World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:5, No:11, 2011

An Agent Oriented Architecture to Supply Multilanguage in EPR Systems

Hassan Haghighi, Seyedeh Zahra Hosseini, Seyedeh Elahe Jalambadani

***Abstract*—**ERP systems are often supposed to be implemented and deployed in multi-national companies. On the other hand, an ERP developer may plan to market and sale its product in various countries. Therefore, an EPR system should have the ability to communicate with its users, who usually have different languages and cultures, in a suitable way. EPR support of Multilanguage capability is a solution to achieve this objective. In this paper, an agent oriented architecture including several independent but cooperative agents has been suggested that helps to implement Multilanguage EPR systems.

***Keywords*—**enterprise resource planning, Multilanguage, software architecture, agent oriented architecture, intelligence, learning, translation.

I. Introduction

ACCORDING to the severe rivalry for obtaining today's target markets, importance of fast respond to the continuously variant requirements doubles. On the other hand, nowadays trying to make added values on each work process in organizations is emphasized and attended. These affairs need new enterprise architecture which as a proper motivation, supplies necessary flexibility for future survival in the current millennium. Enterprise Resource Planning (abbreviated as ERP) systems support such a strong architecture [2], [5], [6]. Although many explanations of ERP are given in the literature [8], we use the definition mentioned in [7] as the next paragraph:

An ERP system is a collection of independent but integrated modules, ready to be operational (designed and engineered before and based on the best practices), but customizable and changeable, which integrates key commercial and management processes (from those processes based on data) in all aspects of the organization, such as administrative, financial, commercial, human resources and production, in order to create added values for the organization. This goal is achieved via effective planning and control of all enterprise resources.

H. H. Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran (phone: 9821-2990-4136; e-mail: h_haghighi@sbu.ac.ir).

S. Z. H. Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran (e-mail: z.hoseini@mail.sbu.ac.ir).

S. E. J. Faculty of Electrical and Computer Engineering, Shahid Beheshti University, Tehran, Iran (e-mail: e.jalambadani@ mail.sbu.ac.ir).

The main purpose of the third generation of EPR systems is to localize these systems in different countries and different industries and organizations with the least price and effort. Therefore, in order to be easily used with individuals from different countries who speak different languages, these systems must be equipped with the Multilanguage capability [7]. Such a capability also supports the implementation and deployment of the ERP system in multi-national companies.

On the other hand, intelligent software agents that act as a part of a software to help a user or another software can be considered as a good alternative for implementing several languages on EPR software. In this paper, to implement several languages on EPR system, a software architecture based on intelligent agents is suggested that relies efficiently on capabilities and features of different kinds of intelligent agents.

In the second part of this paper, in addition to introducing a Multilanguage supporting subsystem, the concept of software agents are reviewed. In the third part, our suggested architecture is explained and some examples of the agents' functionality are presented. Finally, the last part of this paper includes the conclusion and future works.

II. Prerequisites

A. Software Agents

In computer science, software agents are part of software which operate as an interface in order to help users or other software. In fact, users assign the decision authority about what action must be done at each time to agents. These agents are created in order to make an easy and confident way for accomplishing tasks automatically in place of the user interference [1].

According to the Oxford dictionary, agent is defined as somebody who is allowable to do something instead of other person. Between computer and artificial intelligence society, concepts related to agents were defined beforehand with titles like software agents or intelligent agents (at the beginning of 80 decade). Despite the fact that different assumptions about agents existed before, there was a compromise about agents which said that an agent is an isolated computer system set in some environments which can accomplish some tasks flexibly and also automatically in order to reach some planned goals. In 2000, the word "software agent" indicated computer programs having two capability autonomous execution and domain-based reasoning [9].

B. Features of Software Agents

In general, agents have different features. Four important features of them are [1], [9], [10], [11]:

- Autonomy: software agents operate without humans or other agent's direct interference. Autonomy gives agents state of control over their operations and interior states.
- Sociability: agents can interact with other agents and also with human using different interaction languages.
- Reactivity: an agent can understand its environment and react to changes occurred in its environment. The environment can be real world, graphical user interface, other agents, or even Internet.
- Pro-activity: agents can start some goal-based operations without any response to their environment.

C. Capabilities of Software Agents

In order to consider software agents as intelligent agents, they must have seven attributes [1], [10], [12]:

- Interior knowledge extraction and usage
- Fault tolerance against incorrect or unexpected input data
- Usage of special symbolism and also abstraction
- Goal-based behavior
- Learning from environment
- Real time response
- Interaction using the natural language

Of course, sometimes an agent does not need all of these features. For example, application software which only consists of agent to agent interactions does not need interaction by the natural language. Also, real time responding is not necessary for most of applications which require response in a specific time period. Finally, although learning is one of the most favorite features for agents, but we can make capable agents without this feature.

D. Types of Software Agents

Based on the motion capability, ability to thought, roles, learning capability, and ability to autonomous operation, agents are divided in to 7 categories: cooperator, mobile, informative, Internet-based, reactive, composite and intelligent. In continuation of this subsection, some kinds of agents are mentioned, and also it is explained how they help users [1], [10], [11], [12], [13].

- Buyer agents: these software agents help Internet users find their required products and services. For example, when a person tries to buy from eBay, at the bottom of the page, there is a list of products which are interested by users who searched that specific product. This idea is based on this assumption that the user's tendencies are the same relatively, and they search similar products. This technology which is feasible by usage of agents is named cooperative filtering.
- User agents: these agents are used in order to accomplish user's tasks automatically. For example, some of them categorize and order electronic mails according to their requests. Also, some of them fill Internet forms according to the saved user's information.
- Supervisor agents: these agents are used in order to monitor operations of one of equipments like computer systems. For example, agents which record goods quantity in manufactures, monitor contestant's price, or observe changes in stock market are some examples of this kind of agent.
- Data mining agents: this kind of agents is one of the most useful ones in Information Technology. They are used in order to find patterns and procedures from different information resources. Using this kind of agent, users can order existing data based on his/her desired approach in order to access any information. For instance, there may be an agent which always checks changes in market situation and reports changes to the users or incorporations so that they can make decisions more appropriately.

In summary, usages of agents are appropriate for situations in which applications consist of distributed computations, environment realization and monitoring, and autonomous behavior. Since agents have reasoning capability, using their interior knowledge, received messages, and their defined goals, they can accomplish a sequence of complex computations easily. Every process control situation which must monitor real world and perform some actions in response to real time changes in the current state is a very good context for using agents. Sometimes these systems are as simple as thermometer and sometimes as complex as control systems used for atomic reactors.

E. Multilanguage support subsystem

If we want to present an EPR software package to people with different languages from different countries (or at least our neighbor countries), this package should be equipped with a subsystem which provides the ability of showing forms, reports and other software facilities in the least prevalent languages (or at least 2 or 3 of prevalent languages in our neighborhood).

General responsibilities of this system include:

- Find the output result which is presented to the user, such as reports, calendar, pictures and colors.
- Translate the output to the target language.
- Rewrite some part of the basic code of EPR software to the target language.

III. The Suggested Architecture

The suggested architecture in this paper contains the following five software agents:

1) Extraction agent
2) Selection agent
3) Code Rewriting agent
4) Machine Translator agent
5) Query Builder agent

In the suggested architecture, implementing the target language (language of the foreign user) is divided into 2 parts:

1. Assembling and categorizing what is to be translated
2. Machine translating and code rewriting

In the first part, the extraction agent receives EPR codes from the user interface and using its elementary trainings of programming languages, extracts what should be translated from these codes and sends the outcome to the selection agent. The selection agent summarizes this information for the translator, and finally the selected information as well as its translation will be stored in the Main Text summary Database.

The second part of the architecture is the code rewriting phase. In this stage the extraction agent's trainings is transferred to the code rewriting agent, then this agent gives what is to be translated to the machine translation agent, and finally this agent learns how to translate the expression (using the translations saved in the previous stage) and gives the translated text to the code rewrite agent in order to be replaced in the main code. The general scheme of the suggested architecture is presented in Fig. 1.

The functionality and interactions between the agents are described separately in the rest of this paper.

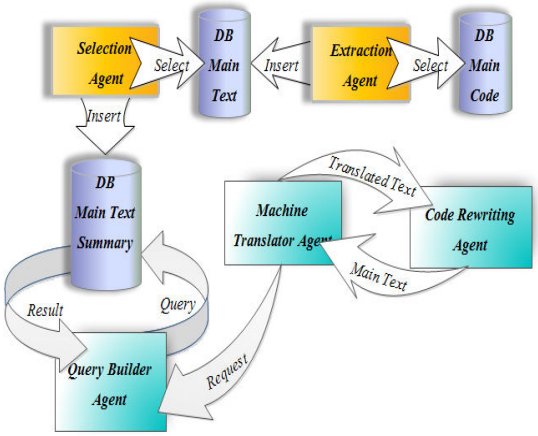

Fig. 1 The proposed agent oriented architecture

A. Extraction Agent

The extraction agent is autonomous, intelligent and trainable. This agent finds the presented output which will be shown to the user in the main code. Therefore, some elementary training has been given to this agent. For instance, the agent knows that a report will be presented to the user after the keywords "write" and "print". Based on these trainings, the agent practices and improves its rate of knowledge. At the end, the agent registers the outcome in the Main Text database which includes the output texts of the software. There is some information that is not inserted directly in the main code and should be inserted during the process of translation; such as manual files that only their address is mentioned in the main code. The content of these files is added by this agent, too.

Example: agent encounters this statement in the main code: Print ("error: not found");

The agent comprehends that the statement written between the quotation marks are probably a message that should be displayed to the user. We use the word "probably" due to the possibility that the statement between the quotation mark is the address of a file for the use by the software itself not the user. We assume that the agent does not know the keyword "Print" yet, and in order to evaluate the mentioned possibility, the agent writes a simple code by its trainings and uses this code to present its own message. If the written code executes accurately, and the agent notices its own message in the output, it will be added to its knowledge that "Print" is the statement to communicate with the user.

B. Selection Agent

The selection agent is autonomous. The responsibility of this agent is to summarize the information acquired from the previous stage. At first, this agent divides the text into some sentences, then categorizes the sentences based on their similarity and selects one sentence as the representative of each category. Finally, this agent stores all the representatives of all categories in a database named Main Text Summary. Thus, Main Text is summarized with this method.

We use a human translator for translating the summarized texts. It means that a translator translates all the available sentences in the Main Text Summary database to the target language and then registers them in the same database. In addition to the text translator, a graphical translator is applied to do the required graphical translations. The responsibility of this human agent is to find equivalents for visual but non-textual objects in the user interface; such as: colors and images.

The advantage of applying human agent and statistical translation is a more understandable translation for the users. In addition, the translator translates only a small amount of data (the representative of the whole available content in EPR) not all the available texts in the EPR; it will not take too much time and effort.

By participation of both extraction and selection agents and partaking of human agent, Main Text Summary Database has been generated so far.

C.Code Rewriting Agent

The code rewriting agent is autonomous, sociable, trainable and with a power of prediction. This agent is self initiated, reacts to the changes of the input language and starts to rewrite the code. This agent uses the trainings of the extraction agent and distinguishes which parts of the main code should be rewritten. Rewriting is the process of replacing the translated text and address of the translated file with the main text and address of the main file, respectively and adjusting the required settings, such as right aligned or left aligned, in the code in order to edit the text.

This agent uses the methods of the extraction agent for determination of dynamic and static texts. A dynamic text is a text with no variable values while a static text is a text whose content is always and in every execution constant and stable. The code rewriting agent gives what is to be translated and its type (dynamic/static) to the machine translation agent and receives the translated code from the same agent.

Using the statistics of the number of the times that every page has been visited, the code rewriting agent finds the highly visited pages and stores their rewritten code into its cache memory. This agent knows that a current visited page is likely to be visited again, so saves the pages accessed by the user into its cache memory from the beginning. After the user exits, these pages will be removed from the memory; it reduces the repetition rate and raises the processing pace.

D.Machine Translator Agent

This agent synchronized with the code rewriting agent is sociable, autonomous, reliable and trainable. This agent's responsibility is to determine an equivalent for an expression or an image. In Main Text Summary, we only have the translation of some sentences, so this agent should find the nearest sentence to the expression which should be translated. As it was expressed, the selection agent do the opposite of this action, so we certainly would find a sentence that has an adequate similarity to the respective expression. After finding the similar sentence, based on the structure of this sentence and its translation, this agent learns how to translate its expression and to find the meaning of unknown words using the DB Vocabulary.

E.Query Builder Agent

This agent is autonomous, sociable and predicts the future requests using the previous history of retrievals. The machine translator agent is not aware of the structure of Main Text Summary Database, so communicates with the database via the query builder agent: it sends its request to the query builder agent, and the query builder agent alters this request in a way that it becomes comprehensible to the database.

Example: From the structure of the main sentence, the machine translator agent comprehends to look for what to be similar to this main sentence. For instance, it requests the query builder agent to find sentences whose number of words is approximately equal to that of the main sentence, whose verb is similar to the verb of the main sentence, the order of sentence components is similar to that of the main sentence and etc. The query builder agent turns this request into several requests in the database and finally finds the one in common between the results. The query builder agent uses the cache memory to store the high frequent requests, too.

IV. Conclusion

In this paper, based on several independent but cooperative agents, a software architecture has been suggested helping to implement Multilanguage EPR systems. In this architecture some features of agents, such as trainability, predictability, intelligence, independence and cooperation to achieve this objective were expressed.

For other researches which are close to what has been offered in this paper and thus can be done in continuing this work in future, we propose to design new agent oriented software architectures in order to supply other features of ERP systems, such as having development environment for providing flexibility, gathering and using best practices and finally software distribution management.

References

[1] Coen, M., SodaBot: A Software Agent Environment and Construction System, MIT AI Lab Technical Report 1493, 1994.
[2] O'Leary, D., Enterprise Resource Planning Systems: Systems, Life Cycle, Electronic Commerce, and Risk, Cambridge University Press, 2000.
[3] Calisir, F., The Relation of Interface Usability Characteristics, Perceived Usefulness, and Perceived Ease of Use to End-User Satisfaction with Enterprise Resource Planning (ERP) Systems, Computer in Human Behavior, Vol. 20. No. 4, pp. 505-515, 2004.
[4] Grabski, S. V., Leech, S. A., Complementary Controls and ERP Implementation Success, International Journal of Accounting Information Systems, Vol. 8, 2007.
[5] Jacobs, F. R., Clay, D., Why ERP? A Primer on SAP Implementation, McGraw Hill, 2000.
[6] Sudzina, F., Johansson, B., Finding ERP Requirements that Support Strategic Management in Organizations, Proc. of Academic International Conf., Increasing Competitiveness or Regional, National and International Markets Development - New Challenges, 2007.
[7] Haghighi, H., Shahhosseini, H. S., Mobasheri, M., Enterprise Resource Planning Software: Development, Evaluation, Selection, and Implementation (In Persian), Tahsin Publisher, 2010.
[8] Sumner, M., Enterprise Resource Planning, Upper Saddle River, NJ, Prentice Hall, 2004.
[9] Henderson-Sellers, B., Giorgini, P., Agent-Oriented Methodologies, IdeaGroup Publishing, 2005.
[10] Chauhan, D., Baker, A., JAFMAS: A Multi-Agent Application Development System, Proc. of the Second International Conference on Autonomous Agents, pp. 100-107, 1998.
[11] Davies, W., Edwards, P., Agent-based Knowledge Discovery, Proc. Of AAAI 1995 Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, pp. 34-37, 1995.
[12] Bose, R., Sugumaran, V., Application of Intelligent Agent Technology for Managerial Data Analysis and Mining, Database for Advances in Information Systems, Vol. 30, No. 1, pp. 77-94, 1999.
[13] Lea, B., Gupta, M. C., Yu, W., A Prototype Multi-Agent ERP System: An Integrated Architecture and a Conceptual Framework, Technovation, ol. 25, pp. 433-441, 2005.