University of Glasgow | Department of Computing Science

A Website Translation Service

Alasdair Campbell
Andrei Mustata
Paul Moore
Stephen Hayton
Wei Zhang

Level 3 Project — 9th February 2012

Abstract

(TBC)

Education Use Consent

We hereby give our permission for this project to be shown to other University of Glasgow students and to be distributed in an electronic format. **Please note that you are under no obligation to sign this declaration, but doing so would help future students.**

Name:	Alasdair Campbell	Signature:	
Name:	Andrei Mustata	Signature:	
Name:	Paul J Moore	Signature:	
Name:	Stephen Hayton	Signature:	
Name:	Wei Zhang	Signature:	

Contents

Chapter 1

Introduction

This article assumes a basic general computing knowledge from the reader. If any words or phrases are not understood, consult the glossary of terms (Section 5.2.2) at the end of the document. All content is owned by the authors of this document or is otherwise referenced in bibliography (Section 5.2.1)

1.1 Background

As part of our degree in third year we are tasked with a team project. This relates to computing disciplines new to us and draws on knowledge gained from the preceding two years at University. Teams were randomly assembled at the beginning of semester and our team received the project to create a website providing a document translation service for a real client. It was a pleasing allocation mainly due to the latter part of the task: the fact we would be working with a real client.

There are many translation services available on the Internet already, a simple Google search for online translation returns around one hundred and fiffty nine million results. We have looked into various different types of translation and interpretation websites during our research and have found pros and cons from each. This vast number of already available websites creates a desire of competing with what's already available, by trying to improve areas where other websites have fallen short. We believe one of the key factors that makes a modern website successful is minimalistic design: it offers simple and effective functionality and is aesthetically pleasing. The combination of these things means users are more likely to use the website after stumbling across it in a search, perhaps, and will ultimately give our client a larger customer base. To clarify, we are not trying to re-invent the wheel with our project. We have used frameworks and other free source components in the development of our website. The system is mainly built around the LAMP structure, i.e., Linux + Apache + MySQL + PHP. We aimed to maintain a user friendly feel and look and believe that is something we accomplished well. One of the things that encourages this notion is the fact we have created a simple 3-stage process in which users can register, upload documents, and request languages to translate to.

1.2 Aims

To briefly summarise our task at hand: we are to develop a website for a free-lance translator. It should allow users to upload documents, request one of the available languages to translate to, and submit a request for translation. Our client should then be able to review those submitted documents and send a quote to the user for the job to be translated. The user should then be able to pay for the job(s) via Paypal, and then receive their translated document after a period of time. Additional required features of the website will be discussed later.

1.3 Motivation

An essential part of our requirements gathering process was the first meeting with our client, Joelle Cimatche. Our team had been forewarned by our supervisor that the client was not very ”technically minded”, so we tried our best to prepare questions that did not assume she had much experience with a computer. At the meeting, the client explained she had a very basic understanding of word processing applications and the Internet, and not much else. This presented us with an additional challenge. We couldn’t simply relay technical jargon to the client and expect her to provide useful feedback. Not only that, our team would have to create a very easy-to-use admin end to the website that she could learn quickly. As we advanced the development of the website, we would have to be very clear and straightforward when updating her on our progress. **Figure 1.1** somewhat illustrates this challenge we have. It is basically the ”adoption rate curve” for a percentage of users of new technologies. In otherwords, how quickly users can say they understand a new technology after it is first released. It is known as Roger’s bell curve. Our client would be at the far right end of the spectrum, in the 16% of users described as ”Laggards” in the graph.

Reflecting on our experiences in our project, working with the client has not been exceptionally easy. One of our main challenges is that our client is a novice computer user. Quite ironically, it has been a task for us to translate regular computing jargon into layman’s terms in order for them to understand. This was critical in our requirement gathering process. In spite of this additional challenge that teams working with other projects might not particularly face, it is not necessarily something that is discouraging for us. When we eventually graduate and face real world software projects, it won’t always be technically minded people like ourselves we deal with, it will people who are more similar to our client. It will provide us something to drawn upon when we are asked to recount our experiences in future interviews. The opportunity to work with a real client will serve us as great preparation for working life.

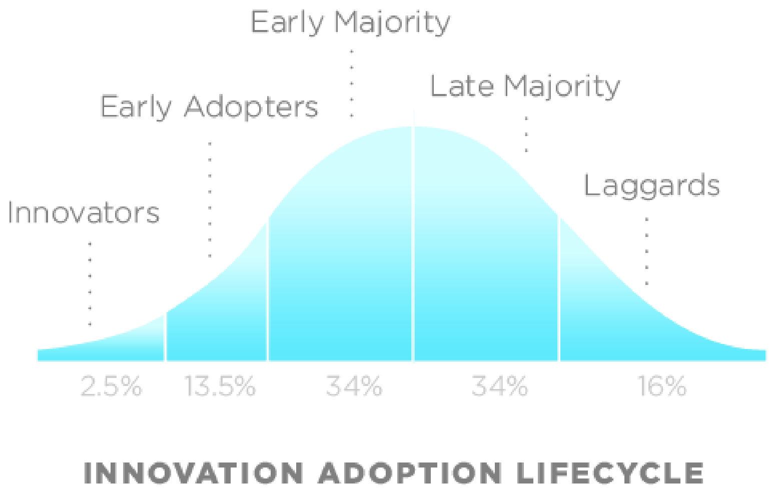


Figure 1.1: Roger's bell curve, source: http://en.wikipedia.org/wiki/File:DiffusionOfInnovation.png

1.4 Preliminaries

To understand this report it is necessary to understand that we do not have to implement a translation algorithm. We are providing customers with an interface to send documents to be translated by a professional translator for a fee, and then returned in the chosen translated language using the same web interface. To understand the process that we have devised it would be advantageous to have a simple understanding of how a database works. We have adopted several frameworks in our project, namely Bootstrap (CSS, HTML), CodeIgniter and phpMyAdmin (all of which are discussed in Section 2 later), which have advanced functionality - functionality which will, in most instances, not be necessary for our project and will therefore be left untouched - however it allows us to show we are efficient software engineers and that we are capable of software re-use.

1.5 Outline

The remainder of this report will go into more detail on the background research of our project, extend on our motivation and set out our group organisation and project plan. After this we will detail our design ideas and methodologies, before moving on to document our implementation, testing and evaluation. We will then discuss any problems encountered and the results of our evaluation before revealing the nal status of the project, giving a detailed outline of the deployed site including nal graphics and information relating to Bethel Translations.

Chapter 2

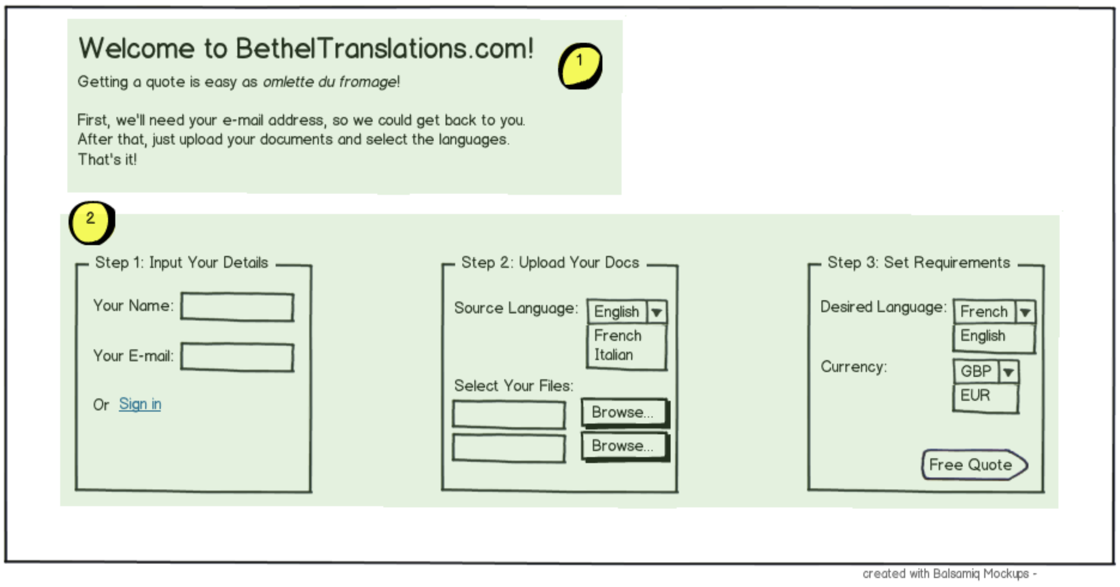
Design

As software developers naturally would when creating anything else from scratch, our team looked for similar websites already in existence. We identified common useful features of each, features that were not so useful, and listed some we thought could be useful but simply did not exist in any of the sites we examined. One recurring theme we noticed in a majority of similar translation websites was that the home page was very cluttered. In other words, the process that the user had to follow to obtain some translation of a document was not extremely clear. Instead, they were met with various registration options, other services and annoying advertisements. From previous modules in our degree, namely IM2 and IS3, our team had experience of applying Jakob Nielsen's heuristics to obtain a successful user interface. We wanted to develop the idea that our website would display the minimal amount of information to a user by dividing the registration, document upload and language selection to a single page, 3 step process. Our interface would then fall in line with the principle that user interfaces should have aesthetic and minimalist design. [1] We believe this is an extremely important aspect of any modern website based on the way that users make a decision of whether or not to use the services offered by the website. For example, imagine a user enters a Google search for ”Translation service” and clicks our website in the results page. If the page they are met with looks too complicated or confusing in nature, the user simply clicks ”Back” on their web browser, and goes to the next appropriate web page. If however the website looks clean, simple and easy to use, the user would be more inclined to use the website properly. We believe that our 3-step process found on the home page encourages anyone that requires a translation service for the provided languages to at least try for a quote, if not go through with the whole process.

With this approach in mind, we began to create some wireframes in order to form a solid idea of how this 3-stage process would physically look on our website. **Figure 2.1** illustrates our early attempt at doing so:

The layout is intuitive, flowing and simple to approach. The user simply enters some details, adds the documents, sets the requirements and clicks a button to request a quote. There is no daunting, large form based entry that some other websites encapsulate. It is minimal and it is undemanding of users. Obviously, the majority of our work done by our system would be when the user clicks ”Get your quote”, and the system would have to register a user (pending email validation), upload their documents to the filestore, associate a jobid and requirements, and place the job in our client's pending work queue. The implementation of such functions are discussed later in Section 3.

[1]http://www.useit.com/papers/heuristic/heuristic_list.html


Welcome to BethelTranslations.com!

Getting a quote is easy as *omlette du fromage*!

First, we'll need your e-mail address, so we could get back to you.
After that, just upload your documents and select the languages.
That's it!

Figure 2.1: The 3-step process

With this 3 stage process as our main design factor of the website, we began to consider the main user groups of the website. We identified the needs of two main user classes: the **clients** and the **translator**

The clients are the users of the service, the visitors of the website. However, as far as the system is concerned, not all visitors are clients, because, in order for a visitor to become a client, he must register with the service. So we decided to have a visitor as a category with a registered user a subcategory. A registered user would then possess all the same abilities as a visitor with some specialised capabilities:

Visitor

- Rationale: The visitor is just visiting. An anonymous visitor of the website.
- Background: "I need to get some documents translated. I came across this website and before I register or send any of my documents, I want to make sure that Im dealing with a serious service."
- He is a potential client, therefore the steps which he must make in order to become one must be as clear as possible.
- His **goal** is to inform himself about the service. In order for him to be converted, he must be convinced that the service provided is of great quality, so the systems goal is to make itself trustworthy. Also, a clear privacy policy regarding e-mail addresses and the documents should be available, since they might contain sensitive data.

Client

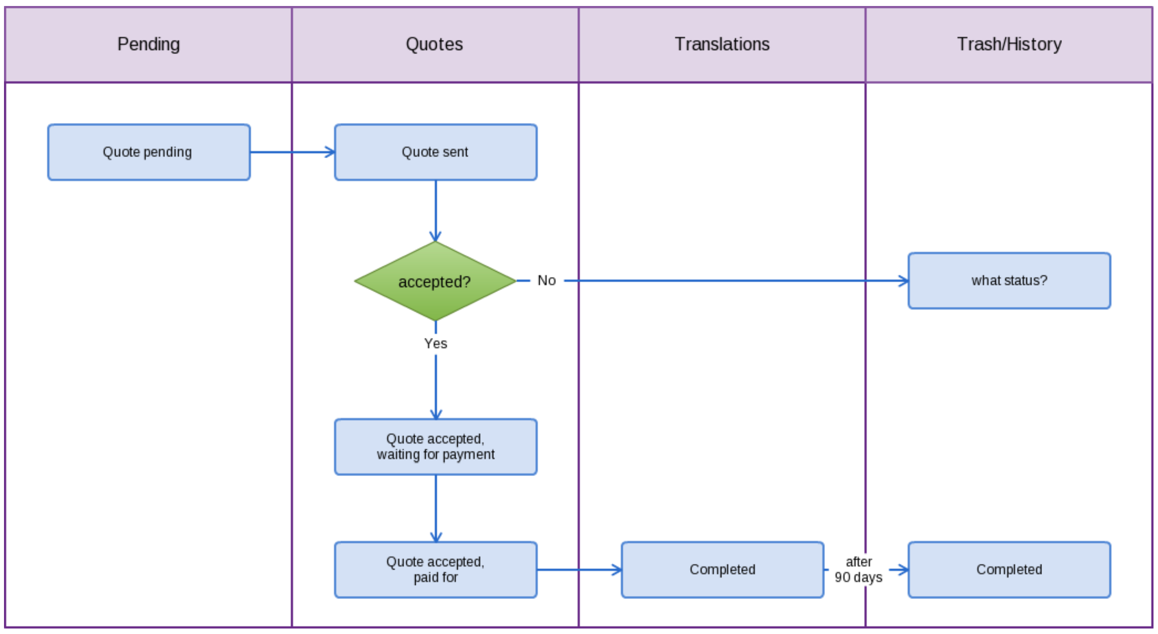


Figure 2.2: The transistion of jobs through ”statuses”

- Rationale: The client is a registered user of the system.
- He has the same goals as the visitor, but, now that he is registered, he trusts the service a bit more. He has access to all the documents that he ever submitted for translation and can view each of the **job statuses** for which his documents are contained within. He can view documents he has paid for, and up to a certain period of time, download both the original copy and the translated copy. He can also view some other useful statistics on his previous jobs.

Administrator/Translator

- Rationale: The translator is the one answering all the translation requests.
- ”As a translator, I must review the documents that my clients send me and quote them. After that, I must also translate them and let them know that I finished work.”
- His goal is to answer to all of the clients requests, i.e. to quote the documents that are received and then translate them and send them back to them.

Building upon the needs of these user classes, we needed to design a flexible system that provided functionality for all of these goals. Our main focal point would be the transistion of **jobs**. In the context of our system, a job is what the website creates when the user uploads one or more documents for translation. Clients (registered users) would submit and pay for them. The translator would review them, download them and upload them. We considered the transition of jobs throughout the lifecycle of a translation, as job statuses, and produced a sequence that is illustrated in **Figure 2.2**.

The diagram is much self-explanatory, but to summarise: Jobs that are submitted by the clients are placed in a **pending** work queue. The translator is able to review these documents and send some quote to the client. After a job has been quoted, it is placed in the **Quotes** queue. Quoted jobs are held in this queue until they are paid for via Paypal. When this has happened, they are moved to **Translations**, which is a breakdown of completed jobs. After a period of 90 days, jobs from this section are moved to **History** in order to reduce space on the server.

The transformation from this design plan to a viable user interface that is built upon these job statuses is described later in Section 3.1.

Chapter 3

Implementation

In this chapter, we describe how we the implemented the system from our design plan and the technologies used in doing so.

3.1 User Interface

TBC

3.2 Database Model

Our database structure was designed after we had thoroughly revised our user registration and job transistion processes. We envisaged there being four seperate entities: one for **users** to become registered, one to represent **documents** being submitted, another for the **jobs** that are comprised of submitted documents and finally one for the documents once they have been **translated**. The attributes of each of these entities and the relationships between them is illustrated in **Figure 3.1**

The majority of activity for these tables occurs in the first three entities, as they are all populated in some way during the 3 stage process discussed earlier.

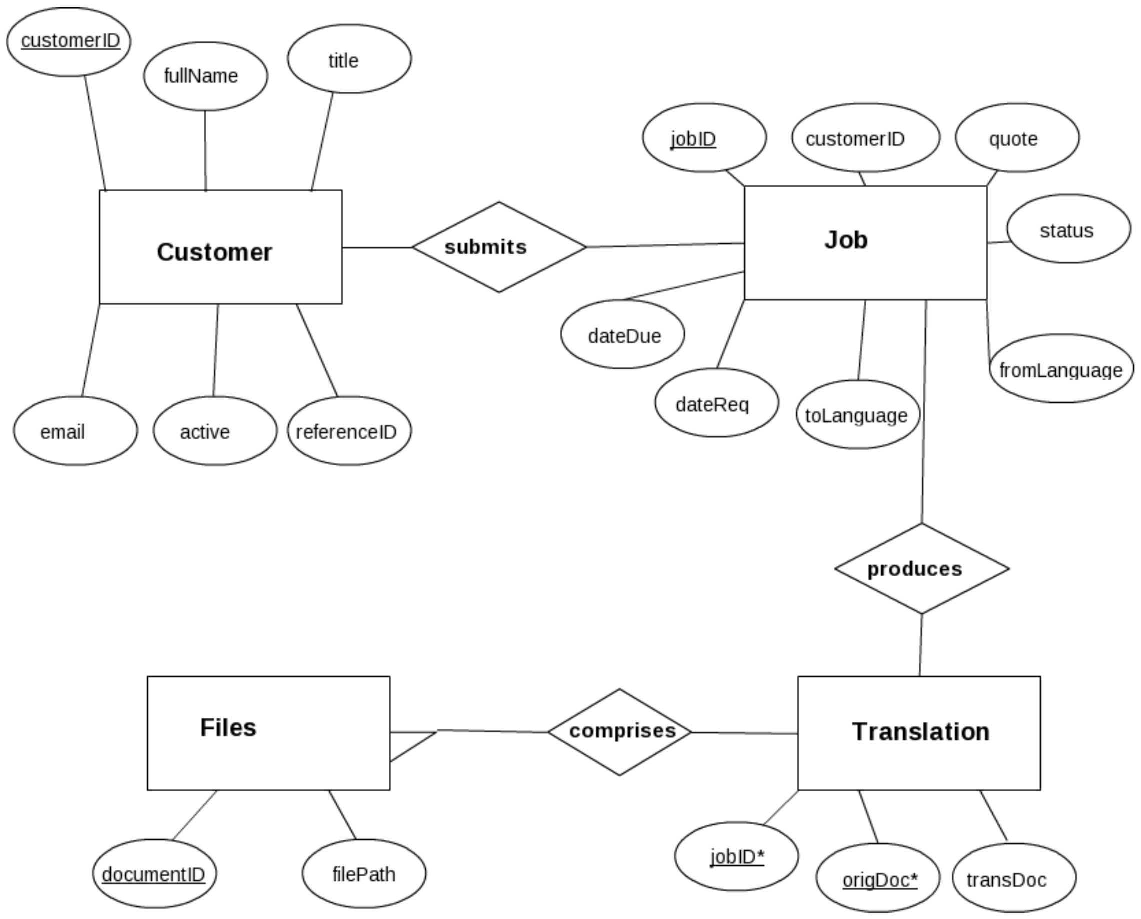


Figure 3.1: ER Diagram for Bethel Translations

Chapter 4

Evaluation

First feedback with Karen incl wireframes etc - Jan 25
Second feedback session with Karen incl more complete website design - Feb 9th

Chapter 5

Conclusion

Team photo goes here? :) A great project! etc..

5.1 Contributions

Alasdair done this... Andrei handled that... Stephen took responsibiliy for... Paul mainly done... Wei was responsible for...

5.2 Appendices

5.2.1 Appendix A - Bibliography

5.2.2 Appendix B - Glossary of Terms

- **Free-lance** - Working for different companies at different times rather than being permanently employed by one company.
- **Client** - A person or organization using the services of a professional person or company.
- **Users** - A set of people who use or operate something, esp. a computer or other machine.
- **Requirement gathering** - Determining the needs of a client through any form of communication.
- **Software project** - Using the surrounding context, a software project aims to create application(s) using programming language(s) by adhering to project management principles.
- **Programming language** - A programming language is an artificial language designed to express computations that can be performed by a computer.
- **Web scripting language (*PHP, Javascript*)** - A scripting language is a programming language that allows control of one or more applications.

- **Website development** - The process of constructing and maintaining a website.
- **LAMP** - LAMP, (Linux, Apache, MySQL and PHP), is an acronym for a solution stack of free, open source software
- **Web application framework** - A software framework that is designed to support the development of dynamic websites
- **Open source** - Computer software for which the code is freely available