

# Localization of Microsoft™ .NET Applications

**A Guide to using Alchemy CATALYST 4.0 to accelerate revenue growth and reduce localization costs.**

---

## **Abstract**

The introduction of Microsoft™ .NET has created new and exciting challenges for the LSP (Localization Service Provider) community and has redefined the technologies and the process of localizing internet/wireless/mobile and desktop applications.

Existing localization technologies that provide environments for the localization of Microsoft™ Window application can no longer be used on these newer generation .NET applications. The build, translate and validate process used by software publishers is also undergoing a remarkable change as the development community embraces the Microsoft .NET computing paradigm.

Gone are the now familiar and easily translated .RC files. These have been replaced with the more versatile yet more complex .RESX files. Using XML as a core technology, while Microsoft are expounding the virtues of this new open-standard, the LSP communities are faced with the difficult challenge of re-tooling and re-educating their localizers. They are challenged with maintaining the efficiency and velocity in which they had previously translated Desktop Applications while grappling with a new technology and localization paradigm.

Alchemy Software Development, the market leader in visual localization environments has just introduced a .NET Visual Component that allows translators and localizers work in the now familiar and the highly visual environment of Alchemy CATALYST. Instead of focusing on the complexities of .NET file formats, localizers and translators can now focus on the quality and precision of their translations, improving their work efficiency and reducing the cost of translating .NET applications.

Initial tests carried out by customers have demonstrated that using the Alchemy .NET Visual Localization Component improves the speed and efficiency of translating .NET applications dramatically, improving quality, reducing costs and making simultaneous release of foreign language applications achievable



*Copyright© 2002 Alchemy Software Development Limited. All rights reserved.*

*The accompanying documentation is the property of Alchemy Software Development Limited and are copyrighted. Any reproduction in whole or in part is strictly prohibited.*

*Microsoft, Visual C++, Windows and Windows NT are either registered trademarks or trademarks of Microsoft Corporation in the U.S.A. and/or other countries.*

*TMX® is a registered trademark of the Localization Industry Standards Association.*

*All other product names or trademarks are the property of their respective owners.*

---

## CONTENTS

INTRODUCTION.....	4
XML Technology	4
WORKFLOW : MICROSOFT™ WINDOW APPLICATIONS.....	5
WORKFLOW : MICROSOFT™ .NET APPLICATIONS .....	6
Resource Generation Process	6
Application Culture Information	7
DEVELOPING A .NET APPLICATION .....	8
Creating a Dialog Form Resource	8
Creating Multiple Resource Files	9
Language .RESX Files	10
Selecting the UI Language	11
In summary	12
LOCALIZING .NET APPLICATION : THE ALCHEMY CATALYST WAY ....	14
Component Plug-In Technology	14
Loading and Viewing .RESX Files	14
Full XML Editing Environment	15
Reducing your costs using Alchemy' Leverage Expert	16
Reducing Engineering Costs and Time	16
The Complete Solution for .NET Applications	17
CONCLUSION .....	18
ABOUT ALCHEMY SOFTWARE DEVELOPMENT LIMITED.....	19

---

## INTRODUCTION

The development paradigm has changed! With the introduction of Microsoft .NET technology, building and deploying Internet and mobile applications has become easier and less complex. Using one technology framework, an easy to learn programming language, leveraging open standards such as XML and SOAP, and integrating it all into one seamless development environment, Microsoft has released its most ambitious development technology yet.

While fundamental approaches to application design remain somewhat consistent with the approach traditionally chosen by Desktop Application developers, the LSP (Localization Service Provider) community face a daunting challenge of upskilling and retooling their localization teams while embracing this new Microsoft technology. Coming to grips with the new open standards and learning the nuances of translating .NET technology will present both a financial and educational challenge.

But given the new complexity of the .NET localization process, will their clients be more willing to forgive slower translation and engineering or higher translation costs? Highly unlikely, so the LSP that learns the quickest and establishes itself as the experts in .NET technology will gain significant competitive advantage and strengthen its market position.

Alchemy Software Development has developed a .NET Visual Localization Component for the market leader in integrated localization environment, Alchemy CATALYST. It's the world's first visual localization for .NET Applications and provides visual translation tools to help translators and localizers focus on the quality and precision of their translations rather than the complexities of the .NET technology.

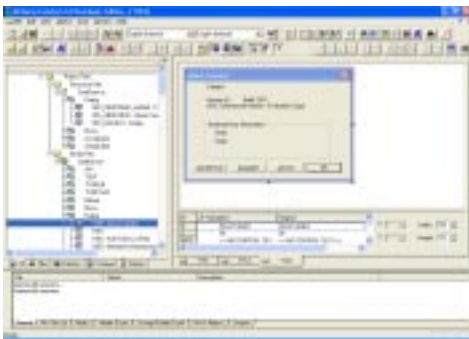
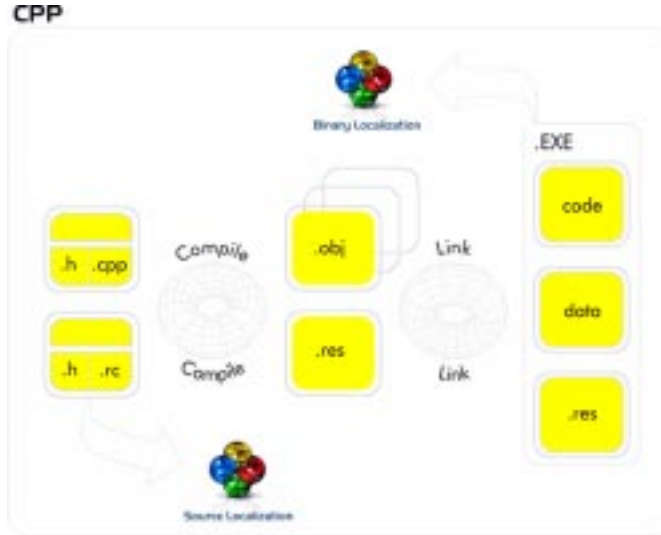
### **XML Technology**

At the core of the Microsoft .NET framework is XML, an open Standard supported by almost all platforms and all major IT players such as IBM, LOTUS, Sun Microsystems, Oracle and Novell. It offers the ability to store data easily and describe the relationships between this data in a common, platform-independent vocabulary. Microsoft has deployed XML as their replacement for the ubiquitous Window resource file formats (RC, DLG) for Dialogs, Menus and Strings. These fundamental UI components are now described in a new XML document referred to as .RESX document.

Building on its robust and superior XML technology, introduced into Alchemy CATALYST 4.0, Alchemy Software Development have been able to provide a full XML editing environment coupled with a visual translation environment for these new Microsoft XML file formats. This ensures faster translation, greater consistency through the re-use of previous translations and glossary management tools and reduced localization costs.

**WORKFLOW :  
MICROSOFT™ WINDOW  
APPLICATIONS**

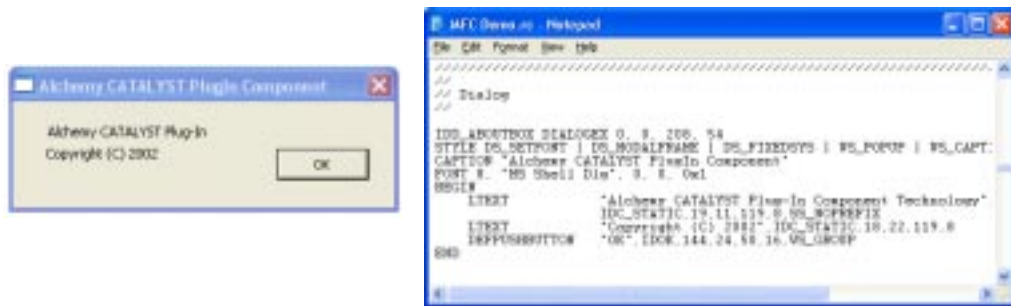
The development and translation process for Microsoft™ Window applications is well known and understood in the localization industry. While some organisations prefer to localize resource files and maintain their translations in eye-readable source code formats, most organizations now opt to translate the Windows binary DLLs or EXEs. This speeds up the translation and localization process, reducing their costs and improving their time-to-market for language products.



Screenshot 1 :Equally at home translating RC and Binary Files.

The most significant engineering benefit derived from translating binary application files is the reduction in project complexity due to the vast reduction of files that need to be processed. Whereas one resource binary file may be made up of 500 smaller resource files, the engineer only has to work on this one binary file. This enormous reduction in the number of files reduces the complexity of managing multiple language translations. Imagine the difference between processing 10 binary DLLs or their equivalent 5,000 source files!

Regardless of the approach development organizations take, Alchemy CATALYST provides a completely integrated solution for the translation of source files, binary files and ancillary program files such as XML documents, RTF help files and Installshield Installer files.

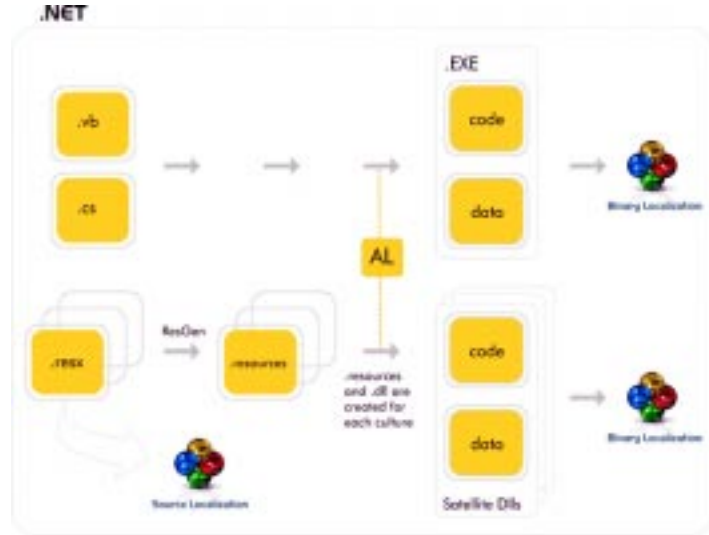


RC Representation of a simple dialog box.

**WORKFLOW :  
MICROSOFT™ .NET  
APPLICATIONS**

The localization process for Microsoft™.NET applications is not well known and poses significant challenges to the localization service providers.

While the Microsoft™ .NET Framework introduces new technologies making the implementation of globalization easier from a development point of view, this technology also redefines the localization workflows.

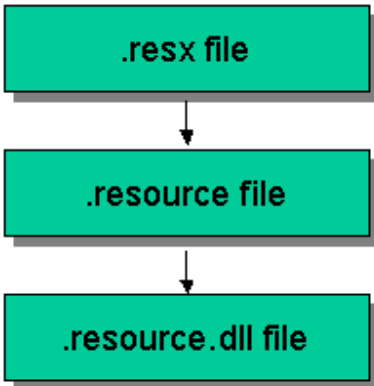


Microsoft™ .NET Applications can be created using Visual Basic or C#. Application resources are defined in a new file type called .RESX. These are XML container documents that describe the elements of the application user interface such as menus, dialogs and strings. There is one .RESX file for each resource type.

**Resource Generation Process**

To create embedded resource files, Microsoft VisualStudio .NET follows two relatively simple steps:-

- (1) A .resx file that contains XML-based description of the application resources is compiled into a .resource file by using the ResGen tool. This tool comes as part of the Microsoft .NET Framework SDK. The .resource file created contains binary resources only.
- (2) The .resource file is embedded into an assembly <sup>1</sup>using either the assembly linker tool or language compiler (such as CSC for the C# Compiler.)



<sup>1</sup> Assembly is the technical term used by Microsoft to describe the combination of application code and application resources that are combined together to form a single .NET application.

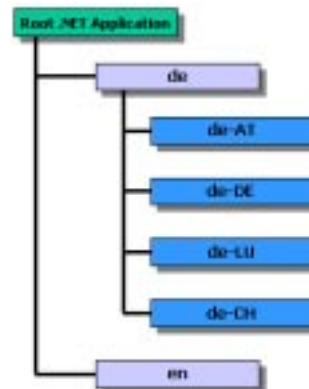
## Application Culture Information

An application culture represents information about specific locales including the writing system, collation system, calendar used, currency, date and time formats etc. The .NET Framework provides a `CultureInfo` class to provide easy programmatic access to these language and text processing fundamentals.

Within this `CultureInfo` class, the `UICulture` data member determines how the application resources will be loaded. In effect this data member specifies the language of the user interface of a .NET application. By changing this data member, the developer can switch the UI language of his .NET application.

By default, if the developer does not set his/her `UICulture` value, a .NET application will set it to the system-installed language, which is the language of the operating system's resources.

Once the `UICulture` is set, a .NET application will load and display the specified language user interface. It will locate the language files for the user interface of a .NET application by loading satellite resource DLLs. Using the RFC 1766 standard, these satellite resource DLLs are stored in a pre-determined directory structure off the root from which the .NET application is launched.



Arrangement of Satellite directory structure containing .NET application resource DLLs.

By enforcing this directory structure, Microsoft has developed a mechanism to have multi-lingual applications reside on servers. The application user interface will be set at runtime when the .NET application requests information regarding the `UICulture` of a client machine.

Using this mechanism, it is possible for an application to reside on a server and different client machines to launch this application using unique and different language User Interfaces. And remember you get this multi-lingual application without having to write one line of code, it's part and parcel of the .NET Framework!

Let's now take a closer look at the Microsoft VisualStudio IDE and how it supports building and deploying .NET applications.

### Creating a Dialog Form Resource

As a dialog form (or Winform) is created, a resource RESX file is created within Microsoft Visual Studio .NET.

```
private void InitializeComponent()
{
    this.button1 = new System.Windows.Forms.Button();
    this.SuspendLayout();
    // button1
    //
    this.button1.Location = new System.Drawing.Point(144, 136);
    this.button1.Name = "button1";
    this.button1.Size = new System.Drawing.Size(136, 40);
    this.button1.TabIndex = 0;
    this.button1.Text = "OK";
    //
    // Form2
    //
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.None;
    this.ClientSize = new System.Drawing.Size(292, 266);
    this.Controls.AddRange(new System.Windows.Forms.Control[] { this.button1 });
    this.Name = "Form2";
    this.Text = "Form2";
    this.ResumeLayout(false);
}
```

Figure: C# Source file with embedded text for a button on a dialog panel. (Embedded text is highlighted)

However the text strings of this Winform are not stored in the RESX file! They are stored in the method **InitializeComponent** within a C# code file. This code is then compiled into an .NET application binary file.

Obviously, this does not help the localization process, as the text strings are essentially 'hard-coded' within the application's data segments. To efficiently translate this WinForm we need to use a method that externalizes the text strings into an external file. This file then can be translated without the need to edit or recompile the C# or VB source code.

## Specifying a resource for localization

Luckily Microsoft VisualStudio .NET comes with a feature to automatically create these externalized resource files. The developer must mark all WinForms as being **Localizable**. This instructs the Microsoft .NET IDE to extract the text and other UI elements of the application and store them in external RESX files.



Figure 1: Marking a Winform as Localizable

```
private void InitializeComponent()
{
    System.Resources.ResourceManager resources = new System.Resources.ResourceManager(typeof(Form2));
    this.button1 = new System.Windows.Forms.Button();
    this.SuspendLayout();
    // 
    // button1
    // 
    this.button1.Name = "button1";
    this.button1.Size = ((System.Drawing.Size) (resources.GetObject("button1.Size")));
    this.button1.Text = resources.GetString("button1.Text");
    this.button1.TextAlign = ((System.Drawing.ContentAlignment) (resources.GetObject("button1.TextAlign")));
    // 
    // Form2
    // 
    this.AutoScaleBaseSize = ((System.Drawing.Size) (resources.GetObject("this.AutoScaleBaseSize")));
    this.ClientSize = ((System.Drawing.Size) (resources.GetObject("this.ClientSize")));
    this.Controls.AddRange(new System.Windows.Forms.Control[] {this.button1});
    this.Name = "Form2";
    this.Text = resources.GetString("this.Text");

    this.ResumeLayout(false);
}
```

Figure : C# method that extracts text from external resource file. (RESX) (Statement that extracts strings from external resource file is highlighted.)

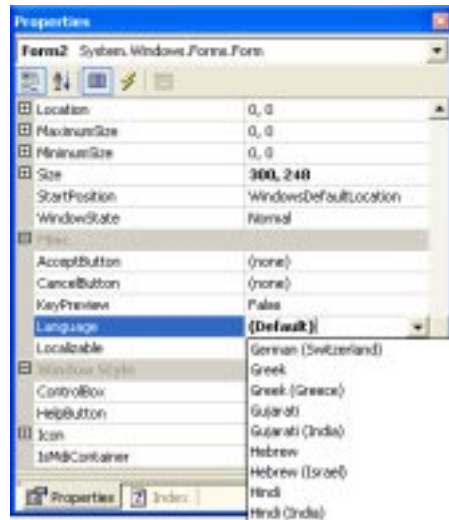
VisualStudio .NET provides the same mechanism for storing external resources, for both C# and VB .NET, so the localization of any application developed using these programming languages should be symmetrical.

If you compare the two code fragments above, you see the difference in the C# code once you change the **Localizable** property of a WinForm. In the first instance the text strings are hard-coded into the C# code. This makes localization difficult and costly. In the second instance, the C# code is modified and uses the C# methods of extracting strings from an external resource file.

### Creating Multiple Resource Files

For each localizable WinForm, a new .RESX file is created. So for example, if you specify that a Winform is to be translated into French and German, Microsoft VisualStudio .NET will create two additional .RESX file for these languages.

To see this in action, simply click on a WinForm and change its **Localizable** property to **TRUE**. Then select the language you wish this WinForm to be localized into.



For each language that you select, a new RESX file will be created by the Microsoft IDE. They are given a pre-determined file name based on the name of the original base RESX file with a language code appended to it. So if the original RESX file was called MainForm.RESX, the German variant of this would be named MainForm\_de.RESX, the French variant of this would be named MainForm\_fr.RESX and so on.

While Microsoft VisualStudio .NET creates all these variant RESX files for the developer, it should be pointed out that this process has to be repeated for each WinForm in an application. With large applications it is not unreasonable to anticipate that this process will prove to be cumbersome and time consuming. Imagine the overhead for a developer if he has to manage 500 base RESX files with 20 variants for each file within the one application! A whopping 10,000 files need to be maintained!

### Language .RESX Files

These language .RESX files are highly optimized to reduce storage space and transmission time over the internet. These files only contain the changes, or deltas, between the base RESX and their language counterpart. While this may seem elegant in terms of economy of disk space, in reality this poses a significant challenge for the localization teams, as each language RESX file is initially created with no content! This means that they cannot be localized without references being made to the original or invariant resource file created by Microsoft VisualStudio.

To see all these invariant and language .RESX files within the Microsoft IDE, click on the **Show All Files** icon to see these file in the Solution Explorer tool bar.



---

## In summary

Undoubtedly, Microsoft .NET provides a very powerful way to develop multilingual applications at the design stage. They have also provided an environment that provides minimal tools for localizing .NET applications within the Microsoft VisualStudio IDE. However, the process soon gets unwieldy when many forms with many controls are required in different languages. It is virtually impossible to manage and localize any kind of large program in this manner. The burden on the developer in managing invariant RESX files and multiple-assemblies is immense and would seriously distract him for developing the main application.

The limitations in this process can be summarized as follows:-

- 1. Incremental resource files cannot be translated directly**

Since the language RESX files and the multiple Satellite Assemblies are not complete copies of the full resources used by an application, they cannot be translated directly. This lack of complete content makes them almost redundant in terms of a viable localization process.

- 2. Complexity inherent in the .NET localization process**

Since each WinForm requires an individual RESX file and each target language requires a variation of this file; managing applications with a large number of Winforms in many languages becomes impossible.

In addition to this, consider the development overhead in maintaining several language variants. The work involved in keeping these in-sync with others would be unmanageable even in a medium sized application.

- 3. No Version Control**

Microsoft VisualStudio .NET provides no version control to help fold back changes into the main project file of the translations and new layouts of language variant RESX files. This has to be done manually and consequently will be prone to a high number of errors and repeated quality assurance problems.

In addition to this, if changes are made to the base application resources, they will not be replicated in the translated variant files automatically. This task will have to be carried out manually, and in most cases will be almost impossible.

This lack of version control makes parallel localization of an application during its development phase impractical and unmanageable even in the smallest of applications.

---

#### **4. New Binary Format excludes current localization technology**

Microsoft .NET applications do not store their resources in standard resource segments as previously used by Win32 desktop applications. This means that all existing translation tools and technologies currently available in the market cannot handle the translation of .NET applications, forcing the development and LSP community to invest in upskilling and retooling their organizations.

In the next section, we'll take a close look at how, Alchemy CATALYST with its robust XML technology, can provide the only viable solution for the translation, engineering and testing of .NET Applications.

LOCALIZING .NET  
APPLICATION: THE  
ALCHEMY CATALYST WAY

Since the introduction of Alchemy CATALYST 4.0, Alchemy Software Development has gained a reputation as the leading localization environment for XML based applications and file formats. The environment used to develop the .NET Visual Localization Component is based on this superior XML technology and provides the only viable solution to the translation, engineering and translation of Microsoft .NET Applications.

### Component Plug-In Technology

As pioneers and market leaders in localization technology, they set about developing a visual environment for the translation of .NET applications that would share many of the benefits of their already firmly established and robust Windows application translation environment. The result was the development of a Plug-in Component for the Alchemy CATALYST 4.0 Integrated Localization Environment.

This component plug-in fits into the existing Alchemy CATALYST 4.0 framework, so that customers' existing investment in Alchemy technology is maintained. This .NET Visual Component will be a familiar environment for professional translators, localizers and testers used to working within the Alchemy CATALYST 4.0 environment, so re-training and re-tooling costs are kept to an absolute minimum.

### Loading and Viewing .RESX Files

Alchemy CATALYST 4.0 recognizes RESX documents and loads them automatically into its localization environment displaying the contents of the file in one of its plug-in visual editors. In the example below, a Winform is loaded into the WinForm editor.



Loading RESX files is automatic and displays .NET resources in WYSIWYG editors

Once a RESX document is loaded, it will be parsed using the Alchemy XML engine.

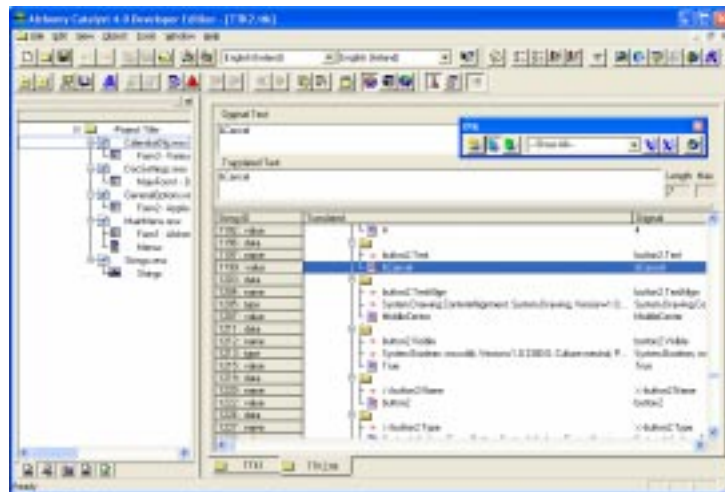
This engine handles all the XML character encoding, entities and validation of the data content according to the XML schema that Microsoft embeds in the header section of RESX documents.

Once the RESX document is parsed completely, a logic layer within the .NET Visual Component interprets the XML data stream and builds a visual representation of the RESX resources within the Alchemy CATALYST 4.0 environment. This is then displayed in one of the .NET WYSIWYG editors.

Providing an intuitive and WYSIWYG environment for RESX documents ensures that translators can focus on the precision and accuracy of their translations without needing to know the underlying XML formatting information and engineers and testers can fix layout and cosmetic localization bugs in an already familiar WYSIWYG environment<sup>2</sup>.

### Full XML Editing Environment

The Localization Engineer is always in complete control of the RESX document. If any element or attribute of the RESX document needs to be changed or altered, the Engineer can use the fully interactive and secure XML editing environment that comes as part of the Alchemy CATALYST 4.0 environment.



Total Control for the Localization Engineer with this powerful and secure XML editing environment

Using Alchemy's secure XML environment ensures that the XML document can be modified at any level within the Alchemy CATALYST environment while maintaining the integrity of the document structure as defined by the XML schema supplied by Microsoft. No other localization environment provides this level of control within XML documents.

---

<sup>2</sup> It should be noted that this process also works for Microsoft Visual Basic .NET applications too. They use the same RESX documents (with some minor changes) as C# applications.

## Reducing your costs using Alchemy' Leverage Expert

To reduce the cost of translating new .NET applications, translations from previously translated desktop applications or glossary files can be leveraged into RESX resource files using Alchemy's Leverage Expert.

In one early adopter test, a client was able to leverage 70% of previously translated text into his newly developed .NET application!

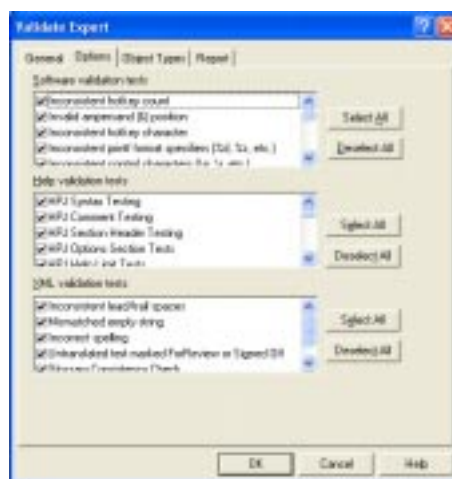
Previous translations can be leveraged directly from TTK files, TRADOS Workbench file, TMX or simple text files, providing the most flexible and versatile options for reducing your future translation costs.

## Reducing Engineering Costs and Time

As the .NET Plug-in uses the Alchemy CATALYST Framework SDK, the Validate Expert can be used to automate the detection and tracking of localization bugs. This reduces the time taken to test applications, log and record localization bugs and then fix them. Automating this phase of the localization workflow can reap tremendous payback, especially in terms of time and deployed Engineering resources.



Reduce your translation costs by leveraging previous translation

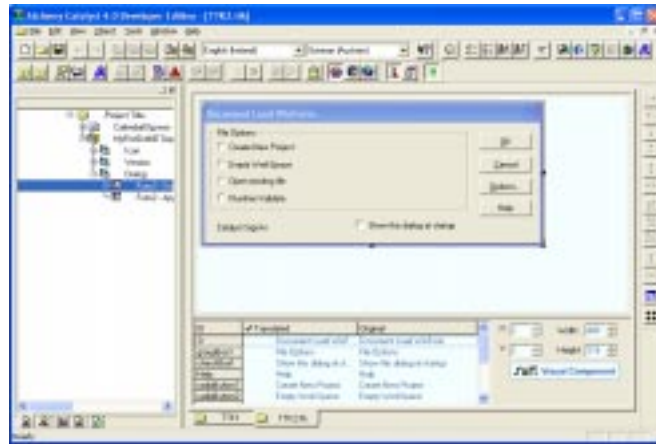


Reduce engineering time by automating the detection of Localization bugs

## The Complete Solution for .NET Applications

As each resource within a .NET Application requires a single RESX document file, the number of files that need to be managed within the localization workflow can be onerous to the Localization Service Providers and software publishers. As updates to the application are released, the complexity of managing files in multiple versions and multiple languages escalates.

To address this problem, the Alchemy .NET Visual Localization Component can also process compiled .NET applications directly.<sup>3</sup>



Equally at home handling RESX files and .NET binaries, Alchemy CATALYST provides a complete localization solution.

Translating, testing and engineering .NET binaries is similar to processing .RESX files, except the number of files that need to be processed tends to be significantly less. This reduces the complexity of managing the translation workflow, reducing costs and project duration.

---

<sup>3</sup> While the .NET Application binary is similar to the Windows 32 PE binaries, the UI resources are stored in different segments. This means that existing localization tools that support binary localization cannot be used to handle .NET binaries. The Alchemy .NET Visual Localization Component can read/write .NET binary files directly.

---

## CONCLUSION

Microsoft .NET is a revolutionary new technology that has redefined the tools and technologies required to localize software applications and content files. Using XML as a core technology to provide an environment to build internet, wireless and web based services, Microsoft .NET presents new challenges to software publishers that view the international market place as a way to accelerate revenue growth and expand their business presence worldwide.

Alchemy .NET Visual Localization Component is the most comprehensive multi-language localization tool for rapidly translating, testing and engineering XML Web Services and applications, significantly increasing translator and localizer productivity, accelerating time-to-market for language products and reducing localization costs. Designed with deep integration of Internet standards such as XML, Alchemy .NET Visual Localization Component dramatically simplifies the localization workflow for Internet and Mobile .NET applications.

Key Benefits
Localize Microsoft .NET Applications in a WYSIWYG environment
Accurately scope translation projects with a detailed breakdown of new and recyclable text in a matter of seconds
Automatically create and manage multilingual glossaries and translation memories
Handle application updates swiftly with advanced ezMatch™ translation memory technology
Distribute and manage complex projects across multiple sites with easy-to-use workflow management utilities
Automate language releases using extensive ezScript™ command line operators and utilities

---

ABOUT ALCHEMY  
SOFTWARE DEVELOPMENT  
LIMITED

Alchemy Software Development is a company formed by the original developers and designers of Corel CATALYST™, an integrated translation environment and a pioneer in visual translation and engineering solutions.

Alchemy' technology is designed to boost the efficiency and quality of globalizing software products and is used by software development and globalization companies worldwide. With over 8,000 licenses installed worldwide, Alchemy CATALYST is used by translators, software engineers, quality assurance specialists and project managers and is referred to as the Gold Standard in Localization. Corel Corporation Limited holds a 24.9% stake in Alchemy Software Development Ltd.

Alchemy Software Development has offices and facilities in Canada, Europe, Asia-Pacific and the United States.

For more information on the company and its products, please refer to [www.AlchemySoftware.ie](http://www.AlchemySoftware.ie) .