A Multi-Language Information Searching Tool

Samira Sadaoui
Department of Computer Science
University of Regina
Regina, SK, S4S 0A2
sadaouis@cs.uregina.ca

Siritorn Srisodsai
Department of Computer Science
University of Regina
Regina, SK, S4S 0A2
siritorn@cs.uregina.ca

Abstract

This paper presents a tool, namely Multi-Language Information Searching (MLIS), for a meaning-oriented search. MLIS allows users to access the right information and provides the search results with knowledge from different cultures and languages. MLIS takes advantage of agent technology as well as web services to enhance the quality of existing search engines in terms of accessibility, usability and flexibility. Their are several advantages in using MLIS, including: MLIS simultaneously performs in different languages translation and search activities as background processes hidden from users; it provides a friendly graphical user interface that instantly displays the search results in separated tabs, categorizing them according to languages chosen by users; it is proposed with a flexible architecture to automatically create multiple web-service agents based on users’ preferences; it is service-independent and can integrate, for the same session, several web services from different service providers.

1 Introduction

Search engines have become one of the most powerful and successful web applications, used for documents and information retrieval over the Internet [1, 3]. Mainly, search services find the exact keywords inside documents without consideration of the keywords’ meaning or the documents’ language [8]. Indeed, the search results usually contain links that are irrelevant to the entered keywords. Therefore, the search service confounds the concept of the Internet which aims to provide interconnected networks for sharing information among people throughout the world [10].

When an Internet user searches for information in a certain language, the entered keywords must be first translated into this language. In addition allowing users choosing the right meaning from different keyword’s definitions will be more profitable. These requests are not successful when using traditional search engines. In addition, the outcomes are not organized according to documents’ languages. Users who have experienced these language issues are unable to take full advantage of those available documents. Consequently, a meaning-oriented search is an important aspect that must be considered.

The primary objective of this paper is to develop a tool, namely Multi-Language Information Searching (MLIS), for a meaning-oriented search in different languages. MLIS is designed to utilize two important concepts in the field of distributed computing: agents [6] and web services [5]. It employs autonomous and cooperative agents as well as web services to enhance the quality of existing search engines in terms of accessibility, usability, and flexibility. These agents not only allow MLIS to fulfill and complete the translation and searching activities in a parallel manner, but also play important roles in transforming passive search engines into active ones. If we want to acquire the same results as in MLIS, users are required to manually perform several tasks for one language and duplicate them for other languages when using existing search services. In contrast, all these tasks are completely hidden and automated in MLIS. More importantly, MLIS provides a friendly graphical user interface that displays the list of URLs in separated tabs, categorizing them according to the languages selected by users. This allows users to easily navigate in the huge amount of results as opposed to existing search engines. Users can now appreciate the valuable and potential of knowledge and information published over the Internet.

The rest of this paper is organized as follows. Section 2 outlines the functional requirements and software qualities of the MLIS system. Section 3 introduces the agent development framework called Jadex used to develop MLIS, and also the BDI model for agent construction and interaction. Section 4 discusses the MLIS design, including MLIS architecture, MLIS agents, their beliefs, plans and event messages. Section 5 presents the MLIS implementation and an
2 MLIS Specification

As shown in Figure 1, the major feature of MLIS is to retrieve documents in different languages based on the keywords’ meaning (for each language). MLIS includes other functionalities to translate both queries and search results. The MLIS functionalities are given below.

- Translate Keywords. Mainly this function translates the entered keywords into another language. It employs web services for keyword translation. In MLIS, agents whose duty is to perform this functionality are called translator agents. These agents simultaneously perform translation activities depending on the language they are responsible for.

- Search Translated Keywords. Using existing search engines, this function searches a list of URLs that are related to the translated keywords. Agents which perform this functionality are called search agents. Each agent is responsible for one language and retrieving activities are simultaneously executed by search agents.

- Translate Web Sites. When users select the URLs that they are interested in, MLIS employs services for web page translation to English. Then, it displays the translated web pages.

In summary, the benefits of MLIS are given as follows.

- Accessibility. MLIS allows accessing the right documents by preserving the meaning of the entered keywords in each language. Indeed, MLIS translates the entered keywords into another language before performing a search throughout available databases of that particular language over the Internet. Moreover, the integration of qualified on-line dictionaries into MLIS can increase the efficiency of searching outcomes with respect to the meaning-oriented aspect.

- Usability. MLIS integrates different web services (keyword and web page translation, keyword search) to provide an ideal unique service. For instance, MLIS includes the translation services in order to provide users with documents that they can understand. In addition, MLIS promotes simultaneous services, most of them are running as background processes. Users can then consume only what they need without any concerns of the hidden processes. Finally, the friendly GUI displays the resulting documents in separated tabs, categorizing them according to the documents’ languages. This facilitates the navigation throughout the huge amount of results.

- Flexibility. MLIS is provided with a flexible architecture to automatically create multiple web-service agents based on users’ preferences. In addition, MLIS is service-independent, i.e. it is not limited to one particular service provider for each service type. In fact, MLIS can use different service providers for one service session. Selecting one from the best service provider or from an accessible one is another possibility supported by the design. For instance, it is possible to use a qualified on-line dictionary for keyword definitions, Alta Vista for keyword translation, Google for search service, and Yahoo for web-page translation.

- Portability. MLIS is platform-independent since it is developed with Java and XML. It can be transformed into a web application that includes existing web services: transferring, searching and translating keywords and web pages.

3 Background

MLIS is composed of multiple agents that simultaneously perform services and work together to complete the system’s objectives. Agents are self-contained, collaborative and concurrent programs [6, 11]. Basically, developing a multi-agent software system requires an agent execution environment (AEE). AEE provides a run-time system for agent execution, a standard interface for agent interaction, and services for agent creation and termination [9, 4].

In this paper, Jadex [2, 7] is selected as the AEE for MLIS, providing the platform architecture, core services and message transport mechanisms based on the FIPA specifications [4]. Jadex allows the implementation of intelligent agents using Extensible Markup Language (XML) and
Java. Jadex agents are developed w.r.t. the goal-orientation and the Belief-Desire-Intention (BDI) model. BDI consists of a set of capabilities, beliefs, goals, plans and events (cf. Figure 2) [7]. In Jadex, the construction and execution of agents require the acknowledgment of agent’s properties defined in the agent definition file (ADF). An ADF describes the structure of each individual agent (cf. Figure 2). It is also used to adequately communicate and respond to incoming messages from neighboring agents. In addition, plain Java classes are implemented for plans and activities belonging to each agent. Plans can be instantiated by other agents’ plans or activated by incoming event messages.

In MLIS, agents possess characteristics [9], such as autonomy (ability to be aware of their assigned objectives), reactivity (ability to adequately respond to others), and collaboration (ability to work with others). For instance, as shown in figure 3, the GUI, translation manager and search manager agents have initial beliefs that indicate the existence of one another on the Jadex platform. Once GUI receives the user’s keywords, it immediately sends a translation-request message to translation manager. This latter acquires the translated keywords for each language and forward them to search manager to perform retrieving activities in each language.

4 MLIS Design

As shown in figure 3, MLIS is designed with a flexible architecture. Indeed with the use of directory service agents, agents can be either attached to or removed from the system. In addition, each web-service agent can employ individual service from different service providers. All the MLIS agents wait for incoming messages and provide responses w.r.t. the events and plans defined in their respective ADFs. In this section, we give more details on MLIS agents.

4.1 Agent Types

MLIS agents are categorized into different types based on the services they perform, including:
Manager agent is responsible for the creation of translation manager, search manager and GUI agents. It requires that translation and search managers are registered and being known to the platform prior to the GUI agent. This allows the appropriate communication to take place. Once these three agents are created, the manager agent deregisters from the platform.

Translation manager agent provides two major services: creating a certain number of translator agents depending on the user’s preferences, and managing the interaction between translator agents and GUI agent.

Search manager agent creates a certain number of search agents depending on the translator agents participating on the platform. It also controls the interaction between search agents and GUI agent.

GUI agent provides a friendly graphical user interface for interaction between users and MLIS. GUI is also responsible for translating and displaying web pages.

Translator agent converts the entered keywords to the language it is responsible for. For example, Chinese translator agent translates keywords into Chinese language.

Search agent provides a searching service of the already translated keywords in the language it is responsible for. For instance, Chinese search agent receives the translated keywords in Chinese, and then searches for documents in Chinese language.

Besides the declaration of agent types in MLIS, there are other elements that must be specified according to the BDI model, for instance, beliefs, plans and events defined in each agent.

4.2 Beliefs, Plans and Events

Each MLIS agent contains a set of beliefs that is used for recognizing facts and knowledge on the agent itself and its environment. Figure 4 is an example of beliefs tag which identifies translation manager and search manager agents on the platform.

In MLIS, plans can be categorized into two main groups: (1) creation plans which are activated during agents’ creation process. All the MLIS agents are created with the default Jadex agent configurations, unless they require some additional procedures, (2) service plans which provide some particular activities, such as sending and receiving messages, or pursuing other plans. In figure 5, we give an example of all the plans used in MLIS when the user searches for information in French and Spanish languages.

Events for each individual agent must be defined. These events are defined w.r.t. FIPA-ACL for message transport mechanisms [4]. Although there are three types of events (message events, gold events and internal events), message events are mostly used in MLIS. For example, Figure 6 is the events tag defined for translation manager agent. It consists of a message event with “name,” “type” and “direction” attributes defined as “request_search,” “FIPA” and “send,” respectively. This means that the request message is a FIPA type and allows translation manager to send a request search message to search manager asking for a search service in different languages. We also illustrate in Figure 7 an example of interaction among MLIS agents based on their ADFs. The interaction consists of activating plans, sending messages and objects.

5 MLIS Implementation

MLIS is proposed with the execution of simultaneous background processes for translating and searching relevant documents. MLIS is implemented in Java (J2SE version 1.5) and XML. The MLIS user interface is implemented with Netbeans IDE environment. In MLIS, we have at least six ADF files, six agent creation plans, eleven service plans, ten result objects, and one interface form object. In addition, other ten classes for passing information via messages are included. For instance “CurrentResultForNextSearch” object is sent when the request of the “Next” is activated.

As shown in figure 8, the following describes a scenario...
when performing a search using MLIS: the user (1) selects languages they prefer, which in this particular scenario are French and Spanish; (2) enters for instance three keywords “sun, king, ring” into the searching field; (3) presses the “Search” button, and the lists of URLs related to the translated keywords in French and Spanish resp. are instantly displayed in separated tabs; (4) for each tab, the user can select multiple URLs and use the “Select” button to open them at once. These links are simultaneously translated into English.

A comparison between existing search engines and MLIS is summarized as follows. First of all, search engines perform a syntactic search while MLIS performs a search based on the keywords’ meaning in each language. Based on the previous scenario, MLIS simultaneously searches for French documents based on the converted keywords “soleil, roi, anneau”, and for Spanish documents based on the keywords “Sol, rey, anillo”. Secondly, instead of displaying the outcomes in an unorganized way, MLIS categorizes them in different tabs based on the documents’ language. This allows users to easily navigate throughout the huge amount of results. On the other hand, if we want to acquire the same results as MLIS using existing search services, the user must manually perform several actions for one language and duplicate them for other languages. In contrast, in MLIS only one step (pressing Search button) is needed to do the translation and search in different languages.

### 6 Conclusion and Future Work

In this paper we proposed an agent-based tool, called MLIS, for supporting a multi-language information search. MLIS offers an ideal unique service for a meaning-oriented search. Moreover, MLIS architecture supports a flexible choice of web services from different service providers.

One future direction consists of providing to users a list of different definitions for each individual keyword using
qualified on-line dictionaries. This allows choosing the right keywords’ meaning before starting the search.

References