SOFTWARE AGENT FOR DESIGN-BUILD PROCESS COOPERATION

Ming-Hsiu Tsai
Department of Construction Engineering
National Taiwan University of Science & Technology
#43,Sec.4,Keelung Rd.,Taipei,106,Taiwan,R.O.C
D8805103@mail.ntust.edu.tw

Min-Yuan Cheng
Department of Construction Engineering
National Taiwan University of Science & Technology
#43,Sec.4,Keelung Rd.,Taipei,106,Taiwan,R.O.C
myc@mail.ntust.edu.tw

Abstract: This paper describes a research project that attempt to develop a process integrating method and finally to create a multi-agent system to support the cooperation in a design-build team. The approach taken is to apply Business Process Reengineering (BPR) philosophy for providing a process-based fast-tracking execution mechanism for design-build teams. An Integrated Design-Build Framework (IDBF) is proposed which consists of three layers; namely, (1) fast-tracking model, (2) integrated process model and (3) design-build multi-agent system (DBMAS). Design-build projects are divided into design-build modules with overlapping relationships firstly, and the specific integrated process model, then, is generated to be the foundation of the performance of fast-tracking model. Finally, applying the multi-agent system technology, the DBMAS is developed for assisting implementations of all processes embedded in design-build modules via information sharing/exchanging and process controlling functions.

Keywords: design-build, fast-tracking, business process reengineering (BPR), multi-agent system (MAS), workflow.

1. INTNTEGRATED DESIGN-BUILD FRAMEWORK

Design-bid-build (D/B/B) was the project delivery process of choice for most of the twentieth century. By the year 2000, traditional D/D/B was still used in nearly two-thirds of projects in the United States [1], and a similar situation prevailed in Taiwan. However, several problems such as “differing goals of designer and constructor,” “defects of lowest bid contracting,” and “segregation between the work of the designer and the input of constructors” bring anxiety when D/B/B methods are applied in major construction projects of greater complexity. For these reasons, design-build (D/B) has also been considered an appropriate method for contracting in Taiwan.

D/B is considered to be the fastest project delivery system as it encourages fast-tracking of design and construction phases. However, the design-builders’ unfamiliarity with the D/B process has decreased the benefits of the D/B method in Taiwan. Not only the front-end problem of defining user needs and translating those needs into a facility program and technical performance requirements has proved a wrinkle for owners, but also changing adversarial processes to collaborative ones for professional designers and general constructors is critical for the design-build team. Furthermore, due to the limitations of Taiwan laws and regulations, A/E and GC keep to their own spheres, so that functional intervals between them bring difficulties in cooperating as a D/B team [5]. For these reasons, it is desirable to increase the efficiency of cooperation and fast-tracking by the D/B team, and toward that end this study aims to reengineer the business processes across professional design (PD) and GC, so that processes within a D/B team can be integrated into more cooperative processes spanning organizational boundaries. Hence, this research applies the Business Process Reengineering (BPR) philosophy [4,6] and extends it to the cross-organizational process to integrate fast-tracking processes between PD and GC.

Accordingly, by combining the fast-tracking mechanism with the process reengineering philosophy, a project of the Integrated Design-Build Framework (IDBF) was invoked in 2005 whose goal is to develop a process integrating method and finally to create a multi-agent system to support the cooperation in a design-build team. Since the IDBF is a long-term project, a two-layer framework aimed at fast-tracking model creation and performing was preliminarily addressed in 2005 [5] (as shown on the top and middle layers in Figure 1).

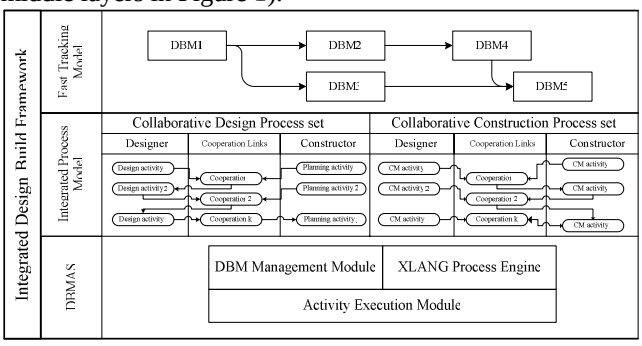

Figure 1 Integrated design-build framework.

The two-layer IDBF proposed a process-based mechanism for fast-tracking construction implementation in D/B projects. A D/B project can be divided into design-build modules (DBMs) with their overlapping relationships in its corresponding fast-tracking model. Theoretically, each design-build module can be fulfilled by its specific collaborative design and construction processes in the integrated process model, the middle layer of the IDBF. However, the complexity in practice, arising from the performing of multiple processes, decreases the feasibility of the two-layer IDBF. To overcome this defect, this paper continuously develops the design-build multi-agent system

(DBMAS, the bottom layer in Figure 1) for supporting the implementation of multiple processes embedded in DBMs so the original IDBF is extended as a three-layer framework. Consequently, this paper is aimed at developing the DBMAS based to the function requirements derived from the fast-tracking model and integrated process model.

2. FAST-TRACKING MODEL

The fast-tracking model is the preliminary for creating IDBF because it is the goal for which the integrated processes of the D/B team will perform. Based on the fast-tracking model, the integrated process model can then be determined. The purpose of the fast-tracking model is to create a scheme to fast-track construction entities and delineate their overlapping relationships. Generally, a construction entity must be fulfilled through its design and building activities. Accordingly, this study packages the design and the building activities into a DBM corresponding to a construction entity, and determines the concurrent relationships among all design-build modules to overlap the construction entities. To this end, based on the axiomatic design methodology as shown in Figure 2, the concurrent relationships of both design proposals (DPs) and of construction process variables (CPVs) need to be determined. By mapping a D/B project into customer, functional, physical and process domains by zigzagging decomposition and creating dependency matrices [5].

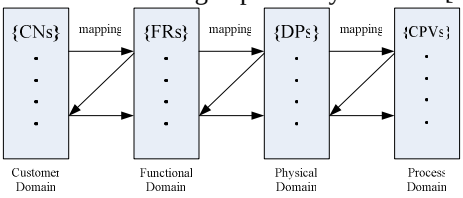

Figure 2 Four Domains in Axiomatic Design

Figure 2 shows the mapping relationships of AD, in which each domain consists of specific characteristic vectors such as customer needs {CN}, functional requirements {FR}, design proposal {DP} and construction process variables {CPV}.

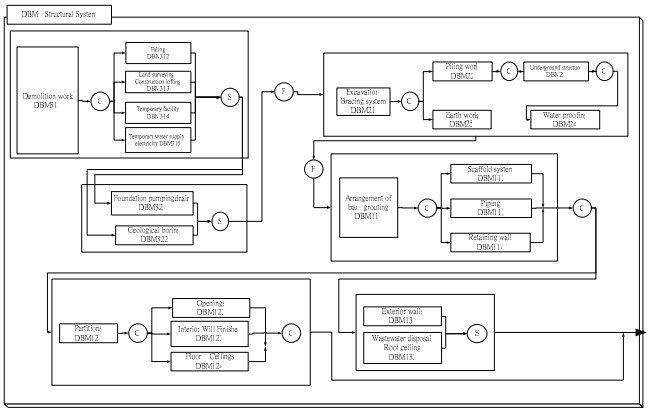

Figure 3 Structural system DBM of the fast-tracking model of case study

Figure 3 shows a partial fast-tracking model derived form a case study, in which the related DBMs are connected with the concurrent relationships so that it provides a guide map for fast-tracking construction.

3. INTEGRATED PROCESS MODEL

Each design-build module in the fast-tracking model is fulfilled through executions of designer's and of constructor's business processes. Therefore, this study subsequently addresses a cross-organizational process reengineering method to integrate processes in a D/B team, since design-build process flows across two strategic business units. To this end, the cross-organization process reengineering method as shown in Figure 4 is addressed to generate the integrated process model for facilitating collaboration on the processes in a D/B project.

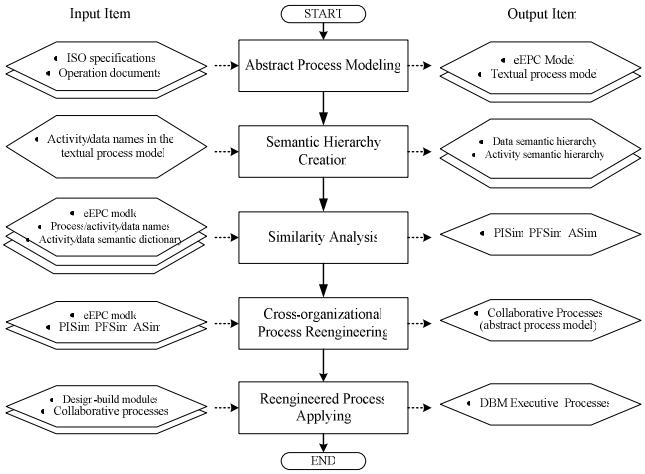

Figure 4 Cross-organizational process reengineering procedure

Based on the input-operation-output paradigm, from an information processing viewpoint, the cross-organization collaborative links between activities/processes can be determined to present the interoperability between two organization's processes. To identify the collaborative links, this research applied semantic similarity analysis to evaluate the data and function similarities between designer and GC's processes so the overlapping and coupling relationships between then can be determined. Based on the determined relationships, the integrated processes model can be created as Figure 5 shows. The integrated process model depicts the coupling processes and the information flows between designer and GC in a D/B team, and can be applied to all DBMs since it is an abstract process model. Therefore, by specifying the particular input/output information to the integrated process model corresponding to a DBM, the model becomes the execution processes corresponding to the DBM. Summarily, DBMs, in the fast-tracking model, with their corresponding integrated processes present the scheme of execution for fast-tracking construction in a D/B project.

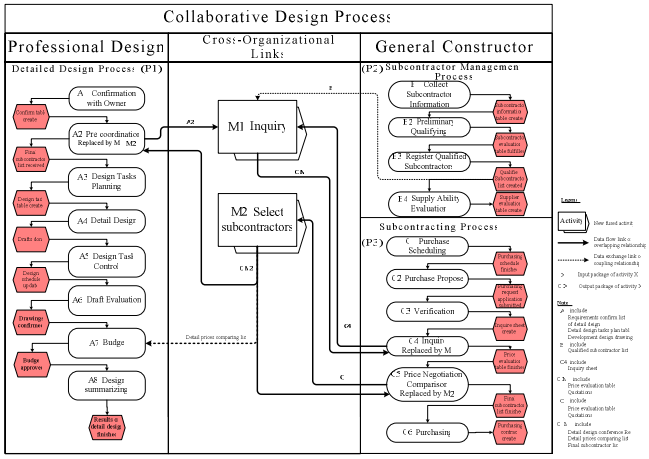

Figure 5 Example of collaborative design process set in the integrated process model

4. DBMAS DEVELOPEMNT

The bottom layer of the IDBF is addressed as the Design-Build Multi-Agent System (DBMAS) which is the infrastructure for implementation of the fast-tracking model and the integrated process model.

The created fast-tracking model and the integrated process model are both concept models and are difficultly implemented due to their complexities. To overcome this problem, the DBMAS is developed for automatic implementation of the fast-tracking model and integrated process model.

Moreover, since the collaborative processes in the integrated process model flow across two independent organizations, the information of the collaborative processes is distributed over heterogeneous information environment. Consequently, the DBMAS needs the capability to exchange data between the legacy systems without interference with the functions of the original systems. Therefore, the DBMAS needs (1) to proceed with the fast-tracking model and manage the procedure of design-build modules' executions, (2) to drive the collaborative processes of the integrated process model to be executed by computer, and (3) to dispatch activities to the corresponding actors and acquire information form the original information system to provide the consistent data to the actors. To these aims, this research combines the Microsoft BizTalk Framework and the multi-agent system philosophy for developing the DBMAS, and Figure 6 shows the system architecture. Basically, the system is composed of three modules; namely, (1) XLANG process engine, (2) DBM management module and (3) activity execution module.

Firstly, the XLANG process engine is the core module of the DBMAS, and it is developed based on the XLANG specification which is an extension of web service description language (WSDL) that defines the behavior and orchestration of business processes [7,8]. All collaborative processes embedded in DBMs are translated into their corresponding XLANG files which are XML-based process definition files, so that the XLANG process engine can drive processes to be executed automatically according to the input XLANG files. Secondly, the DBM management module is responsible for controlling the execution statuses of DBMs and collaborative processes. Two objects, the "DBM manager" and the "process monitor", are involved in the DBM management module, where the DBM manager can activate a new specific DBM according to the activated DBMs' achievement statuses. While a new DBM is activated, the DBM manager will inform the process monitor to generate and input the corresponding XLANG process definition files to the process engine. Then, the process monitor will communicate with the process engine to continuously master conditions of the process executions.

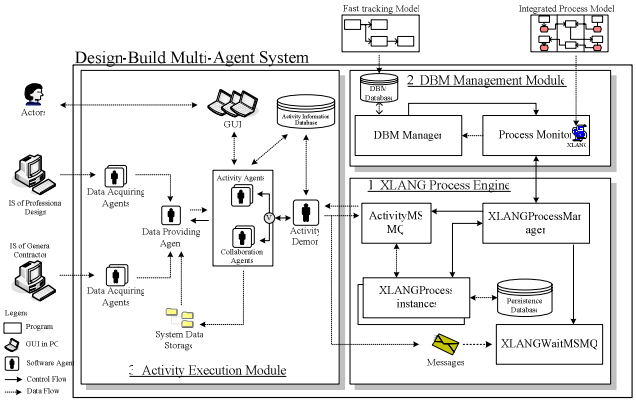

Figure 6 The system architecture of DBMAS

According to the inputted XLANG files, the process engine is responsible for issuing starting messages of the newly activated activities within the XLANG files to the activity execution module and receiving the finished messages from the activity execution module; i.e., the XLANG process engine only drives processes, but the exact processes performance relies on the activity execution module to realize the activities of processes. To assist actors with completing the dynamically activated activities and acquire the corresponding necessary data are the primary missions of the activity execution module. This study developed the activity execution module based on the multi-agent system philosophy due to its delegation and autonomy characteristics. Thus, in the activity execution module, activity agents and collaboration agents are responsible for assisting the actors with the necessary information; the data providing agent and the data acquiring agents are assigned to acquire the necessary data from the existing information systems or databases to activity agents. Summarily, on the one hand the DBM management module cooperates with the XLANG process engine to manage the execution of the fast-tracking model and the integrated process model; on the other hand the activity execution module fulfills activities within the XLANG processes of an activated DBM. The DBMAS can be implemented upon the original information environment without interference with the original information systems.

3.1 DBM Management Module

The DBM management module is the control center of the DBMAS. The primary target of this module is to manage

the execution statuses of the all design-build modules within the fast-tracking model. That is, activating design-build modules and modifying their execution statuses are main responsibilities of the DBM management module. As Figure 6 shows, the management module is composed of a DBM manager, a process monitor and a DBM database. The DBM database not only stores the DBM names and relationships determined in the fast-tracking model, but also records the execution statuses during the runtime of the system. Therefore, the DBM manager can control the statuses of design-build modules of a D/B project according to records within the DBM database. Moreover, when a DBM is determined to be activated, its collaborative design process and construction process are needed to be invoked subsequently. To this aim, the process monitor is assigned in the DBM management module.

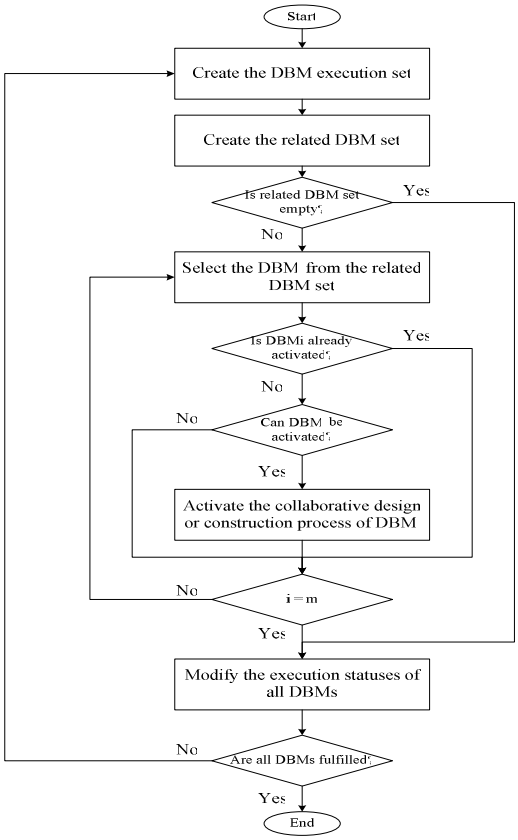

Figure 7 Controlling algorithm of DBM Manager

Figure 7 shows the controlling algorithm of the DBM manager. The DBM manager needs to create a DBM execution set by querying the DBM database. All the activated design-build modules will be involved in the DBM execution set, and the DBM manager, then, can search the database for all the related design-build modules to generate the related DBM set corresponding to the current DBM execution set. The related DBM set is composed of the candidate design-build modules for being activated. The manager needs to check the execution status of the DBM one by one. Four execution statuses are applied for the DBM; namely, (1) design-activated, (2) design-finished, (3) construction-activated and (4) DBM-finished. The DBM manager can activate a DBM's collaborative design or construction process according to the execution statuses of the predecessor and the successor design-build modules and their fast-tracking relationship. Completing the activation task, the DBM manager finally modifies the execution statuses of all design-build modules according to their collaborative processes achievement statuses which are provided by the process monitor.

Moreover, the process monitor is the bridge between XLANG process engine and the DBM management module. Two functions are assigned to the process monitor; namely, (1) gathering the proceeding statuses of the processes in the XLANG process engine and (2) generating the corresponding XLAGN files of the activated DBM.
First, by calling the XLANGProcessManager container of the XLANG process engine, the process manager can receive the proceeding status of an activated process and pass the status information to the DBM manager to be the reference for the DBM status checking. Second, as the collaborative process sets of a DBM needed to be activated, informed by the DBM manager, the process monitor has to generate the corresponding XLANG files and input the files to the XLANG process engine.

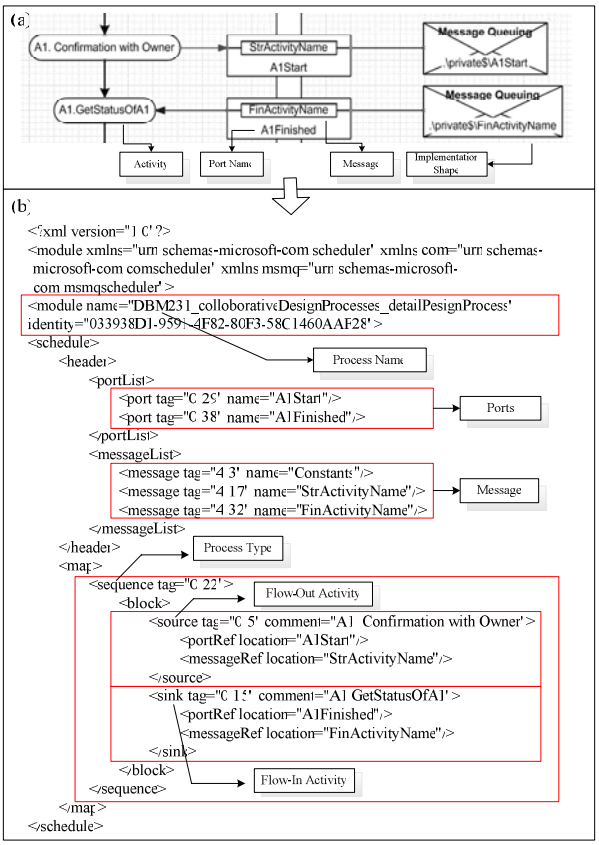

Figure 8 (a) Example of XLANG diagram (b) XLANG file content compiled from the XLANG diagram

XLANG is the underlying orchestration language for Microsoft BizTalk which is expected to serve as the basis for automated protocol engines that can track the state of executing process instances and help enforce protocol correctness in message flows. Thus, to perform processes

automatically, this study compiles the processes into the executable XLANG process files and applies the XLANG process engine to execute the complied processes.

An XLANG process file is an XML-based description file. Basically, four fundamental elements are used to descript a process. Figure 8 shows a part of XLANG diagram of the collaborative design process and its corresponding XLANG file derived from the Figure 5.

In this research, the XLANG process file is applied to send an activity's starting information to the message queuing center and receive its finishing information from the message queuing center. Therefore, each activity in the integrated process model will be mapped to two activities in the XLANG process; i.e., one activity sends the starting information, and the other receives the finished information. Figure 8(a) shows the example of the "A1. Confirm with the owner" activity. Following the XLANG diagram, the XLANG process file can be generated as shown in the Figure 8(b).

3.2 XLANG Process Engine

The XLANG process engine is the core of the DBMAS, which is developed to drive XLANG processes. The primary responsibilities of the XLANG process engine are scheduling the executions of all activated processes and persisting the long-running process. Therefore, seven objects are involved in the XALNG process engine. Figure 9 shows the logic architecture of XLANG process engine.

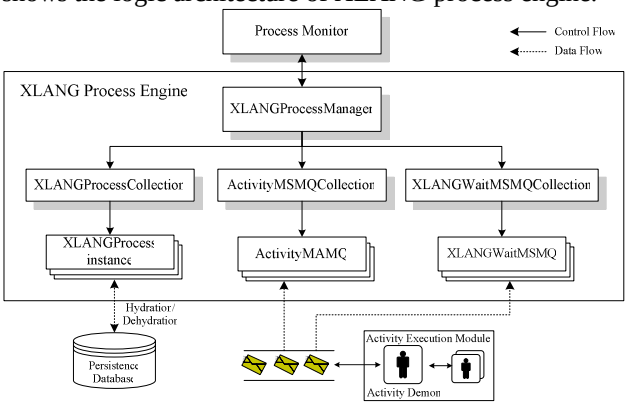

Figure 9 Logic architecture of XLANG Process Engine

1. XLANGProcessManager: is core object of the XLANG process engine because it not only parses the XLANG files to an executable instance in the run time of the system ,but also manage the process execution status in the system
2. XLANGProcessCollection container and XLANGProcess instance: The XLANGProcessManage can parse XLANG files and create the corresponding XLANGProcess instances to be executed in the process engine. All the executing XLANGProcess instances need to be stored in the XLANGProcessCollection container. An XLANGProcess instance will be persisted to the persistence database when it receives a timeout message, which is called "Dehydration". On the contrary, as receiving the waiting message, the dehydration XLANGProcess will be recovered by the XLANGProcessManager, which is called "Rehydration".
3. ActivityMSMQCollection container and ActivityMSMQ: This study applies the Microsoft Message Queuing (MSMQ) technique to be the message gateway between the XLANG process engine and the software agents in the activity execution module. The ActivityMSMQ is responsible for passing the starting and the finished messages of activities. On the one hand, as an activity within an XLANGProcess is started, the XLANGProcessManager will inform the ActivityMSMQ to send a "activity Started" message to the MSMQ server and request the "activity Finished" message while the activity is finished; on the other hand the activity demon will receive the "activityStarted" message from the MSMQ server and will also inform the ActivityMSMQ to send the "activity Finished" message to the MSMQ server.
4. XLANGWaitMSMQCollection container and XLANGWaitMSMQ: Like the mechanism of the ActivityMSMQ, the XLANGWaitMSMQ is applied to pass the "Rehydration" message between activity demon and the XLANGProcessManager.

3.3 Activity Execution Module

The activity execution module is developed based on the multi-agent system philosophy [9,10]. Figure 10 shows the architecture of the activity execution module. Five agents are developed in this module; namely, (1) activity demon, (2) activity agent/s, (3) collation agent/s, (4) data providing agent and (5) data acquiring agent/s. Besides, to collaborate with the agents, a MAS server and an agent name server are needed in this module.

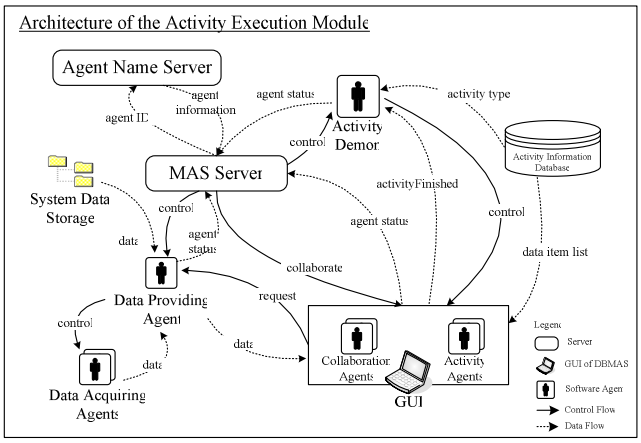

Figure 10 Multi-agent system architecture of Activity Execution Module

Agents will function according to the messages from the other agents or systems. Therefore, this study applies the Agent Communication Language (ACL) to be the specification of the messages. Table 1 shows the responsibilities of the objects in the activity execution module.

Table 1 Agent responsibility of the Activity Execution Module

Agent Name	Responsibility	Communicate with
MAS Server	Collaborate with all agents and delegate tasks.	All other objects
Agent Name Server	Register and record the invoked agent's information, such as ID, name, network address and abilities.	1. MAS server
Activity Demon	1. Acquire the activity starting message form the activityMASQ of the XLANG process engine. 2. Assign the activated activity to its actor and invoke the corresponding activity agent or collaboration agent.	1. MAS server 2. activityMSMQ 3. activity agents 4. collection agents
Activity Agent	1. Provide a GUI for actors to interact with the actors. 2. Provide the necessary data items corresponding to one activity. 3. Submission of achievements of activities	1. MAS server 2. activity demon 3. data providing agent
Collaboration Agent	Similar with the activity agent, but collaboration is corresponding to the collective activities in collective design processes.	1. MAS server 2. activity demon 3. data providing agent
Data Providing Agent	1. Receive the data requests from activity agents and return the results. 2. Delegate data acquiring tasks to the data acquiring agents.	1. MAS server 2. activity agents 3. collaboration agents
Data Acquiring Agent	Acquire the specific data from the existing information systems or databases of professional design and GC.	1. data providing agent

Summarily, the DBMAS provides a platform for execution of fast-tracking model and integrated process model. On the one hand, the collaborative processes embedded in DBMs can be performed as workflows so that the statuses and results of process executions can be monitored can controlled; on the other hand, the software agents in the DBMAS integrate the designer and the GC's information systems based on the integrated process model. Accordingly, the integration between designer and GC can be fulfilled strategically based on IDBF.

5. CONCLUSION

The Integrated Design-Build Framework (IDBF) provides a process-based mechanism for performing fast-tracking construction. Three models are necessarily created in the IDBF, namely, (1) the fast-tracking model, (2) the integrated process model and (3) design-build multi-agent system. Following the IDBF architecture, this study purely aims at the integrated process model creation and design-build multi-agent system development.

To realize design-build modules in the top layer of IDBF smoothly, the integrated process model needs to be generated. Based on the semantic similarity analysis, the overlapping and coupling relationships between professional design and GC's organizations can be identified; therefore, the collaborative processes of integrated process model can then be generated. In short, the integrated process model integrates the professional design and GC's processes into a set of collaborative processes which can serve as a reference model for members of a design-build team to cooperate with each other. In this way, the double-waste efforts and redundant activities of overlapping processes can be decreased.

However, the collaborative processes embedded in all design-build modules will increase complexity in practice. Thus, this study developed the design-build multi-agent system to support the cooperation called for by integrated process model and the fast-tracking model. To drive processes and exchange data between two organizations, the DBMAS is basically composed of a process driving engine called XLANG process engine and an activity execution module consisting software agents. On the one hand the XLANG process engine is responsible for activating the collaborative processes to be executed; on the other hand the activity execution module is responsible for executing activities within the activated processes via interacting with actors and acquiring data from the existing information systems and databases.

This research completes the development of the IDBF by extending a design-build multi-agent system to the IDBF. Thus, the fast-tracking model with its collaborative processes can be implemented by the design-build multi-agent system, and the feasibility of IDBF can consequently be increased.

REFERENCES

[1] J.L. Beard, Design-build: planning through development, New York : McGraw-Hill, 34-36, 2001.
[2] P. Huovila, L. Koskela, and M. Lautanala, "Fast or concurrent: The art of getting construction improved", Proc. of 2nd Workshop on lean construction, Santiago, Chile, 143-158, 1994.
[3] G. Williams, "Fast-track pros and cons: Considerations for industrial projects", Journal of management in engineering, ASCE, 11(5), 24-32, 1995.
[4] M.Y. Cheng, M.H. Tsai, "Reengineering of Construction Management Process", J. Constr. Eng. Manage. 129(1), 105-114, 2003.
[5] M.Y. Cheng, M.H. Tsai, "Cross-Organizational Process Integration in Design-Build Team", Proc. Of The 22th International Symposium on Automation and Robotics in Construction, Ferrara Italy, 2005.
[6] M.Y. Cheng, M.H. Tsai, Z.W Xiao, "Construction Management Process Reengineering: Organizational Human Resource Planning for Multiple Projects", Automation in Construction, accepted on Oct.14, 2005.
[7] "BizTalk Framework 2.0", http://www.microsoft.com/biztalk/techinfo/BizTalkFramework20.doc
[8] "XLANG Web Services for Business process design", http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm
[9] A. Aldea, et al., "The scope of application of multi-agent systems in the process industry: three case studies", Expert Systems with Applications, 26, 39-47, 2004.
[10] F. Bellifemine, et al., JADE Administrator's GUIDE, TILAB S.P.A., 2003