GUI Testing Automated Tools

# Introduction

The following document, wants to give an overall description of those that are the most useful GUI Automated tools. Its objective is evaluating this tools in order to find the one which better fits with the necessity of ORBIS and especially with its MS CRM projects.

# List of tools analyzed

The criteria used for searching and then choose to test and analyze the UI Testing Automated Tools, were the followers:

* Running in multi-platforms and OS and especially running in/for MS Windows infrastructures.
* Specialized on testing Web Application which run in different web browsers, especially for web applications built for IE browser.
* Availability 'of information, documentation and support
* Frequency range of releases and updates
* Costs

This first process of evaluating has produced the following two tools:

* SeleniumHQ – open source software
* Ranorex – proprietary software

Below there will be a more detailed description for each of the tools, with practice examples and images descriptive.

# SeleniumHQ

Selenium is a portable software testing framework for web applications (Wikipedia, 2015).

It is a suite of toolsto automate web browsers across many platforms.

Selenium

* runs in many browsers and operating systems.
* can be controlled by many programming languages and testing frameworks.

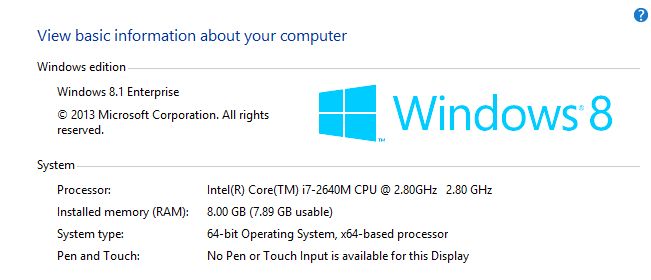
Selenium is composed by different components as will be described below. The objective of this work, is finding a tool that helps on GUI Testing so, the following evaluation are results of effective use of each component.

## Selenium IDE

*The Selenium-IDE (Integrated Development Environment) is the tool you use to develop your Selenium test cases. It’s an easy-to-use Firefox plug-in and is generally the most efficient way to develop test cases*. (Selenium IDE, 2015)

***So, as Selenium IDE is a Firefox AddOn, it doesn’t fit perfect with the necessary of testing MS Dynamics CRM Web Applications, because they mainly run on IE*.** But as the latest version of MS Dynamics CRM runs even in other browsers, the component was tested so to understand better useful and efficiency.

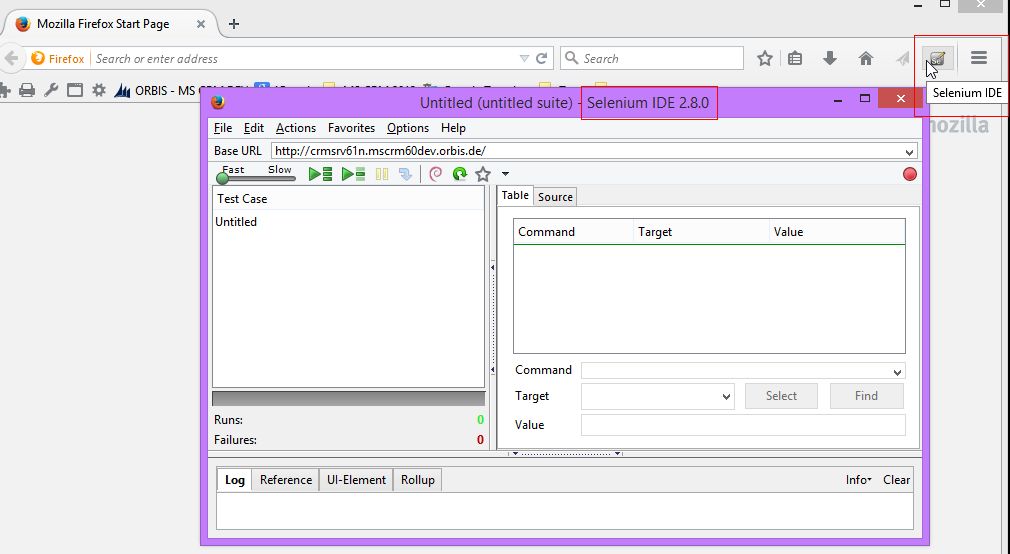
This component was installed in a machine with this infrastructure:



There was installed too the web browser Mozilla Firefox:

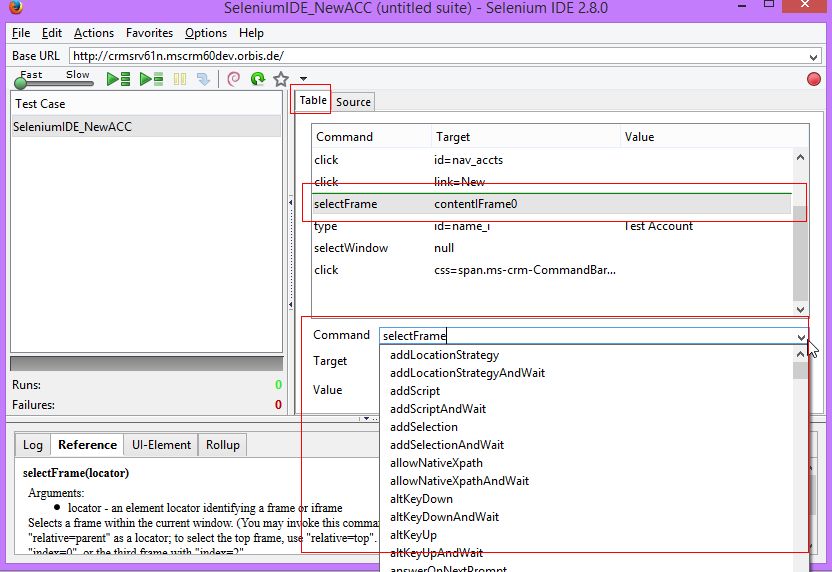


The guide founded on the Selenium Documentation described clearly how to install and make available the Selenium IDE Component. In the end, *there is an easy button directly on the Mozilla Firefox toolbar and clicking on which the Selenium IDE opened*.



The Selenium IDE interface is really simple and intuitive. There was not necessary read more documentation on how to use it. If and when will be necessary have more information about Selenium IDE interface, it is simply found on the Selenium Documentation Section on the Selenium Official web site.

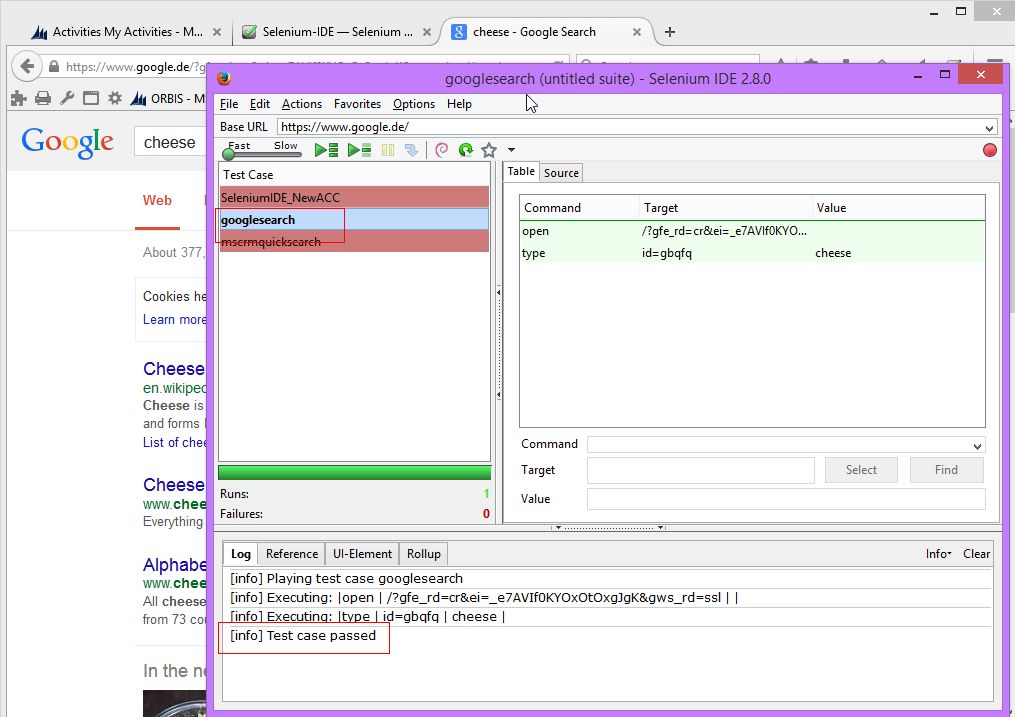
Recording a test case was quite simple. It is sufficient click on the record button (the red ball) and start executing the necessary action on the browser. During this process Selenium IDE records all the actions and furthermore it creates editable commands. They are visible on the “Table” area and in the same, users can update them, helped by a kind of intelli-sense which propose to the user the possible command.



For having more information so to be able to make a more objective evaluation, for this component there were made three kind of test:

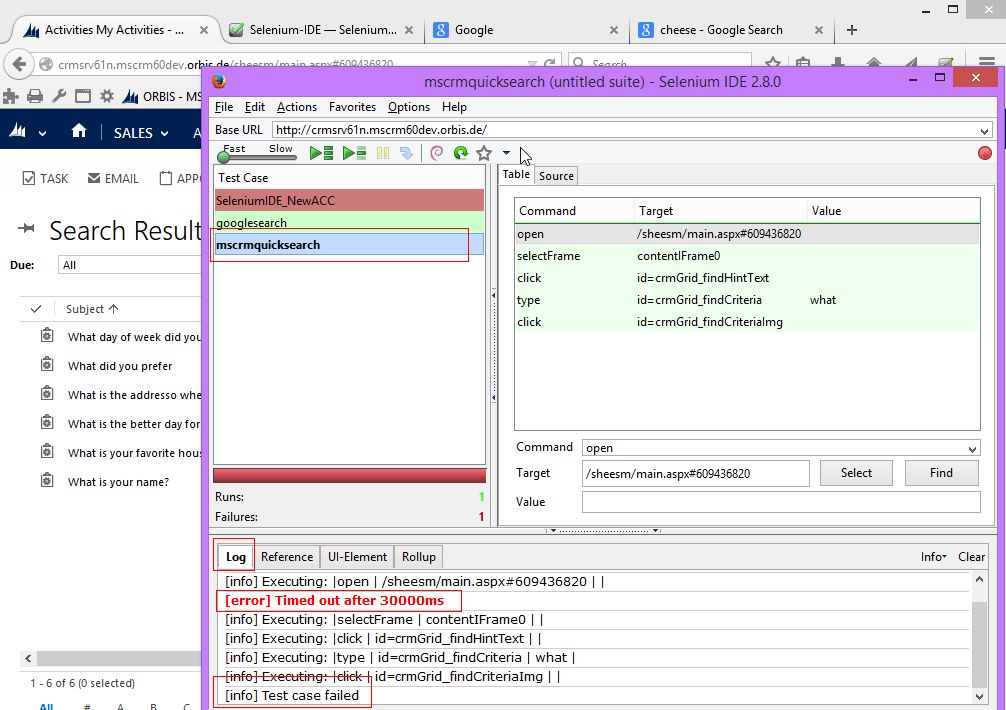
1. recording the action of searching “cheese” on google.com

This kind of test worked perfectly. The Selenium IDE reproduce the action correctly, fast and without errors:

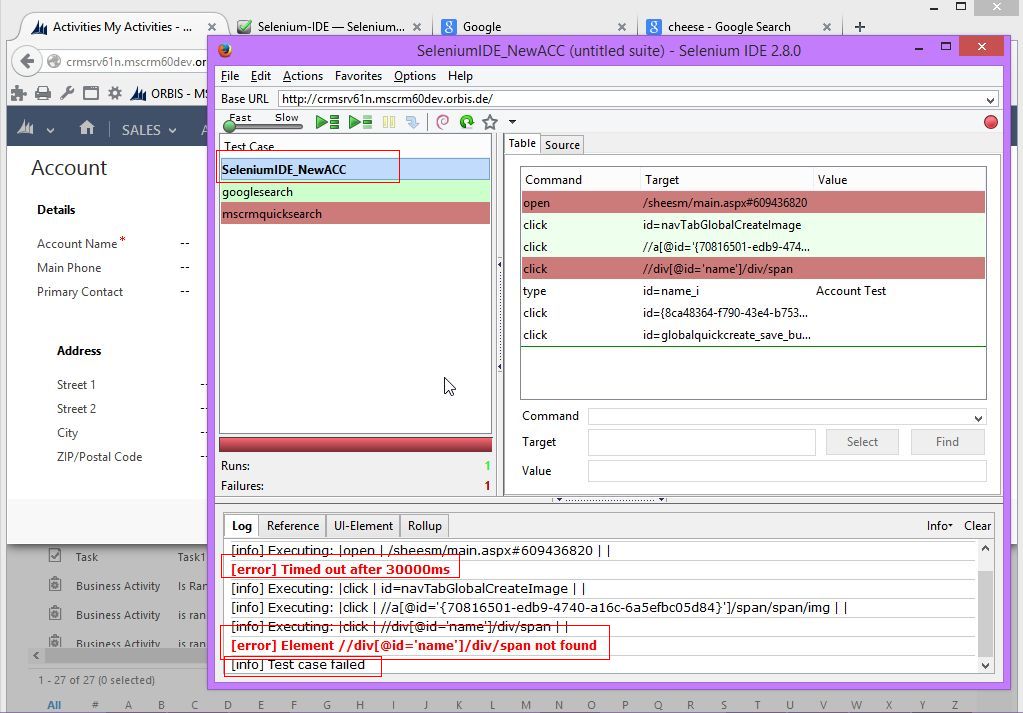


1. recording the action of using “quick find” on MS CRM. For this test case there was used a Microsoft Dynamics® CRM 2013 Service Pack 1 (SP1).

In this case, Selenium IDE try to reproduce all the test recorded but it was really slow and it finished with timeouts errors:



1. Finally, there was tried to record a kind of test some more complex, but in the same time, one of the most common on MS CRM testing: creating a new account through Quick Create. Selenium IDE ***was not able*** to reproduce all the test case and the errors seems to be unjustifiable because the tool was unable to find UI Elements which sure exists in the web page.



#### Conclusions

Selenium IDE is a fast and easy tool to use for recording GUI Test Cases for simple web applications running on Mozilla Firefox browsers. It is free and is easy to install.

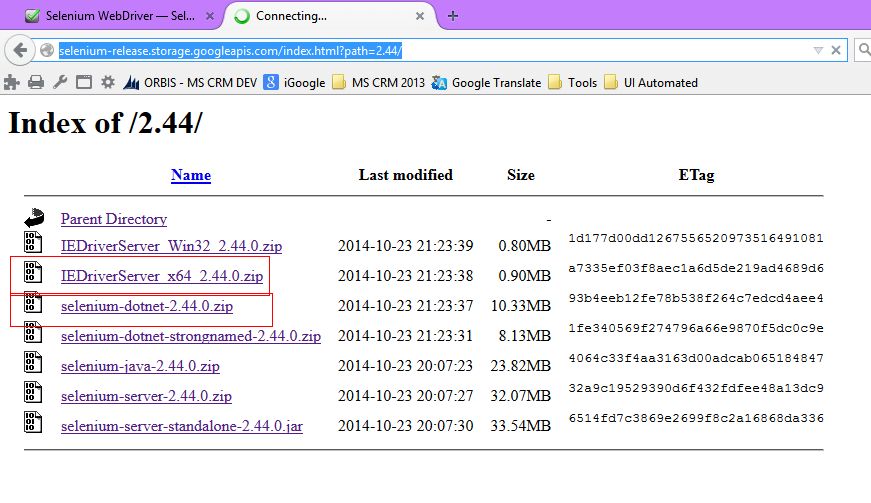
***Selenium IDE is not suggested for the automation of test cases for MS Dynamics CRM web applications.***

## Selenium Web Driver

*WebDriver is designed to provide a simpler, more concise programming interface in addition to addressing some limitations in the Selenium-RC API. Selenium-WebDriver was developed to better support dynamic web pages where elements of a page may change without the page itself being reloaded. WebDriver’s goal is to supply a well-designed object-oriented API that provides improved support for modern advanced web-app testing problems.* (Selenium WebDriver, 2015)

The objective of this section is that of describing the necessary steps for Setting Up a Selenium-WebDriver Project and then use that API’ s to build and run a simple test case on the MS Dynamics CRM Web Application.

**Setting Up the Selenium WebDriver Project** was not immediate and easy. First of all because all the libraries necessary for the IE WebDriver wasn’t available in the same website where is founded the other documentation.



Once founded the folder was sufficient download only:

1-IEDriverServer x64\_2.44.0.zip

2-selenium-dotnet-2.44.0

For this component the tests case made were fundamentally two:

1. The first test consisted in opening IE, navigate to goole.com and search “cheese” as it is suggested by the documentation. So there was developed a simple console application which implement the test:

using System.Text;

using System;

using OpenQA.Selenium;

using OpenQA.Selenium.Firefox;

using OpenQA.Selenium.IE;

using OpenQA.Selenium.Support.UI;

using System.Runtime.InteropServices;

using System.Security.Cryptography;

using System.Threading.Tasks;

using System.Threading;

using System.Collections.Generic;

namespace TestSelenium

{

class GoogleSuggest

{

static void Main(string[] args)

{

IWebDriver driver = new InternetExplorerDriver(@"C:\Users\sheesm\Downloads\IEDriverServer\_x64\_2.44.0\");

driver.Navigate().GoToUrl("http://www.google.com/");

// Find the text input element by its name

IWebElement query = driver.FindElement(By.Name("q"));

query.SendKeys("Cheese");

query.Submit();

WebDriverWait wait = new WebDriverWait(driver, TimeSpan.FromSeconds(10));

wait.Until((d) => { return d.Title.ToLower().StartsWith("cheese"); });

// Should see: "Cheese - Google Search"

System.Console.WriteLine("Page title is: " + driver.Title);

//Close the browser

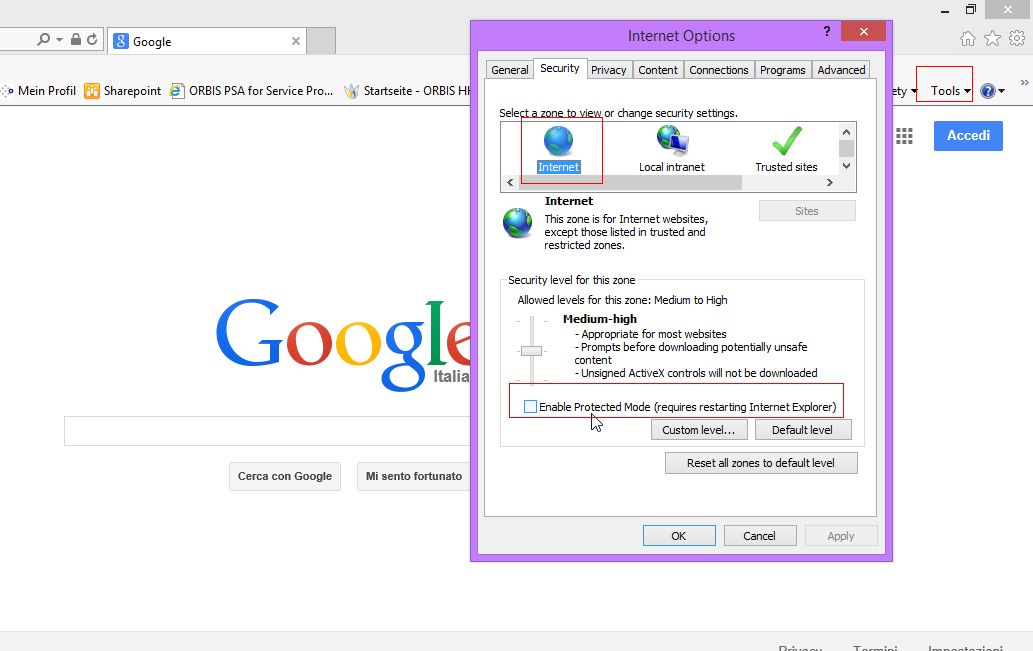
driver.Quit();

}

}

}

When tried to run the console application there were found the first errors which were not well documented. The solution were well described on the web and it consist on putting in an non protected mode *all the zones* in the IE browser:



The work around applied resolved the problem initially, but later, after the operating system made some updates installation, the console application worked no more.

Thanks’ to an expanded community of the selenium users on the web, was possible to find that Selenium Web drive, is incompatible with the windows update installed, the KB3025390 update. Once uninstalled the update, was possible to run the test case with success.

1. The second test case developed consisted on opening from the CRM one business activity, verify it is in an “Open” state and only than set it to “Closed”.

using System.Text;

using System;

using OpenQA.Selenium;

using OpenQA.Selenium.Firefox;

using OpenQA.Selenium.IE;

using OpenQA.Selenium.Support.UI;

using System.Runtime.InteropServices;

using System.Security.Cryptography;

using System.Threading.Tasks;

using System.Threading;

using System.Collections.Generic;

namespace TestSelenium

{

class Program

{

static void Main(string[] args)

{

IWebDriver driver = new InternetExplorerDriver(@"C:\Users\sheesm\Downloads\IEDriverServer\_x64\_2.44.0\");

try

{

driver.Navigate().GoToUrl("http://crmsrv61n.mscrm60dev.orbis.de/");

driver.SwitchTo().Frame("contentIFrame0");

IList<IWebElement> view\_elements = driver.FindElements(By.CssSelector("[id^='gridBodyTable\_primaryField\_']"));

view\_elements[0].SendKeys(Keys.Enter);//open the selected link

driver.SwitchTo().Frame("contentIFrame1");//moves to the new record opened

IWebElement stateCode\_iwe = driver.FindElement(By.Id("statuscode"));

if (stateCode\_iwe.Text == "Open")

{

IWebElement stateCode\_selection = driver.FindElement(By.Id("statuscode\_i"));

var selectElement = new SelectElement(stateCode\_selection);

//select by text

selectElement.SelectByText("Closed");

}

}

catch (Exception ex)

{

System.Console.WriteLine(ex.Message);

}

}

}

}

It resulted to be stressful and slow to search the key of the UI Elements from the web browser, because of the complexly MS CRM pages are. Sometimes elements were immediately founded by CSSClassName, and sometimes their ID was insufficient.

By the way, there was possible to run most of the steps of the test case unless the final step. Setting the state of the business activity to “Closed” generated a generic and unhandled javascript error.

This kind of error resulted to be not documented and furthermore, the web community doesn’t offer more solutions for that error.

#### Conclusions

Selenium Web Drive Component is surely an ambitious project. The APIs are useful, simple and intuitive. The multi-language and infrastructure that it supported give the possibility to create and drive tests case for different browsers and web applications.

But, for MS CRM Web Applications, Selenium Web Drive resulted not compliant because of the complexity of the MS CRM Web pages, and more because of the complexity of the tests needed to develop.

# Ranorex

**Ranorex** is a GUI test automation framework for testing of desktop, web-based and mobile applications. Ranorex does not have a scripting language of its own, instead using standard programming languages such as C# and VB.NET as a base (Wikipedia-Ranorex, 2015)

It supports technologies like Windows Desktop Client Applications, Windows Web Applications, Mobile Apps and runs on Windows platforms, from Win 2003 Server to Win 8.

As the final objective of this document, is finding a tool that permit easy create and manage test case for Microsoft Dynamics CRM Web Application processes, there was defined early a typical test suite, with some typical test case to realize with Ranorex. In this way, there was possible to evaluate the software not only for its features and tools, but also even for its easy to use interface, documentation and support.

The test suite in question consist on the following test cases:

* Enter on CRM
* Create a record from a custom entity
* Update a record of a custom entity, which update fired the creation of another record of a custom entity
* Change the status code of a record of a custom entity, which fired the change status of another record of a custom entity
* Creation of 8 Accounts in hierarchy relationship with each other.

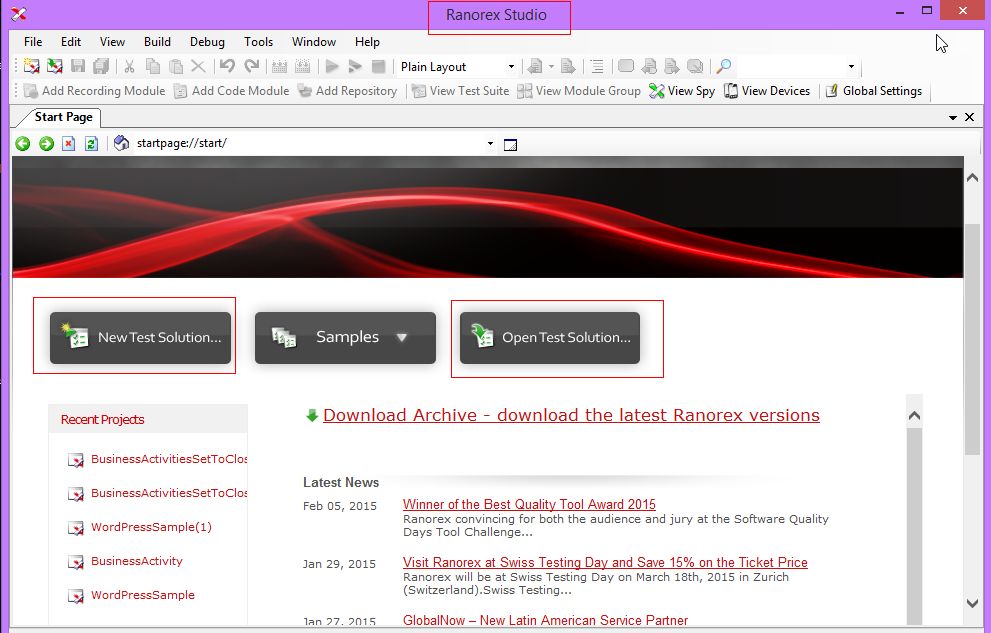
Installing Ranorex Framework, does mean install the set of tools and features that composed it. Each of them is designed for specific situations and needs but at the same time, the results gathering from each one are useable and shareable between the others.

Ranorex is so composed by:

* Ranorex Studio
* Ranorex Recording Module
* Ranorex Spy
* Variables and Data sources – Data Driven Test Case
* User Code Module.

# Ranorex Studio

Ranorex Studio is the main suite, the entry point for building test cases and start working with Ranorex.

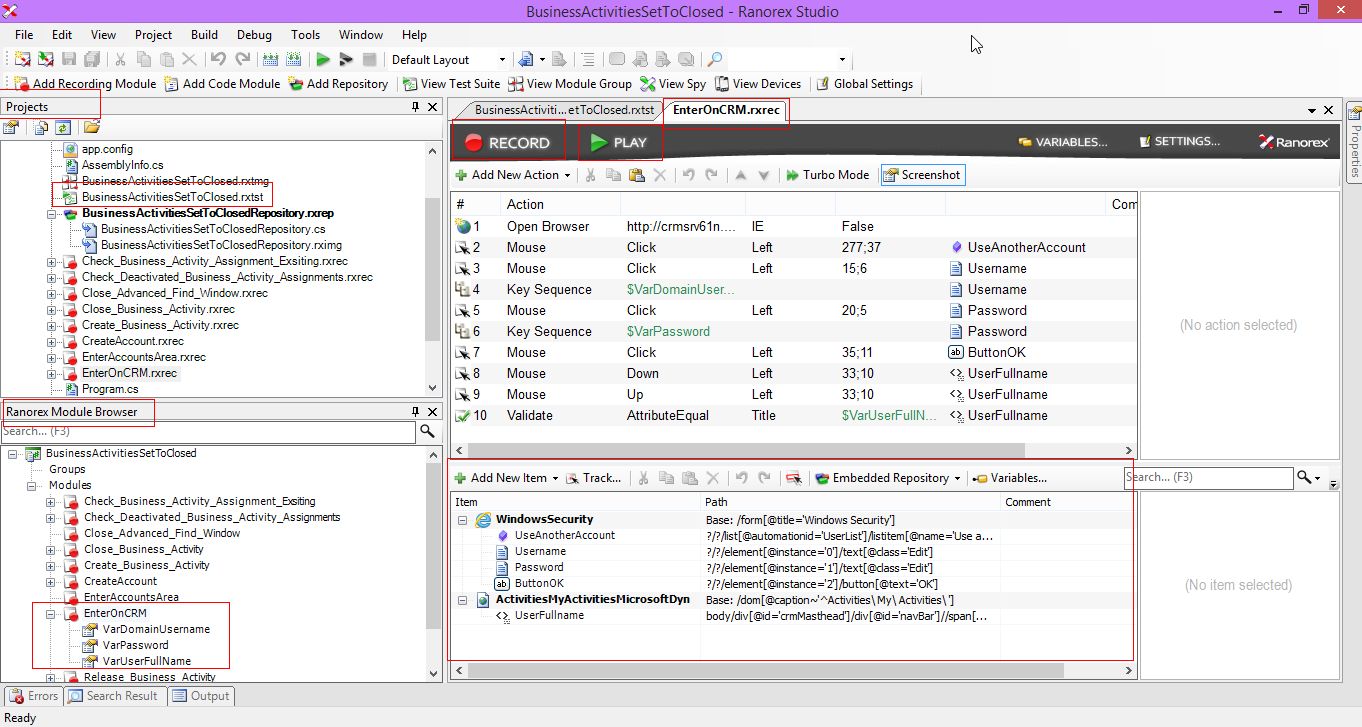


The interface is familiar for those whose normally use MS Visual Studio. The entry point is simple, intuitive and immediate because of the three immediate buttons:

* New Test Solution
* Samples
* Open Test Solutions

which names are self-explanatory.

Opening the Solution, the layout is the same normally founded on the solutions opened with Visual Studio. It is organized by areas, designed for specific scope.

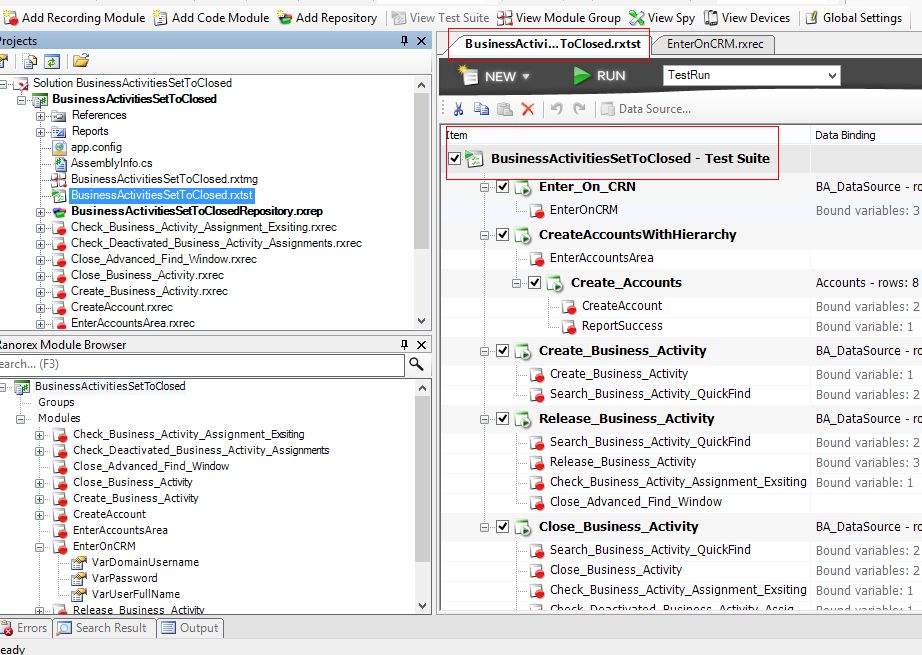


In the ***Projects*** area the user find listed all the files and references currently associated with the projects. The types of files that typically, a Ranorex Studio Solution, can have are:

|  |  |
| --- | --- |
| Test Suite | Represents the project’s test suite (\*.rxtst) |
| Module Group | Represents the project’s module groups (\*.rxtmg) |
| Repository | Used to manage UI elements (\*.rxrep) |
| Recording | Represents an automation module based on capture/replay (\*.rxrec) |
| Code Files | Any type of C# or VB.NET code; typically used to create code based automation modules |

In the ***Module Browser*** area are listed all available modules based on the project’s code files and module groups based on the projects module group file. The view is mainly used to drag and drop and to reuse automation modules and module groups within the test suite view.

When double-clicking a file in the ***’Project View’*** or a module in the ***’Module Browser***’, the associated file will be opened in the ***Studio’s file view***. This view is mainly used for the actions like creating or adapting a recording module (Enter CRM Module Recorder), working with the project’s test suite (BusinessActivitiesSetToClosed.rxtst, test suite solution) and generally all that actions that concern on building a test case.



On Ranorex Studio, there are gathered all the tools and features developed by Ranorex for Automated GUI Tests, so it is sufficient to use it and have easy access of all the specific tool and feature.

#### Conclusions

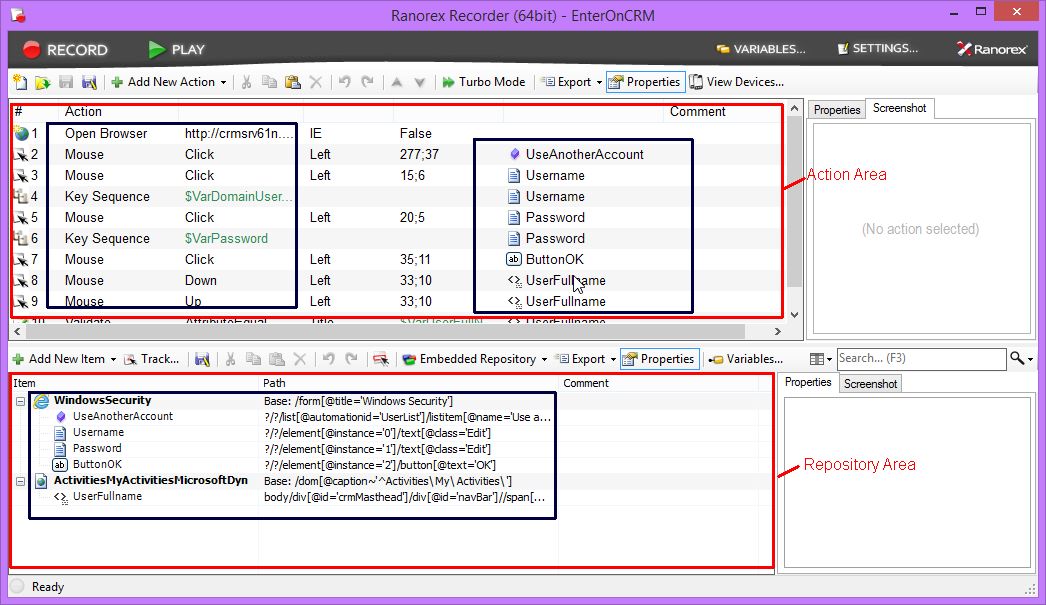
For developing the Test Suite above described, there was used principally Ranorex Studio. Its interface result to be intuitive and immediate. Was not necessary use tutorials of 445 pages, because the name of all the features resulted to be self-explained and the error messages were user friendly and permit to the users to learn from them. ***The tool is surely suggested for building Test Case for MS Dynamics CRM Web Applications.***

Further, it is possible found specific conclusions for the tools and features that composed Ranorex Studio.

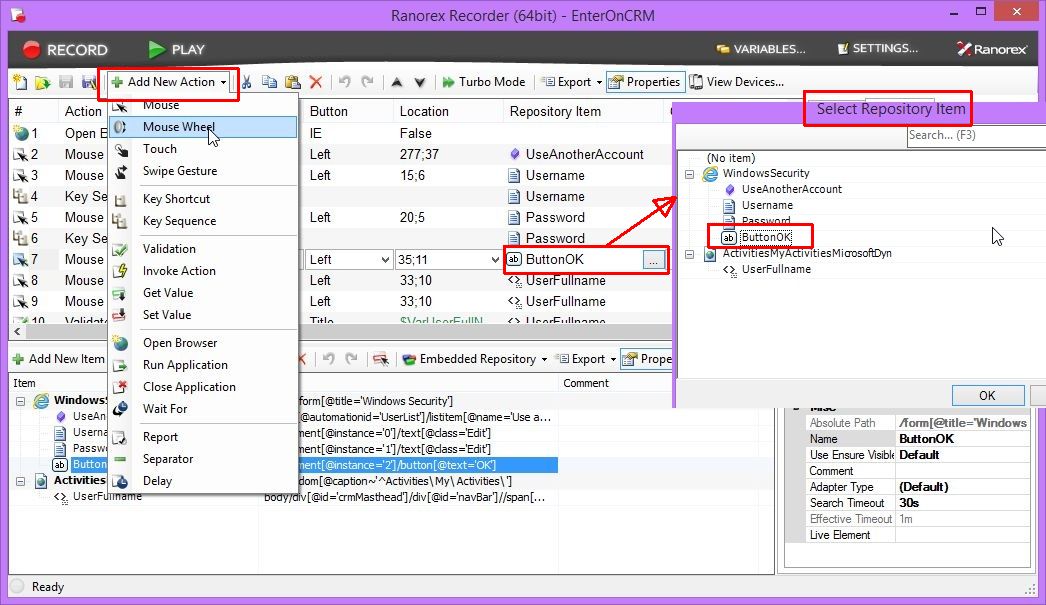
## Ranorex Recording Module

**Ranorex Recording Module** is the most useful tool of Ranorex Framework. It permit to register all the mouse and keyboards action, and reproduce them in the same way.

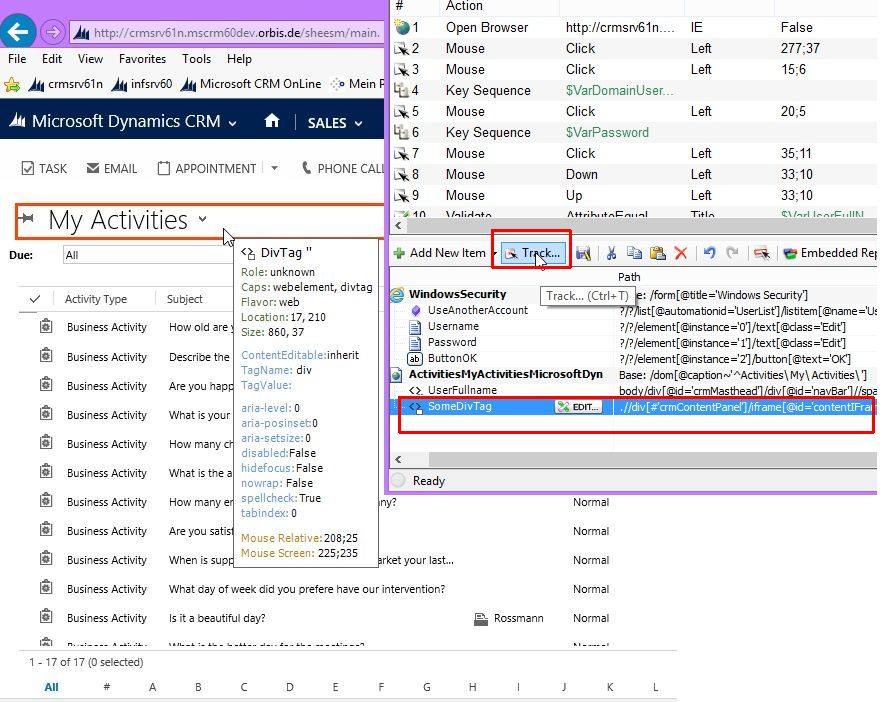
Recording the actions does mean for Ranorex creates two kind of re-usable and updateable elements: Actions and the Repository with the UI Web Elements:



The actions are self-explained and there is possible add other new Actions or updated them and the UI Element where to be executed.



The repository area contains all the UI Elements, tracked during the recording, with the name element and its XPATH. It is possible to add other elements to the repository and re-use all them in other part of the entire solution. Result simple to add elements to the repository because is sufficient use the “Track” features and Ranorex build automatically the XPATH of the tracked element. The Ranorex Spy tool, here later discussed, manages the repository feature.

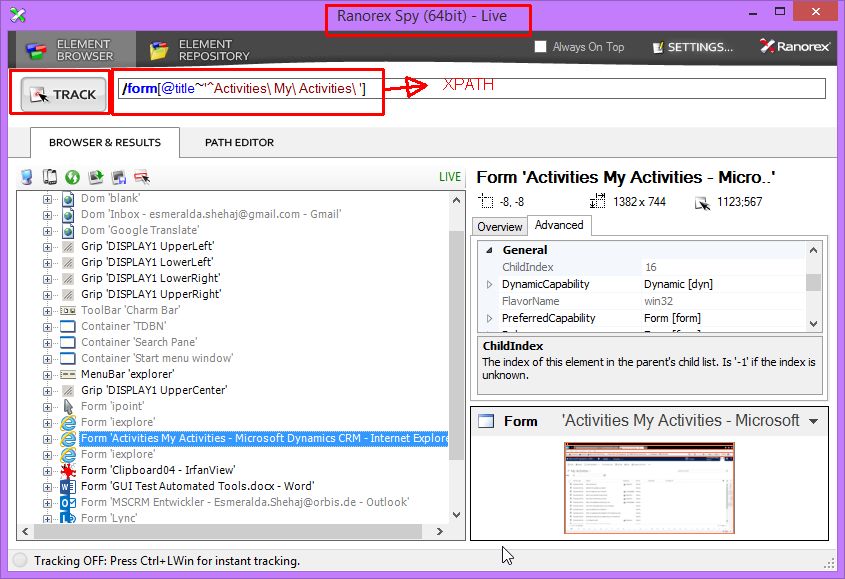


#### Conclusions

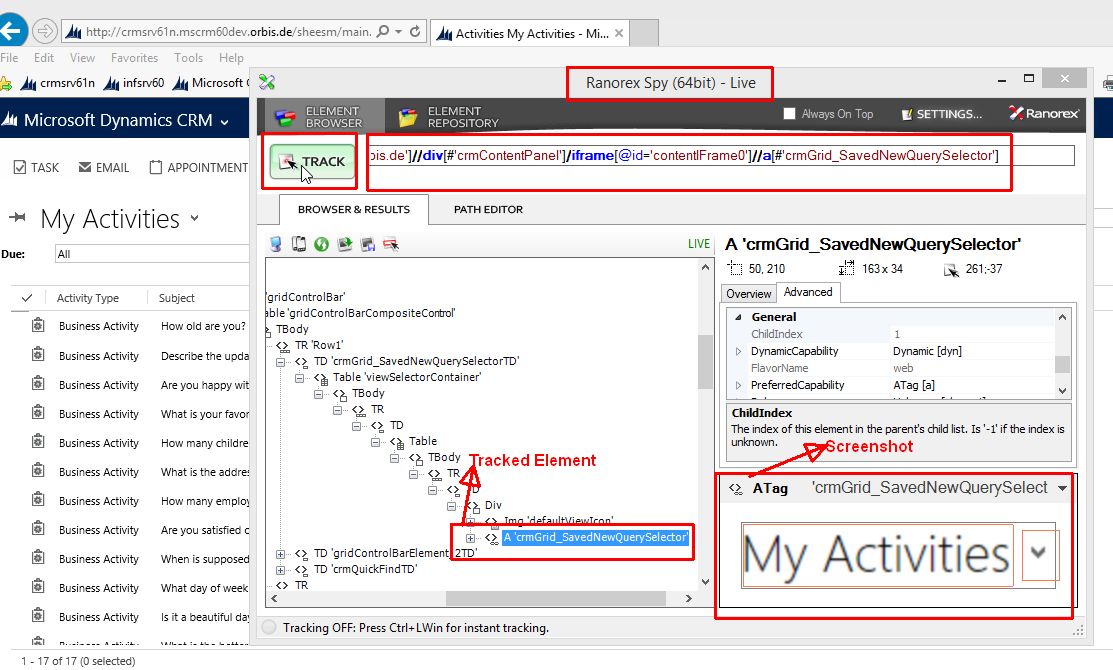
The interface of the Recording Module is clean, user friendly and easy to use. It is also possible to move elements from one area to another using directly drag and drop. The “Action” concept, permit to non-programmer user to program and build their test cases. In the same time, ease and simplicity not make the instrument less powerful.

## Ranorex Spy

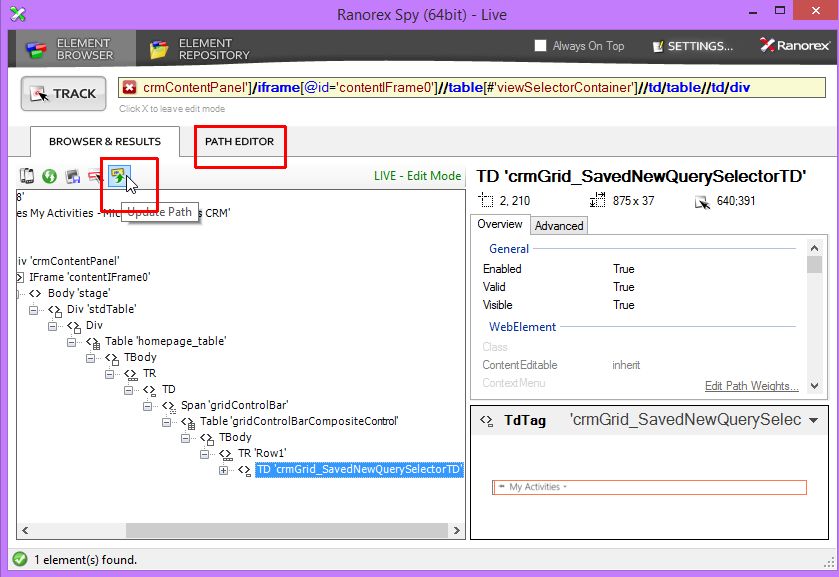
**Ranorex Spy** is the Ranorex integrated tool that permit to identify through their XPATH, all the elements present in all the opened windows in the machine it is running:



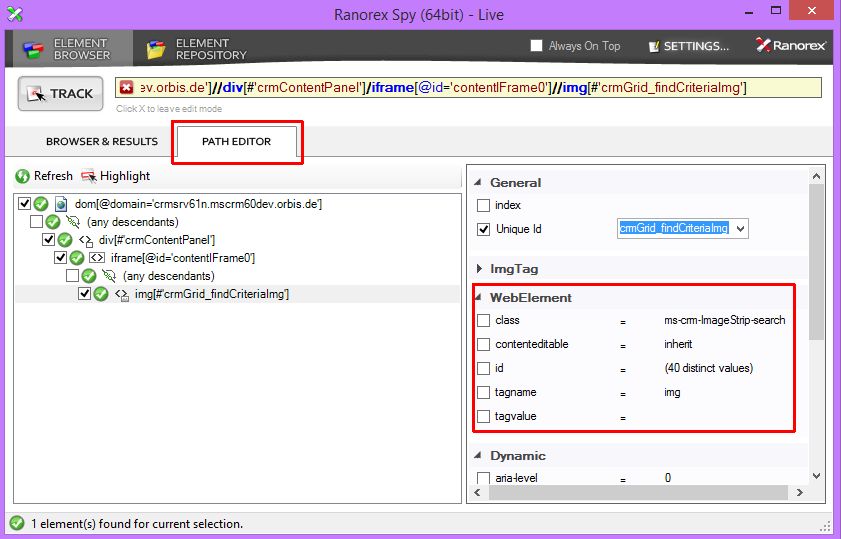
There is possible, using the “Track” button, to easy identify all the necessary elements. Ranorex Spy get not only its XPATH but even a screenshot image of the element making it easier to immediately realize to which element corresponds the selected XPATH.



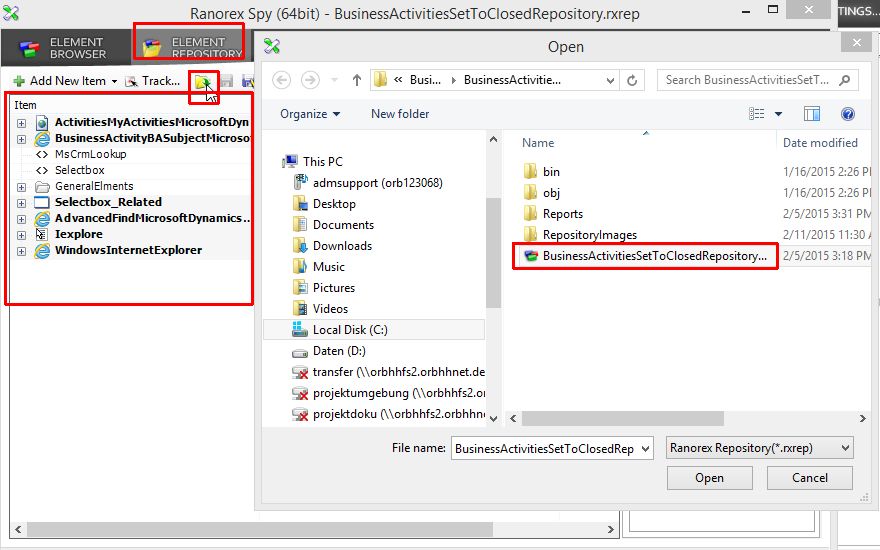
Ranorex Spy is not only a tracker and identifier of the UI Elements. It is also possible to update the XPATH generated, modifying the query elements, through the “Path Editor” area, or the button “Update the path”:



Through the Path Editor feature, is possible to change the query attributes use to identify the elements in case is necessary to found some unique and mostly relative path:



With Ranorex Spy is possible to define new Repository or updated others, so to use them in different Test Solution:



#### Conclusions

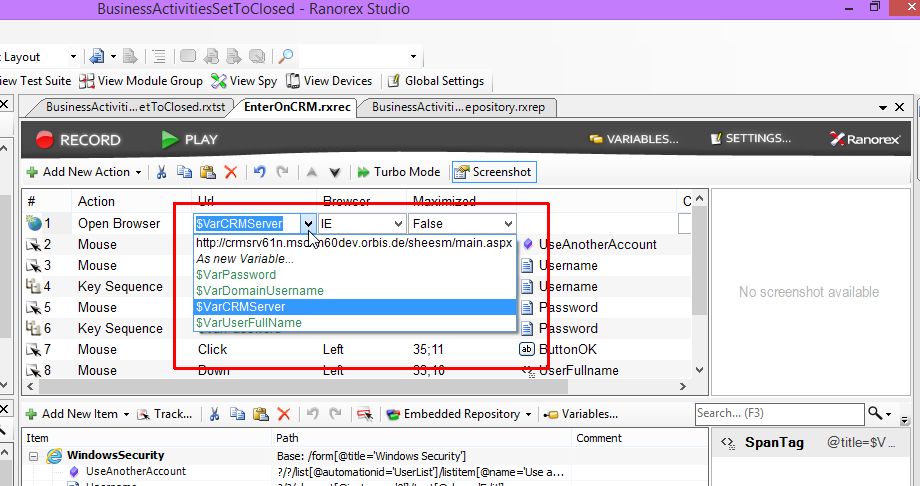
Ranorex Spy seems to be one useful tool for GUI Automated Tested, because its user-friendly interface. Its objective is find and identify univocally UI Elements and it do it with simply features, clear messages which naturally drive the users on how to use it.

## Variables and Data sources – Data Driven Test Case

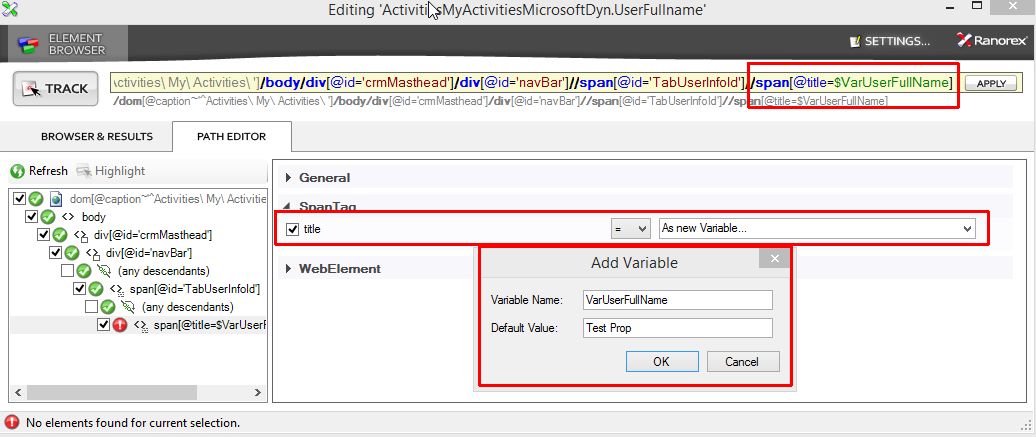
Building the Test Suite above described, there was sometimes necessary define some variables because of dynamic values or because re usability.

Ranorex makes possible define Data Driven Test, because the use of variables and external Data Source. Every time some action consists on data entering, is possible use some variables, which can be binding with external information.

Variables can be defined for the actions:

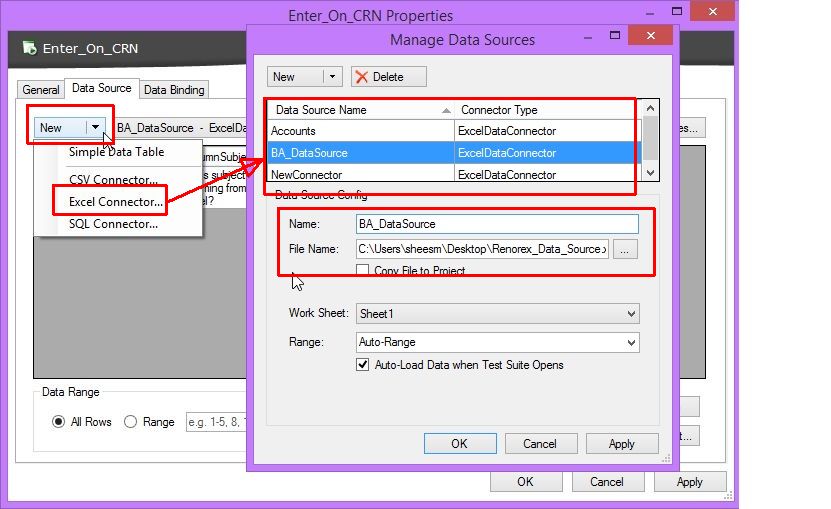


Also it is possible define variables for the UI Elements, on its XPATH for example:



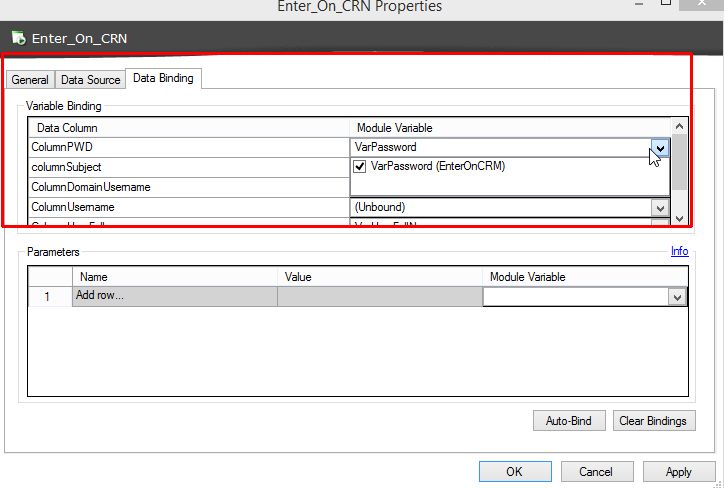
Running the specific test case, it is necessary set some default values. The external values for variables are definable on the Test Suit Area, where the specific test case must be added.

First must be create/uploaded the Data Source:



For each Test Suite, there is possible define more than one Data Source. This permit a better organization of all the necessary external values needed.

Once the Data Source is upload it is possible bind all the variables early defined:

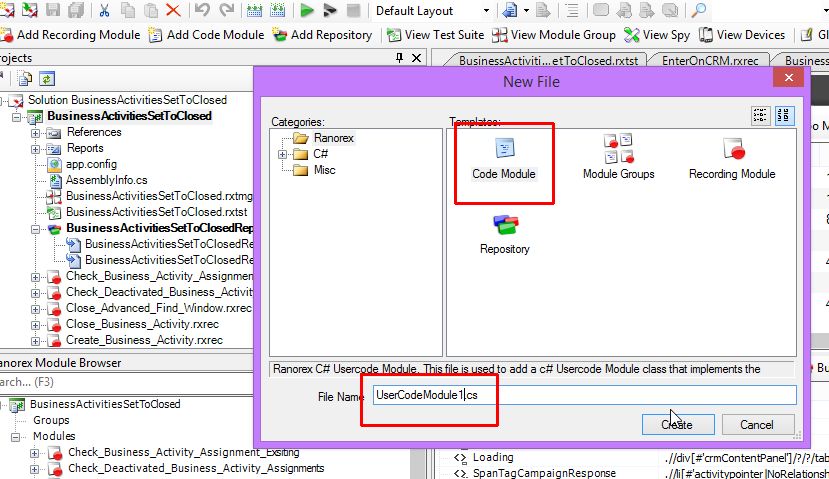


#### Conclusions:

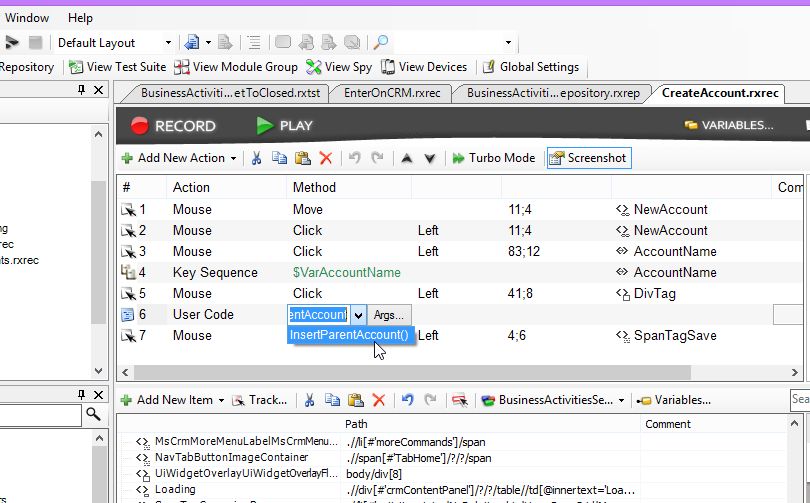
Data Drive Test Cases permit to define test cases with a bigger range of applicability, more re-usable, not only in the same test suite solution, but even on different solutions and environments. Ranorex offers useful features which easy drive users to make use of them in way to build robust and re usable tests case.

## User Code Module

Building the Test Solution here above described, was necessary managed some action which was result of some decision. It is hard working and difficult realize this kind of operation using only the recording module and its *Actions.* With Ranorex is possible write entire test cases with code programming, using as C# as the .net framework from 2.0 to the newer ones.

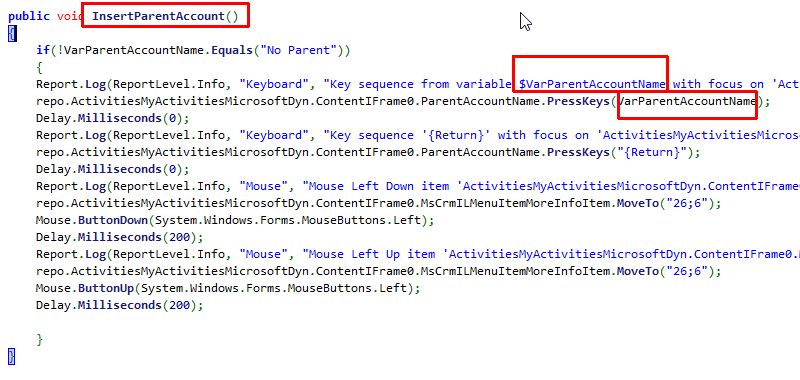


Furthermore, it is possible too to combine Actions with User Code Actions so to better exploit all the Ranorex Potential:



In the specific case, the necessity was that to create different accounts in MS CRM with the hierarchy defined on the Parent Account. The accounts are in a Data Source made by two columns with the account name and the parent account name.

Through the user code there was possible to program the entry value of the parent account field, only if that value is available in the data source:



#### Conclusions

***User Code Module*** add more potentiality to Ranorex, which result to be really full of tools and features. The user code module, for normal test cases is not so necessary, but it become fundamental for test cases made with actions result of logical and higher level decisions.

## Ranorex Conclusions

Ranorex has been tested and used, for quite a month for realizing test cases that represents the 80% and more of all the tests cases normally is make.

It result to be immediately a very powerful tool for Automated GUI Testing. It is composed by different tools and features which perfectly runs even as standalone tools. This permit to create different figures of specialists on building automated test case, which can work alone without hindering each other, and creates so redistributable and reusable elements (repository, user code module, test cases and so on).

Ranorex interface is user friendly, self-explanatory which drive the user on how to better and faster use it.

In Addition to, here must say, that the structure and way Ranorex is made helps to define test cases redistributable and reusable, even for that user which have not much experience on test case building.

# References

*Ranorex*. (2015, 2). Retrieved from Ranorex: http://www.ranorex.com/test-automation-tools/ranorex-studio-automated-testing-software.html

*Selenium IDE*. (2015, 01). Retrieved from Selenium: http://www.seleniumhq.org/docs/02\_selenium\_ide.jsp

*Selenium WebDriver*. (2015, 1). Retrieved from http://www.seleniumhq.org/docs/03\_webdriver.jsp

*Wikipedia*. (2015, 01). Retrieved from Selenium (Software): http://en.wikipedia.org/wiki/Selenium\_%28software%29

*Wikipedia-Ranorex*. (2015, 02). Retrieved from Wikipedia: http://en.wikipedia.org/wiki/Ranorex

.