Intelligent Agent-based Subcontracting System in Construction

Pao H. Lin

Assistant Professor, Dept. of Civil Engineering
Feng Chia University, Taichung, Taiwan, 407
E-mail: paolin@fcu.edu.tw

Will Y. Lin

Ph.D., National Taiwan University, Taipei, Taiwan
E-mail: willsland@hotmail.com

ABSTRACT: Practically, the effective management of subcontracting selection within a construction project has been regarded as one of the critical factors for achieving project success. This study takes advantage of IT initiatives of e-commerce to combine with the use of electronic information exchange standards Java XML (extensible markup language) to propose a "push-strategic" and agent-based online subcontract bidding and negotiation architecture. The design of agent will make it possible to automatically sense the changes in its environment and react accordingly on behalf of the host. The information agent deals with all manipulation of incoming and outgoing messages following the pre-defined communication mechanism. Furthermore, it will communicate with other information agents mounted on all other contract nodes, and also with its local decision support system through the mapping rules. The ultimate goal of this study is to build an agent-based automatic subcontracting environment for Multi-Agent System (MAS) to achieve the effective communication and optimization within the process of subcontractor selection. A proto information system has been designed and implemented in the study. The proposed agent-based subcontracting system, after being made into a widespread and workable process in the future, could expectably become a new paradigm for the subcontracting procurement of the construction industry.

KEYWORDS: E-Commerce, Intelligent Agent, Subcontract, XML

1. INTRODUCTION

Due to the specialized technological divisions of labor in the construction industry, a large majority of engineering functions and values of a project are carried out by specialized engineering firms coordinated and controlled by the master contractor. Traditionally, the practice for selecting subcontractors was to take the way of choosing while working on the project, especially those who were set to work together with or those with whom one had already done business, which gave rise to quasi-firm sub-structures. Nevertheless, because of factors related to limitations on finance, manpower, time, and information in traditional procurement, there is often no way to effectively expend the sample space to try out new clients, which can give rise to inefficiency in subcontractor selection and negotiation processes. However, in recent years, the rise of the Internet and e-commerce has thoroughly changed the traditional market's business rules and has brought a revolution in transaction practices. The use of Internet-based technology initiatives makes the exchange of information simple, fast, omnipresent, and accurate, and brings a new, pivotal opportunity and force to the enhancement of the subcontracting supply chain management. Starting from and looking toward Internet, the subcontractor selection can be categorized as B2B e-procurement in this age of digital commerce. Meanwhile, the present e-procurement is mainly focused on the "pulling strategy." That is, the master contractors who call for subcontracting tenders may establish and maintain a homepage within the specific web-server to post the bidding information and then passively wait for tendering. The pulling way of e-procurement has certainly improved the past flaws of selection from limited candidates, but somehow it is apparently too passive to customer-oriented in tendering process. Recently, agent-based electronic commerce has become even

more widespread as agent and Web technology become more powerful and flexible. Therefore, beyond considering the pull way of e-commerce, the "push strategy" implemented by agent-based subcontracting system is proposed to strengthen the ability of customer-oriented e-procurement and create an automatic mechanism of data acquisition. The push-strategy agent will automatically and precisely transfer tendering information to the specific potential subcontractors, and thereafter deal properly with all relevant responses from them through the Internet. In this research, the overall subcontracting supply chain of a construction project is considered as a global procurement system and an intelligent agent-based subcontracting (ABS) system is developed to obtain optimal combination of subcontractors selection. ABS is programmed to make decisions based on the decision-maker's personal preferences. Combined use of XML (extensible markup language) and Java language allows ABS and other automated processes to access and interact with Web-based information more easily. In the proto-model, ABS shows the ability to facilitate communication, collaboration and coordination, brings more analytical power to bear in the development of solution, and reduces the amount of human intervention in organizational processes. As a matter of fact, the development of ABS is a preliminary trial to take the subcontracting and purchasing process into intelligent re-engineering through omnipresent Internet.

2. INTELLIGENT AGENT

There has been much discussion about whether a certain system is an agent or merely a program. This research does not intend to manifest the historical problem of defining these terms in artificial intelligence. However, intelligent agents could be more easily distinguished from a program due to three major features: autonomy, cooperation, and learning. Nwana defines an agent in terms of the three behavioral attributes, any two of which must be possessed by an intelligent agent [1]. (1) Autonomy: Intelligent agents can operate on their own without the need for human guidance. Therefore, they have individual internal states and goals, and act to meet their goals on behalf of their users. (2) Cooperation: Since agents are designed to act on behalf of their hosts, they would cooperate with other agents or humans. In order to cooperate, agents need to possess a social ability, i.e. the ability to interact with other agents and possibly humans via some communication language. (3) Learning: For agents to be truly regarded as intelligent, they would have to learn as they interact with their external environment since the learning ability has almost been regarded as one element of intelligence. This research adopts Nwana's suggestion to define an intelligent agents: "A program is an intelligent agent if it possess any two of behavioral attributes: autonomy, cooperation, and learning"[1,2,3,4]. Intelligent agent research grew out of Distributed Artificial Intelligence (DAI). The aim of this area was to gain the benefits of modularity when tackling large, real-world problems. With the advent of the wide usage of the Internet, new and somewhat more manageable research areas started to open up, as the Internet is essentially an ideal electronic environment for agents. In the last few years, there has been an explosion in the amount of information available on a daily basis. This information may be stored as passive stored databases and files or it may be information we need to actively request in order to make a decision. Much of this information is stored remotely in a variety of formats and sources; much of it badly labeled, and much of it time-consuming to locate. This has led to a state of affairs where traditional IT systems are increasingly hard-pressed to meet many information gathering challenges. Whereas, previously, humans would take on the role of sifting and coordinating gathered information in order to take decisions, agent-based software technology is rapidly evolving to perform all of these functions. Agents are considered particularly useful for tackling large-scale, real-world problems involving multi -disciplinary perspectives. They are currently applied to a variety of application domains including workflow management, telecommunications network management, air traffic control, business process re-engineering, information retrieval and management, electronic commerce, personal digital assistants, e-mail filtering, command and control, smart databases and so on. [1,5,6,8].

3. APPROACH

Due to the potential for a high number of subcontractors on the market, every business usually employs a certain kind of software system to manage and record documents and information. The formats and standards of each system are possibly different, and without format conversion, it is not easy to share and exchange documents and information. Because of this, an agent-based communication environment for subcontracting is

developed to achieve this goal to conquer the difficulty of problems related to different information system platforms and software compatibility as well as problems related to encryption of data heterogeneity.

Since there has long been a consensus in the industry that a standard ontology is required for efficient data exchange, in this research we follow aecXML's framework developed by World Wide Web Consortium (W3C) for data schema, and define all possible categories of subcontracting information using its terminology, and developed the data structure of XML Schema for Project Subcontracting (SPS) by modifying the aecXML framework. Meanwhile, the Data Acquisition Language for Subcontracting (DALS) is also developed using the syntax of eXtensible Stylesheet Language Transformation (XSLT) as the media for requesting standardized subcontracting information as well as the associated responses. Message Agent is endowed with SPS and DALS, and thus able to autonomously deal with all messaging tasks. Each potential subcontractor equips a Message Agent as a unique information window to automatically acquire external information and also provide relevant responses. While the master contractor receiving responses from subcontractors, the selection process will succeed to be run in the decision support system of general contractor according to the decision-making mechanism of optimal selection. The complete mechanism is some complex to discuss here, and details can be referred to Lin (2001)[7].

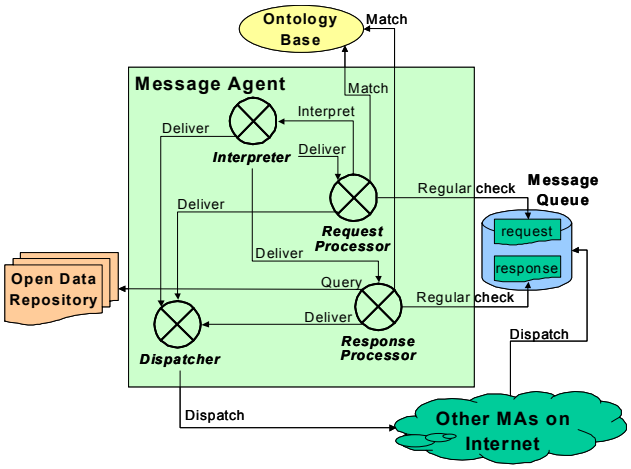

Figure 1. Architecture of Message Agent

This research assumes that all the subcontracting suppliers are equipped with a Message Agent developed by Java language. Figure 1 shows the Message Agent is composed of four major processors: Requester, Responser, Interpreter, and Dispatcher. Requester and Responser check the message Queue regularly, and retrieve the requests and responses, respectively, which are then further processed. Interpreter identifies the request and decides the procedure of request manipulation. Dispatcher dispatches the requests from Interpreter or Requester and the responses from Responser to the next destination. While generating or altering the messages, all processors either check the syntax and data structure of the incoming messages or follow the data schema according to ontology base. Figure 2 is an illustration and the schematic representation of the operation of this ABS system. The general contractor pushes specific tendering information constituted by pre-defined XML formats to the potential subcontractors (A, B, C,….., X). Each project member in this project equips a Message Agent that continuously and regularly monitors its Message Queue and performs proper message manipulation.

4. SYSTEM IMPLEMENTATION

The developing environment of the implementation of the present system is a visual programming tool for Java called Jbuilder 6.0 Personal, Borland Software Corporation. With an integrated, extensible source code editor, graphical debugger, compiler, visual designers, timesaving wizards, sample applications, tutorials, multimedia training, and support for Java standards, JBuilder Personal 6 makes learning and using Java easy. Table 1 and 2 show the details. Message Agent was implemented using Java 2 and tested in IBM PC with Windows 2000 OS. Figure 3 shows the configuration board upon booting Message Agent. The configuration such as folders, Host Name, and Host URL, is set up here at the first time of running. By clicking the button "***Save***" on the board, the configuration is saved for the following use by Message Agent. The content of XML document is cited an instance as Figure 4.

The linkages between the Message Agent and local DALS-speaking DSSs are DALS and the file folders where the DSSs put their requests in and the Message Agent accesses these requests and dispatches them. Figure 5 (a) and (b) illustrates the relationship between them. In Figure 5 (a), DALS-speaking DSSs generate DALS-based requests and deposit them in "Outbox Request Queue", which is regularly checked and dispatched by the Message Agent. The corresponding

responses to original requests are then received by the Message Agent for a certain while later, and are deposited in "Inbox Response Queue", which is checked and accessed by DALS-speaking DSSs.

On the other hand, as shown in Figure 5 (b), the Message Agent passes the incoming requests from other MAs and deposits these request in "Inbox Request Queue", which is regularly checked and processed by specific DALS-speaking DSSs. Responses corresponding to those requests are then generated and deposited in "Outbox Response Queue" by DSSs, which is regularly checked and dispatched by the Message Agent. This cycle of interoperation between Message Agent and DALS-speaking DSSs is beyond the scope of this paper due to the high complexity degree of communication between two software applications. A complete functionality of an automatic communication system should include this cycle.

Table 1. System implementation and developing tools

Language	Java
Developing Tools	Jbuilder
Operating System	Win 2000 or compatible
XML Parser	Xerces 2
XSL Processor	Xalan 2.0

Table 2. Contents of Java packages adopted by this study

Java Package	Content
org.w3c.dom	DOM Level 2 classes and interfaces
org.xml.sax	SAX 2.0 classes and interfaces
javax.xml.parser javax.xml.transform	JAXP interfaces
SAXParserFactory SAXParser	Used in all SAX applications to obtain a SAXParser object
DocumentBuilder-Factory DocumentBuilder	Used in all DOM applications to obtain a Document object (DOM tree)
TansformerFactory Transform	Used in all XSLT applications to obtain a Transformer object
org.apach.crimson	Reference implementation classes for the XML parser
Org.apach.xerces	Reference implementation classes for the XML parser (adopted in this research)

5. PROGRAMMING FEATURES

Instead of description of details of the design of objects formulated by Message Agent, several programming features are addressed here, which are multi-thread processing, parsing with a validating mode using XML Schema, and the use of Remote Method Invocation (RMI).

(1) Multi-thread processing: Multiple threads mean that there are more than one single thread running at the same time and performing different tasks within a program. Not every programming language supports the mechanism of multiple threads. Java not only supports multiple threads but also provides with many utilities including of setting thread priority, synchronizing threads, and grouping threads to manage all threads within a program. Since carrying out various manipulations of a message, the Message Agent is implemented with multiple threads and thus different manipulations of a message are able to proceed independently and smoothly.

(2) Validating a XML document using XML Schema: A DTD defines the data structure of an XML document. It specifies the order in which tags occur, what the tags are, and how many tags are allowed. A DTD provides a uniform format for defining the structure and markup of an XML document. Unlike DTDs, however, XML Schemas adhere to the XML specification and provide better support for XML namespaces and more data types. It is also a recommendation of the W3C. Schemas provide a more flexible means for defining the

structure, content, and semantics of XML than DTDs. In many areas of application, DTD is replaced with XML Schema nowadays although DTDs had been widely adopted for years. Due to the above-mentioned advantages of XML Schemas, the Message Agent adopts a validating parser using XML Schema.

(3) Use of Remote Method Invocation (RMI): Since several major manipulations of a message are involved in passing an XML-based message from a local Message Agent to remote Message Agents, an approach of file transferring from one host to another is required by the Message Agent. Although the protocol File Transfer Protocol is an easy way to be applied to this end, the Message Agent adopts a special remote access mechanism provided by Java called Java Remote Method Invocation (RMI). RMI mechanism is used by the Message Agent to pass an XML-based message from a local host to other remote hosts using specified Uniform Resource Indicators (URIs). RMI system allows an object running in one Java Virtual Machine (VM) to invoke methods on an object running in another Java VM. RMI provides for remote communication between programs written in the Java programming language.

Although the ABS system has so far not been evaluated by the industry, yet there are simple virtual scenarios tested by the authors to validate the feasibility of the whole agent system based on the push-strategy of e-commerce. And ABS shows a performance with positive results from those workable tests..

6. CONCLUSIONS

Amidst rising trends of corporate specialization, the ability to properly manage subcontracting procurement is a key factor in maintaining competitiveness. This research has demonstrated a theoretically practical framework to select the optimally combinatorial team of subcontractors, and brings an all new vision to the development of electronic procurement system in digital economy. Following the macro viewpoint of systematization and standardization, businesses can integrate with their own Enterprise Resource Planning (ERP) system to gain further benefits from their resources, and take a step toward platform systematization for different organizations and work units to meet their ultimately long-term goals.

Accordingly, this research takes advantage of IT initiatives of e-commerce to combine with the use of electronic information exchange standards Java XML to propose an agent-based online subcontract bidding and negotiation architecture. The design of software agent has made it possible to automatically sense the changes in its environment and react accordingly on behalf of the host. The information agent deals with all manipulation of incoming and outgoing messages following the pre-defined communication mechanism. Further, it will communicate with other information agents mounted on all other contract nodes, and also with its local decision support system through the mapping rules. The proposed agent-based subcontracting system based on the active push-strategy of e-commerce, after being made into a widespread and workable process in the future, could expectably become a new paradigm for the subcontracting e-procurement of the construction industry.

7. ACKNOWLEDGEMENTS

The authors would like to acknowledge the National Science Council, Taiwan, for financially supporting this work under contract No. NSC-91-.

8. REFERENCES

1. R. Murch and T. Johnson, *Intelligent Software Agents*, Prentice-Hall Inc., New Jersey, 1998.

2. M.J. Bradshaw, *Software Agents*, AAAI Press/The MIT Press, Menlo Park, CA, 1997.

3. J. Bigus, *Constructing Intelligent Agent with Java*, John Wiley & Sons, Inc., New York, 1997, 23-53.

4. Engeli, Maia, Kurmann and David, *Spatial objects and intelligent agents in a virtual environment,* Automation in Construction Volume 5, Issue 3, September, 1996, pp. 141-150.

5. Lees Brian, Branki Cherif, and Aird Iain, *A framework for distributed agent-based engineering design support,* Automation in Construction Volume 10, Issue 5, July, 2001, pp. 631-637.

6. Anumba C.J., Ugwu O.O., Newnham L., and Thorpe, *A Collaborative design of structures using intelligent agents,* Automation in Construction Volume 11, Issue 1, January, 2002, pp. 89-103.

7. Lin P. H., *An Accelerated Subcontracting and Procuring Model for Construction Projects in Digital Economy*, Ph.D. Dissertation, Department of Civil Engineering, National Taiwan University, 2001.

8. Lin W. Y., *Development of Electronic Acquisition Model for Project Scheduling Using Java-XML*, Ph.D. Dissertation, Department of Civil Engineering, National Taiwan University, 2002.

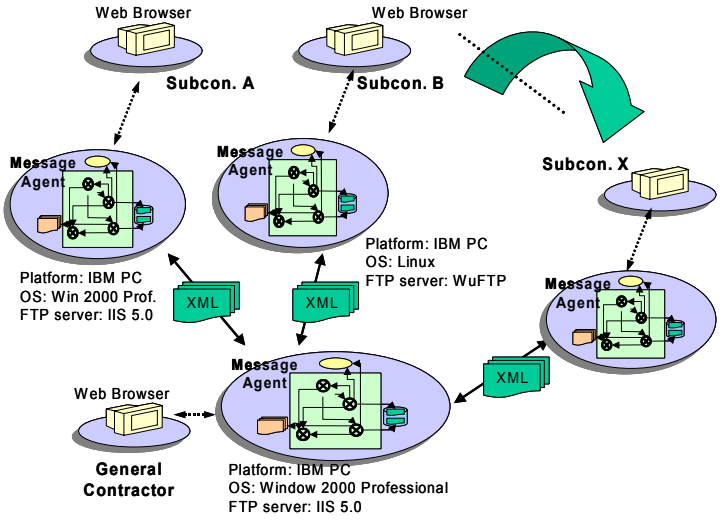

Figure 2. Schematic representation of the system

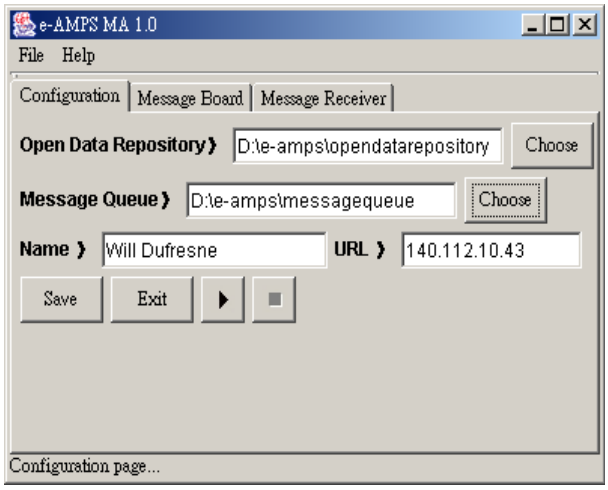

Figure 3. The configuration board of Message Agent

```
<Header>
   <Request requestId="msg101010" date="2001-01-01">
      <Sender role="GC" url="ftp://140.112.10.16/gc">Continental Company</Sender>
      <Receiver role="SC" url="ftp://140.112.10.78/sc1">Smart Excavator</Receiver>
   </Request>
   <Response responseId="msg101010" date="2001-01-02">
      <Sender role="SC" url=" ftp://140.112.10.78/sc1">Smart Excavator</Sender>
      <Receiver role="GC" url=" ftp://140.112.10.16/gc">Continental Company</Receiver>
   </Response>
</Header>
```

Figure 4. An XML instance for a response/request

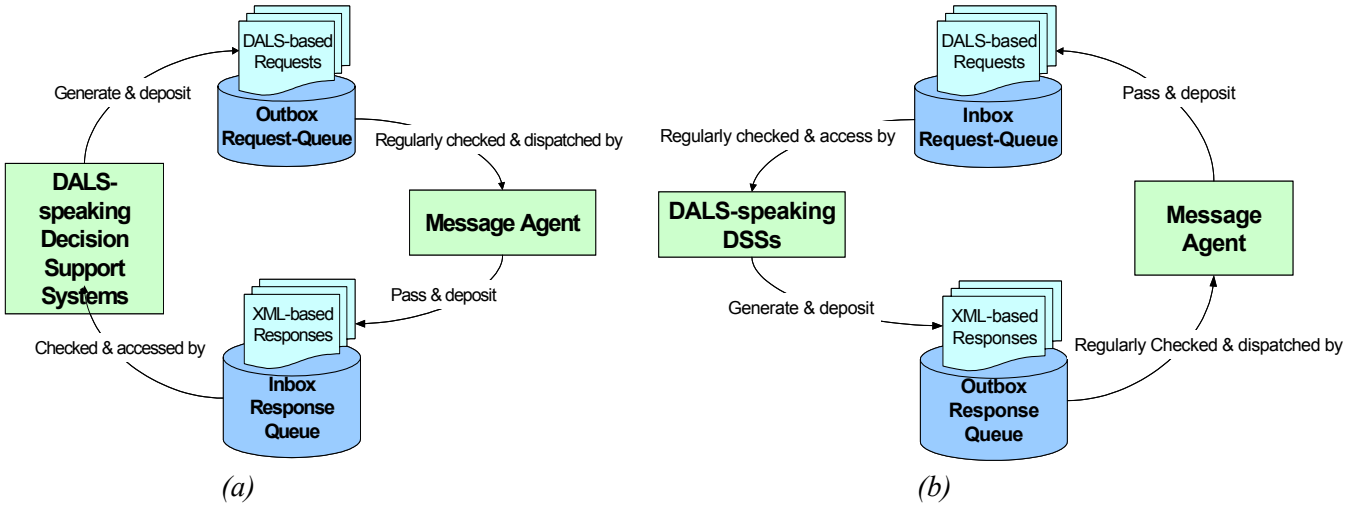

Figure 5. Interoperation between Message Agent and DALS-speaking DSSs