At Your Service: Embedded MT As a Service

Florence M. Reeder
The MITRE Corporation
1820 Dolley Madison Blvd.
McLean, VA 22102
Freeder@mitre.org

Abstract

A growing trend in Machine Translation (MT) is to view MT as an embedded part of an overall process instead of an end result itself. For the last four years, we have fielded (primarily) Commercial-Off-The-Shelf (COTS) MT systems in an operational process. MT has been used to facilitate cross-language information retrieval (IR), topic detection and other, wide-scoped scenarios. These uses caused a fundamental shift in our views about MT – everything from user interface to system evaluation to the basic system structures. This paper presents our lessons learned in developing an MT service for a wide range of user needs.

Introduction

The foreign language material to be handled by the government is increasingly diverse and problematic. Foreign language processing needs are increasing because of the changing conditions of the world. Traditionally, users could focus on just a few foreign languages and a limited number of sources of foreign language materials. As we begin the 21st century, users of online materials are faced with having to process, utilise and exploit documents that may be in one of many languages or a combination of languages. It is not feasible to expect a given user to know all of the languages related to their topic of research. It is equally unrealistic to expect to have on-demand translators available in every language whenever they are needed. Because of the expanding need, tools are being developed to automate the use of foreign language materials.

Unlike previous views of tools, the current vision for machine translation (MT) is as a small part of a larger, mostly automated process. For many users, this does not mean yet another tool with yet another interface, but a nearly invisible companion that incorporates translation and necessary support technologies. One such system, the Army Research Lab (ARL) FALCON system, combines scanning, optical character recognition (OCR), translation and filtering into a single process. Another view of this is the DARPA Translingual Information Detection, Extraction and Summarisation effort (TIDES). TIDES represents the pinnacle of information access and is a real challenge for MT. MT supports the translingual aspects of the effort and can be viewed as an embedded tool which facilitates other technologies. Finally, the integration of MT into the process for intelligence analysis serves as the basis for the CyberTrans project. For this paper, we will discuss the CyberTrans project, the lessons learned and the supporting technologies necessary for the successful integration of MT into other systems.

1 Proposed Architecture

1.1 Original Prototype

The incarnation of CyberTrans grew as a demonstration that MT technology could be useful in the intelligence analysis process. As a result of an MT survey (Benoit et al, 1991), MT technology was believed to be ready for incorporation into an operational environment. Initially, CyberTrans was designed as a wrapper around Commercial-Off-The-Shelf (COTS) and Government-Off-The-Shelf (GOTS) MT systems in Unix environments. A client-server architecture, implemented in a combination of Lisp and C, allowed for uniform user interfaces

to translation engines (Systran, Globalink and Gister). The server software interacted with the translation engines and the client software interacted with the users. The server interacted with client programs through Remote Procedure Call (RPC) passing of translation parameters (such as language, dictionary and output format) and file transfer of translation data. The clients provided were: e-mail, web, FrameMaker and command line. By providing translation through these media, users could translate documents in a familiar interface without having to worry about differences between translation products. The languages provided in the first prototype were those available to the government from Systran (French, German, Spanish, Italian, Portuguese, Russian to English); those purchased from Globalink (French, German, Spanish, Russian to/from English); and those available from the GOTS System Gister (language list is unavailable for publication). At the time of delivery in 1995/1996, this represented a relatively new method for delivering MT technology to general users.

Shortly after the fielding of the initial prototype, the need for additional language services to accompany translation became apparent. As will be discussed in Section 2, the data sent to the translation engines pointed out the differences between translation in an interactive environment and translation in an embedded, automated environment. Interactive translation is much more forgiving of low quality input data while automated processing suffers from poor input data. This forced the designers to implement a series of pre- and post-processing tools to be provided in the translation server. Initially, they were included in the functional architecture as depicted in Figure 1. This addition of language tools caused a necessary re-design of the architecture from a client-server model to an enterprise service model which is characterised by an open architecture view of loosely coupled modules performing services for multiple applications. The newer design will be discussed in the next section. At this time, other system architectures were beginning to be introduced into the community such as those provided by ALIS Technologies; Systran and in FALCON. Because this is a specific lessons learned about the CyberTrans experience, it is beyond the scope of this paper to compare this architecture with other architectures.

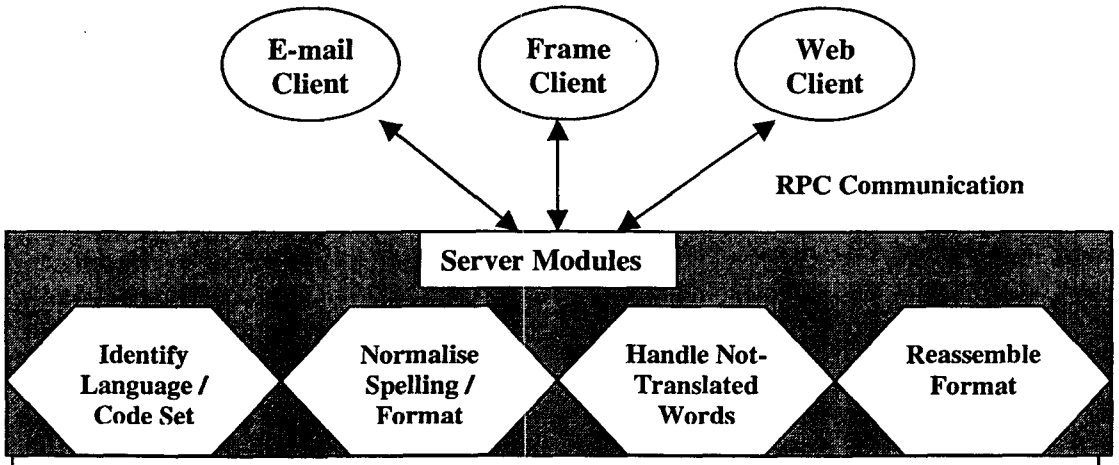

Figure 1: Original Architectural Flow

1.2 Updated Design

Because of the addition of new tools and technologies into the CyberTrans model, it became necessary to re-engineer the server design. As part of the transition of a prototype system into a production-quality system, the reengineering also addressed issues such as system administration support, robust operation for 24/7 service and metrics. As can sometimes be the case, the prototype started being used in

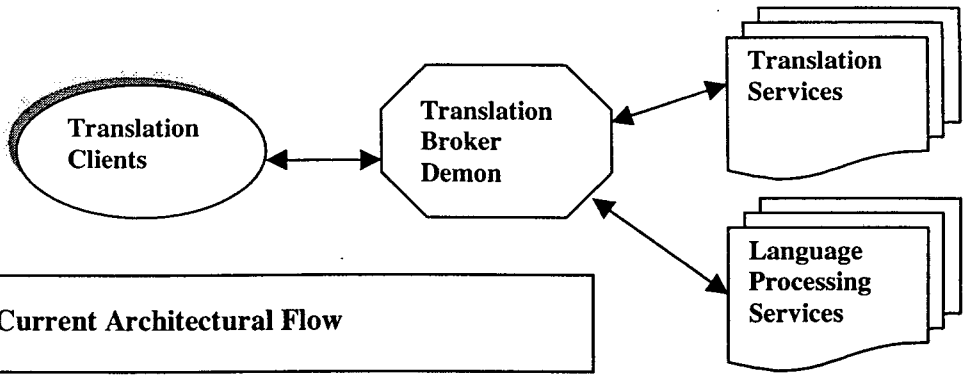

Figure 2: Current Architectural Flow

continuous operation, causing a demand for improvement concurrent with ongoing operation. The reengineering was shaped by the fact that the system had expanded for new capabilities (in pre- and post-processing); the fact that the system had to remain operational all of the time; the fact that the system was being used in ways that were unanticipated by COTS/GOTS MT developers; the fact that the system was to be multi-platform (to include PCs) for an expanding list of languages and the fact that the system was beginning to be seen as providing a service similar to other system services (such as e-mail). These factors caused the system to be reengineered in an enterprise services model as an object-oriented design.

In this architecture, demon processes broker translations – a request for translation is passed to the system by a client program; the translation is planned out as a series of translation-related services; each service is requested from the responsible system object and the resulting translation is then passed back to the client programs. Implemented in a combination of C++, Java and Lisp, the new version represents a service-oriented architecture. Figure 2 shows an updated architecture picture. Translation services include Systran (French, German, Italian, Spanish, Portuguese, Russian, Serbo-Croatian, Ukrainian, Chinese, Japanese, Korean into English); Globalink (French, German, Spanish, Russian to/from English) and Gister (language set list unavailable) with plans to incorporate engines for languages such as Arabic. Language processing services include language/code set identification; code set conversion; data normalisation, including diacritic reinsertion and generalised spell checking; format preservation for Hyper-Text Mark-up Language (HTML) documents; not-translated word preservation and others. The clients remain e-mail, Web and FrameMaker. Platforms include both Unix and PC platforms for clients and with the capability to incorporate PC-based tools as part of the service. Having described the architectures, we turn to lessons learned as a result of having an operational MT capability, running 24/7 for over 6000 translations per month.

2 Implementing Embedded MT

The biggest surprise we encountered in fielding CyberTrans is related to the expectations of the users. The average user initially approaches MT with an almost Star Trek-like view – that it is possible for the system to understand and translate perfectly any given document regardless of content, form, format or even language. While this is an unrealistic expectation of this or any system, an overriding goal which emerges is that embedded MT should be as automated as possible. This represents a fundamental shift from the traditional view of MT as an interactive, user-driven process to as a passive, data-driven process. We will now describe four areas where specific technologies need development for the smooth incorporation of MT into a "real-world" setting: language and code set identification; data normalisation; format preservation and lexicon development. Finally we will describe software engineering issues and challenges which facilitate the straight-forward embedding of MT into existing processes.

2.1 Language / Code Set Identification

Knowing the language and encoding, or code set, of a document is a necessary first step in utilizing on-line text. For automated MT, the identification of the language(s) or code set of a text is necessary for systems to operate effectively. A Spanish-English translation

system will not successfully process an Italian document and will be even less successful in processing a Chinese one. The first requirement, then, which enables automated, embedded processing is the detection of the language(s) and code set(s) of a given document.

In preparing the tools which permit the accurate detection of languages and code sets in an operational setting, we found characteristics of the data which carry throughout all of the processing we discuss. The first, and foremost, is that the data is not clean, well-formed text. Frequently documents will have a mix of languages (both human and machine), code sets (including formatting information) and information pieces (such as e-mail headers, ASCII-art, etc.). For example, chat is very idiomatic and has many pre-defined acronyms. Finally, about 10% of translation materials are very short – between one and ten words. All of these factors contribute to the difficulty of preparing a service for language and code set identification as well as other natural language processing (NLP) tools. The implemented algorithm for language/code set identification is a trainable n-graph algorithm and has been discussed in more detail elsewhere (Reeder & Geisler, 1998). Currently our language and code set identification works for on the order of 30 languages (mostly European) and about 130 code sets (including many ASCII transliterations) yet these numbers are still insufficient for the data routinely processed by CyberTrans. The step after language identification is data normalisation and will be discussed as the next result of lessons learned from CyberTrans.

2.2 Data Normalisation

Machine translation works best with clean, well-formed input text. Operationally, this is an ideal, but not reality. In reality, data that is being translated can suffer from many types of errors including misspellings and grammar mistakes, missing diacritics and transliteration problems, scanning errors and transmission obstacles. With her evaluation of MT systems, Flanagan (1994) describes reasons for errors in translation. MT systems were examined in light of the outputs of translation and the types of errors that can be generated by the translation engine. These include spelling errors, words not translated, incorrect accenting, incorrect capitalisation as well as grammatical and semantic errors. This study does not look at the kinds of inputs that can cause failure in a translation process. A second paper (Flanagan, 1996) examines the quality of inputs to the translation process, arguing for pre-editing tools such as spelling checkers. Yet, this continues to be an interactive view of the translation process. Another study (Reeder & Loehr, 1998) show at least 40% of translation failures (not translated tokens) are attributable to the types of errors, or non-normalised data, presented here. In an embedded process, the system must automatically detect and correct errors.

Language Source
Segmentation
Character omissions
Mixed languages
Input Source
Misspellings
Grammar mistakes
Missing Diacritics
Transliterations
Capitalisation
Production Source
Scanning / OCR
Electronic representation
Conversion errors
Acquisition Source
Network transmission

Table 1 - Categorisation of Error Types

Instead of being random, the errors are regular, especially in generated or automated documents. For instance, a writer of French without a French keyboard will systematically omit diacritics. In this case, the errors in the document are far from random. Along these lines, we have grouped similar error sources together. Operational data can have one or more of these error types: misspellings and grammar mistakes; missing diacritics; mixed language documents; improper capitalisation; transliteration / transcription / code set mismatch; scanning (OCR) errors; web page or e-mail specific standards; conversion errors; network transmission errors; segmentation problems; character omissions. These error types can be categorised by the origination of the problem as in Table 1. Much

current CyberTrans work consists of developing and transitioning tools which can accurately detect and remediate errors, converting documents into a standard (normalised) form. The order in which the normalisation techniques are applied is a subject of ongoing research.

2.3 Format Preservation

Documents arrive in many formats that have meaning in their structure. For instance, web pages contain HTML indicators plus language. A challenge for MT is that the HTML should not be translated whereas the text must be. The file name **rouge.gif** should not be translated to **red.gif** if the web page is to be reassembled. Consider also, the task of translating a bulleted list. It is desirable to maintain a bulleted list with appropriate syntax. Table headings and labels also should be translated without destroying the format of the table. This, too, is a matter of ongoing research.

2.4 Lexicon Update

The highest portion of the cost of providing a machine translation capability reflects the amount of lexicography that must be done – as much as 70% of the cost of a machine translation engine. In addition, the government requires specialised lexical repositories which reflect unique domains such as military, legal, scientific and medical. We must find ways to update lexicons intelligently, using such sources as dictionaries, working aids, specialised word lists and other information reservoirs to provide a broad coverage of words. One current approach is to record the list of words which do not translate and automate the handling of these. An issue in this is determining how to provide sufficient context for lexicographers. Additionally, different translation engines encode lexical entries in widely differing ways, meaning that sharing lexicon entries amongst translation capabilities is problematic. We are working on a lexicon service bureau (LSB) designed to facilitate the sharing of lexical materials. One part of this is the automatic extraction of lexical entries from on-line, machine readable dictionaries. Another part is the analysis of not-translated words. A final portion is research into a specialised category of lexical items – named entities. As with other processes in this section, we are addressing this as part of ongoing research – each advance raises the bar for the level of input text that can be handled.

2.5 Software Engineering Challenges

A final lessons learned from the CyberTrans experience relates to the software engineering challenges of putting together diverse technologies from many vendors for multiple purposes. The first of these is the problem of API's from COTS systems and GOTS systems. Behind our initial command line, file-based interaction lies the fact that translation engines do not routinely provide APIs, presenting an integration challenge. Platform-specific tools also contribute to the integration problem. The second software engineering challenge stemming from this is the amount of time necessary to bring up a translation engine. A good translation engine has a lexicon in the tens of thousands of entries which takes time to load up. Currently, the loading of a translation engine takes as much time as all of the rest of the pre- and post-processing combined. A third challenge is deciding on and enacting a language representation. Although Unicode makes good strides towards the uniform sharing of data, many of the tools needed to convert real data into Unicode need to be improved. Additionally, current implementations of Java and C++ do not have all of the necessary pieces for seamlessly handling of a wide range of languages. Finally, the challenge is in the management and ordering of the translation process. To effectively manage a translation, requires a translation manager which can be a single point of failure.

Conclusion

We have identified the lessons learned from a specific embedding of MT into an overall process. We have identified issues and concerns resulting from this experience. We continue to refine and examine the issues of supporting MT, of making it palatable and viable in multiple applications and frameworks. This system is just one example of embedding MT. Future work must compare this effort to other work in the field.

References

Benoit, J., Jordan, P., Dorr, B. (1991) Machine Translation Technology Survey and Evaluation Report. MITRE Technical Report.

Flanagan, M. (1994) Error Classification for MT Evaluation. In *Technology Partnerships for Crossing the Language Barrier: Proceedings of the First Conference of the Association for Machine Translation in the Americas*, Columbia, MD.

Flanagan, M. (1996) Two Years Online: Experiences, Challenges and Trends. In *Expanding MT Horizons: Proceedings of the Second Conference of the Association for Machine Translation in the Americas,* (pp. 192-197). Washington, DC: AMTA.

Reeder, F. & Geisler, J. (1998) Multi-Byte Issues in Encoding / Language Identification. In *Proceedings of the Embedded MT Workshop, AMTA-98.* Langhorne, PA.

Reeder, F. & Loehr, D. (1998) Finding the Right Words: An Analysis of Not-Translated Words in Machine Translation. In *Proceedings of the 3rd Conference of the Association for Machine Translation in the Americas, AMTA-98.* Langhorne, PA.