MỘT MÔ HÌNH HỆ THỐNG E-LEARNING DỰA TRÊN ĐA TÁC TỬ

Đỗ Văn Nhơn

Nguyễn Trần Minh Khuê

TÓM TẮT

Bài báo nhằm xây dựng một mô hình hệ thống E-Learning dựa trên đa tác tử, bao gồm: xây dựng các thành phần cùng với các nguyên lý và quá trình làm việc của chúng, quản lý các tác tử và giao tiếp giữa các tác tử, cung cấp môi trường và cơ chế để thiết kế và truy vấn cơ sở tri thức phân tán. Chúng tôi đã áp dụng mô hình này trong lĩnh vực hình học phẳng.

Từ khoá: *đa tác tử, hệ thống E-Learning, cơ sở tri thức, biểu diễn tri thức.*

ABSTRACT

The aim of this paper is to construct a model of the E-Learning system based on Multi-Agent technology. The work includes building component parts of the model along with principles and working processes of them, managing Agents and their communication, providing an environment and mechanisms to design and query the distributed knowledge base. We applied our model in the field of plane geometry.

Keywords: *Multi-Agent, E-Learning System, Knowledge Base, Knowledge representation*

I. INTRODUCTION

Since the last decade of 20th century, there have been many tools based on Multi-Agent models, such as VOYAGER [Voy-96], AGLETS [IBM-98], MOLE [Str-99], FIPA-OS [Nor-99], ZADE [Jad-00], ZEUS [BT-00], CAP [She-01], CAPNET [Ege-04], etc. Multi-Agent is one of distributed technologies interested by many scientists, it is useful for solving problems about knowledge.

In 2001, the COKB Model (Computational Object Knowledge Base [Nhon Do-2001]) [9] is a knowledge model that can be used to design and implement knowledge base in practice. The model was a system of many components such as concepts about C-Objects, relations, operators and rules. It proved that Multi-Agent tools and models developed before the year 2000 built an environment to solve issues about knowledge, but they did not mention about knowledge including concepts and a wide diversity of complex relations between concepts. Up to now, they have not advanced in this issue.

Since July 2000, .NET, the last development at Microsoft has introduced [18]. It has been the foundation for Web services with XML (eXtensible Markup Language). NET technology has overcome weakness of DCOM (Distributed Component Object Model) technology. In according to this trend, CAP [She-01] (Component Agent Platform) using Microsoft DCOM and ActiveX technology was developed into a new platform called CAPNET [Ege-04] (Component Agent Platform in .NET) [3], that was fully compliant with the specifications of the Foundation for Intelligent Physical Agents (FIPA) and implemented as 100% managed code in the .NET framework. However, they also did not solve above problems in order that we could apply them in reality.

This paper describes constructing a model of Multi-Agent based E-Learning system. It includes building component parts of the model, principles and working processes of them, managing Mobile Agents and their communication according to the architecture of Aglets Platform. Originally developed at the IBM Tokyo Research Laboratory, Aglets was appreciated for its clear and easy to use API, good modularity and design. Since the initial effort of IBM, several versions of Aglets had been

released, after that the project was hosted at Sourceforge. The development was stopped around 2001. Then it restarted due to the change of the project administrator [13]. The model, which this paper proposes did not use this platform, but we researched its architecture and principles to build a platform by .NET technology, because this technology supports to develop distributed systems strongly. .NET applications can be accessed by various devices across the Internet [18].

We applied this model to design and implement the distributed knowledge base of plane geometry. Besides, we provided principles and methods to represent knowledge with XML and operate distributed knowledge base as inserting, deleting, updating, and querying knowledge. Because knowledge is huge and complex, it needs to be distributed in different locations so that we can solve problems about capability of computers and the speed of processes. Many knowledge engineers can co-operate and develop distributed knowledge base. They can reuse knowledge defined before to build new concepts or relations. E-Learners can query knowledge easily in anywhere.

In section II, a model of the Multi-Agent based E-Learning system, its component parts, general processes are presented along with a short explanation about benefits of Multi-Agent technology to E-Learning system. In section III, working model of Mobile Agent and detailed processes of component parts are described. In section IV, the model of knowledge bases about plane geometry is showed. In section V, structures of XML files representing distributed knowledge about plane geometry and constraints to design XML nodes are regulated. In section VI, methods to query knowledge are presented. Finally, the comparation with other models, results and advantages of this work, along with some future work are discussed.

II. THE MODEL OF MULTI-AGENT BASED E-LEARNING SYSTEM

A. Benefits of Multi-Agent technology to E-Learning system

A multiagent system is one that consists of a number of agents, which interact with one-another. In the most general case, agents will be acting on behalf of users with different goals and motivations. To interact successfully, they will require the ability to cooperate, coordinate, and negotiate with each other, much as people do [5]. Multi-Agent system including Mobile Agents has many special characteristics. A Mobile Agent is not bound to the system where it begins execution. It has the unique ability to transport itself from one system in a network to another. The ability to travel allows a mobile agent to move to a system that contains an object with which the agent wants to interact and then to take advantage of being in the same host or network as the object. They reduce network traffic, overcome network latency, encapsulate protocols, execute asynchronously and autonomously, adapt dynamically. Besides, they are naturally heterogeneous, robust and fault-tolerant [12]. Specially, it has highly flexible capable to solve problems about knowledge. Consequently, Multi-Agent technology is suitable to development E-Learning system.

B. Component parts of the Multi-Agent based E-Learning model

The model includes three component parts: The first part is the graphical user interface. The user has two groups: Knowledge Engineers and E-Learners. The first group has the right to change distributed knowledge base, as loading, designing and saving. The second group is allowed to query distributed knowledge base. The second part is Multi-Agent system. It includes Mobile Agents residing in contexts of computer systems. A computer can have many contexts. A proxy is a class of the interface; it works on behalf of Agent to interact with other interfaces of other Agents. The third part is distributed knowledge base including the system of concepts along with different relations between them.

C. The model of the Multi-Agent based E-Learning system

The model of the Multi-Agent based E-Learning system is shown in Fig 1. E-Learners or

Knowledge Engineers use the graphical user interface to enter Multi-Agent system through Internet by SOAP (Simple Object Access Protocol) or .Net Remoting. Multi-Agent system will query, or load, insert, delete, update distributed knowledge base according to mechanisms proposed. Then, it will return the results to the graphical user interface. Finally, the interface will present the results to E-Learners or Knowledge Engineers.

III. PRINCIPLES AND WORKING PROCESSES OF COMPONENT PARTS

A. Aglet Life-Cycle Model

Aglet Life-Cycle Model is shown in Fig 2. An Aglet is a Mobile Agent residing in a context and migrating from this context to another during working process. A context is defined with properties: host address, port and context name. An Aglet has to create the proxy so that it can communicate with the other agents through this interface. The fundamental operations of an Aglet include creation, cloning, dispatching, retraction, deactivation, activation, and disposal. The creation of an Aglet takes place in a context. The new Aglet is assigned an identifier, inserted into the context, and initialized. The cloning of an Aglet produces an almost identical copy of the original Aglet in the same context. Dispatching an Aglet from one context to another will remove it from its current context and insert it into the destination context, where it will restart execution. The retraction of an Aglet will pull it from its current context and insert it into the context from which the retraction was requested. The deactivation of an Aglet is ability to temporarily halt its execution and store its state in secondary storage. Activation of an Aglet will restore it in the same context. Disposal of an Aglet will halt its current execution and remove it from its current context [12].

B. Principles and working processes of component parts in the model

In fig 3, working processes of component parts in the model are shown. The system includes four Agent Management Centers. Center 1 manage Agents servicing Knowledge Engineers, center 2 manage Agents servicing E-Learners, center 3 manage Agents servicing operations and query distributed knowledge base, center 4 manage Agents servicing access to distributed knowledge base. Knowledge Engineers use graphical user interface to build and design distributed knowledge base. Agent 1A listens to the requirements and interacts with Agent 1B or 1C. If the requirement is loading knowledge base, Agent 1A will interact with Agent 1B. If the requirement is changing knowledge base as inserting, deleting, updating knowledge, Agent 1A will interact with Agent 1C. Agent 1B converts values of properties and properties of nodes in XML files representing knowledge to objects of form as label, text box, check box, group box, combo box, button, etc, after interacting with Agent 3A. With objects of form, knowledge engineers can insert, delete, or update knowledge easily. When they need to save into knowledge base, Agent 1C will convert objects of form to values of properties and properties of nodes in XML files. Then, it interacts with Agent 3B. Agent 3B categorizes the requirement to interact with Agent 3C, 3D, or 3E to insert, delete, or update knowledge.

When E-Learners need to query distributed knowledge base, Agent 2A listens to the requirements and interacts with Agent 2B or 2C. If the requirement is query about defining knowledge, Agent 2B will interact with Agent 3F. If the requirement is a kind of advanced queries, Agent 2B will interact with Agent 3G. Agent 3G interacts with Agent 3H to find the relationships between knowledge, or interacts with Agent 3I to find orders to build knowledge. Agent 3A, 3C, 3D, 3E, 3F, 3H, 3I interact with Agent 4A, 4B, 4C, 4D, 4E, 4F, 4G. Agents in center 4 access to distributed knowledge base including XML files.

Here are some characters of working processes: First, they reduce the network load. After a mobile agent is dispatched to the remote computer, it interacts in the locality. The intermediate results are reduced.

Second, Agents can work with the uncontinuous connection [4]. The network can be disconnected after the computer dispatches an Agent residing in its context into the context of the remote computer. The Agent dispatched becomes independent of the process. When the Agent from remote computer retracts, the computer needs to be reconnected to the network. This shows Agents work asynchronously and autonomously [12]. Third, Agents can interact each other, they do not need communicate with the server as the other paradigms so that they can interact with the other Agents. This decentralizes the network structure [13] (Fig 4). A XML file services a context. The relationships between nodes of XML files are illustrated in Fig 5.

III. THE MODEL OF KNOWLEDGE BASES

The model of Knowledge Bases is a structure consisting of six components: *C, H, R, Op, Func,* and *Rule*. **C** is a set of concepts. The structure of a concept in C includes (*Name, Id, Attrs, Facts*). *Name* is a name of the concept, *Id* is the name of the object, *Attrs* are attributes of the concept, *Facts* are facts of the object. **H** is a set of hierarchies of concepts. The structure of a hierarchy in H includes (*C1, C2*). *C1* is a concept, *C2* is a particular concept of C1. **R** is a set of relations between concepts. The structure of a relation in R includes (*Name, Id, Args, Facts*). *Name* is a name of relation, *Id* is an ordinal number, *Args* are arguments of the relation, *Facts* are facts about relations between objects. **Func** is a set of functions. The structure of a function in Func includes *(Name, Id, Args, Results)*. *Name* is a name of the function, *Id* is an ordinal number, *Args* are arguments of the function, *Results* are results after the function is done. **Op** is a set of operators. The structure of an operator in Op includes *(Name, Id, Args, Results)*. *Name* is a name of the operator, *Id* is an ordinal number, *Args* are arguments of the operator, *Results* are results after the operator is done. **Rule** is a set of rules. The structure of a rule in Rule includes (*Id, Left, Right*). *Id* is an ordinal number, *Left* includes facts of hypotheses, *Right* includes facts of goals.

IV. DISTRIBUTED KNOWLEDGE REPRESENTATION

A. Structures

The structure of a node representing a concept whose level is zero

```
<concept(ordinal) name="Concept-Name" level="Level-Number" >
</concept(ordinal)>
```

The structure of a node representing a concept whose level is one

```
<concept(ordinal) name="Concept-Name" level="Level-Number">
  <variable>
      <variable0 name="Variable-Name" type="Concept- Name" context="Context-Name" />
     <variable1 name="Variable-Name" type="Concept-Name" context="Context-Name" />
     …….
  </variable>
 </concept(ordinal)>
```

The structure of a node representing a concept whose level is more than one

```
<concept(ordinal) name="Concept-Name" level="Level-Number">
    <variable>
     <variable0 name="Variable-Name" type="Concept- Name" context="Context-Name" />
     <variable1 name="Variable-Name" type="Concept- Name" context="Context-Name" />
     …….....
     </variable>
```

```
        <attribute>
          <attributed0 name="Variable-Name" type="Concept- Name" context="Context-Name" />
          <attributed1 name="Variable-Name" type="Concept- Name" context="Context-Name" />
            ……..
        </attribute>
         <relation>
            [<or>]
          <if0 relate="Relation-Name" context="Context- Name" value0="Variable-Name"
              value1="Variable- Name"
               value2="Variable-Name"
                .....
               [flag="true/false"] [out="Out-Value"] />
           <if1 relate="Relation-Name" context="Context-Name" value0="Variable-Name"
              value1="Variable- Name"
               value2="Variable-Name"
                .....
               [flag= "true/false"] [out="Out-Value"] />
            …….
          [</or>]
          ……
      </relation>
  </concept(ordinal)>
```

The structure of a node representing a relation

```
  <relation(ordinal) name="Relation-Name" level="Level-Number">
     <variable>
        <variable0 name="Variable-Name" type="Concept- Name" context="Context-Name" />
        <variable1 name="Variable-Name" type="Concept- Name" context="Context-Name" />
        …….....
     </variable>
    [<out>
      <outvalue0 name="Out-Name" type="Out-Type" />
     </out>]
     <define>
       <supposition>
          <supposition0 name="Variable-Name" type="Concept-Name"
            context="Context-Name"    />
          <supposition1 name="Variable-Name" type="Concept-Name"
            context="Context-Name"    />
         …….
       </supposition>
      <relation>
```

```
[<or>]
<if0 relate="Relation-Name" context="Context- Name" value0="Variable-Name"
    value1="Variable- Name"
     value2="Variable-Name"
      .....
     [flag="true/false"] [out="Out-Value"] />
<if1 relate="Relation-Name" context="Context-Name" value0="Variable-Name"
    value1="Variable- Name"
     value2="Variable-Name"
      .....
     [flag= "true/false"] [out="Out-Value"] />
  …….
[</or>]
……
  </relation>
 </define>
</relation(ordinal)>
```

B. Example

```
<?xml version="1.0" encoding="utf-8"?>
<Root>
<context host="10.24.1.205" port=40 name="context1"
<concept0 name="POINT" level=0>
 </concept0>
 <concept1 name="LINE" level=0>
 </concept1>
<concept2 name="LINE SEGMENT" level=1>
   <variable0 name="A" type="POINT" />
   <variable1 name="B" type="POINT" />
</concept2>
</context>
 </Root>

<?xml version="1.0" encoding="utf-8"?>
<Root>
<context host="10.24.1.207" port=45 name="context3"
<concept5 name="TRIANGLE" level=2>
<variable>
      <variable0 name="A" type="POINT" context="context1" />
      <variable1 name="B" type="POINT" context="context1" />
      <variable2 name="C" type="POINT" context="context1" />
</variable>
```

```
<attribute>
        <variable0 name="AB" type="LINE SEGMENT" context="context1" />
        <variable1 name="BC" type="LINE SEGMENT" context="context1" />
        <variable2 name="AC" type="LINE SEGMENT" context="context1" />
       <variable3 name="ABC" type="ANGLE" context="context2" />
        <variable4 name="BCA" type="ANGLE" context="context2" />
        <variable5 name="BAC" type="ANGLE" context="context2" />
 </attribute>
<relation>
  <if0 relate="STRAIGHTNESS" context="context5" value0="A" value1="B" value2="C"
                flag="false" />
</relation>
</concept5>
<concept8 name="RIGHT-TRIANGLE" level=2>
<variable>
        <variable0 name="ABC" type="TRIANGLE" />
  </variable>
    <relation>
      <or>
        <if0 relate="LINE SEGMENT_RIGHT ANGLE_LINE SEGMENT" context="context5"
                value0="AB" value1="AC" flag="true" />
         <if1 relate="LINE SEGMENT_RIGHT ANGLE_LINE SEGMENT" context="context5"
                value0="AB" value1="BC" flag="true" />
        <if2 relate="LINE SEGMENT_RIGHT ANGLE_LINE SEGMENT" context="context5"
                value0="AC" value1="BC" flag="true" />
      </or>
  </relation>
</concept8>
</context>
 </Root>

<?xml version="1.0" encoding="utf-8"?>
<Root>
<context host="10.24.1.208" port=41 name="context5"
<relation45 name="STRAIGHTNESS" level=0>
     <variable>
        <variable0 name="A" type="POINT" context="context1" />
       <variable1 name="B" type="POINT" context="context1" />
       <variable2 name="C" type="POINT" context="context1" />
     </variable>
     <define>
```

```
      <supposition>
        <supposition0 name="d" type="LINE" context="context1" />
       </supposition>
       <relation>
        <if0 relate="POINT_LIES ON_LINE" context="context2" value0="A" value1="d"
              flag="true" />
        <if1 relate="POINT_LIES ON_LINE" context="context2" value0="B" value1="d"
              flag="true" />
        <if2 relate="POINT_LIES ON_LINE" context="context2" value0="C" value1="d"
              flag="true" />
       </relation>
     </define>
   </relation45>
<relation52 name="LINE_RIGHT ANGLE_LINE" level=1>
   <variable>
     <variable0 name="a" type="LINE" context="context1" />
     <variable1 name="b" type="LINE" context="context1" />
   </variable>
   <define>
     <supposition>
            <supposition0 name="A" type="POINT"   context="context1" />
            <supposition1 name="B" type="POINT" context="context1" />
              <supposition2 name="C" type="POINT" context="context1" />
     </supposition>
     <relation>
       <if0 relate="POINT_LIES ON_LINE" context="context2" value0="A" value1="a" flag="true"
/>
       <if1 relate="POINT_LIES ON_LINE" context="context2" value0="A" value1="b"
flag="false" />
       <if2 relate="POINT_LIES ON_LINE" context="context2" value0="B" value1="a"
flag="false" />
       <if3 relate="POINT_LIES ON_LINE" context="context2" value0="B" value1="b" flag="true"
/>
       <if3 relate="LINE_INTERSECT_LINE" context = "context4" value0="a" value1="b"
flag="true" />
       <if4 relate="ANGLE MAGNITUDE" context = "context4" value0="A" value1="C"
value2="B" out="90" />
     </relation>
   </define>
  </relation52>
<relation55 name="LINE SEGMENT_RIGHT ANGLE_LINE SEGMENT" level=1>
   <variable>
     <variable0 name="AB" type="LINE SEGMENT" context="context1" />
```

```
    <variable1 name="CD" type="LINE SEGMENT" context="context1" />
   </variable>
   <define>
    <supposition>
            <supposition0 name="a" type="LINE"   context="context1" />
            <supposition1 name="b" type="LINE" context="context1" />
      </supposition>
    <relation>
     <if0 relate="LINE SEGMENT_LIES ON_LINE" context="context3" value0="AB"
value1="a" flag="true" />
     <if1 relate="LINE SEGMENT_LIES ON_LINE" context="context3" value0="CD"
value1="b" flag="true" />
     <if2 relate="LINE_RIGHT ANGLE_LINE" value0="a" value1="b" flag="true" />
     </relation>
   </define>
  </relation55>
</context>
 </Root>
```

C. Constraints

Constraints to build a concept whose ordinal is N and level is M

Constraint 1: A context includes many concepts, but a concept only exists in a context

Constraint 2: Names of concepts are different

Constraint 3: Concepts in the tag <variable> are concepts whose ordinals are less than N and levels are equal or less than M

Constraint 4: Names of variables in sub tags of the tag <variable> are different

Constraint 5: Concepts have to be based on the set of rules

Constraints to build a relation whose ordinal is N and level is M

Constraint 1: A context includes many relations, but a relation only exists in a context

Constraint 2: Names of relations are different

Constraint 3: Relations in the sub tags of the tag <relation> are relations whose ordinals are less than N and levels are equal or less than M

Constraint 4: Names of variables in sub tags of the tag <variable> are different

Constraint 5: Names of variables in sub tags of the tag <supposition> are different

Constraint 6: The quantity of the values of the relation in the tag <if> is equal to the quantity of variables in this relation

Constraint 7: The set including names of variables in sub tags of the tag <relation> is equal to the set including names of variables in the tag <variable> and the sub tags of the tag <supposition>

Constraint 8: Relations have to be based on the set of rules

V. QUERY DISTRIBUTED KNOWLEDGE BASES

A. The first kind of query

Description of the query: Define a concept or a relation

Principle to query: Find the node represents the concept/relation and translate values and attributes of its sub-node into a normal sentence

Example 1:

Query: What is a RIGHT ANGLED TRIANGLE?

Answer: Let ABC be a TRIANGLE. If the relation between LINE SEGMENT AB and LINE SEGMENT AC is LINE SEGMENT CONGRUENCE or the relation between LINE SEGMENT AB and LINE SEGMENT BC is LINE SEGMENT CONGRUENCE or the relation between LINE SEGMENT AC and LINE SEGMENT BC is LINE SEGMENT CONGRUENCE, ABC is a RIGHT ANGLED TRIANGLE

If you want to know about relating concepts or relations, please click on the links below them.

Example 2:

Query: What is STRAIGHTNESS?

Answer: Let A be a POINT, B be a POINT, C be a POINT, and d be a LINE. If the relation between POINT A and LINE d is POINT_ LIES ON_ LINE, the relation between POINT B and LINE d is POINT_ LIES ON_ LINE, and the relation between POINT C and LINE d is POINT_ LIES ON_ LINE, the relation between A, B and C is STRAIGHTNESS

If you want to know about relating concepts or relations, please click on the links below them

B. The second kind of query

Description of the query: Find the relationship between concepts and relations

Principle to query: Base on rules to find the paths between concepts and relations. These paths represent the relationship between them. If do not find any path, there is no relationship between them.

Example 1:

Query: Find the relationship between a PARALLELOGRAM and a SQUARE

Answer: There is the relationship between a PARALLELOGRAM and a SQUARE

Path 1: PARALLELOGRAM -> RECTANGLE -> SQUARE

Path 2: PARALLELOGRAM -> RHOMBUS -> SQUARE

Example 2:

Query: Find the relationship between LINE SEGMENT CONGRUENCE and LINE SEGMENT _RIGHT ANGLE_ LINE SEGMENT

Answer: There is no the relationship between LINE SEGMENT CONGRUENCE and LINE SEGMENT _RIGHT ANGLE_ LINE SEGMENT. There is no path between them.

C. The third kind of query

Description of the query: How to build a concept or a relation

Principle to query: Find the total orders from partial orders. Results are linear orders to build a concept or a relation

Example:

Query: How to build a RIGHT ANGLED ISOSCELES TRIANGLE

Answer: To build a RIGHT ANGLED ISOSCELES TRIANGLE, you need to know one of linear orders of following knowledge

Linear order 1: POINT, LINE, LINE SEGMENT, POINT_LIES ON_LINE, STRAIGHTNESS, TRIANGLE, LINE_ INTERSECT _LINE, ANGLE MAGNITUDE, LINE_ RIGHT ANGLE _LINE, LINE SEGMENT_ LIES ON _LINE, LINE SEGMENT _RIGHT ANGLE_ LINE SEGMENT, RIGHT TRIANGLE, LINE SEGMENT CONGRUENCE, ISOSCELES TRIANGLE, RIGHT ANGLED ISOSCELES TRIANGLE

Linear order 2: LINE, POINT, LINE SEGMENT, POINT_LIES ON_LINE, STRAIGHTNESS, TRIANGLE, LINE SEGMENT CONGRUENCE, ISOSCELES TRIANGLE, LINE_ INTERSECT _LINE, ANGLE MAGNITUDE, LINE_ RIGHT ANGLE _LINE, LINE SEGMENT_ LIES ON _LINE, LINE SEGMENT _RIGHT ANGLE_ LINE SEGMENT, RIGHT ANGLED TRIANGLE, RIGHT ANGLED ISOSCELES TRIANGLE

Linear order 3: POINT, LINE, LINE SEGMENT, LINE SEGMENT CONGRUENCE, POINT_LIES ON_LINE, STRAIGHTNESS, LINE_ INTERSECT _LINE, ANGLE MAGNITUDE, LINE_ RIGHT ANGLE _LINE, LINE SEGMENT_ LIES ON _LINE, LINE SEGMENT _RIGHT ANGLE_ LINE SEGMENT, TRIANGLE, RIGHT TRIANGLE, ISOSCELES TRIANGLE, RIGHT ANGLED ISOSCELES TRIANGLE

VI. CONCLUSION

Although the world appeared many models of E-Learning systems based on many different technologies, the model of the Multi-Agent based E-Learning system is a model interested by its especial characteristics. Example, in 2004, Nanyang Technological University (NTU) in Singapore developed a model of E-Leaning system [4] based on Aglets platform [12]. But things which NTU concerned weren't problems about knowledge including concepts and a wide diversity of relations between concepts. These applications are Course Information Distribution, Customized Course Construction, Interactive User Tracking, System Security, Load Balancing, Response Time Evaluations, Domain Classification of Course Documents Keyword Selections and Document Indexing, Keyword-Based Searching. Other models of Multi-Agent based E-Learning system did not mention about this issue.

In conclusion, our work solves three problems. The first is building a Multi-Agent platform. The second is constructing an E-Learning model based on this platform along with its principles and working processes. The third is applying this model to build the distributed knowledge bases in plane geometry, providing methods to design, and query knowledge. The model can be applied not only for plane geometry but also for algebra, graph theory, physics, etc. Besides, we can extend the scope of knowledge including not only concepts, relations and rules but also operators, functions, etc. We can build Agents solving mathematical problems automatically.

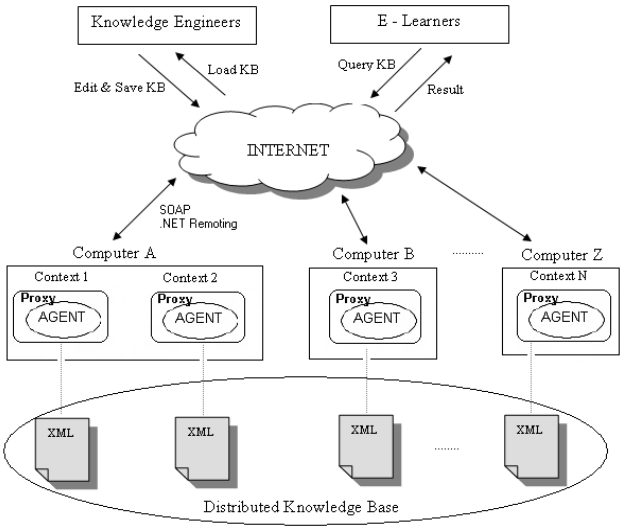

Fig 1. *The model of the Multi-Agent based E-Learning system*

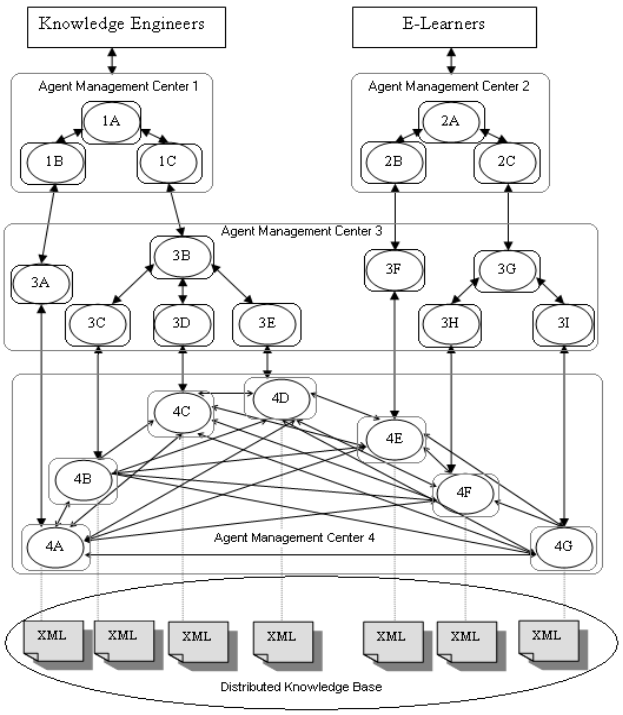

Fig 3. *Working processes of component parts in the model*

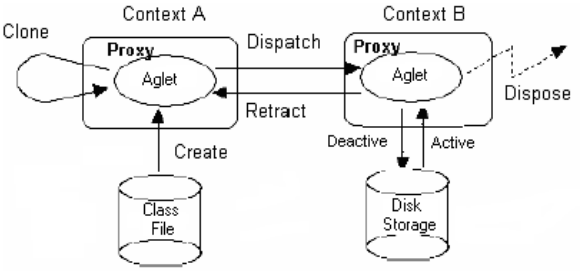

Fig 2. *Aglet Life-Cycle Model [12]*

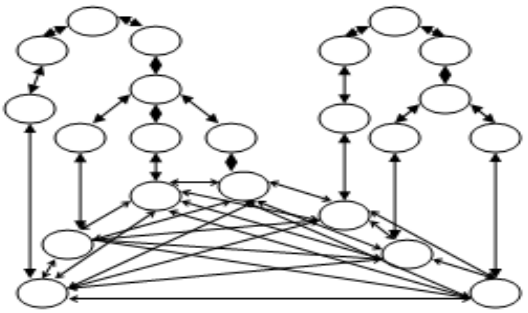

Fig 4. *Network structure*

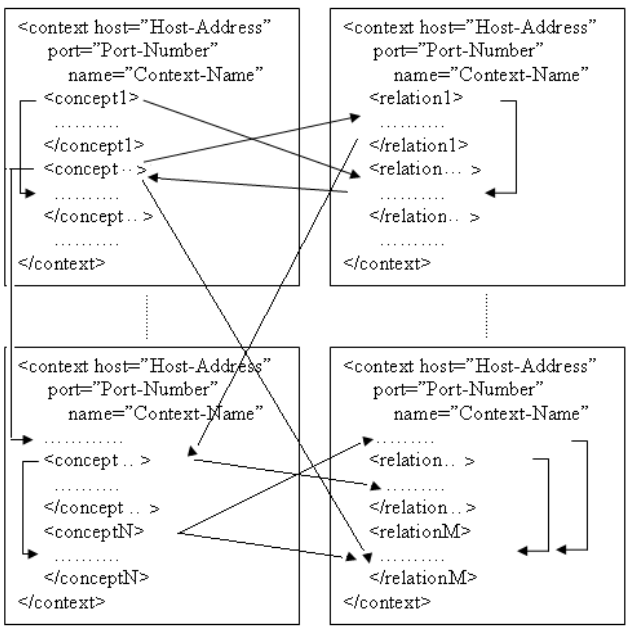

Fig 5. *Relationships between nodes of XML files*

REFERENCES

[1] Tetsuya Yoshida, "Cooperation learning in Multi-Agent Systems with annotation and reward", *International Journal of Knowledge-based and Intelligent Engineering System 11*, pp 19-34, 2007.

[2] Anatoly Gladun, Julia Rogustshina, "An ontology based approach to student skills in Multi-Agent e-Learning systems", I*nternational Journal Information Technologies and Knowledge Vol.1*, 2007.

[3] Ernesto German, Manuel Chi, Leonid Sheremetov, *Design and implementation of a FIPA compliant Agent Platform in .NET*, 2004.

[4] Jon T.S. Quah, Y.M. Chen, Nanyang Technological University, Winnie C.H. Leow, Singapore Polytechnic, Singapore*, E-Learning System*, 2004.

[5] Michael Wooldridge, John Wiley & Sons Ltd, *An Introduction to Multiagent Systems*, 2002.

[6] Marc J.Raphael, Scott A.Deloach, *A Knowledge Base for Knowledge-based Multi-Agent System Construction*, 2000.

[7] Katia Sycara, Multi-Agent System, 2000.

[8] Roberto A.Flores Mendez, *Towards a Standardization of Multi-Agent System Frameworks*, 1999.

[9] Nhon Do, *Construction and development models of knowledge representation for automatically resolving systems, Ph.D Thesis, University of National Science*, Ho Chi Minh City, 2001.

[10] Nhon Do, Nhan Do, *A model of Ontology and Application*, 2005.

[11] Midas and Agentcities Research Projects, *Zeus Agent Realization Guide & Zeus Technical Manual*, 2000.

[12] Danny B.Lange, Mitsuru Oshima *Programming and deploying Java Mobile Agents with Aglets,* 1998

[13] Luca Ferrari, *The Aglets 2.0.2 User's Manual*, 2004.

[14] Cormen, T. H., Leiserson, C. E., and Rivest, R.L., *Introduction to Algorithms*, The MIT Press, 1997.

[15] Max Bramer and Vladan Devedzic, *Artificial Intelligent Applications and Innovations*, 2004.

[16] Natalya Fridman Noy, Carole D. Hafner, *Ontology Design*, 1997.

[17] Natalya F. Noy, Deborah L. McGuinness, *Ontology Development*, 2002.

[18] Aptech, *.NET programming*, 2004.

[19] Simon Robinson, Christian Nagel, Jay Glynn, Morgan Skinner, Karli Watson, Bill Evjen, Wiley Publishing, *C# Professional Programming*, 2004.