An overview of a multilingual machine translation system based on the e-speranto interlingua

Grega Jakus, Sašo Tomažič

University of Ljubljana, Faculty of Electrical Engineering

Abstract - The paper presents an overview of a multilingual system based on the e-speranto. E-speranto is a computer language intended for recording multilingual documents on the Web. It can also serve as an intermediate language, or interlingua, in multilingual translation. Its advantage over similar approaches is the compatibility with HTML and the intelligibility of documents both to computers as well as to people. The development of e-speranto-based translation system consists of four stages. They include the development of language, tools for writing documents directly in e-speranto, interpreters of e-speranto and translators from a natural language to e-speranto.

The implementation of the system is roughly divided into two parts: the client side and the server side. On the client side, the composition of the e-speranto documents takes place. On the server side, server handles the requests for the interpretation of the documents written in e-speranto and returns the result to the client. The database serves as a source of language-specific content such as language rules, word and phrase dictionaries. The database can be located on the client or on the server.

The interpretation of an e-speranto document is performed in two steps. In the first step, the replacement of e-speranto concepts and their attributes with the words in a target language (the so-called lexical transformation) takes place, while the structural transformation is performed in the second step. The interpreters have a modular architecture. The modules are divided into three layers of abstraction. A level of abstraction refers to the degree of abstractness of language structures that enter a certain module as data and on which transformations are performed. The procedures in the modules in the first layer perform language-independent operations that are common to rule-based machine translation. The procedures in the second layer are typical of a group of languages, while the ones in the third layer are language-specific.

One of the demanding problems related to the multilingualism on the Web is the number of units (translators and interpreters), needed to perform the transformation among texts in different languages. The problem is burning since there are practically no political, geographical or any other kind of obstacles for interaction among individuals. One can come across any language while using the Internet. In order to develop translators for all known 6900 languages that are spoken in the World today, about 47,610,000 of translators should be made.

One of the possible solutions to this problem is the use of the so-called intermediate language or interlingua. Interlingua is an abstract presentation of the content that is independent from any natural language [1]. The record in interlingua must contain the whole information required for generating text in a natural language. Thus, the entire meaning we want to express in a natural language must be captured in interlingua. The advantage of using the interlingua is a two-phase course of translation between two natural languages. During the process, the modules that perform the conversion from a natural language to the interlingua (translators) are independent of those that perform the opposite conversion (interpreters). Moreover, the interpreters and translators of different languages are also mutually independent. The effect of this independence is the reduction of the number of units that would be needed in case of direct mapping among the individual languages. The cost of the latter approach is as high as $n(n-1)$, where $n$ denotes the number of languages among which we want to translate. By using the interlingua approach, the cost of the interpreter development reduces to $2n$, since only a translator and an interpreter for each language must be made.

2. RELATED WORK

In the past, numerous attempts of creating an interlingua-based multilingual system were conducted. Some more notable implementations are presented in the next paragraphs.

DLT (Distributed Language Translation) [2] was a project of the development of a multilingual system in the 1980s that used an adapted version of Esperanto as an interlingua. The document written in Esperanto would be carried over the network and interpreted in a chosen language by the target computer. Although DLT presented a novel and interesting approach to machine translation, the results were not promising in practice.
3. E-SPERANTO

E-speranto [6, 7] is a formal computer language for recording multilingual texts. Its main goal is to overcome language divides in the Internet. A document in e-speranto is interpreted in a chosen language, when the user requests the document to be displayed.

Figure 1: Record of a sentence in e-speranto. The basic building element in e-speranto is a clause. A clause is a semantic unit that corresponds to a sentence in a natural language. Clauses are composed of sentence elements introduced by XML tags. The grammatical characteristics are expressed explicitly by means of XML attributes. The concepts representing the essence of e-speranto are marked in English for the sake of better intelligibility.

The basic syntax of e-speranto is based on XML (eXtensible Markup Language) which is an important technology on the Web today. XML is compatible with HTML and therefore e-speranto can be incorporated into Web pages. Syntactic and grammatical rules are taken from Esperanto, but are expressed explicitly by means of metadata.

The main advantage of e-speranto over similar approaches is especially the intelligibility of documents both to computers as well as to people and compatibility with existing Web technologies, such as XML and HTML.

4. DEVELOPMENT OF A TRANSLATION SYSTEM BASED ON THE E-SPERANTO

The development of the e-speranto-based translation system consists of four stages (Figure 2). In the first stage, the e-speranto language is developed, including the development of syntax, grammar and vocabulary. The syntactical and grammatical rules are specified in a XML Schema [8] document, which serves as a reference for validation when composing e-speranto documents.

In the second stage, the tools for writing documents directly in e-speranto are developed. For this purpose an integrated development environment (IDE) based on the Eclipse platform was developed. The IDE provides the user with the content assistance, document validation, access to the dictionaries and their effective use at the composition of a document in e-speranto.

The interpreters of e-speranto in a natural language are developed in the third stage. Currently, there is an effort taken in developing the interpreters in Slovenian, Russian, Serbian and English language. When the development in the third stage is completed, the displaying of Web pages in e-speranto will be possible.

The first three stages are already in progress. The fourth stage is optional. It includes the development of the translators from a natural language to e-speranto which is a bit more demanding than translation from e-speranto into a natural language. The reason for this lies in the fact that, in the second case, the source text is structured and computer-friendly, which simplifies the process of conversion. For this reason, the development of translators into e-speranto is not foreseen until the fourth stage of the development. If good tools for writing documents in e-speranto are developed, the fourth stage is, in fact, not needed. When the translators from natural languages into e-speranto are available, both multilingual translation between different pairs of languages as well as translating e-speranto texts into different languages will be possible.

5. OVERVIEW OF THE SYSTEM

The implementation of the translation system (Figure 3) is roughly divided into two parts: the client side and the server side. The server side is only present during the development of the e-speranto language and the interpreters. When the language is standardized, the interpreters will be moved from server to client. The functionality of the interpreters will be contained in the browsers through the plug-ins.
Figure 2: Stages of the development of the e-speranto-based machine translation system. In the first stage, the language is defined. In the second stage the tools for writing texts directly in e-speranto are developed. The interpreters of e-speranto are developed in the third stage. The fourth stage is optional. It includes the development of the translators between a natural language and e-speranto.

Client side
On the client side, the composition of the e-speranto documents takes place. For this purpose an integrated development environment (IDE) based on the Eclipse platform was developed. The IDE provides the user with the content assistance, document validation, access to the dictionaries and their effective use during the composition of a document.

In the next step of the development, publishing features that would enable automatic publishing of e-speranto documents on the Web server will be added to the IDE.

Server side
On the server side, Apache Tomcat, a Java HTTP Web server, handles the requests for the interpretation of the documents written in e-speranto. The requests are passed to the e-speranto interpreter, named INES (the INterpreter of E-Speranto). The operation of the INES is described in one of the following chapters in detail.

Figure 3: The implementation of the multilingual translation system based on the e-speranto.
Figure 4: Composition of an e-speranto document in the Eclipse-based development environment. The e-speranto IDE consists of XML editor and various views with contextual tools.

Database

The database is primarily located on the server, but can also be located on the client, where the composition of the e-speranto documents takes place. Database serves as a source of language-specific content such as language rules, word and phrase dictionaries.

6. PROCESS OF THE INTERPRETATION

When the user gives a request for the interpretation of a specific e-speranto document, Web server passes the document to the interpreter. The interface between the Internet and the core of the interpreter is written in Java programming language. In this part, the input document (Figure 1) is parsed and transformed into a symbolic tree (Figure 5). The tree is presented in a syntax common to Mathematica programming language in which the core of the interpreter was made. Beside the transformation of data structures (e-speranto documents to symbolic trees), the compilation of a “starter script” is also carried out. The script comprises symbolic trees that need to be transformed; specifications of the modules that are needed to perform the interpretation and various processing instructions.

The compilation phase is the main step of the interpretation in a selected natural language. As is the case with the majority of the interpreters of the interlingua, in INES this phase is also realized in two steps. In the first step, the replacement of e-speranto concepts and their attributes with the words in a target language (the so-called lexical transformation) takes place, while the structural transformation is performed in the second step. The interpretation in both steps is carried out with the modules that are arranged into three levels of abstraction. A level of abstraction refers to the degree of abstractness of language structures that enter a certain module as data and on which transformations are performed. The architecture of INES with some distinctive procedures on individual layers is shown in Figure 6.

The first layer comprises modules that dictate the course of interpretation and are independent of the target language. The layer is only aware of the fact that a sentence in a natural language contains the elements that
express an action or activity (i.e. the predicate) and the holder of this action or activity (i.e. the subject). This layer also contains the algorithms for movements in the tree structure. Among the various possible methods the top-bottom, left-right approach is implemented in INES. The individual subtrees are identified according to their type; their transformation is then performed by the lower layers.

The modules in the second layer are closer to the language families. These modules in general perform transformations of particular subtrees in accordance with their type. The type of a subtree is determined by the syntactic and/or semantic role the root element is performing according to the parent element in the tree representation. In general, a subtree of a certain type corresponds to a particular clause or its part in a sentence of a natural language. Figure 5 shows a subtree that corresponds to a predicate noun in a natural language.

The procedures with language-specific rules can be found in the third layer. These procedures map the parts of a tree structure to the elements of a natural language in a way that is specific to the language of interpretation. An example of such a transformation is the replacement of the e-speranto concepts and their attributes with the words of the target language or the rearrangement of the tree edges in accordance with the word order in which particular clauses appear in the target language. The access to word and phrase dictionaries is also implemented in this layer.

7. CONCLUSION

In this paper we introduced a brief overview of the multilingual machine translation system, based on the e-speranto. Our major concerns during the development are the compatibility of e-speranto with Web pages, its expressiveness and unambiguity and user friendly composition of the documents in this language.

Our further research will be towards the following fields. We intend to define the format and the content of the dictionaries in detail. Moreover, we intend to upgrade the integrated development environment with features that would enable automatic publishing of the composed e-speranto documents on the Web server. We also want to perform a more thorough research in the optimal number of layers in the layered architecture of the interpreters and determine the content that needs to be placed into individual layers. In this context, we will pursue the aim of optimally high factor of module reuse when interpreting into similar languages.
REFERENCES


