recognition parameters of a DOM element contained in a Site Object
* Script composition and validation
  + Allow the user to create the automation script and validate it.
* Script management
  + Allow the user to rename, create or delete scripts
* Script scheduling
  + Allow the user to schedule the execution time and frequency of the script.
* Script Execution
  + Actual execution of the actions in the scripts which include monitoring and manipulating websites and notifications to the user.
* Logging
  + Log each step of the script execution in an application log.

For complete software requirements description see SRD (Software Requirements Document)

# Software Specifications

## System Model

The following Drawing describes the main system components and the relationships between them. The Drawing lacks the UI component for clarity sake (All the components have a UI component except the persistency module).

illustration ‑: System Model

Data Access Layer

DB

Persistency Module

Embedded Browser

DOM Analysis

Mapping Control

Mapping Module

Script Builder

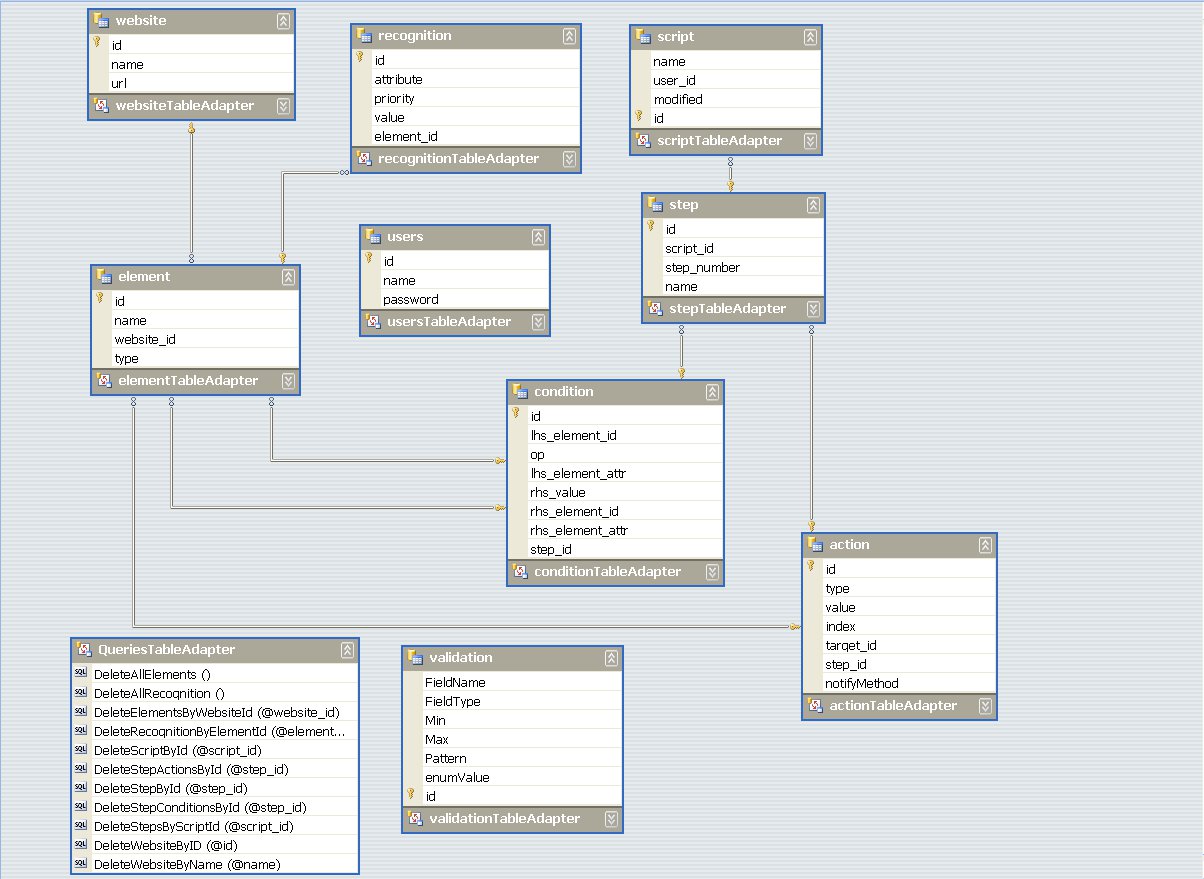
Script Scheduler

Script Logger

Script Runner

Scripting Module

The ERD has somewhat changed from the last report, following performance and refactoring, and here is the updated ERD:



Drawing ‎5‑2: ERD

## Functional Specifications

Following is the main user operations flow, described as a DFD. For a complete data flows drill down see SRD (2.7.1)



Drawing ‑: DFD-0

The following are the updated Functional requirements described in the SRD section 3.1:

* The system shall save all its data in a DB based on MS SQL SDF format. DB schema described by ERD Drawing 5-2. Script scheduling data will not be saved and determined at runtime.
* The system DB will arrive to the client preset with username and password.
* The system shall have a login form.
* The system shall have Browser control form via which the user will select the relevant elements
* The system shall have Map editing form where the user will tweak the table that will help in recognizing DOM elements. The table will map between a site object and DOM element attributes.
* The system shall have Script scheduling form where the user will determine the time and frequency of script execution.
* The system shall have Script execution status and log form.
* Element selection will be done using mouse hover over the element and pressing the middle mouse button for confirmation. When the mouse hovers over an element it shall be surrounded by a red rectangle.
* Script execution shall reuse the browser control form.
* Log of execution progress shall be displayed and written to a file.
* Workflow shall be implemented according to the DFD Drawings in section 2.7.1 of the SRD

## Major Performance Requirements

The performance requirements remained the same as in the SRD section 3.2:

* The system will support at least 3 simultaneous website interactions.
* Element interaction (click/text fill) shall be done in 1 second max.

Data base capacity is proportional to the amount of websites/scripts complexity it contains.

## Environmental Constraints

The Environmental Constraints remained the same, See section 2.4 in the SRD.

## Technology Alternatives Analysis

In order to implement the AWA system there is need in several different technologies:

* Programming language: criteria for selection would be language efficiency, ease of use, and learning curve.
* Development environment: criteria would be cost, ease of use, learning curve, support
* Persistency technology: criteria for selection would be cost, implementation complexity, data access efficiency
* DOM access technology (DOM library)

The following table summarizes the different alternatives; the grade given is between 1 to 10 when 10 is the best:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| item | Alternative 1 | | | | Alternative 2 | | | | Alternative 3 | | | |
| **Name** | **(+)** | **(-)** | **grade** | **Name** | **(+)** | **(-)** | **grade** | **Name** | **(+)** | **(-)** | **grade** |
| Coding language (25%) | C++ | Efficient, most libs are in c++ | complex | **7** | JAVA | portable, easy to write | not efficient, clumsy UI library | **6** | C# | best known to me, easy to write, rich libraries | windows only – not portable | **9** |
| Dev. Env.  (15%) | MSVS | professional, easy UI, reliable | cost | **9** | Eclipse | free, large user base | mostly for java | **8** | Code Blocks | free | mostly for CPP, small user base | **5** |
| DOM Library  (40%) | Native (mshtml.dll shdocvw.dll) | full DOM control | only for CPP, no dialog control | **8** | WatiN | in C#, supports dialogs | no full DOM control | **9** | selenium | java script based, simple, wide user base | hard to integrant with windows forms | **6** |
| Data Base  (20%) | MS SQL | proffesional, C# adaptor | complex, overkill | **8** | sqlite | easy, light, known, free | partial support in sql | **9** | SDF | free, large user base | local save | **9** |

Table ‑

From the table it obvious that the combination of C#, MSVS, WatiN and sqlite/SDF is optimal. Sqlite is poorly supported with MSVS so the final optimal solution is C#, MSVS, WatiN and SDF.

# Software Design

## System Architecture

The system architecture is composed of 3 main modules:

* Persistency Module
* Mapping module
* Scripting Module

The Persistency module contains the database, and the components to manipulate and query its data (Data Access Layer). The DB is responsible to persist element mappings and scripting information.

The Mapping Module is composed of UI in which the user can map web page elements to scriptable objects (also known as SiteObjects), which are saved in the DB.

The scripting module is composed of a UI in which the user can create a script to manipulate and monitor a website pages displayed in IE. This module is responsible to schedule the script run time and to notify the user of various events.

The following drawing illustrates the high level architecture which will be discussed in detail in section of the SDD.

GUI

Mapping Module

Scripting Module

Persisting Module

Drawing ‑: System Architecture

## Detailed Design

Some UI changes and enhancements were made. See section 2 of the SDD

## System Design Alternatives

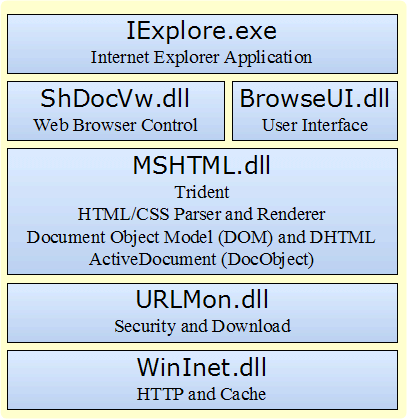
### Browser Interfacing

In order to control and monitor a browser programmatically, Several possibilities were explored.

There are several Browser APIs that enables access and control over the browser functions and data structures.

* + - 1. mshtml.dll module of IE

This module which is also called "Trident" [‎19], is at the core of Internet Explorer browser,as seen in Drawing ‎6‑2, and is responsible for manageing the analysis and rendering of HTML and CSS. The module exposes interfaces which allow control over the controls the page presents.



Drawing ‑: IE Components

This Module is implemented as COM Object which means it can be used as a reference in any .NET language.

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| know and tested interface | interface that is pretty low level, thus controlling an element is complex |
| fast | supported only in IE |

Table ‎6‑1:Trident

* + - 1. Open source library WatiN

WatiN[http://sourceforge.net/projects/watin] is an open source library that enables to automate IE utilizing .NET languages.

It allows running automatic tests of websites using browser interactions. It is written in C# but can be userd in any .NET language. The following is its main features:

* Automates all main HTML elements.
* Search elements by multiple properties
* AJAX support
* Supports IFRAMES and FRAMES (cross domain)
* Supports dialogs: alert, confirm, login, file.
* Supports HTML Dialogs.
* Implements Page and Control model.
* Supports screenshots
* Logging
* Integrates with unit test tools
* Supports IE 6,7,8/Firefox 2,3 browsers
* Open source (apache license 2.0)

|  |  |
| --- | --- |
| Advantages | Disadvantages |
| simplifies complex operations | open source – no reliability guaranteed |
| supports multiple browsers |  |
| added value as described in feature list |  |

Table ‑:WatiN

### Script writing

Running operations script, that the user composed, is a main part of the project. Several ways of script entry and analysis were explored, where the main criteria were usability and functionality:

* + - 1. Standard scripting language

Python is an object oriented scripting language' which has extensive capabilities' and is used as a scripting language in many application (such as XEN virtualization server, blender 3D). utilizing a module called IronPython [http://ironpython.net] one could enable python scripts to access .NET assemblies.

For example:

so=SiteObject('q')

if so.value == "expected value":

so.type("new value")

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| standard and known language | programming knowledge is required |
| complex scripts can be composed due to language richness | not secured enough (script can access any object that exists in the application) |

Table ‑:Standard Scripting

* + - 1. Specialized scripting language

The idea to create a specialized parser was explore, so as to fit it to the terms and object world of the application. The parser would receive a text string, analyze it and convert it into script objects.

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| specialized language that exactly fits the application needs | user must learn new language |
| complex scripts can be composed | parser implementation is very complex |
| secure: language will allow aces only to selected application objects |  |

Table ‑:Specialized Scripting

* + - 1. Graphical script composition

Interface forms through which script can be composed by selection an minimal text entry (see SDD).

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| no programming skills required | complexity of scripts is limited |
| relatively easy to implement |  |
| secure: the interface will allow access to controlled objects |  |
| the script is saved in the database |  |

Table ‑:GUI Scripting

### Data Base

### MS SQL Compact 3.5

MS SQL Server based database but with limited features and capacity.

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| visual studio supported | limited (file based so no stored procedures) |
| free |  |

Table ‑:SDF

### Sqlite

Open source, file based database.

|  |  |
| --- | --- |
| **Advantages** | **Disadvantages** |
| visual studio pro supports it with a plugin | limited (file based so no stored procedures) |
| open source |  |
| low memory footprint |  |

Table ‑:Sqlite

### Browser Type

### IE (Standalone)

### Firefox (Standalone)

### Embedded IE [[2]](#footnote-2)

## System Drawing

See illustration ‎5‑1: System Model

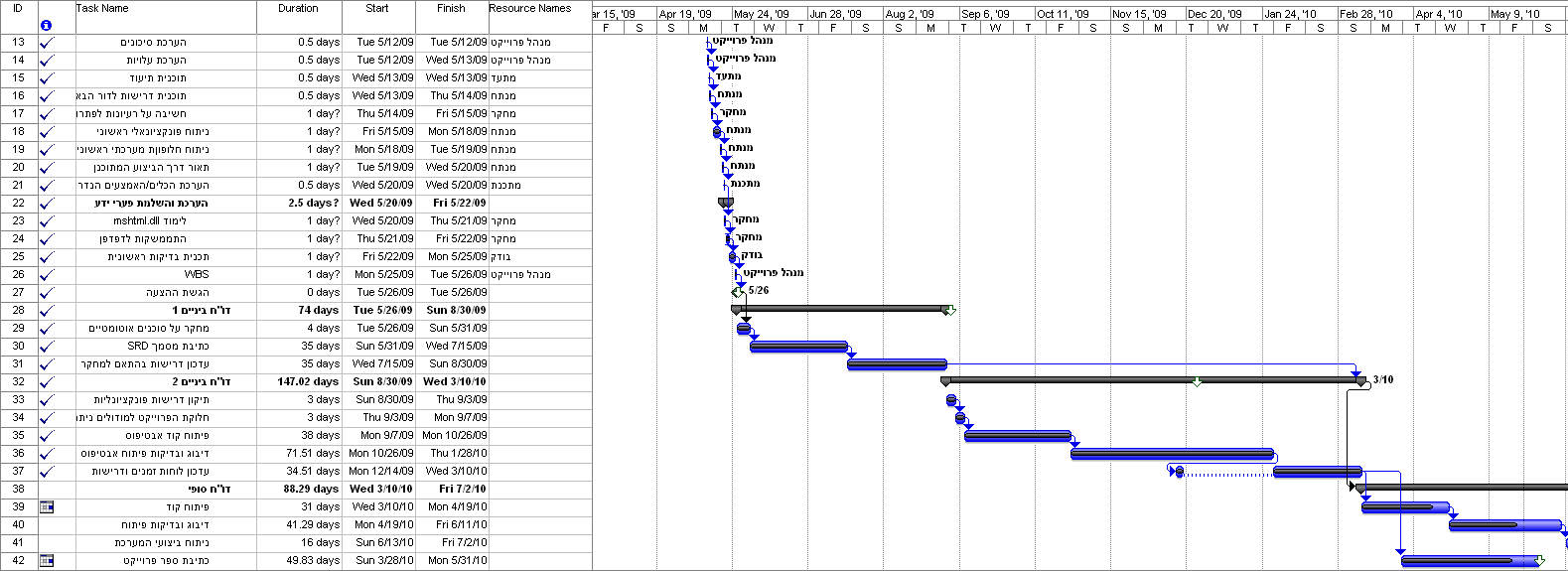
# Project Planning

## Risk Management

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Risk Category** | **Description** | **Risk Probability** | **Risk Effect** | **Solution** |
| Inadequate Initial Planning | wrong estimate of system size | Average (25%-50%) | missing project time tables (Serious) | complete planning before beginning of project |
| wrong project management | implementation phase before or during of planning and design | Low (10%-25%) | implementation and requirements misaligned, missing project time tables (catastrophic) | track project with well defined milestones |
| Technical factors | Development environment problems | Low (10%-25%) | missing project time tables (Serious) | learn development environment well, prepare to use alternatives |
| Technical factors | website technology change to flash/html5 | Low (10%-25%) | product incompatibility (catastrophic) | make a technical survey before selecting technology |
| human factor | developer personal issues | 50%-75% | missing project time tables (Serious) | NA |
|  |  |  |  |  |

Table ‎7‑1:Risk Management

## Work Plan



Drawing ‑:Ghant Chart

# Software Testing and Evaluation

## Software Test Plan

See STD (Software Test Documentation)

## Software Test Report

All completed features were tested. See STR in the appendix

## End to End operating examples

This example will show the system operation from application invocation to script execution and analysis.

The relevant application screenshots will be show as needed.

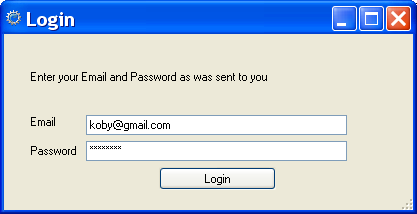
In this example we'll create a script that will search Google for Afeka College and notify us by popup if Afeka College website is in the first place.

This script will execute every day for a month.

### Login

After the user has launched the application only the login screen appears.

The user must enter his email(username) and password he received with the installation materials. (the applications DB is pre-initialized per user)



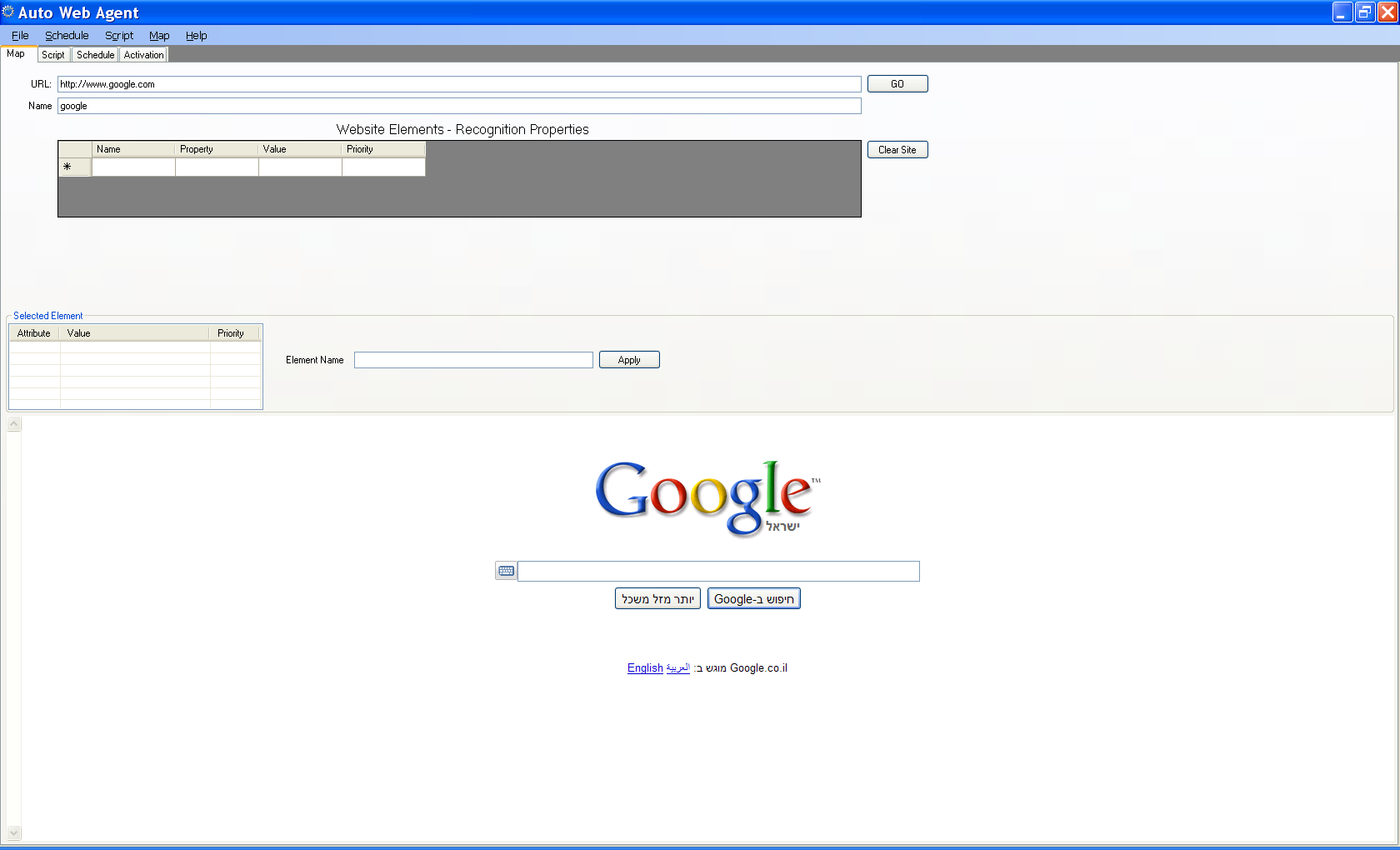
Drawing ‑:Login Screen

### Website Mapping

The main application form appears and the map tab is selected.

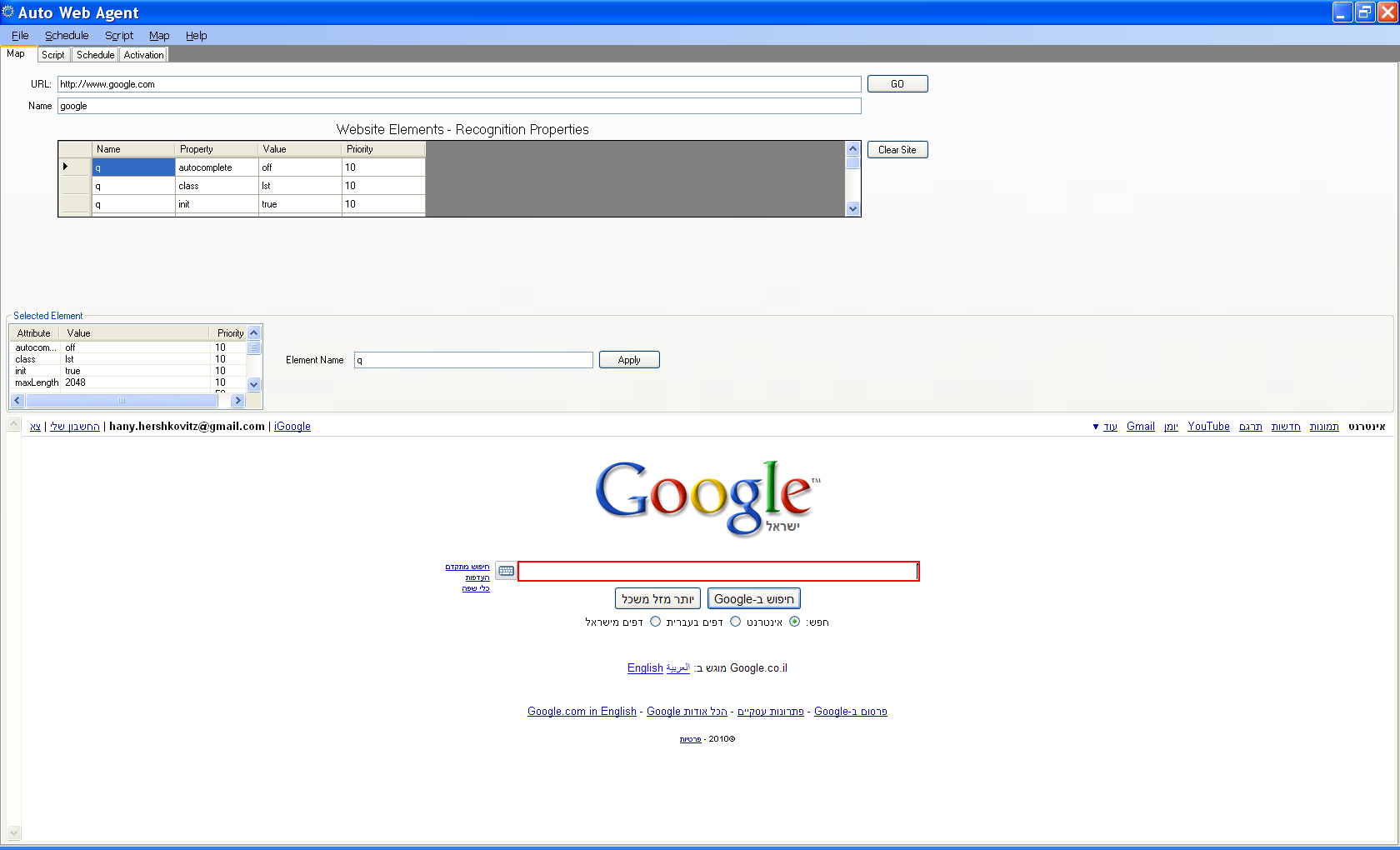
We enter [www.google.com](http://www.google.com) in the URL text box. We have an option to give the url a friendly alias in the name text box, if we don't the name is the same as the URL after pressing the go button. In this example well name it "google"

When the go button is pressed the browser control navigates to the entered website if it is valid. Drawing ‎8‑2:Navigate to target website Shows it:



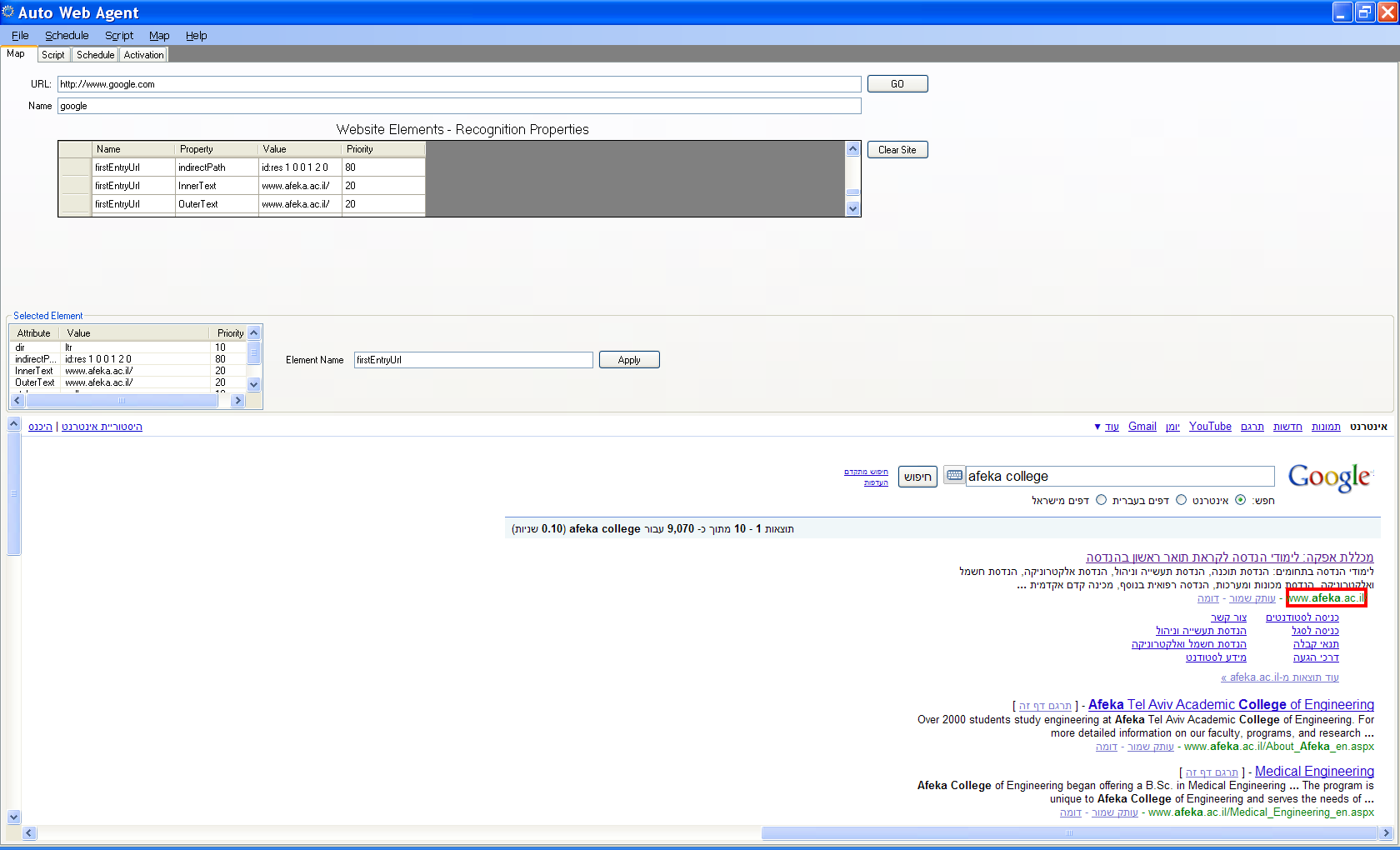
Drawing ‑:Navigate to target website

Hovering over the web page elements highlights them with red rectangle. Double pressing the SHIFT key will select the highlighted element, and it's discovered recognition properties will be shown in the 'Selected Element' panel.pressing the apply button will insert the element into the data base with the given 'Element Name'. if non was given the application will try to gues it by naming it as the ID element and if it doesn't exist then by it's NAME attribute. If the NAME attribute does not exist then it will prompt the user to enter a name. Drawing ‎8‑3 Shows the selection of the search text box:



Drawing ‑:mapping an element

We map in the way the search button, then enter Afeka college in the search textbox and press enter. The search results page loads and there we map the first entry url:

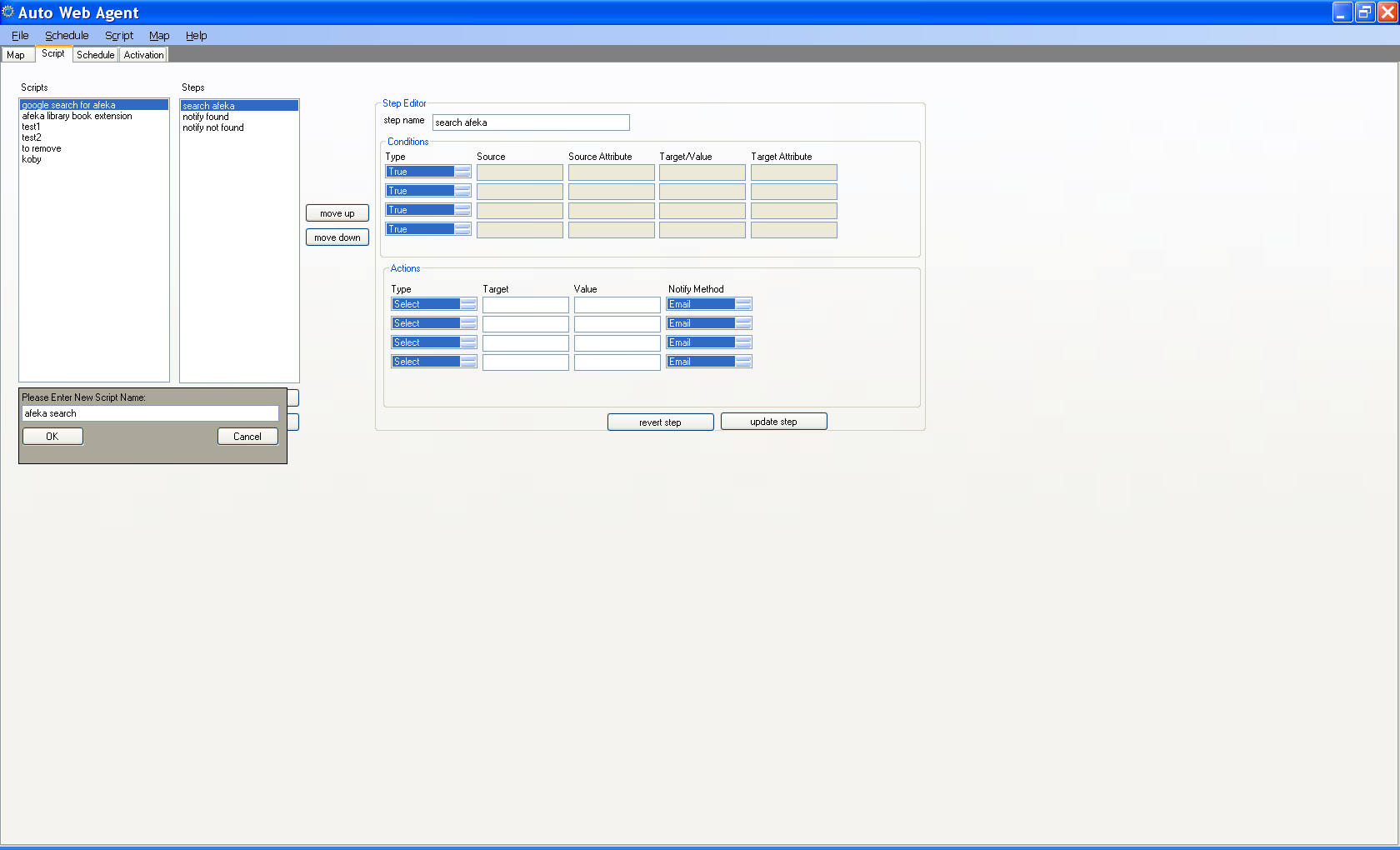


Drawing ‑:map first entry url

We are now set to compose a script.

### Script composition

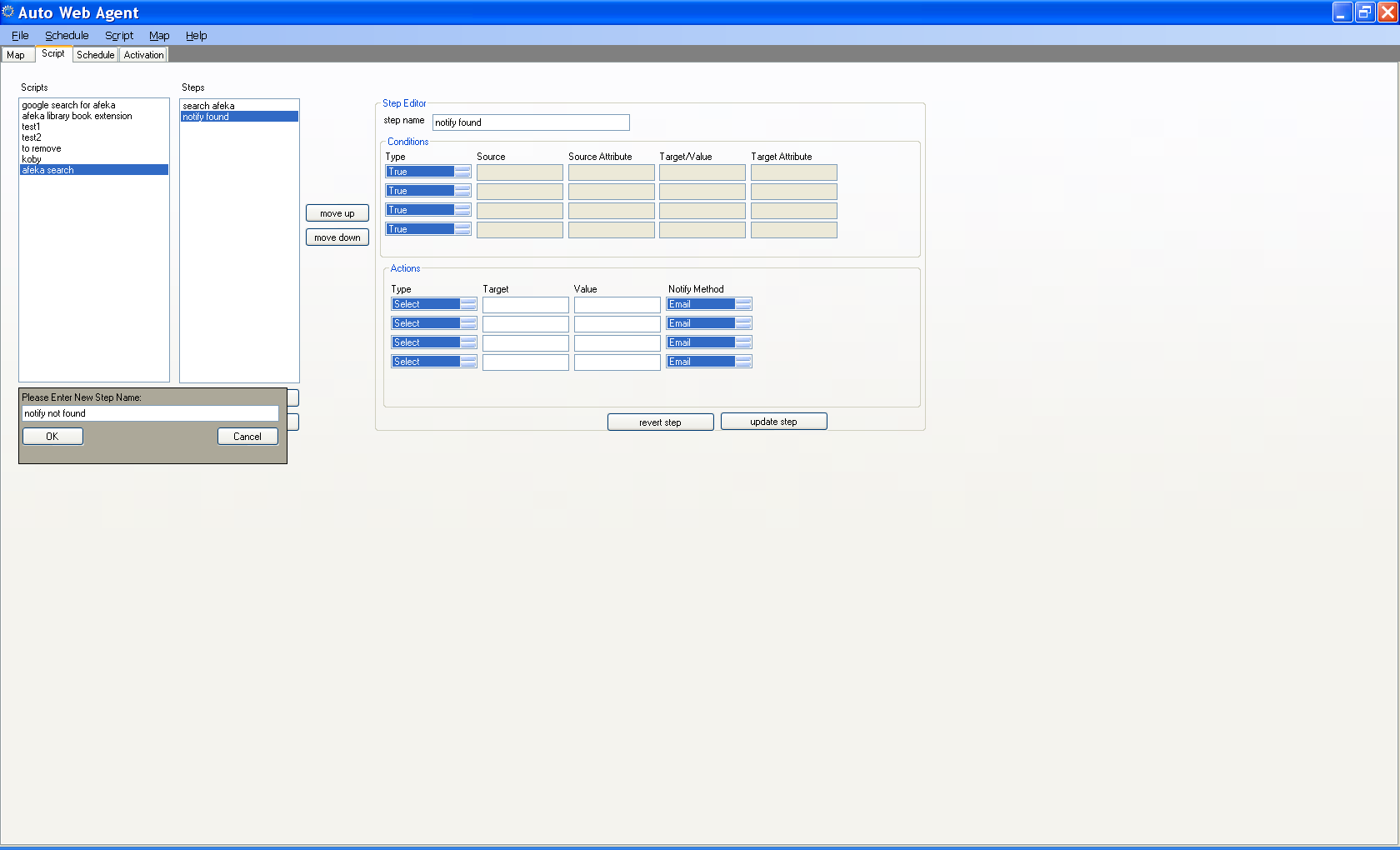
Moving to the Script tab, press the 'add script' and in the popup dialog we enter a new script name ("afeka search"):



Drawing ‑:adding new script

The script is added to the list.

Similarly, we add 3 steps ("search Afeka","notify found", "notify not found") to the script by pressing the "add step" while the "afeka search" script is selected:

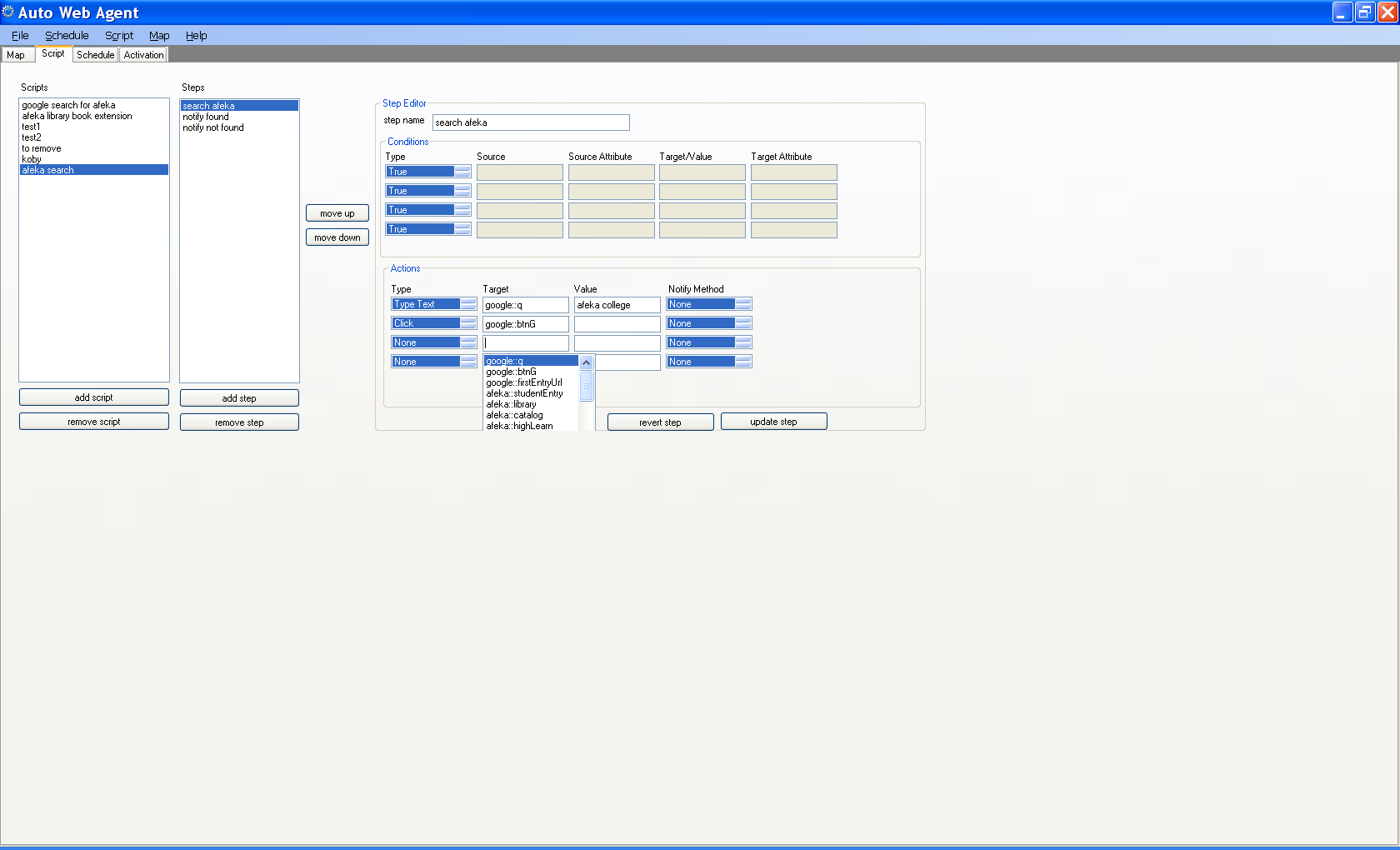


Drawing ‑:adding steps to script

Now we configure each step as follows:

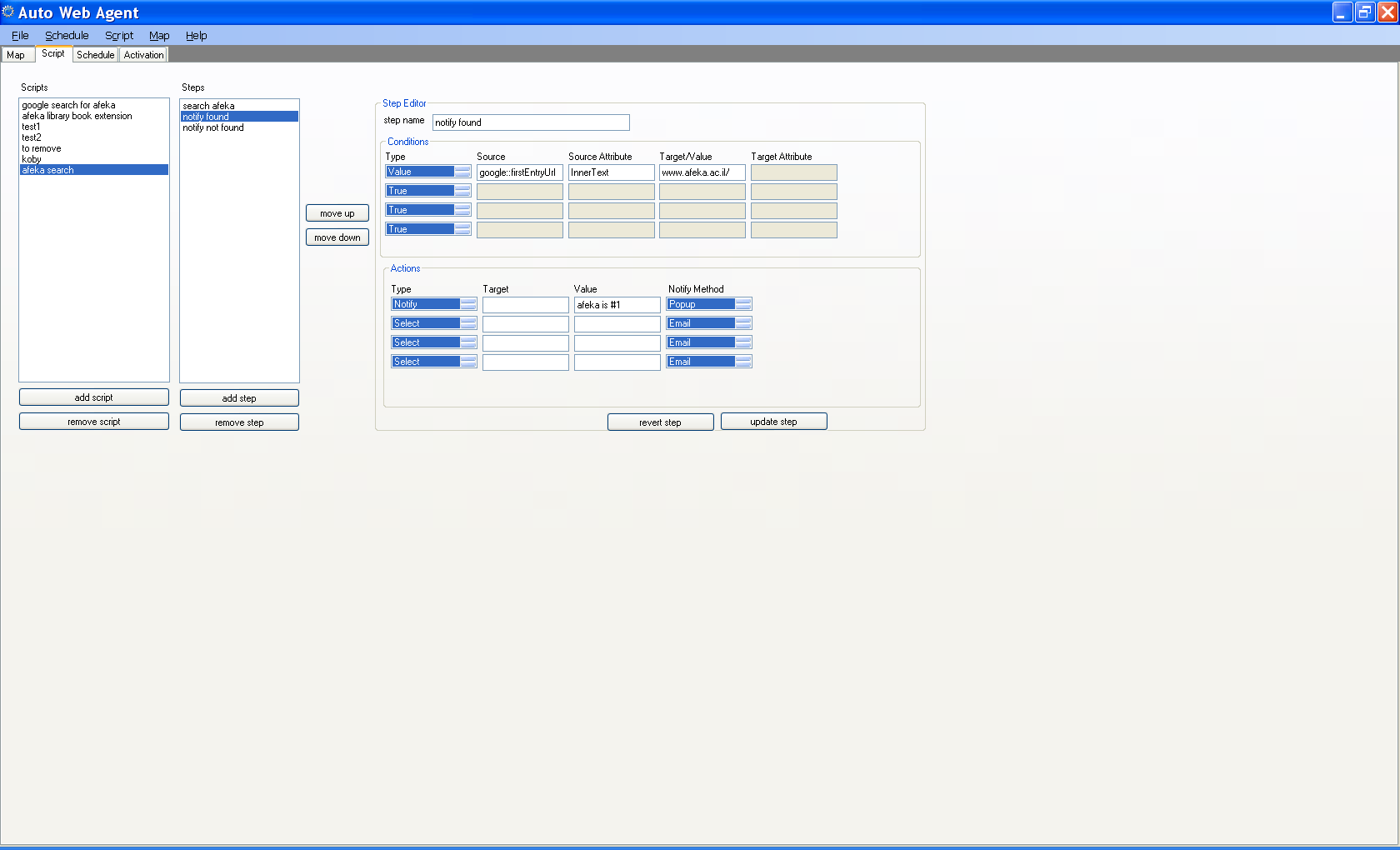
* Select step
* Fill in conditions that must evaluate to true in order for the actions to be performed
* Fill in the actions

In our example the first step has no condition (all true) and 2 actions (filling the search text box with "search afeka" and pressing the search button):



Drawing ‑:step 1 config

In the second step we'll set a notify action on the "Value" condition that FirstEntryUrl innerText attributes value is [www.afeka.ac.il/](http://www.afeka.ac.il/):

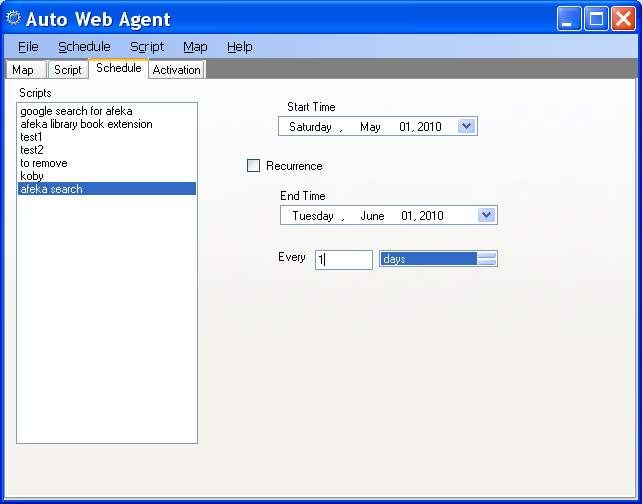


Drawing ‑:step 2 config

In a similar manner we set the last step but on a "NotValue" condition.

### Script scheduling

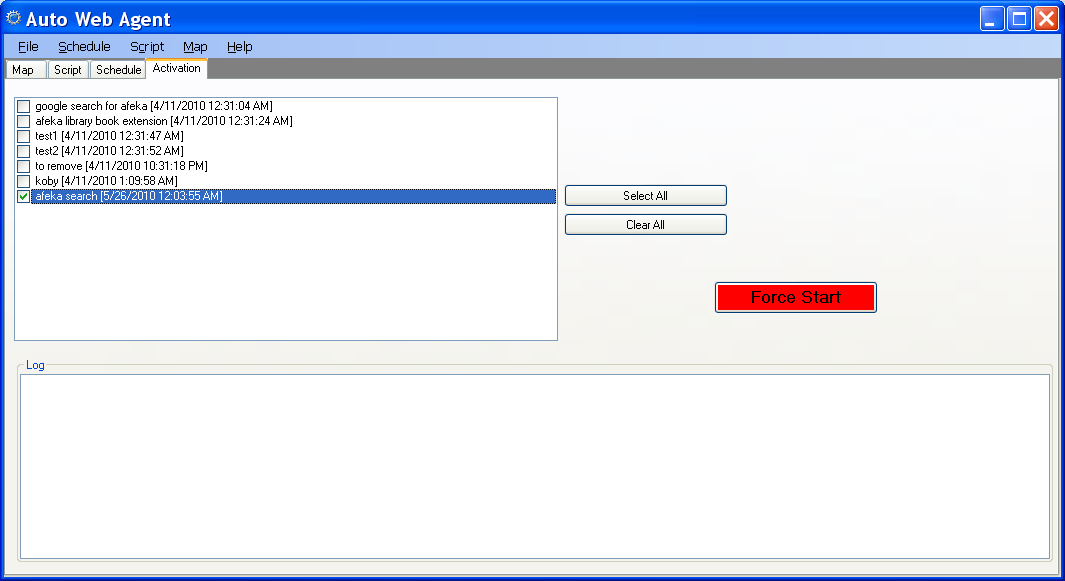
We want to run the script every day for a month, so we need o configure it in the schedule tab:



Drawing ‑:script scheduling

### Script activation

We activate the script by marking its checkbox in the activation tab:



Drawing ‑:script activation

## Efficiency analysis

* The system is minimizing DB access by saving in memory dataset of the DB.
* The mapping process time is heavily dependent on the internet connectivity.
* Script run time is dependent on the WatiN library. In average locating and manipulating a control take about 500ms.

## System Performance

## System Cost

The system is a pure software product.

## System reliability

All data is backed up to the DB, which makes the system very reliable.

## System integrity

The system over all work well, but it has problem with sites with IFRAMES (like "Afeka library" or "hapoalim bank"), and flash.

## Originality and uniqueness

This is the first product that tries to make browser automation accessible to non programmers by presenting an intuitive graphical interface, and simplifies and abstract the technical details involved.

## Data Security

The database used is password protected and encrypted.

# Derivatives

The derivative of the project is installation application that needs to be installed on the user's PC.

The required infrastructure is:

* Any modern desktop (latest decade will do) running windows XP and above.
* Internet connection

# Epilogue

## Summary and Conclusions

The work process in the project was in the waterfall paradigm as dictated by the process.

Most of the system was well defined prior to implementation.

During the project I learned various aspects of the inner workings of a browser.

A major conclusion from the project is that the management part of the project (documentation and report writing) took most of the project time, I think this is due to the waterfall method. I think an Agile approach would have produced a better quality product as documentation task are less intensive.

About 90% of the system is implemented and will reach 100% by the due date.

The system produced is in prototype grade but further development can get it to commercial status.

## Future Work and Development

* Add support for flash
* Add support for IFRAMES
* Extend the condition and action types
* Examine commercialization options

## Works cited

1. Jon Kleinberg. **Authoritative sources in a hyperlinked environment**. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Extended version in Journal of the ACM 46(1999). Also appears as IBM Research Report RJ 10076, May 1997. [[pdf]](http://www.cs.cornell.edu/home/kleinber/auth.pdf)
2. Soumen Chakrabarti, Martin van den Berg, Byron Dom, **Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery**, WWW8 [[html]](http://www8.org/w8-papers/5a-search-query/crawling/index.html)
3. Steve Lawrence and Lee Giles, **Searching the World Wide Web,** Science 1998 [[pdf]](http://www.neci.nj.nec.com/~lawrence/papers/search-science98/search-science98.pdf)
4. Jeffrey Dean, Monika R. Henzinger, **Finding Related Pages in the World Wide Web**, WWW8 [[html]](http://www8.org/w8-papers/4a-search-mining/finding/finding.html)
5. Oren Zamir and Oren Etzioni, **Grouper: A Dynamic Clustering Interface to Web Search Results**, WWW8, [[html]](http://www8.org/w8-papers/3a-search-query/dynamic/dynamic.html)
6. Rajan M. Lukose and Bernardo A. Huberman, **Surfing as a Real Option** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/options.html)
7. Bernardo A. Huberman and Rajan M. Lukose, **Social Dilemmas and Internet Congestion** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/storms.htm)
8. Bernardo A. Huberman, Peter L.T. Pirolli, James E. Pitkow, and Rajan M. Lukose, **Strong Regularities in World Wide Web Surfing**, Nature ??? [[abstact]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/surfing.html)
9. William W. Cohen, Wei Fan, **Web-Collaborative Filtering: Recommending Music By Spidering the Web**, WWW9 [[html]](http://www9.org/w9cdrom/266/266.html)
10. Mike Perkowitz Oren Etzioni, **Towards Adaptive Web Sites: Conceptual Framework and Case Study**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/towards/towards.html)
11. Marc Langheinrich, Atsuyoshi Nakamura, Naoki Abe, Tomonari Kamba, Yoshiyuki Koseki, **Unintrusive Customization Techniques for Web Advertising**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/unintrusive/unintrusive.html)
12. Jim Hendler, **Is there an intelligent agent in your future?** Nature, Web matters, 11 March 1999. [[html]](http://helix.nature.com/webmatters/agents/agents.html)
13. Steve Kirsch, **The future of Internet search (keynote address)**, SIGIR 1999 [[pdf]](http://www.acm.org/pubs/articles/proceedings/ir/290941/p1-kirsch/p1-kirsch.pdf)
14. G. Pandurangan, P. Raghavan, and E. Upfal. Using PageRank to Characterize Web Structure, Proceedings of the 8th International Computing and Combinatorics Conference (COCOON), 2002.
15. [G. Pandurangan](http://www.cs.brown.edu/people/gopal) , [P. Raghavan](http://theory.stanford.edu/people/raghavan/index.html) , and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Building Low-Diameter P2P Networks**](http://www.cs.brown.edu/research/webagent/focs-2001.pdf). Proceedings of the 42th IEEE Symp. on Foundations of Computer Science. 2001.
16. S.R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**The Web as a graph**](http://www.cs.brown.edu/research/webagent/pods-2000.pdf). Proceedings of the 19th ACM Symposium on Principles of Database Systems, pp 1-10, 2000.
17. Thomas Hofmann, [**Learning Probabilistic Models of the Web**](http://www.cs.brown.edu/research/webagent/Hofmann-SIGIR00.pdf), ACM SIGIR 2000
18. R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Stochastic models for the Web graph**](http://www.cs.brown.edu/research/webagent/focs-2000.pdf). Proceedings of the 41th IEEE Symp. on Foundations of Computer Science. 2000.

# Appendix

## SRD (Software Requirements Document)

**Introduction**

**1.1 Purpose.**

The purpose of this document is to describe the requirement needed for the Automatic Web Agent System system.

The document is divided to sections, where as each section will describe the requirements from a single module.

**1.2 Scope of the software**

Nowadays web use has become common to practically every household. The web enables to perform many diverse tasks, ranging from acquiring entertainment data (movies, songs, gossip etc…) to financial transactions.

Most of the tasks a user might perform, require the user to authenticate his identity, track data on the website and perform actions based on knowledge acquired on other websites. These actions are very similar in most of the tasks regardless of their nature, all of which require the attention of the user and might be extremely time consuming.

The Automatic Web Agent application project was devised to relieve the average/advanced user from these routine and time consuming tasks.

The Automatic Web Agent will enable the user to compose a script of actions to be taken on specific websites according to data sampled on other sites and conditions that the users will set, and determine the scheduling of the script execution.

For example, there can be a script that will sample the users favorite sky website every 4 hours and will monitor the snow depth of the sky track. Once a preset depth is detected the script will extract the prices of plane tickets to the sky track and email them to the user.

Another example can be a script that will extend the lending period of a book in the Afeka college library, which is not equipped today with a built-in agent, once every 3 days.

Alternatively, an example from the business sector could be a for a company which sells products via the web, a script can track the competitors web sites for price changes and update the prices of the products accordingly.

**1.3 Definitions, acronyms and abbreviations.**

|  |  |
| --- | --- |
| DOM | Document Object Model |
| WWW | World Wide Web |
| DB | Data Base |
| Site Object | a C# Object that represent a specific element in the DOM of the website |
| HTML |  |
| DHTML |  |
| URL |  |
| HTTP |  |
| SQL |  |
|  |  |

Table ‑

**1.4 References**.

1. Jon Kleinberg. **Authoritative sources in a hyperlinked environment**. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Extended version in Journal of the ACM 46(1999). Also appears as IBM Research Report RJ 10076, May 1997. [[pdf]](http://www.cs.cornell.edu/home/kleinber/auth.pdf)
2. Soumen Chakrabarti, Martin van den Berg, Byron Dom, **Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery**, WWW8 [[html]](http://www8.org/w8-papers/5a-search-query/crawling/index.html)
3. Steve Lawrence and Lee Giles, **Searching the World Wide Web,** Science 1998 [[pdf]](http://www.neci.nj.nec.com/~lawrence/papers/search-science98/search-science98.pdf)
4. Jeffrey Dean, Monika R. Henzinger, **Finding Related Pages in the World Wide Web**, WWW8 [[html]](http://www8.org/w8-papers/4a-search-mining/finding/finding.html)
5. Oren Zamir and Oren Etzioni, **Grouper: A Dynamic Clustering Interface to Web Search Results**, WWW8, [[html]](http://www8.org/w8-papers/3a-search-query/dynamic/dynamic.html)
6. Rajan M. Lukose and Bernardo A. Huberman, **Surfing as a Real Option** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/options.html)
7. Bernardo A. Huberman and Rajan M. Lukose, **Social Dilemmas and Internet Congestion** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/storms.htm)
8. Bernardo A. Huberman, Peter L.T. Pirolli, James E. Pitkow, and Rajan M. Lukose, **Strong Regularities in World Wide Web Surfing**, Nature ??? [[abstact]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/surfing.html)
9. William W. Cohen, Wei Fan, **Web-Collaborative Filtering: Recommending Music By Spidering the Web**, WWW9 [[html]](http://www9.org/w9cdrom/266/266.html)
10. Mike Perkowitz Oren Etzioni, **Towards Adaptive Web Sites: Conceptual Framework and Case Study**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/towards/towards.html)
11. Marc Langheinrich, Atsuyoshi Nakamura, Naoki Abe, Tomonari Kamba, Yoshiyuki Koseki, **Unintrusive Customization Techniques for Web Advertising**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/unintrusive/unintrusive.html)
12. Jim Hendler, **Is there an intelligent agent in your future?** Nature, Web matters, 11 March 1999. [[html]](http://helix.nature.com/webmatters/agents/agents.html)
13. Steve Kirsch, **The future of Internet search (keynote address)**, SIGIR 1999 [[pdf]](http://www.acm.org/pubs/articles/proceedings/ir/290941/p1-kirsch/p1-kirsch.pdf)
14. G. Pandurangan, P. Raghavan, and E. Upfal. Using PageRank to Characterize Web Structure, Proceedings of the 8th International Computing and Combinatorics Conference (COCOON), 2002.
15. [G. Pandurangan](http://www.cs.brown.edu/people/gopal) , [P. Raghavan](http://theory.stanford.edu/people/raghavan/index.html) , and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Building Low-Diameter P2P Networks**](http://www.cs.brown.edu/research/webagent/focs-2001.pdf). Proceedings of the 42th IEEE Symp. on Foundations of Computer Science. 2001.
16. S.R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**The Web as a graph**](http://www.cs.brown.edu/research/webagent/pods-2000.pdf). Proceedings of the 19th ACM Symposium on Principles of Database Systems, pp 1-10, 2000.
17. Thomas Hofmann, [**Learning Probabilistic Models of the Web**](http://www.cs.brown.edu/research/webagent/Hofmann-SIGIR00.pdf), ACM SIGIR 2000
18. R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Stochastic models for the Web graph**](http://www.cs.brown.edu/research/webagent/focs-2000.pdf). Proceedings of the 41th IEEE Symp. on Foundations of Computer Science. 2000.

1. <http://en.wikipedia.org/wiki/Trident_%28layout_engine%29> , downloaded 1/9/2009
2. <http://msdn.microsoft.com/en-us/magazine/cc163723.aspx> , downloaded 1/9/2009

**1.5 Overview of the document.**

This document is mainly aimed for the developers and technical staff involved in the project, and will provide a basic understanding of the requirements to be met on project completion. The main points of interest for a developer would be:

1. The functional requirement listing, in section 3, interface requirements and the security and performance requirements near the end of that section.

2. The model description in section 2.7.

For the IT personnel, the primary points of interest are:

1. Environment considerations – section 2.4

2. Resource requirements – sections 3.4, and

3. Security requirements – section 3.6

**2 General Descriptions**

**2.1 Relation to current projects.**

This project is somewhat dependent on the open source Web Test API called WatiN [[3]](#footnote-3) the current version used (WatiN-2.0.10.928-net-2.0) should be enough for this project.

**2.2 Relation to predecessor and successor projects.**

NA – this is a new project.

**2.3 Function and purpose**

|  |  |
| --- | --- |
| Function | purpose |
| DOM element recording | allow a method for the user to populate the DB with mappings between DOM elements and Site Objects |
| Site Objects management | allow user to change recognition parameters of a DOM element contained in a Site Object |
| Script composition and validation | Allow the user to create the automation script and validate it. |
| Script management | Allow the user to rename, save or delete scripts |
| Script scheduling | Allow the user to schedule the execution time and frequency of the script. |
| Script Execution | Actual execution of the actions in the scripts which include monitoring and manipulating websites and notifications to the user. |

Table ‑

**2.4 Environmental considerations.**

The product will be used on the customer PC as a managed windows application.

The target audience of this product is intermediate to advanced browser user with beginner knowledge in scripting concepts.

The product is designed to run on:

* X86 architecture
* 1GB RAM / HD space TBD
* windows XP OS.

**2.5 Relation to other systems.**

Any website is hosted on a hosting server (HTTP server)

HTTP Server

Auto Web Agent application

WWW

Context diagram 1

When a browser navigates to a website it actually sends a request for a page from the HTTP server (via the URL entry in the browser which translates to a HTTP request). The server then sends the html document to the browser which constructs from it a DOM tree and then renders it to the screen.

In this project the browser is embedded in the Auto Web Agent application.

**2.6 General constraints.**

* Only pure DHTML web sites will be supported (i.e. no flash/flex/SilverLight/activex support).
* The project provided is intended as a proof-of-concept only
* The product will be graphically designed to some extent but the development effort will be put into functionality and features as a priority.

**2.7 Model description.**

The following Drawings will describe the data flow between the various processes in the system.

**2.7.1 DFD Drawings**

Context Drawing (DFD0):

The system consists of 2 main processes: acquiring data to help recognize the DOM elements of the website and a script composition/execution process



Drawing - context Drawing

Element Mapping:

The user selects elements in the browser control The elements are then saved in the DB.



Drawing – element mapping

Edit element map:

The user (script programmer) selects and marks the priority of the DOM element attribute according to which the element will be identified in the script.



Drawing - edit element map

Edit Script:

In this process script is being composed and stored in the DB



Drawing – edit script

Script run:

DOM element mapped to site objects are being monitored and manipulated according to the script commands (steps)

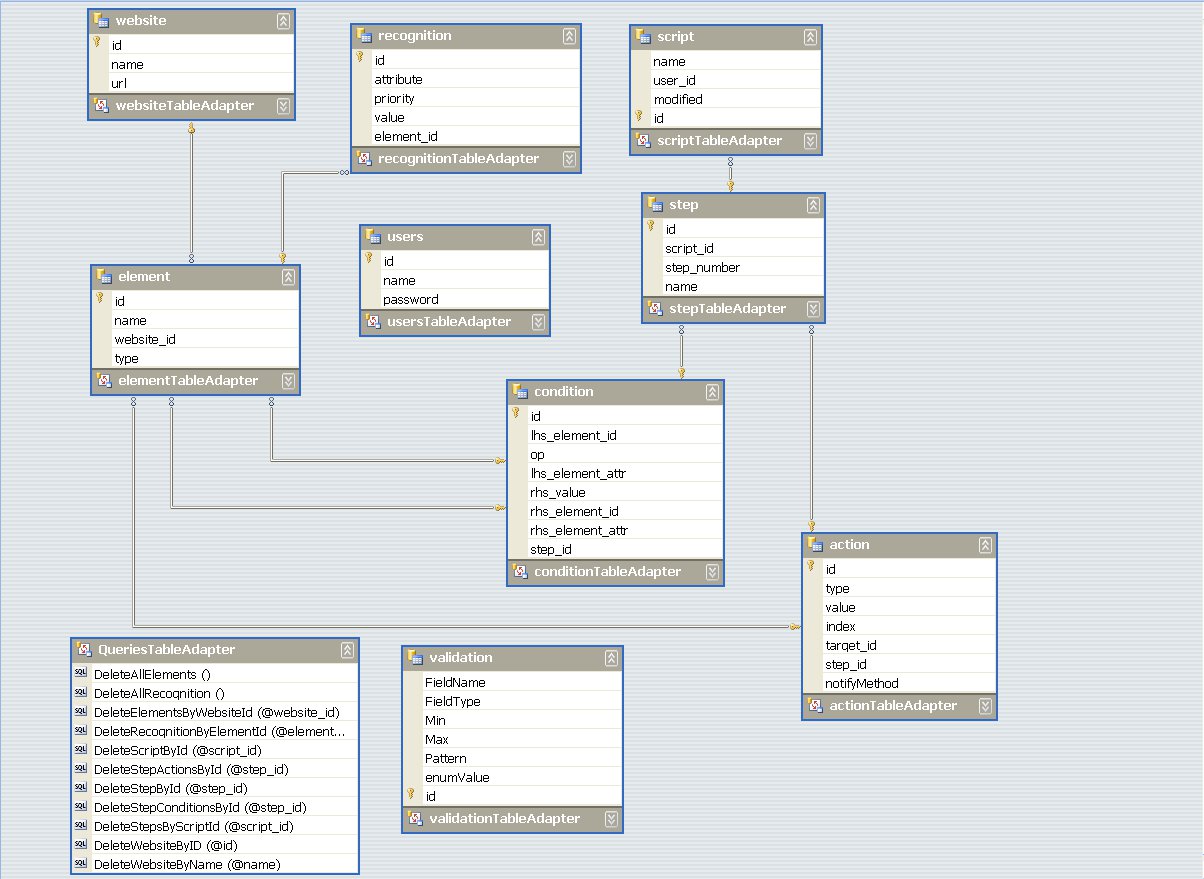


Drawing – script run

**3 Specific Requirements**

**3.1 Functional requirements.**

* The system shall save all its data in a DB based on MS SQL sdf format (see Drawing 6):
  + Element mapping between application objects and DOM elements. This will be done utilizing the website/element/recognition tables where a match of several site object properties will be used to select the corresponding DOM element
  + Script elements that compose scripts, which consist of script/step/condition/action tables



Drawing - ERD

* The application shall have 4 main forms:
  + Browser control form via which the user will select the relevant elements
  + Map editing form where the user will tweak the table that will help in recognizing DOM elements. The table will map between a site object and DOM element attributes.
  + Script scheduling form where the user will determine the time and frequency of script execution.
  + Script execution status and log.
* Element selection will be done using mouse hover over the element and pressing the middle mouse button for confirmation. When the mouse hovers over an element it shall be surrounded by a red rectangle.
* Script execution shall reuse the browser control form.
* Log of execution progress shall be displayed and written to a file.
* Workflow shall be implemented according to the DFD Drawings in section 2.7.1

**3.2 Performance requirements.**

* The system will support at least 3 simultaneous website interaction.
* Element interaction (click/text fill) shall be done in 1 second max.

**3.3 Interface requirements.**

NA

**3.4 Resource requirements.**

* X86 based cpu
* MS windows XP or greater
* 2 GB memory
* 20MB hard disk space

**3.5 Verification requirements.** As in test plan document

**3.6 Security requirements.**

All site and user data shall be stored in a password protected DB file.

## SDD (Software Design Description)

1. Introduction.
   1. **Purpose.**

The purpose of this document is to specify all aspects of the software system design for the "Automatic Web Agent" Project.

The main audience for this document is developers, basing system implementation on the application interfaces and functional flows laid out in this document.

This document will also provide sections aiding management or external consulting staff by describing system architecture, referenced components and modules and system interfaces, by which a general understanding of the system may be attained.

* 1. **Definitions, acronyms and abbreviations.**

Table ‑

|  |  |
| --- | --- |
| DOM | Document Object Model |
| WWW | World Wide Web |
| DB | Data Base |
| Site Object | a C# Object that represent a specific element in the DOM of the website |
| HTML | Hyper Text Markup Language |
| DHTML | Dynamic HTML |
| URL | Universal Resource Link |
| HTTP | Hyper Text Transfer Protocol |
| SQL | Simple Query Language |
| API | Application Programming Interface |
| CSS | Cascading Style Sheet |

* 1. **References.** May extend/delete information from SRD Section 1.4.

1. Jon Kleinberg. **Authoritative sources in a hyperlinked environment**. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Extended version in Journal of the ACM 46(1999). Also appears as IBM Research Report RJ 10076, May 1997. [[pdf]](http://www.cs.cornell.edu/home/kleinber/auth.pdf)
2. Soumen Chakrabarti, Martin van den Berg, Byron Dom, **Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery**, WWW8 [[html]](http://www8.org/w8-papers/5a-search-query/crawling/index.html)
3. Steve Lawrence and Lee Giles, **Searching the World Wide Web,** Science 1998 [[pdf]](http://www.neci.nj.nec.com/~lawrence/papers/search-science98/search-science98.pdf)
4. Jeffrey Dean, Monika R. Henzinger, **Finding Related Pages in the World Wide Web**, WWW8 [[html]](http://www8.org/w8-papers/4a-search-mining/finding/finding.html)
5. Oren Zamir and Oren Etzioni, **Grouper: A Dynamic Clustering Interface to Web Search Results**, WWW8, [[html]](http://www8.org/w8-papers/3a-search-query/dynamic/dynamic.html)
6. Rajan M. Lukose and Bernardo A. Huberman, **Surfing as a Real Option** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/options.html)
7. Bernardo A. Huberman and Rajan M. Lukose, **Social Dilemmas and Internet Congestion** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/storms.htm)
8. Bernardo A. Huberman, Peter L.T. Pirolli, James E. Pitkow, and Rajan M. Lukose, **Strong Regularities in World Wide Web Surfing**, Nature ??? [[abstact]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/surfing.html)
9. William W. Cohen, Wei Fan, **Web-Collaborative Filtering: Recommending Music By Spidering the Web**, WWW9 [[html]](http://www9.org/w9cdrom/266/266.html)
10. Mike Perkowitz Oren Etzioni, **Towards Adaptive Web Sites: Conceptual Framework and Case Study**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/towards/towards.html)
11. Marc Langheinrich, Atsuyoshi Nakamura, Naoki Abe, Tomonari Kamba, Yoshiyuki Koseki, **Unintrusive Customization Techniques for Web Advertising**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/unintrusive/unintrusive.html)
12. Jim Hendler, **Is there an intelligent agent in your future?** Nature, Web matters, 11 March 1999. [[html]](http://helix.nature.com/webmatters/agents/agents.html)
13. Steve Kirsch, **The future of Internet search (keynote address)**, SIGIR 1999 [[pdf]](http://www.acm.org/pubs/articles/proceedings/ir/290941/p1-kirsch/p1-kirsch.pdf)
14. G. Pandurangan, P. Raghavan, and E. Upfal. Using PageRank to Characterize Web Structure, Proceedings of the 8th International Computing and Combinatorics Conference (COCOON), 2002.
15. [G. Pandurangan](http://www.cs.brown.edu/people/gopal) , [P. Raghavan](http://theory.stanford.edu/people/raghavan/index.html) , and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Building Low-Diameter P2P Networks**](http://www.cs.brown.edu/research/webagent/focs-2001.pdf). Proceedings of the 42th IEEE Symp. on Foundations of Computer Science. 2001.
16. S.R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**The Web as a graph**](http://www.cs.brown.edu/research/webagent/pods-2000.pdf). Proceedings of the 19th ACM Symposium on Principles of Database Systems, pp 1-10, 2000.
17. Thomas Hofmann, [**Learning Probabilistic Models of the Web**](http://www.cs.brown.edu/research/webagent/Hofmann-SIGIR00.pdf), ACM SIGIR 2000
18. R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Stochastic models for the Web graph**](http://www.cs.brown.edu/research/webagent/focs-2000.pdf). Proceedings of the 41th IEEE Symp. on Foundations of Computer Science. 2000.
19. <http://en.wikipedia.org/wiki/Trident_%28layout_engine%29> , downloaded 1/9/2009
20. <http://msdn.microsoft.com/en-us/magazine/cc163723.aspx> , downloaded 1/9/2009
21. System Architectural Design
    1. System Architecture

The system architecture is composed of 3 main modules:

* Persistency Module
* Mapping module
* Scripting Module

The Persistency module contains the database, and the components to manipulate and query it's data (Data Access Layer). The DB is responsible to persist element mappings and scripting information.

The Mapping Module is composed of UI in which the user can map web page elements to scriptable objects (also known as SiteObjects), which are saved in the DB.

The scripting module is composed of a UI in which the user can create a script to manipulate and monitor a website pages displayed in IE. This module is responsible to schedule the script run time and to notify the user of various events.

The following drawing illustrates the high level architecture which will be discussed in detail in section .

Drawing ‑

Data Access Layer

DB

Persistency Module

Embedded Browser

DOM Analysis

Mapping Control

Mapping Module

Script Builder

Script Scheduler

Script Logger

Script Runner

Scripting Module

The data flow Drawing can be found in the SRD appendix.

* 1. System Interface Description.

The system has a single external interface which is the windows GUI, described is section

1. Detailed Description Of Components. Gives detailed component information.

**3.n** **[Component identifier]** Fill in name of component here.

**3.n.1 Type**. Could be a module, an input/output/temporary file, a program, a class, a script, a web page, etc.  
**3.n.2 Purpose**. Describe the purpose of the component.  
**3.n.3 Function Inputs and Outputs**. Describe the functionality of the component, including its interface properties (call and return types) and logical behavior.  
**3.n.6 Interfaces.** Define the control and data flow to and from the object. Gives a detailed picture of its context in the overall system architecture. **3.n.7 Data.** Describe in detail (where possible) the data values and data structures belonging to this component. Otherwise give an outline description.

* 1. Database
     1. Type: module
     2. Purpose

The type of the database is sdf (Compact relational database developed by Microsoft, also known the SQL Server Compact format; designed for applications that run on mobile devices and desktops; contains the complete database contents and can be up to 4GB in size.)

The database persist several functions:

* Users and password
* Mappings between DOM elements and SiteObject elements.
* Scripts
  + 1. Function Input And Outputs

All functions working on the database are in the DAL class.

* + 1. Interfaces

In order to connect to the database a data access object is required. The system uses a dataset (AutoWebAgentDBDataSet.xsd) to map the database to objects.

The dataset contains a table adapter object for each table in the database and through it allows read/write access to the database.

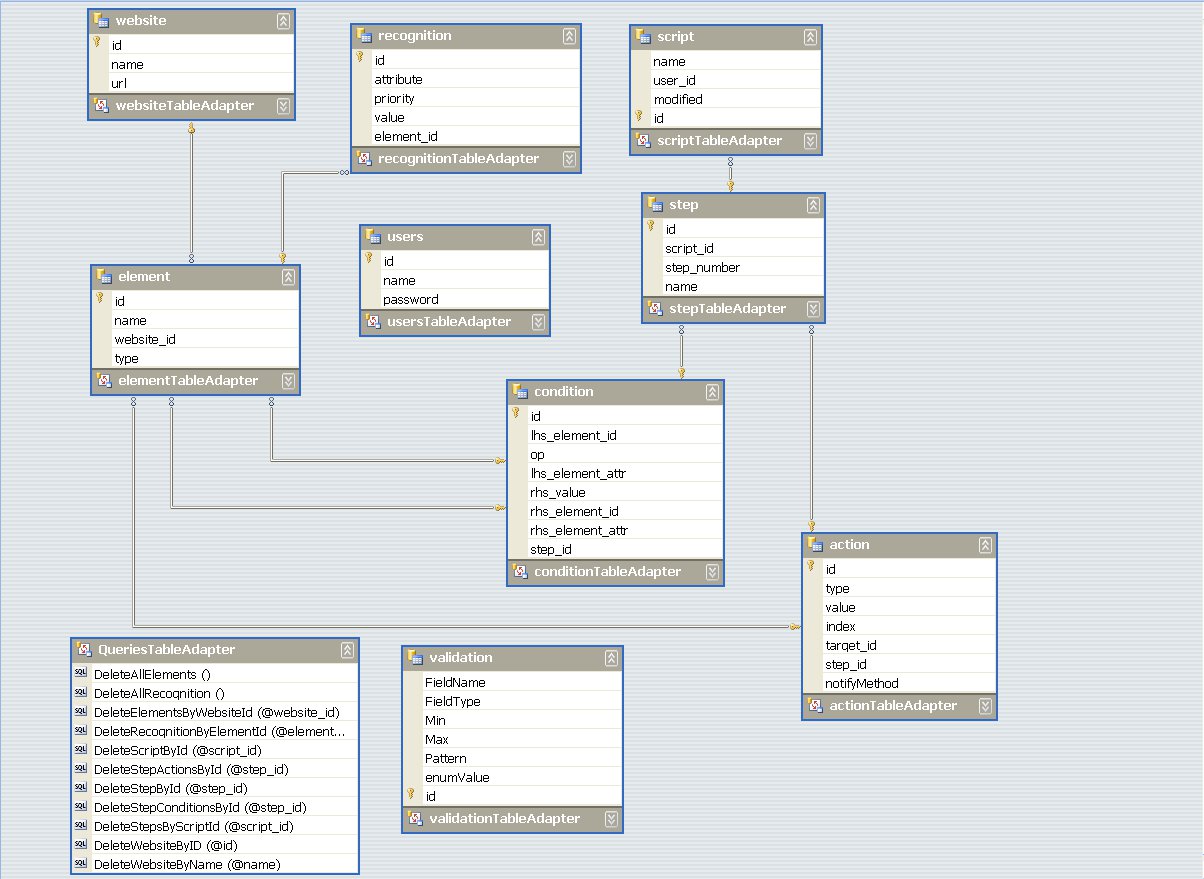
The data in the file is encrypted and in order to decrypt it a connection string must be used with the format:

Data Source=<path to .sdf file>;Password=<password>;Persist Security Info=True

* + 1. Data

The database contains the tables in and their corresponding adapters. The QueriesTableAdaper is used to have custom operation on the tables.

Drawing ‑



* 1. Data Access Layer (DAL)
     1. Type: Class
     2. Purpose

Provide a simplified access to the database, such that each table will be represented by an object, and also will contain methods to save long and complicated SQL sentences.

* + 1. Function Input And Outputs
* public IEnumerable<RecognitionProperty> GetElementRecogitionProperties(int wid, int eid): this function gets the website id and element id and returns a list of matching element recognition properties.
* public IEnumerable<RecognitionProperty> GetElementRecogitionProperties(string wName, string eName): this finction gets a website name and an element name and returns a list of matching element recognition properties.
* public int GetWebsiteID(string url): this function gets a website url and return its id in the DB.
* public int GetWebsiteIDByName(string name): this function gets a website id by searching it's name.
* public string GetWebSiteUrlByName(string wName): this function get a website id by its URL.
* public AutoWebAgentDBDataSet.websiteRow GetWebsiteRow(string url): this function gets a site URL and returns its matching rows in the website table.
* public AutoWebAgentDBDataSet.websiteRow GetWebsiteRow(string name,out bool found): this function gets a site name and returns its matching rows in the website table.
* public void SaveChanges(): this function commits changes to the DB.
* public int SaveChanges(params string[] tableNames): this function commits changes to the given tables
* public DataTable CreateElementRecognitionView(int WebsiteID): this function gets a website id and returns a DataTable for the Web Recorder View.
* private void CreateDB(string name): this function gets a DB name and creates a new DB.
* public int SetElement(AutoWebAgentDBDataSet.websiteRow WebsiteRow, string elementName, string elementType, List<RecognitionProperty> elementProperties ): this function gets a website row, an element name, an element type, and a list of recognition properties update the proper tables and return a 1 for success.
* public void ClearWebsiteData(int wid): this function gets a website id and deletes all entries relating to it.
  + 1. Interfaces

The mapping or scripting module may take an instance of this class in order to gain access to the DB.

* + 1. Data

The DAL::DB property holds an instance of awaDAL.AutoWebAgentDBDataSet dataset, which allows access to the DB.

* 1. Mapping Control
     1. Type: Method
     2. Purpose

The purpose of this module is to translate the element selected by the user in the GUI to element mapping in the data base (this maps a name of a DOM element and a set of recognition properties that will help identify it when the script runs and search this element in a page).

* + 1. Function Input And Outputs

The method implementing the function is contained in the webRecorderForm class and is called private void saveElementButton\_Click(object sender, EventArgs e)

It has the standard "event raised" arguments (sender object, and EventArgs object) and return no value.

* + 1. Interfaces

Each time the user presses the 'apply' button on the recorder GUI this method is called. If the user entered an element name it will take it otherwise it will generate a name if possible, then it will gather the recognition properties of the selected element in a list and map between the list and the name using the DAL::SetElement method described in

* + 1. Data

The method holds an instance of List<DAL.RecognitionProperty> list which contains RecognitionProperty entries.

* 1. Embedded Browser
     1. Type: module
     2. Purpose

The embedded browser is a Microsoft provided WebBrowser Control found in System.Windows.Forms assembly. It is part of the web recording form GUI and is used for helping the user select the element he wants to map. When the user hovers the mouse over an element it is surrounded by a red rectangle. When the user double click the SHIFT key the surrounded element is sent to the DOM analyzer to extract it's recognition properties.

* + 1. Function Input And Outputs

The following function are given to their respective events in the web browser control:

* void webBrowser\_DocumentCompleted(object sender, WebBrowserDocumentCompletedEventArgs e)
* void Document\_MouseOver(object sender, HtmlElementEventArgs e)
* void webBrowser\_DocumentCompleted(object sender, WebBrowserDocumentCompletedEventArgs e)
* private void webBrowser\_PreviewKeyDown(object sender, PreviewKeyDownEventArgs e)
  + 1. Interfaces

The web browser control exposes a Document object to interact with the page and also exposes the following events which are used by the web recorder form:

* DocumentCompleted – Is raised when the browser finished loading a page.
* Document.MouseOver – Is raised when the mouse hovers an element
* Document.MouseLeave – Is raised when the mouse leaves an element.
* PreviewKeyDown – Is raised when a key is pressed on the keyboard (before processing it)
  + 1. Data

The only relevant data the web browser control holds is the Document object which is used to interact with the page's DOM.

* 1. DOM Analysis
     1. Type: method
     2. Purpose

The purpose of the DOM analysis method is to analyze the selected DOM element and extract from recognition properties, such that it will be recognized uniquely and could be search and found by these properties at script runtime.

Recognition properties can be attributes of the element (for example 'id' or 'name') or calculated path to the element from another know element which is called 'indirectPath' in the case the element has no 'id' or 'name' attributes.

The algorithm to calculate the 'indirectPath' is as follows:

*if element has no id/name*

*1. determined position in parent children by counting the*

*depth of nextSibling loop*

*2. push count to stack*

*3. then check if parent have id attribute: if true then push*

*id to stack else if parent=null stop else repeat 1-3*

*4. serialize stack content and create a fake attribute:*

*indirectPath => [stack contents] with the format:*

*id:<id>|<name:<name>> <children indexes seperated by space>*

* + 1. Function Input And Outputs

The method signature is:

private void analyzeElement(HtmlElement currentElement)

it gets an HtmlElement object which represents the element the user selected, and doesn't return any value.

* + 1. Interfaces

This method is called when the user hovers over an html element on a page and the presses the SHIFT key twice in a row.

* + 1. Data

The method holds an instance of List<DAL.RecognitionProperty> list which contains RecognitionProperty entries.

* 1. Script Builder
     1. Type: Class
     2. Purpose

The purpose of this class is to enable the user to visually build a browser automation script.

The process of building a script is composed of several steps (terms are taken from the GUI):

* Add a script and set its name in a 'add script' dialog
* Add steps to the script, each step will require the following:
  + Determine the step name in the 'Add step' dialog.
  + Set the conditions to activate the step. the condition types can be:
    - True – always perform the actions.
    - False – never perform the actions.
    - Value – perform the actions only if source attribute is equal to 'Value'.
    - Equal – perform the actions only of source attribute equals target attribute
    - Checked – perform the actions only if source has a 'checked' attribute on.
    - Selected – perform the action only if source has the 'selected' attribute on.
  + Set the actions. Action types can be:
    - Select – selects source (valid for list box)
    - Check – checks source (valid for checkbox)
    - Uncheck – uncheck source (valid for checkbox)
    - Type Text – types 'Value' in source
    - Notify – generates a notification using the selected method(email, log, popup)
    1. Function Inputs and Outputs
    2. Interfaces
    3. Data
  1. Script Scheduler
     1. Type: Class
     2. Purpose

The purpose of this class is to set the scheduling parameters of a script. A script may run once, at a designated time, or run periodically in an assigned time span. The period time is set by the user, which select the amount of time and the unit for each period.

* + 1. Function Inputs and Outputs
    2. Interfaces
    3. Data
  1. Script Runner
     1. Type: Class
     2. Purpose

The purpose of this class is to enable the user select which of the scripts will be active. Only active scripts will be checked for scheduling and running.

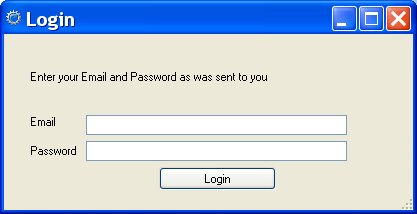
* + 1. Function Inputs and Outputs
    2. Interfaces
    3. Data
  1. Script Logger
     1. Type: class
     2. Purpose

The purpose of the logger class is to log script activations, script runs, script results. The log is written to the log text box and to the windows event log.

* + 1. Function Inputs and Outputs
    2. Interfaces
    3. Data

1. User Interface Design
   1. Login Screen
      1. Appears on application startup.
      2. Screen Image

Drawing ‑:login screen



* + 1. Objects and Actions

This screen will prevent unauthorized access to the application.

It contains standard email(user)/password fields and a login button.

* 1. About Screen
     1. When selecting Help-> About this screen will appear with the copyright details of the application
     2. Screen image

Drawing ‑

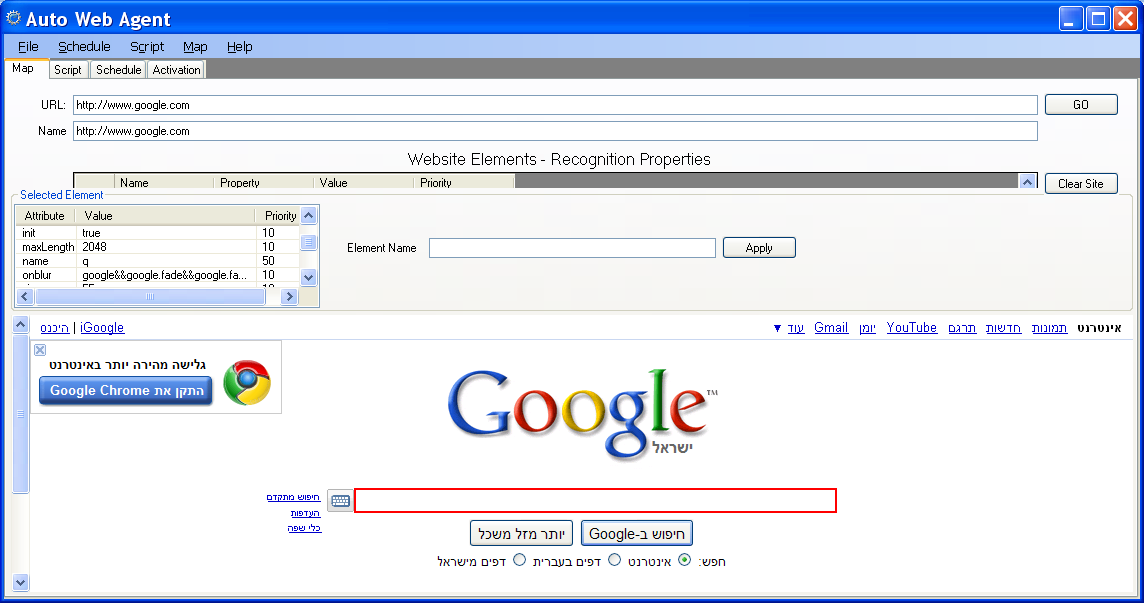


* + 1. Objects and Actions

This is a plain and standard 'About' screen with application name, author, copyright, logo and an 'OK' button to make it go away.

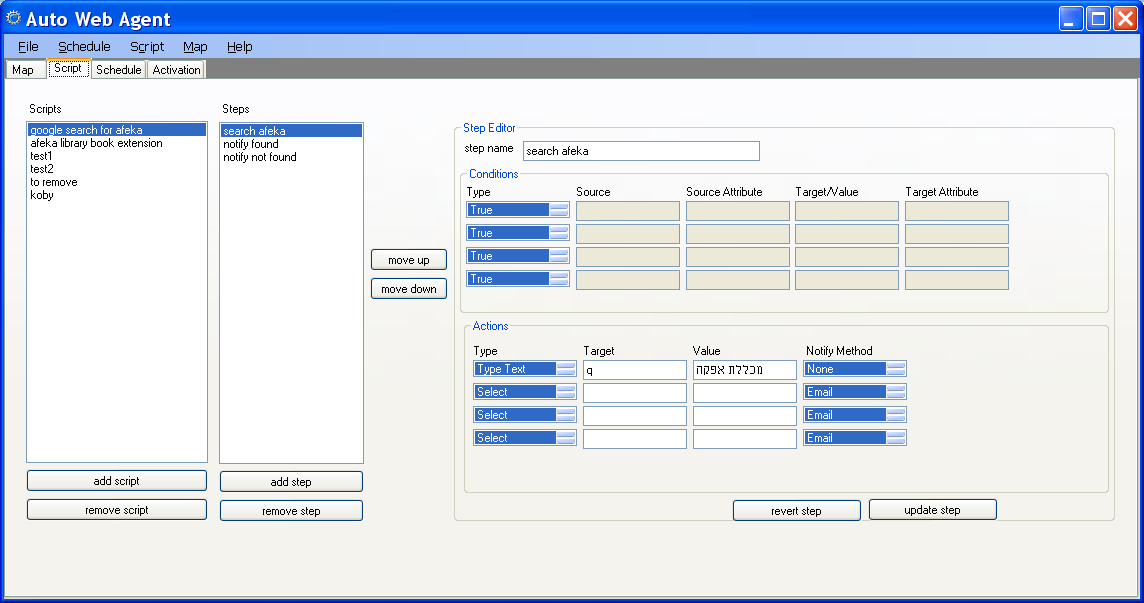
* 1. Mapping Screen
     1. This screen appears when selecting the Map tab
     2. Screen image

Drawing ‑



* + 1. Objects and Actions
* URL text box: the user enters the website URL. Pressing enter or the 'Go' button will cause the web browser control to navigate to the URL.
* Name text box: the associated name to the URL. This name will be used to address the website in the scripts. It defaults to the URL if the user leaves it blank.
* Website Elements table: displays the recognition properties for the elements saved in the DB. It contain the following fields:
  + Name: the element's name
  + Property: element property (such as id, name, class etc)
  + Value: property value
  + Priority: priority assigned for recognition algorithm in the scripting module.
* Clear site button: deletes all data related to a website.
* Selected Element table: this table shows the selected element properties. The user may change the values discovered before applying element to the DB.
* Element name textbox: used to assign a user defined name to the selected element. If empty it defaults to the element id. If element id doesn't exist then it defaults to the element name and if if that doesn't exist then an error popup is displayed.
* Apply button: commit selected element to the DB
* Web browser control: used to display the web page and to select elements on it by hovering over an element a red rectangle is displayed around an html element, double pressing the shift key will load its properties to the 'Selected Element' table.
  1. Script Screen
     1. This screen appears when selecting the Script tab
     2. Screen image

Drawing ‑

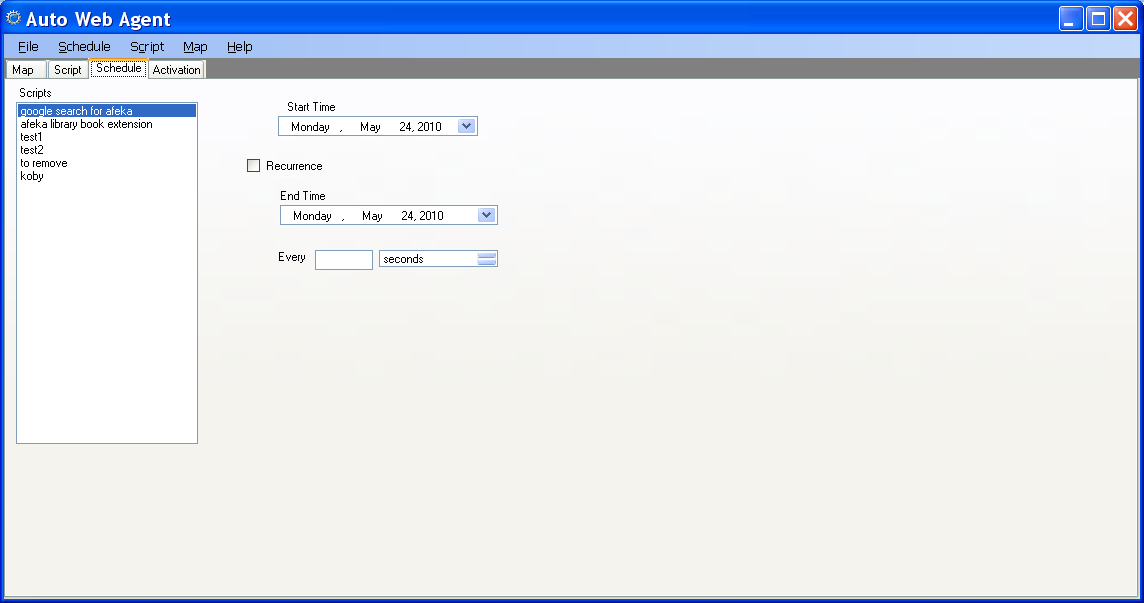


* + 1. Objects and Actions
* Scripts list box: list box that contains all the defined scripts.
* Add script button: opens a dialog which asks for the new script name. After pressing the ok the script is added to the scripts list box.
* Remove script button: delete the selected script along with its steps.
* Steps list box: list box that contain the steps in the selected script.
* Add step button: opens a dialog which asks for the new step name. After pressing the ok the step is added to the steps list box.
* Remove step button: delete the selected step.
* Move up/down buttons: moves up/down the selected step in the Steps list box.
* Step editor displays information about the selected step. A step is composed of 2 parts: the condition part and the action part. Only if the condition part evaluates to true the action part will be executed. The GUI supports 4 conditions and 4 actions per step. The condition part has 4 fields:
  + Type – true, false, equal, value, selected, checked
  + Source – the element name under test
  + Target/value – target is the element name for comparison in 'equal' condition, value is valid for value condition
  + Attribute –the attribute under test

The action part has 4 fields:

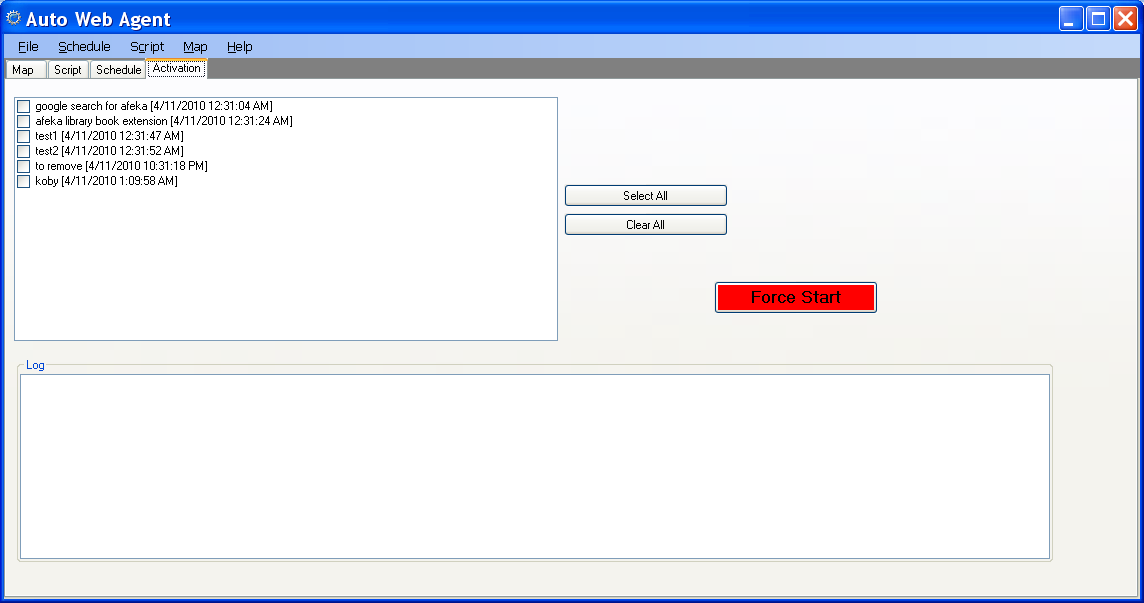
* Type – select, check, uncheck, type text, notify
* Source - the element which will be changed
* Value – the value for a 'type text' or 'select' action
* Notify method – email, log, popup. This field is valid only for the notify action.
* Update step button: commit step to DB.
  1. Schedule Screen
     1. This screen appears when selecting the Schedule tab
     2. Screen Image

Drawing ‑



* + 1. Objects and Actions
* Scripts list box: used for the user to select the script for which scheduling parameters will be set.
* Start time picker control: used to select the start time of the script run
* Recurrence checkbox: if checked enables recurrence of script run
* End time picker control: last time the script can run
* Interval text input box: holds the time interval between runs
* Time unit list box: used o select the unit of the interval time.
  1. Activation Screen
     1. This screen appears when selecting the Activation tab
     2. Screen Image

Drawing ‑



* + 1. Objects and Actions
* Checked list box: used for selecting script for activation. Only selected scripts will be considered for scheduling/run.
* Log list box: used to log events in the system, like depicted in .
* Select all button: selects all scripts
* Clear all button: clears all script selection
* Force start button: immediately start running the selected scripts regardless of scheduling.

## STD (Software Test Documentation)

1. **Introduction**

This test plan will verify the entire requirements in the Auto Web Agent SRD were implemented correctly. The philosophy of testing is to test each component as isolated as possible (Unit test each component by building a test class for each component class) and then perform tests on the system as a whole.

Three synthetic websites (i.e. html files) with assorted types of elements will be used as the test subjects.

**Test items**

* + - Element processing
    - Script processing

1. **Features to be tested**

* Test functionality of element recognition:
  + Select text input element and verify it is entered in the DB correctly.
  + Select radio button and verify it is entered in the DB correctly.
  + Select a list element and verify it is entered in the DB correctly.
  + Select button and verify it is entered in the DB correctly.
  + Select checkbox and verify it is entered in the DB correctly.
* Test functionality of map editor
  + Change name of several elements and verify they are changed in the DB
  + Change the recognition priorities/attributes of an element and verify it is reflected in the DB
* Test script editor
  + Create new script
  + Add several steps
  + Delete several steps
  + Save script to DB.
* Test script scheduling and execution
  + Schedule a script to run after 10 minutes, check it executes correctly.
  + Schedule a script to run after 1 hour for 3 times, check it executes correctly.
  + Schedule a script to loop indefinitely, check it executes correctly.

1. **Features not to be tested:**
   1. the open source WatiN library.

1. **Environmental needs –** as in project proposal section 4.

**Schedule** – as in Gantt chart.

## STR (Software Test Report)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Case | Purpose | Inputs | Expected Result | Test Procedure | Test Result | Pass /Fail |
| 1 | Test functionality of element recognition | testPage.htm | Text input surrounded by red rectangle. it is entered in the DB correctly | Select text input element and apply it. | works as designed | p |
| 2 | Test functionality of element recognition | testPage.htm | Radio button surrounded by red rectangle. it is entered in the DB correctly | Select radio button and apply it. | works as designed | p |
| 3 | Test functionality of element recognition | testPage.htm | List element surrounded by red rectangle. it is entered in the DB correctly | Select list element and apply it. | problem marking with red (minor) | f |
| 4 | Test functionality of element recognition | testPage.htm | Button surrounded by red rectangle. it is entered in the DB correctly | Select button and apply it. | works as designed | p |
| 5 | Test functionality of element recognition | testPage.htm | Checkbox surrounded by red rectangle. it is entered in the DB correctly | Select checkbox and apply it. | works as designed | p |
| 6 | Test functionality of map editor | testPage.htm | the elements are changed in the DB correctly | Change name of several elements | works as designed | p |
| 7 | Test functionality of map editor | testPage.htm | priorities/attributes reflected in the DB correctly | Change the recognition priorities/attributes of an element | works as designed | p |
| 8 | Test script editor | testPage.htm | script is saved in the DB correctly | Create new script | works as designed | p |
| 9 | Test script editor | testPage.htm | steps are saved to the DB correctly | Add steps to script | works as designed | p |
| 10 | Test script editor | testPage.htm | only deleted steps are removed from DB | Delete several steps | works as designed | p |
| 11 | Test script editor | testPage.htm | only changed steps are updated in the DB | Change several steps | works as designed | p |
| 12 | Test script scheduling and execution | testPage.htm | check it executes correctly | Schedule a script to run after 10 minutes, |  |  |
| 13 | Test script scheduling and execution | testPage.htm | check it executes correctly | Schedule a script to run after 1 hour for 3 times, |  |  |
| 14 | Test script scheduling and execution | testPage.htm | check it executes correctly | Schedule a script to loop every 5 minutes for a month, |  |  |

Table ‎11‑4:STR

## SPMP (Software Project Management Plan)

1. Introduction - Project Management Plan Overview

This appendix will describe the progress made and the progress planed of the project execution.

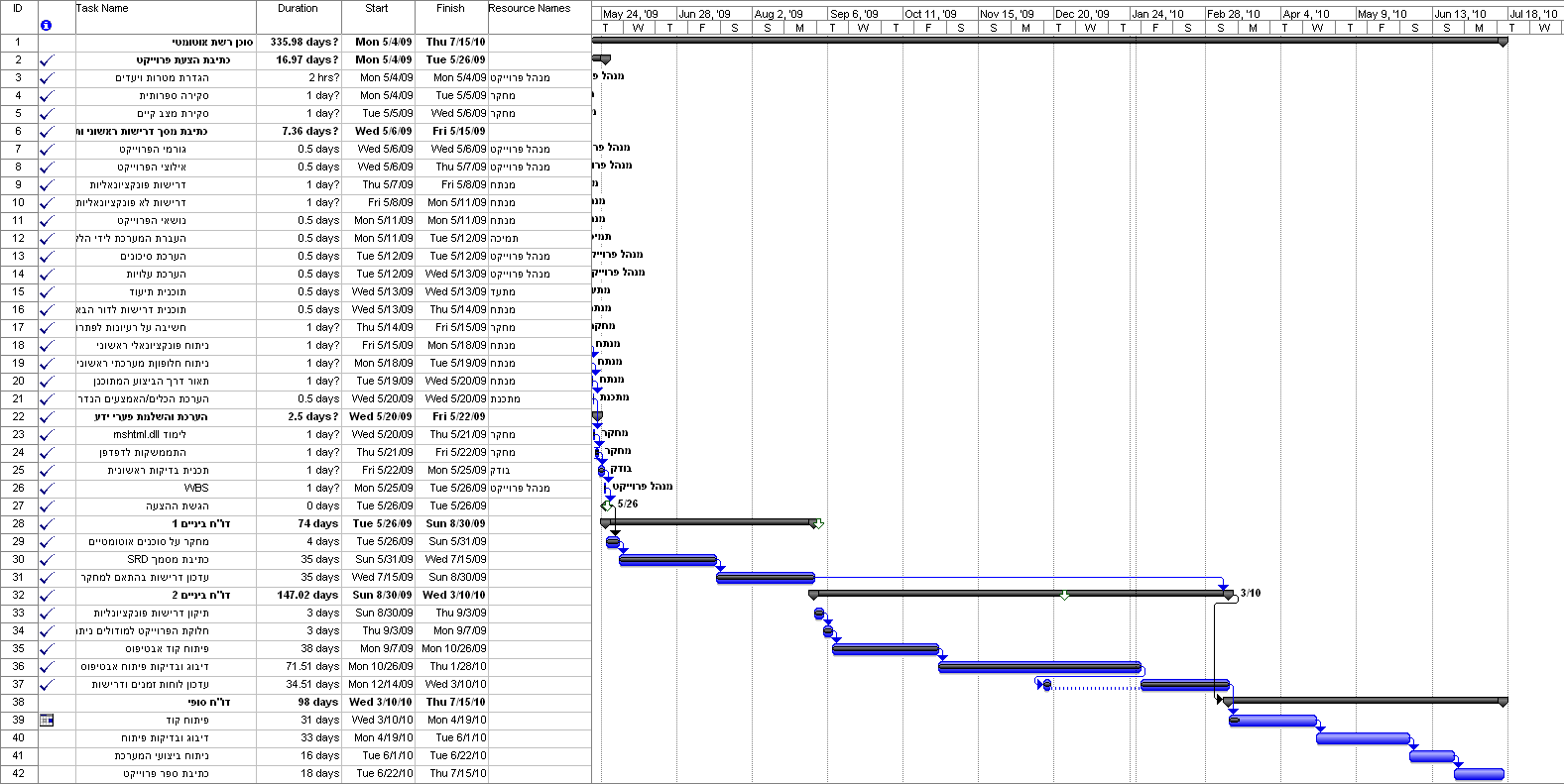
1. Risk Management - Describes updates of risks and managing them.

Due to main developer (me…) birth of new daughter, time tables may be delayed by factor of 3-4 weeks.

1. Evaluation of the SPMP

The progress of the project is more or less on track except for a 2 week delay in deliver the second report. The prototype coding was begun module by module and then integrated into an application. The next thing is to make it robust, complete all features, test it and fix all found bugs.

1. Schedule – Updated and Detailed Work Plan.



## Project Proposal

****

**מחלקת הנדסת תוכנה**

שם הפרויקט: סוכן רשת אוטומטי

Project Name: Automatic Web Agent

הצעת פרויקט

|  |  |
| --- | --- |
| שם הסטודנט: | קובי הרשקוביץ |
| מספר תעודת זהות: | 025632654 |
| שם המנחה: | אלעד הוגן |
| תאריך ההגשה: | 25/05/09 |

**תוכן ענינים**

[מבוא 4](#_Toc229036130)

[מטרות, יעדים ומדדים 5](#_Toc229036131)

[סקירה ספרותית 6](#_Toc229036132)

[סקירת מצב קיים 7](#_Toc229036133)

[מסמך דרישות ראשוני ותיחום של המערכת 7](#_Toc229036134)

[גורמי הפרויקט 7](#_Toc229036135)

[תכלית הפרויקט 7](#_Toc229036136)

[לקוחות הפרויקט 7](#_Toc229036137)

[משתמשי הפרויקט 7](#_Toc229036138)

[אילוצי הפרויקט 7](#_Toc229036139)

[אילוצים ראשיים/קריטיים 7](#_Toc229036140)

[הגדרות שמות, קיצורים וראשי תיבות 7](#_Toc229036141)

[עובדות והנחות הקשורות לפרויקט 8](#_Toc229036142)

[דרישות פונקציונאליות 8](#_Toc229036143)

[תחום העבודה 8](#_Toc229036144)

[תחום המוצר 8](#_Toc229036145)

[דרישות מידע ופונקציונאליות 8](#_Toc229036146)

[דרישות לא פונקציונאליות 8](#_Toc229036147)

[דרישות עיצוב 8](#_Toc229036148)

[דרישות שימוש וממשק משתמש 8](#_Toc229036149)

[דרישות ביצועים 8](#_Toc229036150)

[דרישות הפעלה 8](#_Toc229036151)

[דרישות תחזוקה ותמיכה 9](#_Toc229036152)

[דרישות אבטחה 9](#_Toc229036153)

[דרישות תרבותיות ופוליטיות 9](#_Toc229036154)

[דרישות חוקיות 9](#_Toc229036155)

[נושאי הפרוייקט 9](#_Toc229036156)

[נושאים פתוחים 9](#_Toc229036157)

[פתרונות מדף 9](#_Toc229036158)

[בעיות חדשות 9](#_Toc229036159)

[משימות 9](#_Toc229036160)

[העברת המערכת לידי/לניהול הלקוח 9](#_Toc229036161)

[סיכונים 9](#_Toc229036162)

[עלויות 9](#_Toc229036163)

[תיעוד ולימוד משתמשים 10](#_Toc229036164)

[דרישות לדור הבא 10](#_Toc229036165)

[רעיונות לפתרון 10](#_Toc229036166)

[ניתוח פונקציונאלי ראשוני 10](#_Toc229036167)

[ניתוח חלופות מערכתי ראשוני 13](#_Toc229036168)

[תאור דרך הביצוע המתוכנן 13](#_Toc229036169)

[האמצעים/הכלים הנדרשים 13](#_Toc229036170)

[פערי ידע שעל הסטודנט להשלים 14](#_Toc229036171)

[תוצרי הפרויקט 14](#_Toc229036172)

[תכנית בדיקות ראשונית 15](#_Toc229036173)

[ניהול סיכונים 15](#_Toc229036174)

[תוכנית ניהול הפרויקט 16](#_Toc229036175)

[Work Breakdown Structure 16](#_Toc229036176)

[תוכנית עבודה של הפרויקט 17](#_Toc229036177)

[רשימת מקורות: 17](#_Toc229036178)

# מבוא

כיום השימוש באינטרנט נעשה רווח כמעט בכל בית ובאמצעותו ניתן לבצע משימות רבות ומגוונות החל מקבלת מידע בידורי וכלה בפעולות בנקאיות. במרבית מן הפעולות נדרש המשתמש להזדהות, לבצע מעקב אחר נתונים באתר, ולבצע פעולות על סמך ידע שלמד מאתרים אחרים. פעולות אלו במרבית הזמן זהות/דומות בין האתרים וכמו כן ביצוען עלול לדרוש תשומת לב וזמן רב מן המשתמש.

הפרוייקט המוצע הינו סוכן רשת אוטומטי הבא להחליף מטלות שגרתיות וגוזלות זמן של משתמש דפדפן ממוצע/מתקדם. באמצעות המערכת המוצעת יוכל המשתמש לתכנן תסריט של פעולות מסויימות באתרים מסויימים בהתאם למידע הנדגם מאתרים אחרים ולקבוע את זמן ביצוע התסריט.

לדוגמא ניתן לקבוע תסריט שבו נדגם כל 4 שעות אתר סקי שהמשתמש מחבב במיוחד (לדוגמא www.champery.com) , התסריט מכוון לזהות את השדה של גובה השלד באתר וברגע שגובה השלג מגיע ל-110cm התסריט שולף מאתר ה-"דקה ה-90" את מחירי חבילות הנופש לאתר ושולח דוח מפורט עם הנתונים באי-מייל למשתמש.

דוגמא נוספת יכולה להיות תסריט שיאריך את זמן השאלת ספר באתר הספריה של מכללת אפקה ,שאינו מצוייד כיום בסוכן אוטומטי להארכת השאלות, פעם בשלושה ימים.

יחודיות הפרוייקט היא שבניגוד לסוכנים אוטומטיים הנמצאים באתר מסויים ונשלטים מתוך האתר הפרוייקט הנ"ל מציע סוכן שיתוכנת בצד הלקוח ולא בצד השרת, דבר אשר יאפשר לו נוחויות ושליטה מירבית להשגת מטרותיו.

# מטרות, יעדים ומדדים

**מטרות:**

רבות מהפעולות הפיננסיות, הלימודיות והבידוריות נעשות באמצעות דפדפן ברשת האינטרנט:

* רישום לאתרים
* קניה/מכירה של מניות
* הזמנת כרטיסים
* הגשת מטלות
* בחירת קורסים

רבות ממטלות אלו הן מחזוריות, שגרתיות ועלולות להתיש את המשתמש ו/או לגרום לו לבצע שגיאות באופן הכנסת הנתונים.

מטרת הפרוייקט הנגזרת מבעיה זו היא בניית אפליקציה אשר תבצע מטלות שגרתיות וצפויות מראש בצורה אוטומטית במקום משתמש דפדפן אנושי

**יעדים:**

ניתן לפרוט מטרת על זו ל-3 יעדים:

* ניטור אתר בצורה מחזורית
* ביצוע פעולות אוטומטיות בדפדפן
* התראה למשתמש ע"פ בקשתו על אירוע באתר מסויים.

**מדדים:**

המדדים ע"פ תימדד הצלחת הפרוייקט הם:

* אחוז האתרים הנתמכים מתוך עשרת האתרים הראשונים בחיפוש בגוגל יעמוד על 80% בקטגוריות פיננסים, בידור, ספורט (דוגמא לחיפוש יכולה להיות המילה 'פיננסי')
* נוהל כניסה לאתר הספרייה במכללה והארכת השאלה אם קיימים ספרים מושאלים יסתיים בהצלחה.
* נוהל דגימת אתר לשינויים בתדירות של 6 פעמים בדקה יסתיים בהצלחה (ידווח בצורה נכונה על שינוי).
* משתמש חדש יוכל להגדיר תסריט אוטומטי של דגימת 2 אתרים ואינטראקציה ביניהם בתוך כשעה.

# סקירה ספרותית

סוכני רשת הם מערכות תוכנה מורכבות אשר פועלות ברשת האינטרנט העולמית (www), ברשת הארגונית או ברשתות מקומיות. סוכני רשת מתוכננים לבצע מגוון משימות החל מאגירה וניתוב של מידע וכלה בחיפוש קיטלוג וסינון מידע.

הספרות מתארת מסגרת לתכנון וניתוח של סוכני רשת ומערכות סוכנים בהתבסס על מודלים מתמטיים של הסביבה בה הם פועלים. יש 3 אבני בניין מרכזיות לבניית סוכני רשת:

* מודלים סטוקאסטיים של גרף הרשת אשר משקללים את הפילוג והקישוריות של דפי רשת ונותנים קוים מנחים מרכזיים בתכנון סוכנים, ע"י הפקת תכונות ייחודיות של הסביבה בה הסוכנים פועלים.

הרשת יכולה להיראות כגרף מכוון אשר הצמתים בו הם דפי HTML סטטיים והקשתות בו הם קישורים מאתר אחד למשנהו. מאמר [ ‎18] מציע ומנתח מודלי גרפיים אקראיים שנוצרו ע"י תצפיות נסיוניות ברשת. המודלים שונים מהמודלים המקובלים בכך ש קשתות שנבחרו בצורה בלתי תלויה לא משתקפות בסטטיסטיקה שנצפתה ברשת, כך שקשתות במודלים אלו הינם תלויות סטטיסטית ביניהן. כמו כן נוספים צמתים לגרף כתלות בזמן. דבר זה משקף את העובדה שהרשת משתנה עם הזמן.

* שיטות למידה סטטיסטית המאפשרות לסוכנים ללמוד על סביבתם ע"י הפקת מודלים סטוקאסטיים של דף רשת ומבנה קישורים מקומיים. מאמר [ ] מתאר את הרשת כעולם שבו המוני קישורים מחברים בין מסמכי ודפי אתרים כך שנוצר גרף בעל סיבוכיות גבוהה וללא תקדים וזהו גרף הרשת. מאמר זה מציג גישה חדשה ללימוד מודלים הסתברותיים שך הרשת אשר יכולים לשמש כחלק בתהליך חיזוי קישוריות ומידע של מסמכי רשת. השיטה המתוארת במאמר זה היא שיטה הסתברותית לצמצום מימדים (probabilistic dimension reduction) אשר גוזרת ומאחדת את שיטת הניתוח הסמנטי המאוחר (Latent Semantic Analysis) ואת אלגוריתם קליינברג (Hubs-and-Authorities) בסביבה סטטיסטית.

דבר זה אמור להיות הצעד הראשון לפיתוח תשתית סטטיסטית של טכנולוגיות מידע ברשת. למרות שמאמר זה לא מתמקד ביישום מסויים, מגוון אלגוריתמים הפועלים בסביבת רשת יכולים לנצל את היתרונות בטכניקות המוצגות למטרות שונות כגון מנועי חיפוש, "זוחלי רשת" (Web crawlers) ומערכות סוכני מידע.

* אלגוריתמים לתכנון אוטונומי וקבלת החלטות בסביבת רשת לאפשר לסוכנים להשיג מטרות ולהסתגל לסביבה משתנה [16]

ניקח לדוגמא בעיות קלאסיות בתחום בסיסי נתונים/איחזור מידע כגון חיפוש בטקסט, כריית מידע וסיווג. הקלט לבעיות מסוג זה הוא בדרך כלל אוסף של נתונים/מסמכים. הרשת עם המבנה הנוסף שלה כגרף, מאפשרת את השיפור של טכניקות קיימות עם טכניקות מעולם הגרפים. במאמר הודגם שיפור זה באמצעות פתרונות מבוססי גרפים לבעיות הבאות: חיפש נושא, מיספור נושא, סיווג וזחילה (crawling).

# סקירת מצב קיים

כיום כאשר משתמש רוצה לבצע מעקב/שינוי נתונים באתר מסויים פרושות לפניו מספר אפשרויות:

תמיכה של האתר עצמו בסוכן רשת אשר מיידע את המשתמש לגבי נתון ייחודי לאתר אליו הוא שייך. לדוגמא האתר ebay מציע סוכן שיציע במכרז מחיר עבור פריט מסויים עד תקרה שיקבע המשתמש.

מעקב מחזורי אישי של המשתמש אחר שינויים באתר.

שימוש בתוכנות אוטומציה יקרות כדוגמת QFT או RFT.

QTP(Quick Test Pro ) היא חבילת תוכנה מבית מרקורי שבה ניתן לבצע אוטומציה לאפליקציות חלונאיות ורשתיות כגון אוטומציה לכתיבת דואר אלקטרוני ושליחתו או חקירת דף רשת והכנסת נתונים אליו.

RFT (Rational Functional Tester) היא גם חבילת תוכנה לבדיקות אוטומטיות מבית IBM המאפשרת גם לבצע בדיקות על אלמנטים מסוג Flex וטרמינלים כגון VT102.

מחירם של כל אחת מחיבולות אלו מגיע לכמה אלפי דולרים עבור רשיון למשתמש אחד.

# מסמך דרישות ראשוני ותיחום של המערכת

## גורמי הפרויקט

### תכלית הפרויקט

להוות כלי למעקב, והתראה על שינויים באתרי אינטרנט וביצוע פעולות מתוכנתות מראש כתגובה למידע הנכרה מהם, באותם אתרים עצמם או אתרי מטרה אחרים.

### לקוחות הפרויקט

משתמשי דפדפן מתקדמים הרוצים לבצע פעולות אוטומטיות בדפדפן ללא מגע יד אדם.

### משתמשי הפרויקט

ראה לקוחות

## אילוצי הפרויקט

### אילוצים ראשיים/קריטיים

תמיכה ב-IE בלבד.

### הגדרות שמות, קיצורים וראשי תיבות

|  |  |  |
| --- | --- | --- |
| שם/קיצור/ר"ת | פירוש באנגלית | פירוש בעברית |
| IE | Internet Explorer | דפדפן מבית מיקרוסופט |
| HTML | Hyper Text Markup Language | שפה לתיאור טקסקט עשיר |
| DOM | Document Object Model | מודל אובייקט מסמך |
| אלמנט | Element | פריט בדף HTML |
| URL | Uniform Resource Locator | מאתר משאבים אחיד |
| Hostname | Server name part of the URL | שם השרת ב-URL |
| WWW | World Wide Web | רשת האינטרנט העולמית |

### עובדות והנחות הקשורות לפרויקט

הדף הנבדק לא יכיל אובייקטי Flash ו-Java

למשתמש יש ידע בסיסי ב-HTML

הדפדפן יהיה מסוג IE.

## דרישות פונקציונאליות

### תחום העבודה

רשת האינטרנט (WWW)/אינטרהנט.

### תחום המוצר

ניטור ואוטומציה של אתרים.

### דרישות מידע ופונקציונאליות

בסיס נתונים לשמירת מיפוי אלמנטים בדף לאובייקטים, ולשמירת טבלת מצב-פעולה (trigger-action table)

ממשק משתמש:

מיפוי אלמנטים בדף לאובייקטים בבסיס נתונים.

כתיבת תסריט פעולה.

קביעת תזמון ריצה.

יכולת זיהוי אלמנטים בדף HTML.

יכולת הכנסת נתונים אוטומטית בדף HTML.

* יכולת דגימה מחזורית של שדות מסויימים באתרים מסויימים
* יכולת התראה על אירוע למשתמש במצעות חלון /או דואר אלקטרוני
* המשתמש יוכל להגדיר באילו תנאים תבוצע פעולה, אותה יוכל להגדיר גם כן (לדוגמה לחיצה על כפתורים ומילוי שדות כאשר ערך של שדה מסויים ישתנה)

## דרישות לא פונקציונאליות

### דרישות עיצוב

הממשק יעוצב כך שמשתמש דפדפן ממוצע יוכל להבין בצורה אינטואיטיבית את פעולתו.

### דרישות שימוש וממשק משתמש

הממשק יהיה מעוצב כממשק לשוניות כאשר:

הלשונית הראשונה תכיל את ממשק מיפוי האלמנטים

הלשונית השניה תכיל את ממשק כתיבת התסריטים

הלשונית השלישית תכיל את ממשק תזמון והרצת התסריטים.

### דרישות ביצועים

המערכת תפעל במהירות גבוהה ממהירות עבודה של משמתמש אנושי.

### דרישות הפעלה

חיבור לרשת.

IE מותקן.

### דרישות תחזוקה ותמיכה

תיקוני באגים ובוצעו ע"פ דרישות הלקוח.

### דרישות אבטחה

סיסמאות ופרטים אישיים יוצפנו בבסים הנתונים

### דרישות תרבותיות ופוליטיות

NA

### דרישות חוקיות

NA

## נושאי הפרוייקט

### נושאים פתוחים

האם יוכלו לרוץ כמה תסריטים במקביל

האם להכניס מנגנון שיתמוך בשינויים באתר/ים

### פתרונות מדף

Newbie (<http://www.newbielabs.com/webautomation.html>)

iMacros (<http://wareseeker.com/publisher/imacros-website-testing-and-we/38865/>(

iRobot (<http://www.irobotsoft.com>)

Selenium (<http://seleniumhq.org/>)

Rational Functional Tester (<http://www.ibm.com/software/awdtools/tester/functional/index.html>)

QTP (<http://mercuryquicktestprofessional.blogspot.com/>)

### בעיות חדשות

NA

### משימות

המשימות הנדרשות מתוארות ב-WBS ובתוכנית העבודה.

## העברת המערכת לידי/לניהול הלקוח

מוצר מדף – יתבצע באמצעות תיעוד ודוגמאות שימוש.

## סיכונים

ראה ‏13 (ניהול סיכונים)

## עלויות

פרוייקט תוכנה בלבד – ההשקעה היא זמן פיתוח בלב

## תיעוד ולימוד משתמשים

התיעוד לפרוייקט יהיה בהערות בגוף הקוד, בדוחות הביניים ובספר הפרוייקט.

## דרישות לדור הבא

לא ידוע עדיין.

## רעיונות לפתרון

ראה ניתוח פונקציונלי ראשוני לפירוט הרעיון

# ניתוח פונקציונאלי ראשוני

פעולת המערכת נחלקת לשלושה שלבים:

* איסוף/מיפוי האלמנטים ה-"מעניינים"

בשלב זה המשתמש גולש לאתר המטרה באמצעות הדפדפן המוטמע באפליקציה, ומסמן אלמנטים רלוונטיים בדף באמצעות העכבר. בלחיצה על מקש במקלדת ייטען השדה המסומן לטופס מיפוי האלמנטים באפליקציה. טופס זה מכיל מיפוי בין האלמנט ב-DOM לבין הייצוג הפנימי שלו במערכת אשר יישמש ביצירת התסריטים, וכן מידע נוסף על האלמנט כגון הטיפוס שלו, הערך הנוכחי שלו וכו'.

מיפוי המחדל עבור שם אובייקט יהיה ה-id של האלמנט, אך המשתמש יוכל לשנות זאת בטופס זה.

* הגדרת תסריט פעולה

בשלב זה המשתמש מגדיר תסריט פעולה המורכב מזוגות סדורים של תנאי קיום ופעולה (להלן יקרא זוג כזה "כלל"). תנאי הקיום יכול לבדוק ערך של אלמנט (המיוצג ע"י אובייקט) ואם הוא עומד בתנאי אז יתבצע חלק הפעולה של הכלל.

לאחר הגדרת כל כלל יתבצע מעבר בדיקה לוודא את תקינות הכללים בתסריט.

* תזמון/הרצה של תסריטים

בשלב זה בוחר המשתמש תסריט להרצה וניתנת לו האפשרות לקבוע מתי ירוץ התסריט ובאיזו מחזוריות.

מפת האלמנטים וכן התסריטים ישמרו בבסיס נתונים לצורך המשכיות (Persistence).

## תרשימי DFD

תרשים הקשר:



תרשים ‏6‑1 Context Drawing(DFD0)

מיפוי אלמנטים:



תרשים ‏6‑2 מיפוי אלמנטים

עריכת מפת אלמנטים:



תרשים ‏6‑3 עריכת מפת אלמנטים

עריכת תסריט:



תרשים ‏6‑4 עריכת תסריט

הרצת תסריט:



תרשים ‏6‑5 הרצת תסריט

# ניתוח חלופות מערכתי ראשוני

|  |  |  |  |
| --- | --- | --- | --- |
| נושא לחלופה | QTP | iMacros | RFT |
| מחיר($) | 35000-70000 | 50-500 | 6500-9000 |
| כיסוי פונקצינאלי | מושלם | חסרה שליטה בתזמון | מושלם |
| קלות שימוש | מסובך – יש ללמוד VBScript | פשוט | בינוני – יש לדעת JAVA ברמה בינונית/נמוכה. |
| Scripting | VBScript | Proprietary | JAVA |

# תאור דרך הביצוע המתוכנן

* הגדרת סכמה לבסיס הנתונים – יש להגדיר את מבנה הטבלאות כגון מבנה טבלת המיפוי ומבנה טבלאות התסריטים
* הגדרת שפת תסריט – יש להחליט על האופן בו המשתמש יתכנת תסריטים במערכת. האפשרויות הנשקלות הן שימוש באבני בניין קבועות ליצירת צעדי תסריט כך שכל צעד יורכב מבחירת מספר אלמנטים קבוע באמצעות פקדים חלונאיים. אפשרות נוספת היא כתיבת תסריט בשפת C# וניתוחו באמצעות מודול הידור תסריטים.
* חלוקת המערכת למודולים – רכיבי המערכת יחולקו למודולים הניתנים להכלה במחלקות.
* הגדרת המחלקות הנדרשות בכל מודול ומודול – כל מחלקה תכיל פונקציה לוגית בודדת.
* קידוד ובדיקת כל מודול בזה אחר זה- הקידוד יעשה בשפת C# ובדיקתו תיעשה ב-NUNIT.
* אינטגרציה של כל המודולים – בשלב זה תיבנה בפעם הראשונה כל האפליקציה כמערכת אחת.
* בדיקות – ביצוע בדיקות המערכת ווידוא שכל המטרות הושגו. כמו כן יזוהו כל הבאגים ויתועדו עבור שלב התיקון.
* תיקוני באגים ובדיקות חוזרות – ניפוי השגיאות עד לפעולה מיטבית של המערכת.
* תיעוד – כתיבת הדוחות וספר הפרוייקט. כמו כן התיעוד בתוך הקוד יעשה במהלך כתיבת הקוד.

# האמצעים/הכלים הנדרשים

**סביבת פיתוח:**

סביבת פיתוח VS2008.

דפדפן IE8

חיבור לרשת

**סביבת ריצה:**

חיבור לרשת

דפדפן IE8

.NET runtime framework 3.5

# פערי ידע שעל הסטודנט להשלים

לימוד הממשקים הנדרשים (כגון mshtml.dll, docview.dll, DOM) בעיקר מאתר MSDN

# תוצרי הפרויקט

דוח ביניים 1 – יכיל את כל אשר נדרש ע"פ ההגדרה במסמך "מבנה תוצרים – הנדסת תוכנה"

אבטיפוס.

תקציר

סקירה ספרותית

מטרות העבודה

סקר מצב קיים

ניתוח חלופות מערכתי

ניתוח חלופות טכנולוגיות

תיאור פיתרון נבחר

תוכנית בדיקות תוכנה

לוחות זמנים

ניהול סיכונים

רשימת מקורות.

מסמכים מצורפים SRD ,STP , SPMP. הצעת פרויקט.

דוח ביניים 2 - יכיל את כל אשר נדרש ע"פ ההגדרה במסמך "מבנה תוצרים – הנדסת תוכנה"

תקציר

מטרות העבודה

סקירה ספרותית

אלטרנטיבות לתיכון המערכת

הפתרון שנבחר לתיכון המערכת

תוכנית בדיקות תוכנה

לוחות זמנים מעודכנים להמשך העבודה

הערכת ביצוע המשימות

ניהול סיכונים

רשימת מקורות

מסמכים: SRD, SDD, STD, SPMP. הצעת פרויקט.

אפליקציה מושלמת (קובץ התקנת אפליקציה, חוברת הפעלה).

ספר פרוייקט - יכיל את כל אשר נדרש ע"פ ההגדרה במסמך "מבנה תוצרים – הנדסת תוכנה"

יסכם את דו"חות הביניים

ישלב דרישות מערכת ואפיון מערכת

יפנה למסמכים שכתבנו בדוחות הביניים

ארכיטקטורת המערכת

דווח בדיקות תוכנה

דוגמאות להפעלת המערכת

יתאר ביצועים עלות ואמינות

יתאר את הצלחת המערכת או אי ההצלחה.

המסמכים שנעשו במשך העבודה על הפרויקט.

מערכת

קבצי התקנה

קבצי עזרה והדרכה למשתמש

אבטיפוס ראשוני

מצגת פרויקט

מערכת סופית

# תכנית בדיקות ראשונית

בניית דפי דמה המכילים מגוון רב של סוגי אלמנטים וניסוי המערכת עליהם.

בדיקת אוטומציה של חיפוש באתר הבית של גוגל.

בדיקת ניטור שדה באתר ושליחת מייל כאשר שדה זה עובר ערך מסויים

בדיקת מנוע יצירת תסריטים

# ניהול סיכונים

|  |  |  |  |
| --- | --- | --- | --- |
| סיכון | סיכוי | השפעה | דרך מניעה |
| חריגה מלוחות זמנים | 50% | אי הגעה לאבני דרך בפרוייקט | ניסיון להקדים ביצוע של משימות והקדשת זמן נוסף לפרוייקט |
| כשל חומרתי בסביבת הפיתוח/ריצה | 20% | עיכוב של עד כשבועיים | גיבוי כל איטרציה במערכת ניהול קוד |
| אי יכולת לזהות אלמנטים מיוחדים בדף | 50% | אי יכולת להשתמש באלמנטים אלו – ירידה בפונקציונאליות המובטחת | עדכון סעיף האילוצים בשלב מוקדם ככל האפשר |
| דפים דינמיים (Ajax, DHTML) | 80% | שיבוש יכולת זיהוי האלמנטים | התקנת מנגנוני המתנה לאלמנטים לא קיימים. |

# תוכנית ניהול הפרויקט

## Work Breakdown Structure

## תוכנית עבודה של הפרויקט



תרשים ‏14‑1

# רשימת מקורות:

1. Jon Kleinberg. **Authoritative sources in a hyperlinked environment**. Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998. Extended version in Journal of the ACM 46(1999). Also appears as IBM Research Report RJ 10076, May 1997. [[pdf]](http://www.cs.cornell.edu/home/kleinber/auth.pdf)
2. Soumen Chakrabarti, Martin van den Berg, Byron Dom, **Focused Crawling: A New Approach to Topic-Specific Web Resource Discovery**, WWW8 [[html]](http://www8.org/w8-papers/5a-search-query/crawling/index.html)
3. Steve Lawrence and Lee Giles, **Searching the World Wide Web,** Science 1998 [[pdf]](http://www.neci.nj.nec.com/~lawrence/papers/search-science98/search-science98.pdf)
4. Jeffrey Dean, Monika R. Henzinger, **Finding Related Pages in the World Wide Web**, WWW8 [[html]](http://www8.org/w8-papers/4a-search-mining/finding/finding.html)
5. Oren Zamir and Oren Etzioni, **Grouper: A Dynamic Clustering Interface to Web Search Results**, WWW8, [[html]](http://www8.org/w8-papers/3a-search-query/dynamic/dynamic.html)
6. Rajan M. Lukose and Bernardo A. Huberman, **Surfing as a Real Option** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/options.html)
7. Bernardo A. Huberman and Rajan M. Lukose, **Social Dilemmas and Internet Congestion** [[abstract]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/storms.htm)
8. Bernardo A. Huberman, Peter L.T. Pirolli, James E. Pitkow, and Rajan M. Lukose, **Strong Regularities in World Wide Web Surfing**, Nature ??? [[abstact]](http://www.parc.xerox.com/spl/groups/dynamics/abstracts/InternetEcology/surfing.html)
9. William W. Cohen, Wei Fan, **Web-Collaborative Filtering: Recommending Music By Spidering the Web**, WWW9 [[html]](http://www9.org/w9cdrom/266/266.html)
10. Mike Perkowitz Oren Etzioni, **Towards Adaptive Web Sites: Conceptual Framework and Case Study**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/towards/towards.html)
11. Marc Langheinrich, Atsuyoshi Nakamura, Naoki Abe, Tomonari Kamba, Yoshiyuki Koseki, **Unintrusive Customization Techniques for Web Advertising**, WWW8 [[html]](http://www8.org/w8-papers/2b-customizing/unintrusive/unintrusive.html)
12. Jim Hendler, **Is there an intelligent agent in your future?** Nature, Web matters, 11 March 1999. [[html]](http://helix.nature.com/webmatters/agents/agents.html)
13. Steve Kirsch, **The future of Internet search (keynote address)**, SIGIR 1999 [[pdf]](http://www.acm.org/pubs/articles/proceedings/ir/290941/p1-kirsch/p1-kirsch.pdf)
14. G. Pandurangan, P. Raghavan, and E. Upfal. Using PageRank to Characterize Web Structure, Proceedings of the 8th International Computing and Combinatorics Conference (COCOON), 2002.
15. [G. Pandurangan](http://www.cs.brown.edu/people/gopal) , [P. Raghavan](http://theory.stanford.edu/people/raghavan/index.html) , and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Building Low-Diameter P2P Networks**](http://www.cs.brown.edu/research/webagent/focs-2001.pdf). Proceedings of the 42th IEEE Symp. on Foundations of Computer Science. 2001.
16. S.R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**The Web as a graph**](http://www.cs.brown.edu/research/webagent/pods-2000.pdf). Proceedings of the 19th ACM Symposium on Principles of Database Systems, pp 1-10, 2000.
17. Thomas Hofmann, [**Learning Probabilistic Models of the Web**](http://www.cs.brown.edu/research/webagent/Hofmann-SIGIR00.pdf), ACM SIGIR 2000

R. Kumar, P. Raghavan, S. Rajagopalan, D. Sivakumar, A. Tomkins, and [Eli Upfal](http://www.cs.brown.edu/people/eli), [**Stochastic models for the Web graph**](http://www.cs.brown.edu/research/webagent/focs-2000.pdf). Proceedings of the 41th IEEE Symp. on Foundations of Computer Science. 2000.

1. AWA – Automatic Web Agent [↑](#footnote-ref-1)
2. As a windows forms control [↑](#footnote-ref-2)
3. WatiN (pronounced "what-in") aims to bring an easy way to automate Web Application Testing in .Net. Inspired by Watir, WatiN has grown into a feature rich and stable framework, offering automation of Internet Explorer and Firefox. [↑](#footnote-ref-3)