Prof. Luka Nerima
Asist. Asheesh Gulati

NTIC

Twic Projet on iphone

Group : Daniela Sinjari
Eris Ago

(II-year, master in computer science)

May 2011

Introduction

The goal of this project is to develop a stand- alone application on iphone in order to make functional the TWIC tool.

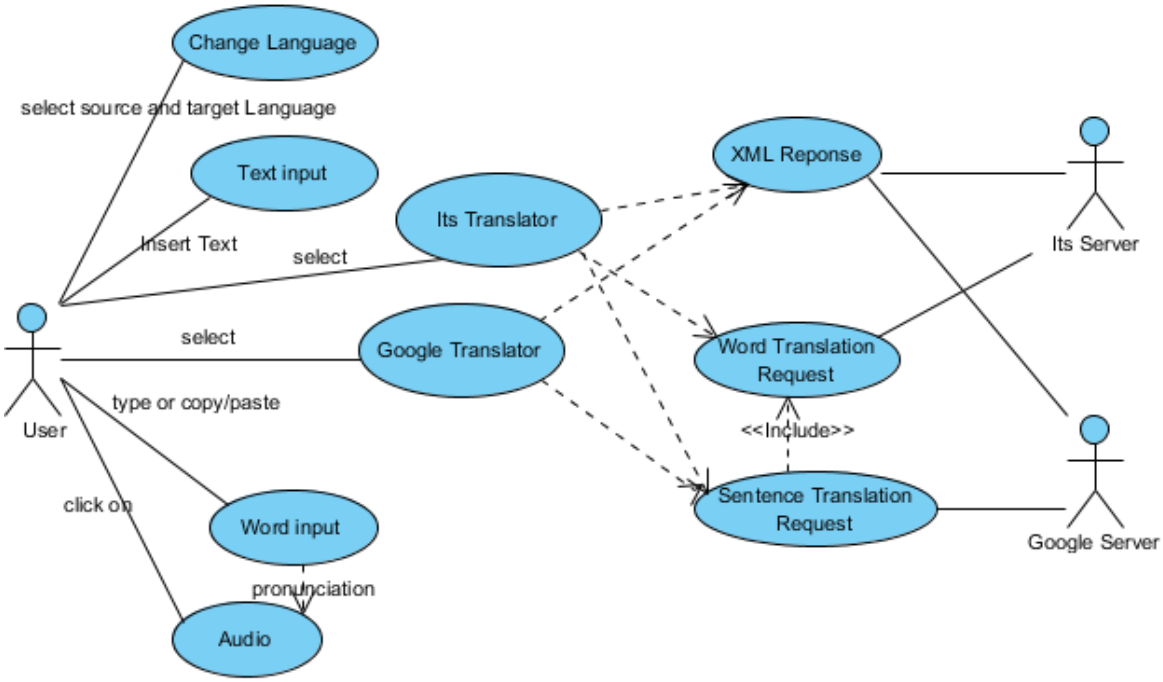
TWIC (Translation of Words in Context) is a dictionary developed by LATL which uses a technology for deep parsing several natural languages (English, French, Italian, German etc). This tool comes in the form of a standalone application that runs on Windows PC, as a plugin to install in the Firefox browser and a plugin for Google Chrome.

In this standalone version, we have used iphone sdk and Xcode platform. During the development, tests have been done in iphone simulator which comes as integrated part with iphone sdk.

Architecture

The client application implements an interactive interface that permit the user to insert a text or to paste a copied text from an web page, then to select the source and the target language and finally choose one of the two available dictionaries: Twic or Google Translator to display the text translation. The phrase entered by the user will be sent to the LATL server that perform the analysis of the sentence and return the translated words to the client.

Uml Use Cases



Platform && Version for application development

At the beginning, we have started to implement Twic Extension for Firefox Mobile (Fennec) for mobile devices, trying somehow to make an adaption of the existing Firefox version of TWIC. Since Fennec is almost recent and there exist a poor documentation (tutorials, forums etc.) for developers online, we've faced with some difficulties in adapting the overlay page. Since Fennec is a totally separate host application, there are some Fennec-specific issues that you need to handle when creating the extension. Most notably, the differences in the user interface (UI) structure - the XUL used to build the UI. We've started from modifying the overlay, but we've encountered a problem with the text recuperation from a random web page in order to translate it in Twic window. Basically, we were able to recuperate text from the first window of Fennec when it's launched for the first time. If we would navigate through other web pages, there was not possible to recuperate the text on click or selection event listeners .

For these reasons, we've been redirected toward a platform which was more reliable and promising a full documentation such as iphone developement.

To develop this iOS application, we have used the XCode 4 developer toolset which includes the Xcode IDE (Apple's integrated development environment), performance analysis tools, iOS Simulator, and the iOS SDKs. Xcode provided all the necessaries tools to design our application's user interface and write the code that brings it to life. While developing an iOS application it's also possible to run it on computer, an iPhone, an iPad, or an iPod touch.

The language of choice for iPhone development, is **Objective-C.**

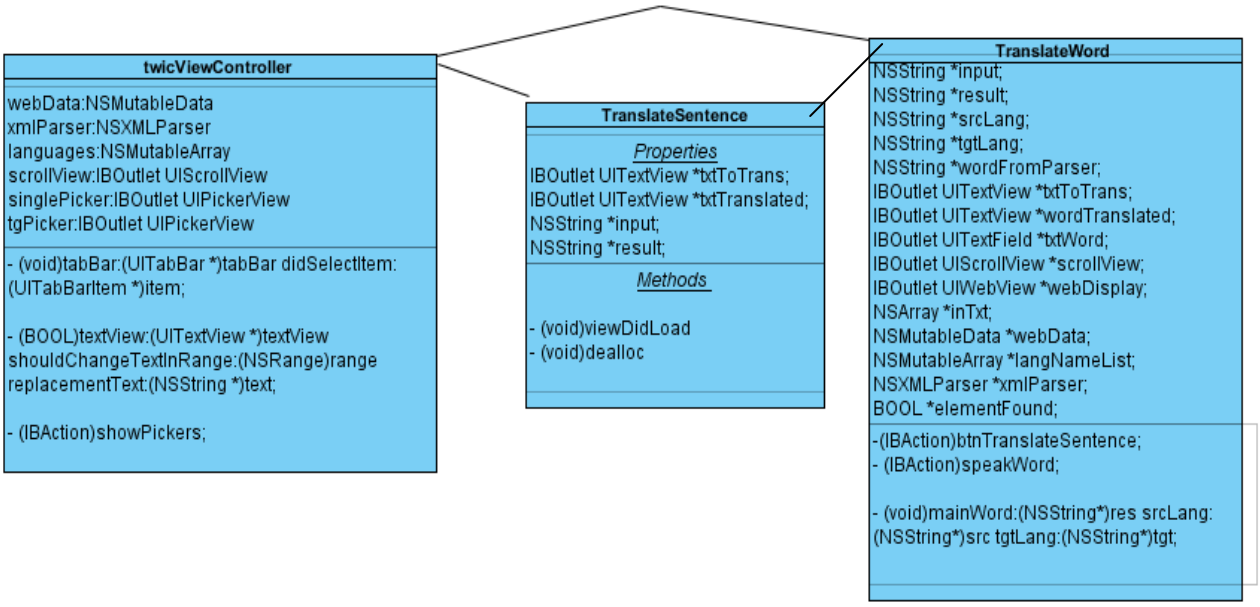
In order to test or deploy the application on real devices, it's obligatory to become a member of the iOS Developer Program,.The reference link in the last section of report describes in details the iOS application development process.

After finishing the development of the iOS application, we can submit it to the App Store, the marketplace where iOS users obtain their applications. However, we have tested the application on iphone simulator to cover a wide variety of usage patterns.

Software requirements: This application has been implemented using the iOS SDK 4.2 (with Xcode 3.2.3) distribution on Mac OS X v10.6.7.

Uml Classes

The uml classes below display the relationships such as containment, inheritance,associations and others.



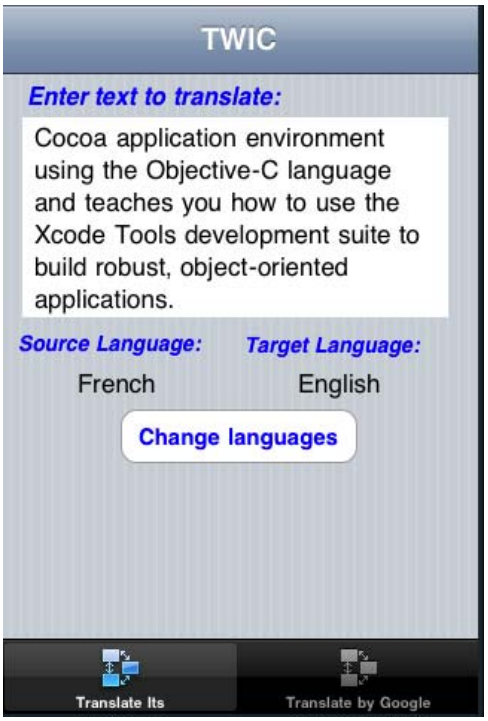
The instance of twicViewController corresponds to the main interface designed in twicViewController.nib, it will be instantiated in twicAppDelegate.m class. Inside the method -(void)tabBar:(UITabBar*)tabBar didSelectItem:(UITabBarItem*)item it's treated the event of tabBar Item selection. If it's selected 'Translate by Google' tabItem, it will be initialized an object of TranslateSentence class, else it will be initialized an object of TranslateWord class responsible for Its Translation.

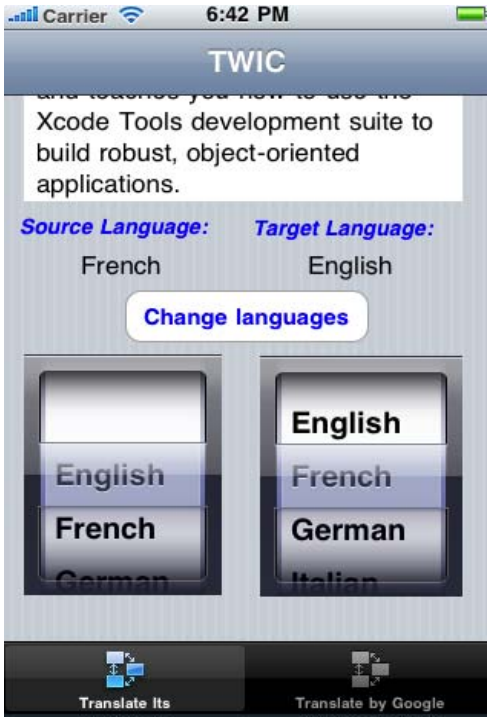
Inside the method –(void)mainWord:(NSString*)res srcLang: (NSString*)src tgtLang: (NSString*)tgt , it will be sent a query UrlRequest to Its server and it will receive as output a Xml string that will be treated in order to get data and attribute values inside the tags. The output will be formatted in UTF8Encoding to avoid problems with characters accents especially for French language.

User Interface Conception

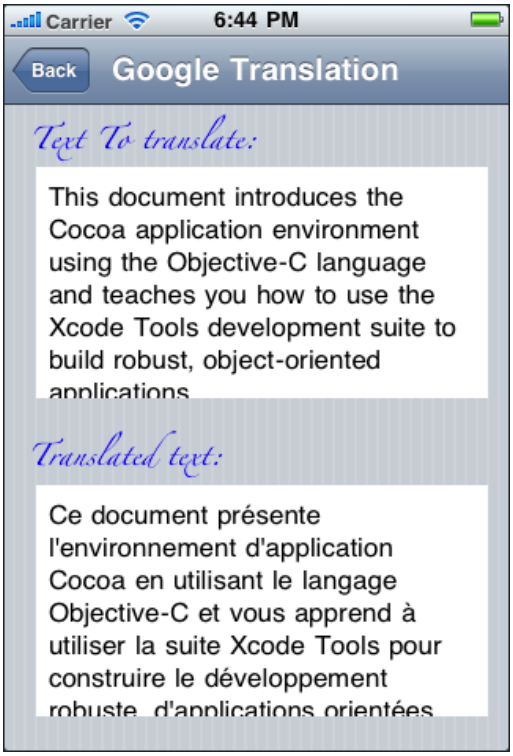
To design the user interface, we have used the **Interface Builder application** which is included in Xcode toolset. It lets you design your application's user interface graphically and save those designs as resource files that your application loads at runtime. Some of user interface layout have been implemented programmatically, f.e in cases of dynamic objects creation.

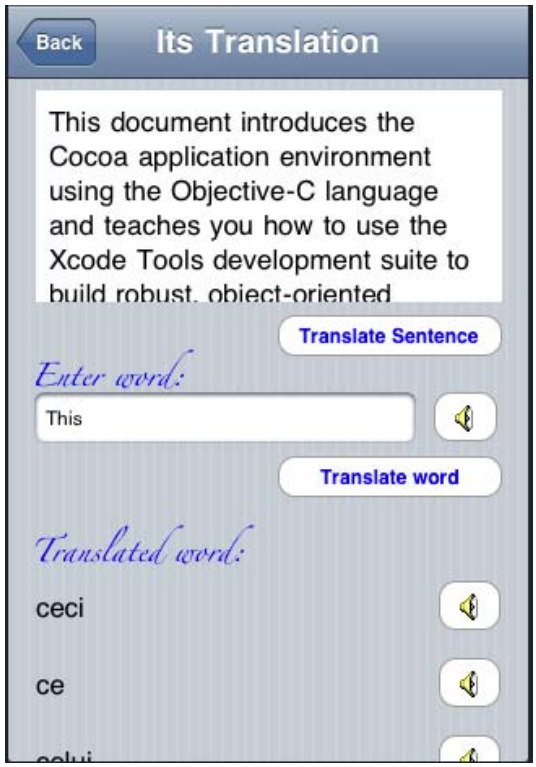
The main interface is concepted to give the possibility to the user to insert a text or paste it (copied f.e from a eb page) in order to translate it by Google or Its Translator. The source and target languages are set as French-English by default, but the user can change them by clicking in 'Choose language' button.





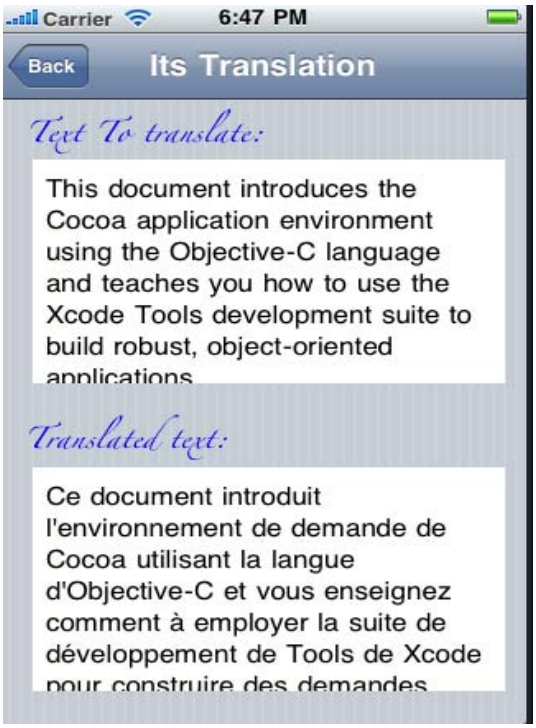
If user clicks on 'Translate by Google' button, another interface named 'Google Translation' with be opened with the translated text. If user clicks on 'Translate Its' button, 'Its Translation' interface will be opened.



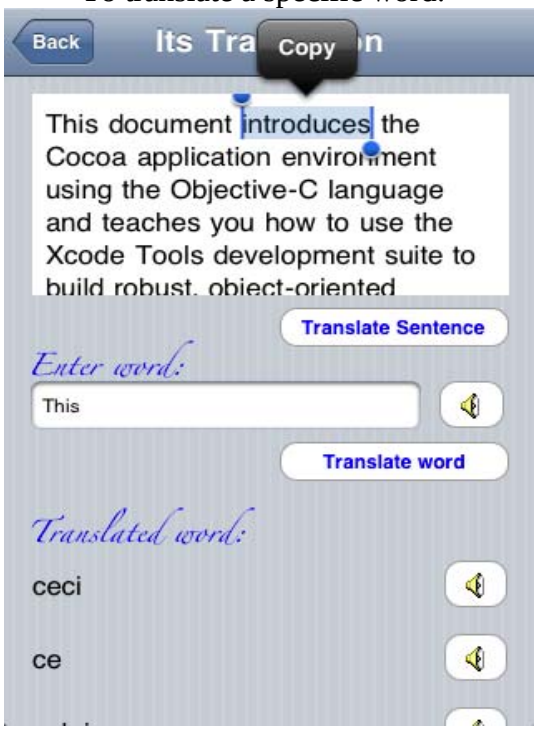


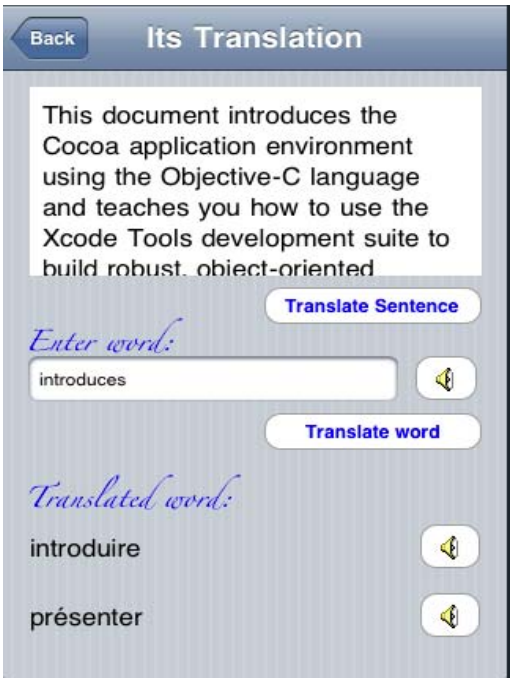
Here user may want to translate all the sentence by clicking on ‘Translate Sentence’ or translating a specific word to see its possible translations by inserting a word in text box, and clicking on ‘Translate word’.

On ‘Translate sentence’ click:



To translate a specific word:





On ‘Translate word’ button click, it will be displayed a list of possible translation, associated with audio pronunciation.

System limitations and difficulties

Due to the small dimensions of iphone display, we have decomposed the modules and designed interfaces to adapt them to the iphone dimensions in order to be more practical for the user. By inserting a scroll view, user can scroll the view to see the whole information. Even the text areas which keep the text to translate or translated text, have a scroll view.

One of the limitations of system is that you have to use only a Mac OS with Intel processor in order to install XCode toolset and to develop an iphone application. But using some tricks found in forums, we were able to install finally a Mac OS in a virtual machine in an Acer laptop with intel processor.
For the audio service, it was not possible to use an iphone shared application for the audio, instead of that we have used google services for word pronunciations.

Another obstacle we had to deal with, was the parsing of the xml data returned by Its (Twic) server. In the case of having French language as a target language, the output contained normally characters with accent like é,è,à etc and we had to use UTF8encoding, in order to display them correctly.

Another important thing to emphasize, while using a navigation controller to navigate from interface to interface using also Back option it's important to not forget to release interface objects like instances of UIView, UIScrollView, UILabel, UIButton etc. Otherwise, it will cause interface blocking generating a segmentation fault, with problems to allocate memory.

Conclusions

This project allowed us to explore practical techniques necessary for the realization and implementation of a stand alone application iphone.

It was really an interesting experience using Xcode toolset and iphone SDK to develop a stand-alone application.

The language of choice for iPhone development, **Objective-C** is a language based on C with extensions for object-oriented concepts, such as classes, inheritance, interfaces, messages, dynamic typing, etc. Pointers in Objective-C, though powerful, are also another time-waster.

The project may need some other enhancements like improving the selection of specific words from a text, with the option of highlighting the selected word, and on highlighted event, the system could display the translated word. Using textArea object to keep the text, it's not possible to highlight specific items in it. Instead, using a UIWebView to keep the content would permit to set css for font-style for text items.

References

A very useful link for Deploying iPhone Apps to Real Devices:

http://mobiforge.com/developing/story/deploying-iphone-apps-real-devices