MULTIAGENT COORDINATION IN E-UNIVERSITY

Thiri Haymar Kyaw, Ni Lar Thein

thirihaymarkyaw@gmail.com

ABSTRACT

This paper describes a coordination approach for e-university that is hierarchical composite multiagent architecture. There are many heterogeneous composite agents each of which is composed of several individual agents and two types of coordinator agents: Intelligent Coordinator agent (ICo) and Assistant Coordinator agent (ACo). When any agent has a task, it creates a token and sends to ICo through ACo. Then, ICo chooses correct composite agent and sends the token to it. Each composite agent has one ACo that is responsible for selecting correct individual agent, and communicating with ICo. Coordinator agents use token based coordination to route the tokens to correct receiver agents efficiently and directly. We developed coordinator algorithms for passing the tokens. This approach can be able to assign the tasks or send a message to appropriate agents successfully while reducing extra routing and so provide efficient and effective coordination among large agent teams with very little communication. We present the results comparing the number of messages in our proposed model and typical decentralized model.

***Index Terms*—** Multiagent System, Token based Coordination, E-University, Composite Agent and Coordinator Agent.

1. INTRODUCTION

Automated coordination is still a very active research area because it can decrease operational cost, risk, time and human activities while increasing efficiency, flexibility and cooperation performance. Coordination is the process of managing the possible interaction between activities and process. There are many widely used coordination approaches such as auction based coordination, token-based coordination and hybrid approach [4], [7], [15], etc. Some approaches have extra routings that tokens are passed to agents, which do not need to pass or bids, which cannot get the task or resource, are sent.

The proposed approach uses token-based hierarchical coordination because hierarchical coordination gives the best results for large-scale systems and token-based approach finds the reasonable solution rather than searching for the optimal. New coordination algorithms are developed to determine the tokens are passed to where. In previous, token-based algorithms use the local models for token movement and a token is routed to all agents within the teams. In this approach, coordinator agents provide token movement decisions by using token passing algorithms called coordinator algorithms. In this paper, we show a multiagent architecture for a university and apply the coordination approach for informing and task allocation between agents. This concept offers a range of coordinated services, tasks allocation and access to resources through the coordinator agents and supports on the fundamental computerized subsystems for the university's administrative, teaching, library and student affairs.

Multiagent system approach has been widely used in the development of large and complex systems. Multiagent approach is appropriate for university architecture while coordinating and cooperating between agents effectively support for the university system. In our multiagent architecture, each subsystem is constructed as a composite agent and Intelligent Coordinator agent (ICo) communicates and coordinates among composite agents. A composite agent is composed of agent teams or individual agents. If any agent appears a new task, it creates a token and sends to ICo through the Assistant Coordinator agent (ACo). ACo has responsibilities of passing the token to correct destination agent within the same composite agent and uses coordinator algorithm like ICo. ICo determines when and where to pass the token directly using its coordinator algorithm with the help of Matching Inference Engine (MIE). MIE is a simple rule based reasoning engine. The main purpose of this approach is to provide efficient and effective coordination among large agent teams with very little communication.

In the rests of the paper, we discuss the proposed hierarchical composite multiagent architecture and university architecture in section 2. We present token-based coordination with proposed coordinator algorithms and how to work coordination in the prototype e-university in section 3. The evaluation results of comparing our model and traditional decentralized model in section 4. Finally, in section 5, we conclude the paper with discussion and our future work.

2. AGENT ARCHITECTURE FOR E- UNIVERSITY

Hierarchical architecture is a typical control-oriented architecture of multiagent system. It well matches human organization management procedures and provides effective feedback and control ways [3]. In this multiagent architecture, there have many heterogeneous composite agents and two types of coordinator agents. Hierarchical composite multiagent architecture based on [10] is shown in figure 1.

We create a composite agent team CA = $\{CA_1, CA_2, ..., CA_n\}$ and CA_i is composed of a Assistant Coordinator agent (ACo) and member agents, $CA_i = \{ACo, A_1, A_2, ..., A_n\}$. ACo is responsible for selecting appropriate agents to pass the token and communicating within CA_i. Intelligent Coordinator agent (ICo) is responsible for token movement decisions between composite agents.

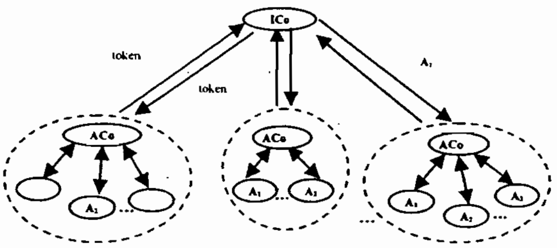

Figure 1. Coordination in Hierarchical Composite Multiagent Architecture.

Multiagent system approach is effective for tackling the management of the entire university system. The abstract structure of multiagent university architecture is shown in figure 2. There are six composite agents: AdminAgent, StuAffairsAgent, DeptAgent FinanceAgent, LibraryAgent and StudentAgent.

AdminAgent performs administrative tasks, staff management, initiation for task assignment, etc. AdminAgent is composed of ACo, administrator agent and other staff agents. Most of tokens are created by AdminAgent. StuAffairsAgent represents the Student Affairs department. Its tasks are student registration, examination, inquiring student information, creating and updating students' records including attendance.

DeptAgent represents Faculties in the university. It performs teaching, supervising, scheduling courses or timetabling classes and planning lectures, etc. DeptAgent is composed of ACo and teacher agents in faculties. Teacher agents represent teachers who are recorded in TeacherProfiles and active as an agent during the time DeptAgent's interface is active and they attend to their departments.

LibraryAgent represents the library system in the university. Its own tasks are member registration, book reservation, book lending, etc. FinanceAgent represents Finance department. It provides preparation of budget, financial statements and reports, payments and payroll system. StudentAgent is on behalf of all students in the university. ICo also provide as the coordinator between different university multiagent systems.

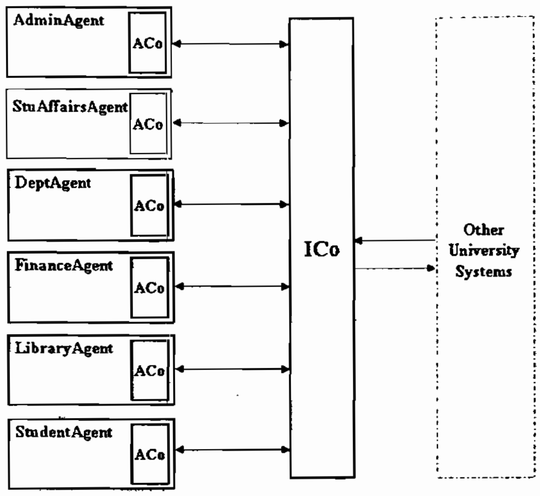

Figure 2. The general Architecture of Multiagent University.

3. TOKEN BASED COORDINATION

Coordination in the activities of the agents in a multiagent system is a key element of Multiagent System organization. Token-based algorithms have been developed in [1], [11], [12] and [14]. A token encapsulates control and information needed for message passing, task allocation, and resources sharing and so on. The general format of a token used in our approach is shown in figure 3.

TokenID	Title	Description	
SenderID	Sender Address		
ReceiverID	Receiver Address		
Attachment	Deadline	Signature	Reply

Figure 3. General Format of a Token.

3.1. Task Assignment

AdminAgent creates a token for course assignment and send to ICo. When ICo receives the token, it resets the correct receiver agent according to the Matching function in MIE and sends to the related DeptAgent. When DeptAgent receives the task token, it's ACo chooses appropriate teachers for each courses based on the preferences and other profile facts of the teachers. After selecting the appropriate agents, it sends the task assignment token to selected teacher agents for assigning the course. Each receiver agent receives the task assigned token and reply accept or reject. If any conflicts arise, they are resolved by sending negotiate tokens each other. Token passing procedures for a task token are shown in Figure 4 and 5.

```
token t = getToken(t);

if ( t.tokenID = 1 )          // Inform token
{ for  ∀ CAi ∈ CA            //composite agents
  {   replicate t;
      t.Receiver.Add ( CAi ) ;
  }
    Send all replicas t to  ∀ CAi ∈ CA
}

if ( t.tokenID = 2 )  // Task allocation token
{   SCA = ϕ ;      // selected CA
     for ∀ CAi ∈ CA
         SCA = TitleMatching(t.TitleContent)
     t.Receiver = CAi;
     send  t  to SCA ;
 }

if ( t.tokenID = 3 )       // Resource token
{   if ( CheckRequestedCA ( ) = 1 )
     {      t. Receiver = RCAi ;
             send  t  to RCAi  ;
     }
    else if ( CheckRequestedCA ( ) > 1 )
     {    if( task . Priority = MAX)
           { t. Receiver = RCAi ;send  t  to RCAi ;   }
     }
}

if ( t.tokenID = 4 )       // Request token
{   for  ∀ CAi ∈ CA
     {   if    ( SearchResourceAvailable( t.Title ) )
              send t to CAi  ,
          take Resource AND return;
      }
      reply  t  to requested agents;
}

if ( t.tokenID = 5 )          // Negotiate token
{   NRule=TitletoNegotiateRule(t.TitleContent);
    Append NRule to t.DetailDescription;
    reply  t  to  t.Sender;
}
```

Figure 4. Token passing procedure used by ICo.

```
token t = getToken(t);

if ( t.tokenID = 1 )    // Inform
{      for ∀ Agi ∈ AG        ; AGi ∈ CA
          send t to Agi ;
}

if ( t.tokenID = 2 )   // Task allocation
{      SAg = ϕ ;
       for ∀ Agi ∈ AG
       if ( TitleToRoleMatching ( ) = 1 )
           SAg = SAg ∪ Agi ;
       if (SAg > required no. of agents)
           SAg = SelectOptimal( );
       for  ∀ Agi ∈ AG
           Send  t  to Agi ;
       if  (Agi  reject the task )   Negotiate( );
      else  Assign Task and Update Workload;
}
 if ( t.tokenID = 3 )   // Resource
{    if ( CheckRequestedAgents ( ) = 1 )
         send  t  to Agi;
     else if ( CheckRequestedAgents ( ) > 1 )
        if( task . Priority = MAX)  send  t  to Agi  ;
     else  send  t  to  ICo ;
}

if ( t.tokenID = 4 )     // Request
{   if (CheckResourceAvail(t.titleContent)=true )
     {   t.Attachment = Resource OR Data;
         send  t to  senderAgi ;
     }
     else  send  t  to  ICo ;
}

if ( t.tokenID = 5 )     // Negotiate
{   while ( negotiateResult ≠  OK )
     {    send  t  to  Agent  AND
          Negotiate ( );
     }
     t.Reply = negotiateResult;
     send  t  to ICo;
}
```

Figure 5. Token passing procedure used by ACo.

The interface of the AdminAgent and sent message is shown in Figure 6.(a) and individual agent with received message is shown in Figure 6.(b).

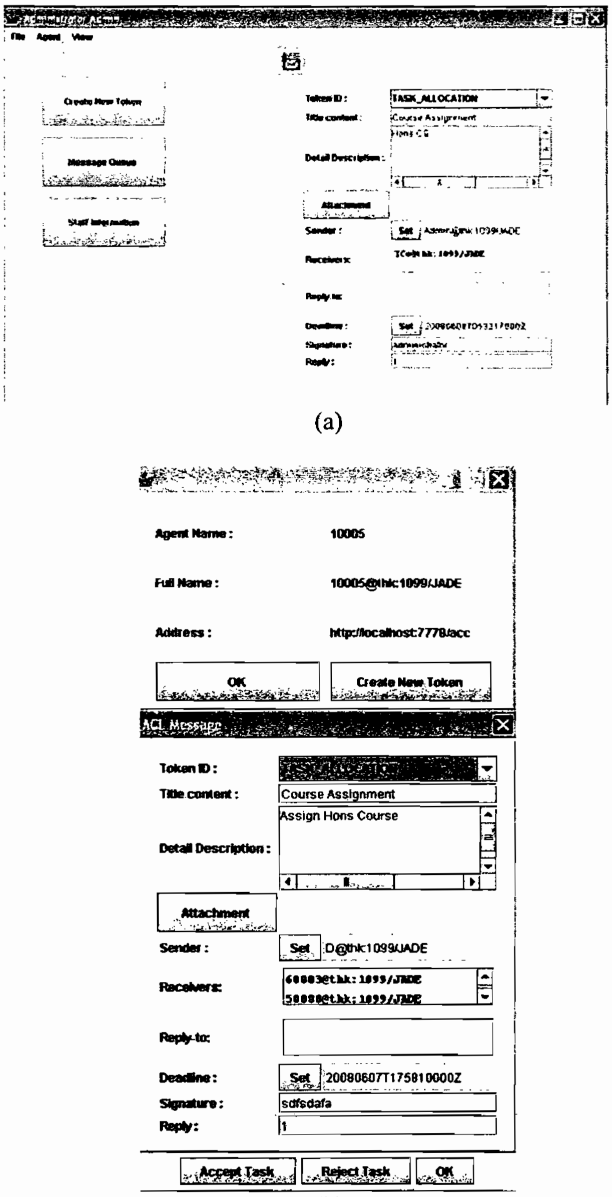

Figure 6. (a) AdminAgent creating a task token. (b) Individual agent receiving a token.

3.2 Inform Token

When AgentICo receives an inform token, which title is "Meeting for all teachers and staffs", it's MIE checks the title and sends to all related composite agents (Admin, Dept, StuAffair and Library). When ACo in each composite agent receives this inform token, it passes to all related individual agents in its team. When AgentICo receives an inform token, which title is "To attend Seminar", it's MIE checks the title and sends to related composite agent (DeptAgent). When ACo in DeptAgent receives. this inform token, it passes to related teacher agents in its team.

3.3 Requesting the Resource

Individual agents may request the resource to perform their tasks, for example, request a scanner or web cam or books in the library. When ICo receives a request token, ICo checks the availability of the resource using MIE which searches the detail facts of the resource in the related databases and checks whether the other agents currently used or not. If the resource is available, the resource is packed into the attachment and the token is sent to the agent that requested the resource.

When ICo receives the resource token, ICo searches the requested agents. If there are more than one requested agents, ICo checks the tasks' priorities that each agent have to do using this resource and chooses one agent that has maximum priority of the task. Then ICo sends the resource token to the selected composite agent and ACo passes the token to the requested agent. If there is more than one, ACo chooses appropriate agent based on the task's priority like ICo.

3.4 Other Features of E-University

DeptAgent interface provides course timetable for all classes, teachers' information and course information. StuAffairAgent provides exam scheduling and showing staff's information. LibraryAgent handles Booklist and Memberlist database and perform reserving and lending the books in the library. All ACo agent stored the sending and receiving messages in the message queue in the running time. AgentICo handles the AgentProfiles for selecting the correct receiver agent. Using this system not only provides coordination in university subsystems but also all students can see the course information, timetables, exam information and easy access to the library.

4. EVALUATION RESULTS

We run our system in six personal computers (Pentium (R) D 3.20 GHz and 1 GB memory) with java Sun SDK 1.5, MySQL Server 5.0 and JADE version 3.4.1. AdminAgent, DeptAgent, StuAffairAgent, StudentAgent, LibraryAgent and AgentICo are run in different computers and mySQL server is located in one computer which runs AgentICo.

We generate different number of agents and send a task assignment token for 9 agents. We test 10 rounds with 10 agents, 20 agents up to 100 agents for assigning the task to 9 agents. In typical decentralized model, task token is passed around the team until to get the 9 agents got their task. So, there are more messages to accomplish the task. In contrast, our model needs only 2 messages for coordinator agents (ICo and ACo) plus 9 messages for 9 agents that can perform the task. The number of messages for two models is shown in Figure 7.

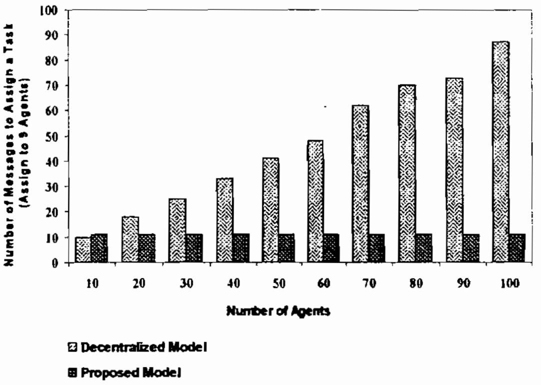

Figure 7. Number of Messages for Typical Decentralized Model and Proposed Model

Manually or traditionally task allocation to teachers in a university takes more time including negotiating time until to reach the global agreement. In our approach, task assignment can be performed choosing optimal agents directly without extra routing to inappropriate agents with very little negotiating time. The results show that the proposed approach takes less time for message passing. It can reduce the communication time for interaction between multiple agents and intends to scalability for large agent teams.

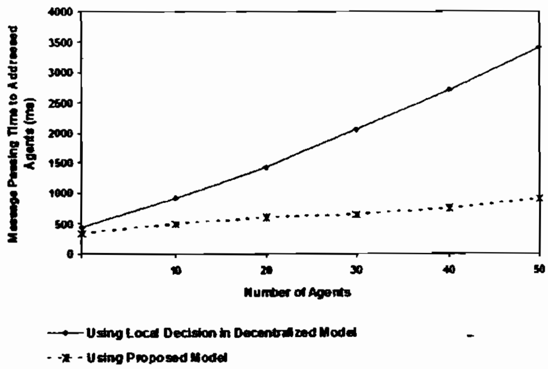

Figure 8. Message Passing Time for Typical Decentralized Model and Proposed Model

In addition to analytical evaluation, we generated 50 agents and applied to decentralized model and our model. In typical decentralized model, agents receive the token from its neighbors, checks for appropriateness and then pass to its neighbor if it is not appropriate for itself. For a task assignment token, it is passed around until reaching to all require agents and so there is some extra routings for passing to unrelated agents. In our proposed model, there is no extra routings and can reach the token directly to the correct receiver agents. So, message passing time is reduced more linearly than the decentralized model. The comparison result on message passing time for two models is shown in Figure 8.

Moreover, we consider about rewards of the typical decentralized model and proposed model. A reward is obtained by accepting the task or reaching the correct receiver agent. We generated different number of agents and sending the messages to these agents. In our proposed model, there is 8 rewards get on the 10 messages, 18 rewards get on the 20 messages and so on. There is no extra routing to unrelated agents and just only two messages for coordinator agents (ICo and ACo). So, the total number of reward received is linearly increased.

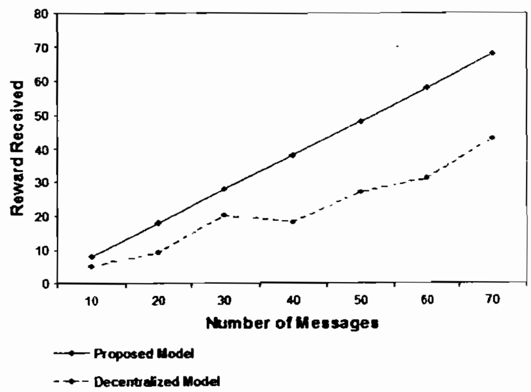

Figure 9. Reward Received Vs. Number of Messages

In the typical decentralized model, there is more fluctuated in reward received. Based on the local model to decide the token passing decision, each agent accepts the token or rejects and passes to the neighbors. Therefore, sometimes there is a little more rewards got and sometimes there is decrease in reward received. The comparison result of two models for reward received on the number of messages is shown in Figure 9.

5. CONCLUSION

We presented a token-based coordination approach for multiagent e-university through middle coordinator agents. In this paper, coordinator algorithms are developed for message informing, task allocation, resource allocation and requesting the resource within e-University. Our proposed multiagent system architecture includes heterogeneous composite agents and two coordinator agents: Intelligent

Coordinator (ICo) and Assistant Coordinator (ACo). The proposed approach can be able to assign the task directly to the most appropriate agent that can perform the task, without extra routing to other agents and reduce message passing time for coordination. Integrating the proposed coordination with the e-university architecture can reduce human involvements in processes and accelerates university workflows. As a future work an interesting extension would be to communicate and coordinate between different multiagent e-university through ICo for different types of tokens.

6. REFERENCES

[1] A. Farinelli, L. locchi, D. Nardi and V. A. Ziparo, "Assignment of Dynamically Perceived Tasks by Token Passing in Multi-Robot Systems", Proceedings of the IEEE, Vol. 94, Issue 7, July 2006, pp. 1271-1288.

[2] F. Bellifemine, G. Caire and D. Greenwood, "Developing Multi-agent Systems with JADE", Wiley Series in 'Agent Technology, 2007.

[3] H. Tianfield, J. Tian and X. Yao, "On the Architectures of Complex Multi-Agent Systems", Proceedings of the Workshop on Knowledge Grid and Grid Intelligence, IEEE/WIC International Conference on Web Intelligent/ Intelligent Agent Technology, Oct, 2003, pp. 195-206.

[4] J. M. Vidal, "Multiagent Coordination Using a Distributed Combinatorial Auction", AAAI Workshop on Auction Mechanisms for Robot Coordination, 2006.

[5] J. Tian and H. Tianfield, "A Multi-agent Approach to the Design of an E-medicine System", Springer Lecture Notes in Computer Science, 2003, pp. 85-94.

[6] M. Dastani, F. Arbab and F. de Boer, "Coordination and Composition in Multi-Agent Systems", Proc. of the Fourth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'05), 2005.

[7] M. Ham, G. Agha, "Market-based Coordination Strategies for Large-scale Multi-Agent Systems", System and Information Sciences Notes, Vol. 2, No. 1, pp. 126--131), 2007.

[8] M. Oprea, MAS_UP_UCT: "A Multi-Agent System for University Course Timetable Scheduling", International Journal of Computers, Communications and Control. vol. 2, no. 1. pp. 94-102, 2007.

[9] M-W. Jang, "Efficient Communication and Coordination For Large-Scale Multi-Agent Systems", Ph.D Thesis, University of Illinois at Urbana-Champain, 2006.

[10] N. Bensaid and Ph. Mathieu, "Integration of hypothetical reasoning in hierarchical multi-agent architecture", 1996.

[11] P. Scerri, Y. Xu, J. Polvichai, B. Yu, S. Okamoto, M. Lewis, and K. Sycara, "Challenges in Building Very Large Teams", Lecture Notes in Economics and Mathematical Systems, Vol. 558, Springer Berlin Heidelberg, 2007, pp. 197-228.

[12] P. Scerri, A. Farinelli, S. Okamoto and M. Tambe, "Token Approach for Role Allocation in Extreme Teams: analysis and experimental evaluation", 2003.

[13] X. Zhang, H. Xu and B. Shrestha, "Developing Multi-Agent Systems with Automatic Agent Generation and Dynamic Task Allocation Mechanisms", Proc. of the sixth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'07), May, 2007.

[14] Y.Xu, P.Scerri, B.Yu, S. Okamoto, M. Lewis, and K. Sycara, "An integrated Token-Based Algorithm for Scalable Coordination", Proc. of the Fourth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'05), 2005, pp. 407 – 414.

[15] Y. Xu, P. Scerri, K. Sycara and M. Lewis, "Comparing Market and Token-Based Coordination", Proc. of the Fifth International Conference on Autonomous Agents and Multiagent Systems (AAMAS'06), 2006.