行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※　　營建業經驗累積與學習模式之建立(2/2)　　※
※　　　　　　　　　　　　　　　　　　　　　　　　※
※※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：■個別型計畫　　□整合型計畫
計畫編號：NSC 90-2211-E-002-074
執行期間：　90 年 8 月 1 日至 10 年 7 月 31 日

計畫主持人：曾惠斌　　國立臺灣大學土木工程學系

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

執行單位：國立臺灣大學土木工程學系

中　華　民　國　91　年　10　月　31　日

行政院國家科學委員會專題研究計畫成果報告

營建業經驗累積與學習模式之建立(2/2)

A Study on the Construction Experience Accumulation and Lesson-learn Model (2/2)

計畫編號：NSC 90-2211-E-002-074
執行期限：90 年 8 月 1 日至 91 年 7 月 31 日
主持人：曾惠斌　　國立臺灣大學土木工程學系
電子郵件信箱： hptserng@ce.ntu.edu.tw
計畫參與人員：林威延　　國立臺灣大學土木工程學系

I. Abstract

本計畫擬從營造廠的角度，建立一套「中小型營造廠商資訊及經驗累積模式」的示範性資訊架構，該架構概分為(1)基本業務型與(2)決策支援型兩大類。其中，「決策支援型」架構部份，其基本概念乃是將營造廠過去累積的專案工程經驗視為企業的資產，使其不因人事的變遷而流失，以利企業長期發展。本計畫前期應用物件關聯式資料庫、網際網路等資訊技術與架構，以及資料交換標準(XML)。以工程進度管控為例，建構一套資訊共享及決策支援的雛型系統，作為本計畫最後成果之試金石。 基本上，這工程進度管控經驗累積系統需要進度管理資訊架構及自動管控機制。本研究建立一工程進度管控經驗累積系統(e-AMPS)包括XML式的工程進度資料標準(XSS)、工程進度資料自動擷取機制(DALS)和自動搜尋(HSA)及自動傳遞(MTC)等四子系統，而這系統主要是透過一個本研究建立的資代理人(MA)來達成資料自動累積及管理之目的。

關鍵詞：決策支援、物件關聯式資料庫、網際網路, XML

In this project, a construction experience accumulation and lesson-learn system will be developed to meet the requirement of contractor's short/long-term benefit and the framework of governmental on-going IT projects. In this project, several information technologies, such as, relational database and the Internet, and an information exchange standard (XML) are adopted. We established a prototype of data sharing and decision support for construction schedule control as the touchstone for the final result of this research project. Basically, there are two major obstacles to project participants gaining efficient access to external information in a distributed data environment: the variety of data structures that project members may use, and the lack of an automatic mechanism for data acquisition. Based on the ontology defined by the eXtensible markup language Schema for Scheduling (XSS), the Data Acquisition Language for Scheduling (DALS), the Hierarchy Searching Algorithm (HSA), and an automatic mechanism called Message Transfer Chain (MTC), an Electronic Acquisition Model for Project Scheduling (e-AMPS) centralized in an information agent, Message Agent (MA), was developed. Each participant equips a Message Agent as his unique information window to automatically acquire external information and provide other participants with scheduling information as well.

II. Introduction

In this project, a construction experience accumulation and lesson-learn system will be developed to meet the requirement of contractor's short/long-term benefit and the framework of governmental on-going IT projects. In this project, several information technologies, such as, relational database and the Internet, and an information exchange standard (XML) are adopted. We established a prototype of data sharing and decision support for construction schedule control as the touchstone for the final result of this research project. While focusing on scheduling issues, this paper presents a novel methodology for automating communication among participants

during the construction phase of multi-contract projects, which will solve these two abovementioned problems. The ultimate goal of this study is to develop an automatic communication environment for multi-contract projects and thus achieve effective communication among project participants to ensure project success. In this paper, an agent-based communication environment called Electronic Acquisition Model for Project Scheduling (e-AMPS) is developed to achieve this goal. e-AMPS is centralized in an information agent called Message Agent endowed with the Hierarchy Searching Algorithm (HSA) and an automatic mechanism of Message Transfer Chain (MTC), both of which are based on the ontology defined by eXtensible markup language Schema for Scheduling (XSS) and the Data Acquisition Language for Scheduling (DALS). Each participant equips a Message Agent as a unique information window to automatically acquire external information and provide other participants with scheduling information.

In the following paragraphs of this paper, several issues on scheduling for multi-contract projects are discussed first, and problems encountered with automating scheduling in a distributed information environment are also addressed. Secondly, the framework of e-AMPS centralized in the Message Agent is introduced in order to give an overall picture of the proposed concepts. The methodology embedded in the Message Agent consisting of the ontology defined by eXtensible markup language Schema for Scheduling (XSS), the Data Acquisition Language for Scheduling (DALS), the Hierarchy Searching Algorithm (HSA), and an automatic mechanism called Message Transfer Chain (MTC) is explained in detail. Finally, a prototype is implemented using Java, and an example is provided to illustrate the effects of the automatic communication for scheduling proposed in this study.

III. 3. e-AMPS Model and Message Agent

To solve the difficulties involved in sharing scheduling information among project participants in a data-distributed environment, an agent-based communication environment called Electronic Acquisition Model for Project Scheduling (e-AMPS) has been developed. The model is centralized in an information agent called Message Agent, which is embedded with two environmental prerequisites and endowed with two intelligent capabilities as mentioned earlier. Basically, Message Agent is a computer program that deals with all messaging tasks involved in automatic communication, and will be introduced in the following sections. Each participant in the same project, named a Host or Contact Node in the following paragraphs, equips a Message Agent as a unique information window so that Message Agents in the same project can automatically communicate with each other. In this section, we introduce the basic framework of e-AMPS and the functions of Message Agent in order to give an overall picture of the proposed concepts. Figure 1 illustrates the complete architecture of e-AMPS. Figure 2 illustrates the architecture of the Message Agent, which is composed of four major processors: RequestProcessor, ResponseProcessor, Interpreter, and Dispatcher.

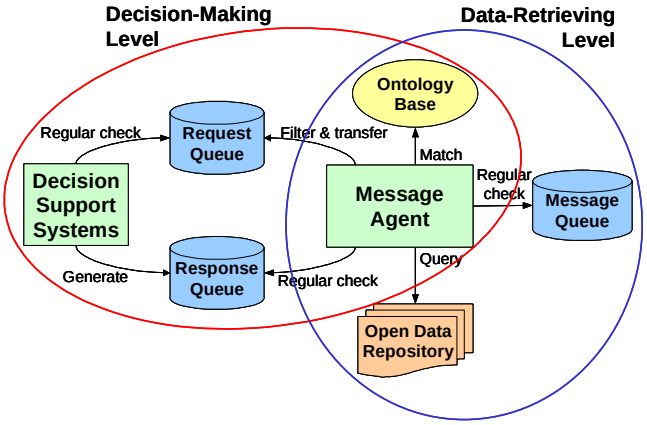

Fig.1. Complete architecture of e-AMPS

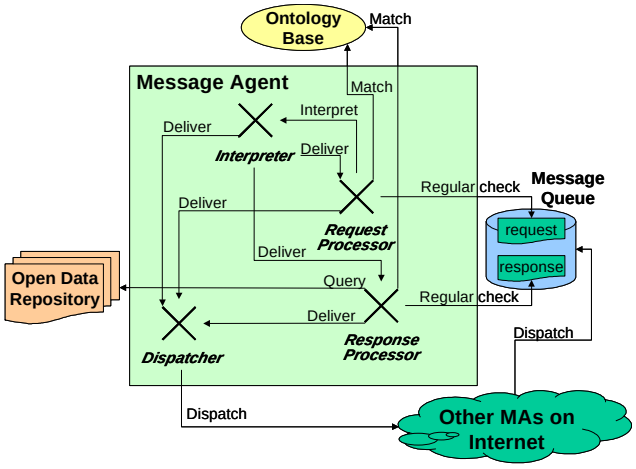

Fig.2. Architecture of the Message Agent

IV. 4. Ontology and Data Acquisition Language for Scheduling

In the framework of e-AMPS, Ontology Base, physically a local or global file folder, contains the data schema encoded by XML syntax and physically is an XML file "Scheduling_Schema.xml". All incoming information to be shared will be examined or parsed by Message Agent according to this schema file. If any violation is detected, the information will be returned to the original sender. The Ontology base ensures that all exchanged information is generated under a common data schema and therefore able to be reused after it is exchanged.

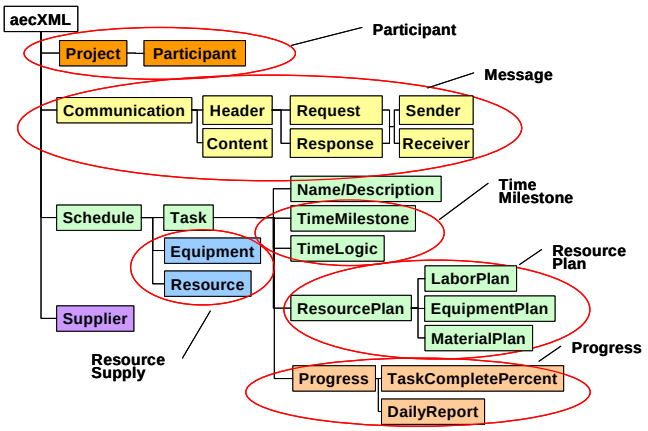

Fig.3. Data Structure of XML Schema for Scheduling (XSS)

V. 6. Implementation and an Example

A prototype of the Message Agent was implemented using Java 2 and tested in IBM PC. Operating systems included Windows 2000 Professional and Linux, with the FTP servers IIS 5.0 and WuFTP, respectively. A simple architecture of 3 project members is shown in Figure 4, which develops a typical distributed data management environment. Since Message Agent is an autonomous system that automatically detects the incoming requests and responses from other Message Agents, there are no user interfaces available. However, applications developed for project management can retrieve the information acquired by the Message Agent for further uses, such as representing progress information in web browsers. Several hypothetical original requests were made to trigger and test the whole message transfer process, which will be illustrated in the next section.

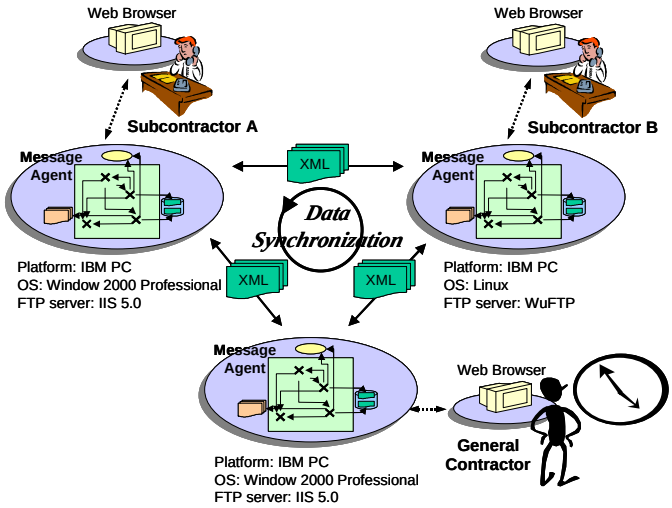

Fig. 4 A simple architecture of 3 project members in a DDM environment

Table 1 details the major module objects and their methods implemented in a Message Agent.

Module Object	Method	Function
Dispatcher	**boolean** checkTimeConstraint()	Filter out the tasks that doesn't meet time constraint
	boolean checkScopeConstraint()	Filter out the tasks that doesn't meet scope constraint
	string getMessenger()	Get the contact information of upper or lower messengers in Host's contract accordingly.
	void updateHeader()	Add headers for bypass request and response
	void sendMessage()	Send message to the receiver of the last message node in the header
RequestProcessor	**string** unwrapStylesheet()	Unwrap the wrapped query pattern using time and scope functions to a standard XSLT document
	void updateRequest()	Update the incoming request
Interpreter	**string** interpretDuring()	Interpret the time constraint function *During()* into standard XSLT syntaxes
	string interpretThrough()	Interpret the time constraint function *Through()* into standard XSLT syntaxes
	string interpretWithin()	Interpret the time constraint function *Within()* into standard XSLT syntaxes
	string interpretMeet()	Interpret the time constraint function *Meet()* into standard XSLT syntaxes
	string interpretStartBefore()	Interpret the time constraint function *StartBefore()* into standard XSLT syntaxes
	string interpretFinishAfter()	Interpret the time constraint function *FinishAfter()* into standard XSLT syntaxes
ResponseProcessor	**void** performQuery()	Perform the transformation between Schedule.xml and unwrapped request.xsl
	void updateResponse()	Update the incoming response

Figure 5 is a hypothetical project to illustrate more fully the concept of autonomous data acquisition proposed by this paper. The contractual relationships and corresponding schedules of the general contractor and eleven subcontractors are shown in the figure. The entire project starts on Jan 1st, and finishes on Apr 10th.

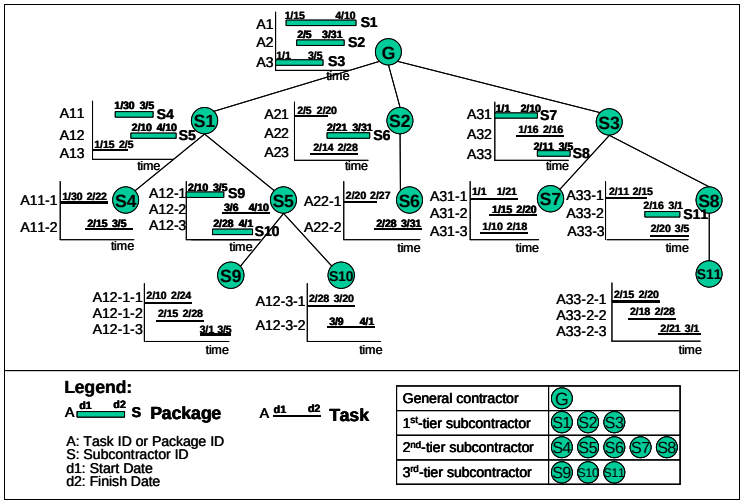

Fig. 5 Example Project for Illustrating the Automatic Data Acquisition

On Feb 1st, Subcontractor S5 originates a request for the progress information of all tasks whose duration overlaps a period between Feb 1st and Feb 9th, which is about one week before S5's task A12-1 starts. The original request without the header created by a pre-programmed process at S5's site is deposited in the Message Queue, as shown in Table 6, and waits for S5's Message Agent to dispatch it. S5's Message Agent detects this request and automatically performs the HSA. Since none of the tasks of S5's meet the Time Constraint (between Feb 1st and Feb 9th), S5's Message Agent decides to dispatch the request to its upper messenger, S1, and generates a complete request as shown in Table 2.

Table2. Request from Subcontractor S5

```
<?xml version="1.0" encoding="BIG5" standalone="yes"?>
<aecXML xmlns="x-schema:D:\ASCE\schema\Schedule_Schema.xml">
  <Communication>
    <Header>
      <Request requestId="msg101010" date="2001-01-01">
        <Sender role="SC" url="ftp://140.112.10.16/sc">S5</Sender>
        <Receiver role="SC" url="ftp://140.112.10.78/sc">S1</Receiver>
      </Request>
    </Header>
    <Content>
      <TimeConstraint>During("2001-02-01","2001-02-09")</TimeConstraint>
      <ScopeConstraint>Ancestor("P0001")</ScopeConstraint>
      <Pattern>
          <xsl:template match="/">
          <xsl:apply-templates>
          <xsl:value-of select="aecXML/Schedule/Task/Progress/DailyReport
                                [@date=During("2001-02-01","2001-02-09")]"/>
          </xsl:template>
      </Pattern>
    </Content>
  </Communication>
</aecXML>
```

In summary, by sending the original request to the communication network, S5 received five pieces of responses generated by S7, S3, S2, S4, and S5 through Message Transfer Chain. 19 messages including the original request, five bypassed requests, five original responses, and eight bypassed responses were created in total.

e-AMPS is centralized in an information agent called Message Agent endowed with the Hierarchy Searching Algorithm (HSA) and an automatic mechanism of Message Transfer Chain (MTC), both of which are based on a common data schema for scheduling (XSS) by eXtensible Markup Language (XML) and the Data Acquisition Language for Scheduling (DALS). Each participant equips a Message Agent as a unique information window to automatically acquire external information and provide other participants with scheduling information. Since there has long been a consensus in the industry that a standard ontology is required for efficient data exchange, in this paper we follow aecXML's framework for data schema, and define all possible categories of scheduling information using its terminology, and developed the data structure of XML Schema for Scheduling (XSS) by modifying the aecXML framework, which included two XML-based files: contract.xml and schedule.xml. Meanwhile, the Data Acquisition Language for Scheduling (DALS) is also developed using the syntax of eXtensible Stylesheet Language Transformation (XSLT) as the media for requesting standardized scheduling information as well as the associated responses.

VI. Self-evaluation of Research Results

The project result have been published in 3. Tserng, H. P. and Lin,W.Y., (2003). " Developing an Electronic Acquisition Model for Project Scheduling (e-AMPS) Using XML-based Information Standard," International Journal of Automation in Construction, Vol. 12, pp.67-95. (SCI, EI)

VII. Reference

[1] aecXML web-site (2000), http://www.aecxml.org/

[2] Gijsbertus, T. Bart Luiten & Frits P. Tolman, Automating communication in civil engineering, Journal of Construction Engineering and Management, ASCE, 123(2)(1997), 113-120