Research on Applications of Multi-Agent System Based on Execution Engine in Clinical Decision-Making

Zhenzhen Yan[*], Liang Xiao, Jianzhou Liu, Xing Liu, Yumin Hu,
Qiuju Wei, and Xusong Liu

Department of Computer Science,
Hubei University of Technology, Wuhan 430068, Hubei,
People's Republic of China
48453626@qq.com, healthcloud@126.com, 791282452@qq.com

Abstract. Medical errors have become a common concern of social problems. One important reason is the lack of clinical experience. Clinical guidelines are the solution to this problem; however, they are always kept being updated. Let the system adapt to the changing clinical guidelines is a challenging idea. In this paper we propose MAS (Multi-Agent System) based on the Execution Engine can achieve this goal. In the MAS, clinical guidelines are mapped into rules, which are stored in the rule repository and could be processed by the Execution Engine, to guide agents' behaviors. Rules in the system are configurable and Execution Engine can always obtain the latest data at runtime without interrupting system running, so it implements the system's adaptability. Agents, which could be deployed in distributed environments [2], simulate doctors' roles to do aided diagnosis by collaborations which implements data sharing and improves the accuracy of decision-making greatly.

Keywords: Multi-Agent System · Adaptability · Clinical guidelines · Execution Engine·Clinical Decision Making.

1 Introduction

It is well known that human doctors can make mistakes in diagnosis especially in the treatment of complex diseases. In 1999, IOM (U.S. National Institute of Medicine) published a landmark report "To err is Human" (people will make mistakes) [3], the report showed that: first, the quantity of medical errors is huge; second, most medical errors are caused by human factors which can be avoided by means of a computer system[14]. Therefore, improving the quality of care, control ling medical errors, improving patient safety are feasible and imminent.

However, doctors diagnosing the disease is still in the stage of using traditional experience in current medical procedures, they mainly depends on the experience, the diagnostic indicators and laboratory test results. It usually takes several years of

[*] Corresponding author.

working for a full-time physician to accumulate a certain diagnosis experience. It is significance to abstract the experience and knowledge. We can provide them for the doctors to make decisions in a convenient form, it will reduce the subjective blindness in medical activities, make the diagnosis more scientific, and thereby improve the level of disease diagnosis and treatment [19]. Thus, clinical guidelines came into being.

Clinical Decision Support System can effectively address the limitations of clinician knowledge[11], reduce human negligence, and reduce medical costs relatively [1], [3], so as to provide a guarantee for the quality of care. Despite the CDSS has many advantages, but just a few of them are really accepted by a doctor and put them into use, the main causes are listed below: 1) different doctors' needs are different. Very experienced doctors only need few critical pieces of advice from computers; while unexperienced doctors may need detailed advice. (2) System cannot adapt to the rapid changes of the clinical guidelines [2]. We are in a era of knowledge explosion, emerging evidence-based medicine, clinical guidelines are constantly being updated. A perfect decision support system should support and adapt to the rapid changes in clinical guidelines. But most of the existing systems failed to do so. (3) During the diagnosis of complex diseases, it often needs a medical team made up of some experts to discuss the problem, but in real life these doctors work in different environment, collaborations has become an important obstacle to solve the problem[15],[18].

This paper propose that a MAS based on the Execution Engine can effectively address the above deficiencies.

First, in the MAS, the conclusions deduced by the system shows in a user friendly interface that also displays the ranking of advice, the doctors can choose and pick the most valuable advice for their situations or make their own decision.

Second, in the MAS, clinical guidelines is described by XML format, we designed a multi-agent Execution Engine to parse this type of XML file, and the system can have timely access to these updated content at runtime and do not interrupt the operation of the system.

Third, the paper presents the idea of Multi-Agent Systems, which can divide a problem into many small parts in order to achieve the goal that agents complete tasks through discussing in the way of communication in the distributed environment.

2 Approach Overview

2.1 Agent Technology Proposing

Object-oriented software development is mainstream, although it has a lot of advantages, also some shortcomings are exposed, the most important deficiencies can be summarized as: object-oriented technology is not the most appropriate for the real world simulation. Agent software development is designed to solve this problem, which is closer to the reality of the human society and the general problem-solving methods and habits of human beings [13].

The basic idea of object-orientation is starting from the existing things of the real world, stressed that understanding and thinking of problem should be centered in things of the problem itself. According to the essential characteristics of these things we abstract things as objects of the system and the object works as a basic unit in the system. Through the basic concepts of object, class, inheritance, encapsulation, messages, etc. to program.

The basic idea is of agent-orientation is starting from the people and the surrounds in the real world. It thinks that the properties of things, especially the dynamic ones, are heavily influenced by the related people and environment.It emphasizes the interaction among understanding, thinking and objective things. It combines the subjective and objective characteristics of things and abstracts them into agent of system. The agent works as a basic unit in the system, through the team work of the agents to release the goal the system is aimed at [8].

With the development of computer science and technology, agent provides the possibility for cooperating among people to solve complex problem.

2.2 Agent Concepts

Agent originates in a conceptual model of distributed artificial intelligence. It is a new method of computation to solve complex, dynamic, distributed intelligent applications. It usually refers to a target, behavior and knowledge, solving problem by planning, deducing and decision-making according to their capacity, resources, status, and related knowledge in an uncertain environment. It is a kind of entity can complete specific tasks and achieve a certain goal independently [12].

Agent generally has the following major features [5]:

1) Autonomy (autonomy): After initialized, without user intervention, the Agent can make some decisions independently.

2) Reactivity: Agent can react in time to the changes of environment.

3) Collaboration: Agent has the ability to resolve conflicts through negotiation, which is the key of MAS (Multi-Agent System) systems working successfully.

4) Adaptability: With the interaction between the user and the environment, Agent can actively adapt to the environment and expand their knowledge.

5) Communication: This feature is extremely important in MAS, Information between the different Agents can be exchanged each other by communication and does not affect the independence of the Agents. In the MAS system, The ability of communication is the foundation of their ability to learn, and also is the basis for coordination, exchange and competition between different Agents [10]. We select FIPA as the communication language. It draws up a series of technical specifications which contain the architecture, communication language, content language and interaction protocols [20].

Currently, the vast majority of Agent-based systems are single agent, but with the proposing of increasingly complex application problems and the maturation of technology, the single agent systems cannot solve these problems because of the individual Agent's limitation of knowledge and computing resources. Typically, the most powerful tool to deal with complex problems is abstracting and modular that the multi-Agent system just provides. If a problem is very complicated, the only way to solve this problem is to develop a large number of modular components with special functions (i.e., Agent) designed to solve a specific aspect of the problem. This decomposition allows each Agent to resolve corresponding special problem using appropriate examples. When interrelated problems appear, each Agent in the system must be coordinated to ensure proper handling of this correlation.

Multi-Agent System has the following characteristics: Each member of the Agents only has the incomplete information and problem-solving ability, lack of a global perspective to coordination; Knowledge and Data are dispersed or distributed; the process of calculation is asynchronous and concurrent.

Meeting the needs of Multi-Agent Systems theory and practical application, a new MAS development platform –JADE arises at the historic moment. JADE (Java Agent Development Framework) is a multi-Agent Development Framework fully coded by Java [5], which follows the FIPA rules. It provides the basic naming services, yellow pages services, communication mechanisms, etc. can be effectively integrated with other Java development platforms and technology, which greatly simplifies the development of Multi-Agent systems in all aspects. JADE mainly includes the following sections: 1) A runtime environment on which agent to survive;2) a runtime library on which programmers to develop;3) a series of graphical tools, which can help users manage and monitor Agent status in runtime.

3 Case Study

Assuming that the patients have been enrolled in the scope of Urgent referral in the stage of breast cancer referral, this case study is about Triple assessment of the patients. Triple assessment is a common procedure in the UK National Health Service to determine whether or not a patient with suspected breast cancer does or does not have malignant or benign breast disease [4]. The process involves taking a comprehensive clinical history followed by decisions about imaging (typically mammography and ultrasound); biopsy (needle sampling) and the management of confirmed cancer. Genetic risk assessment is also included in this application.

The application includes decision support for four areas crucial to the evaluation of a patient attending a triple assessment clinic (updated from the original workflow). Triple assessment workflow is shown in Fig. 1.

1） Genetic risk assessment

The application helps assess familial and genetic risk of breast cancer. The RAGs applet is used here to construct a pedigree and calculate genetic risk for the individual patient, expressed as high, medium or low. Support for this task is based on NHS NICE Familial Breast Cancer guidelines.

2） Selection of correct mode of imaging (mammography, ultrasound or neither). The medical knowledge for this task is derived mainly from ACR guidelines.
3） Selection of the right mode of biopsy (if any). The knowledge for this task is derived from NHSBSP guidelines.
4） Selection of the optimum way to manage the patient based on examination, imaging and cytology results (it is assumed that core biopsy results if any would not be available on same day). The knowledge for this task is derived mainly from BASO guidelines [6, 7], [9].

'triple_assessment'defines the main tasks:examination, imaging_decision, ultrasound_enquire, biopsy_decision,and so on. These may need to be tagged with roles,for example Examination (Surgeon), Imaging_Decision/Ultrasound_Enquiry/ Mammography_Enquiry(Radiologist), Biopsy_Decision(Pathologist). Management Decision (Management).

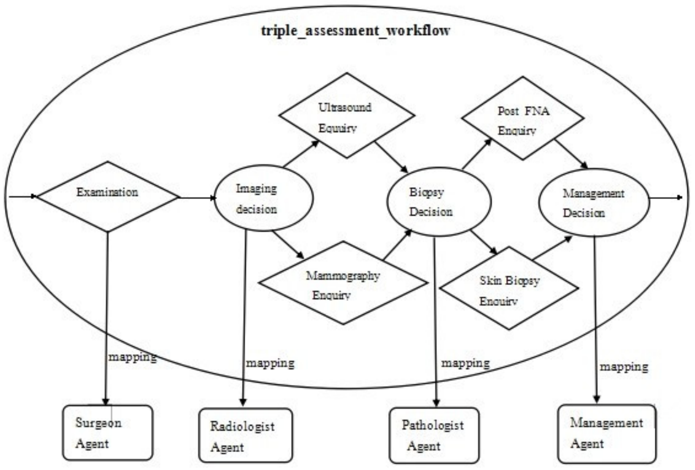


Fig. 1. Triple_assessment_workflow shows the main tasks and the role of necessary agents

4 Design and Implementation

4.1 The Architecture of the System

Multi-agent system architecture is shown in Fig. 2, In the MAS, when an event occurs (for example, a patient reports a symptom), it will trigger the first agent's behavior and process the message by performing the Execution Engine. The results produced by the Execution Engine will be sent to the next agent which is needed for cooperation(the collaborative agent is defined in the Behavioral Rule), the next agent processes received messages in the same way, until the completion of the task.

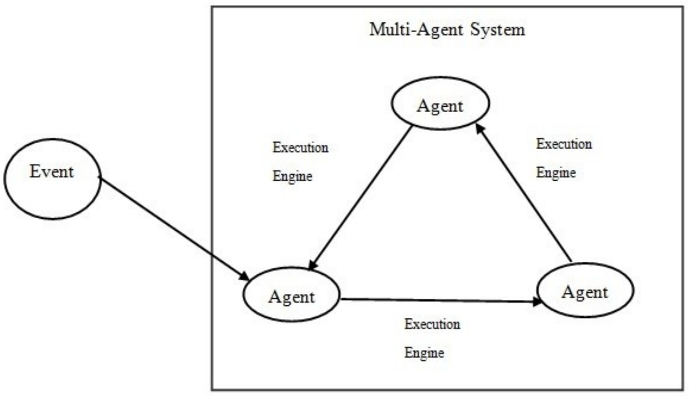


Fig. 2. Multi-agent system architecture

4.2 Agent Design

In the case of Triple assessment for suspected breast cancer, we design the following four agents: SurgeonAgent, RadiologistAgent, PathologistAgent, and ManagementAgent.

SurgeonAgent: SurgeonAgent is the doctor who patient firstly visits; he inquires basic information about the patient's symptoms, and makes the appropriate decisions.

RadiologistAgent: the function of this Agent is doing a mammogram of both breasts, or Do an ultrasound of the affected area based on the diagnosis made by the previous doctors.

PathologistAgent: the function of this Agent is doing further examination of the patient, such as Skin biopsy, Fine needle aspirate and so on.

ManagementAgent: the function of this Agent is making the final decision according to the comprehensive diagnosis made by all the doctors.

4.3 Execution Engine

Execution Engine is designed to get the latest XML content at runtime, without affecting the operation of the system. The structure of Execution Engine is shown in Fig. 3. It is made of Rule Repository, Rule Matching and Rule Interpretation Engine. When event occurs, the trigger agent first performs Behavioural Rule matching, and find the rules belonging to its own and can handle the current event from Rule Repository by Rule Matching component, and then return the rules to Rule Interpretation Engine component, which can parse the rules.When parsing <Processing> tag, it will do a derivation of the event according to Production Rule and get a result, and then select a branch which the condition equals the above result in Decision Tree. In accordance with the condition, the corresponding action will be performed and send the details to the defined receiver agent of the decision-making branch.

This design method not only can be used for diagnosis and treatment of breast cancer but also can apply to other diseases with the modification of the content in the BR and PR. We will introduce the Execution Engine in the following three parts and state the principle of the three components respectively.

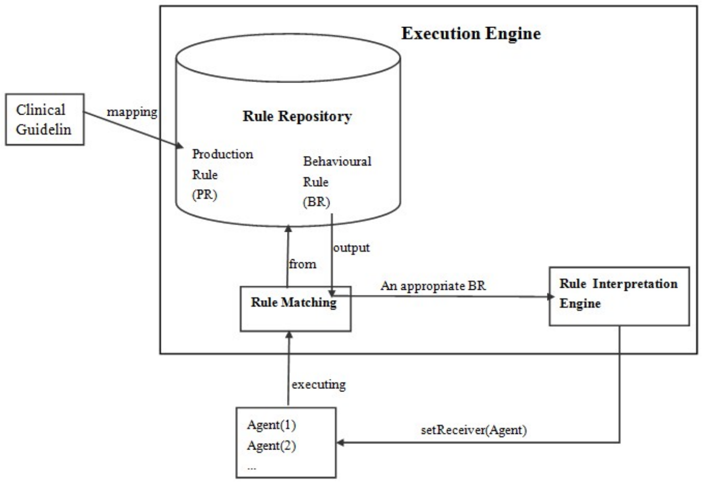


Fig. 3. Execution Engine Architecture, the key of the MAS

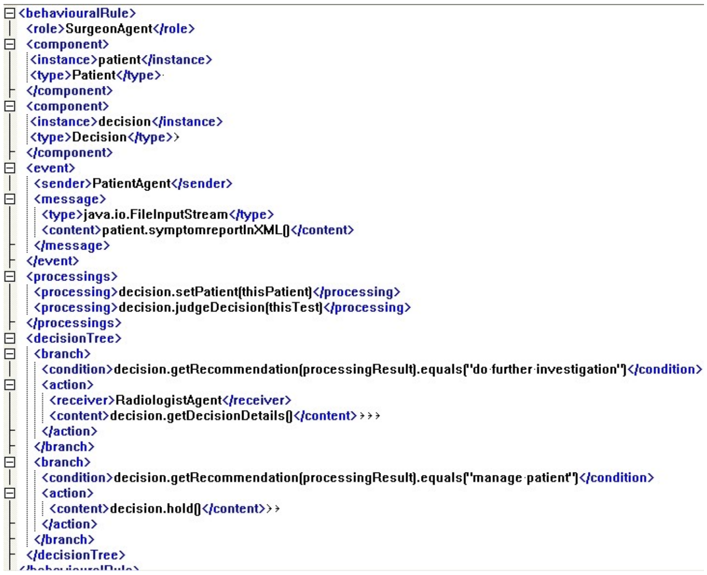
Rule Matching

When an event occurs, the agent will select a BR which can handle the current event according to the following conditions in Rule Repository: Sender, MessageContent and so on. It can be implemented in the following method:

```
//Get the current BR’s Sender, MessageContent
for (int j=0;j<list.size();j++){
Element event=root.getChild ("event");
String Sender=event.getChildText ("sender");
Element message=event.getChild ("message");
String MessageContent =message.getChildText ("content");
// comparison
ACLMessage msg=receive ();
If(sender.equals(msg.getSender().getLocalName())&&Message
Type.equals(msg.getContent())){
    return list.get(j);
}
}
```

Rule Repository

Behavioural Rule is used to guide the behavior of the agent which is defined in the form of XML. All Behavioural Rules are placed in Rule Repository, each Behavioural Rule is defined for some agents to dealing with a certain events, and each agent has its own rule. The expressions of Behavioural Rule are as follows: {event, processing, decision (condition, action) n}.An XML-based specification of a Behavioural Rule is shown in Fig. 4.The definition of BR with the similar structure describes that when an event occurs, if it matches the event which is defined in a Behavioral Rule, then the agent will handle the message by the pre-defined <Processing>, and then perform an action according to condition. Finally, send the output to the receiver agent and this output becomes the event which triggers the next collaboration agent to participate in.



```
<behaviouralRule>
  <role>SurgeonAgent</role>
  <component>
   <instance>patient</instance>
   <type>Patient</type>
  </component>
  <component>
   <instance>decision</instance>
   <type>Decision</type>>
  </component>
  <event>
    <sender>PatientAgent</sender>
    <message>
      <type>java.io.FileInputStream</type>
      <content>patient.symptomreportInXML()</content>
    </message>
  </event>
  <processings>
    <processing>decision.setPatient(thisPatient)</processing>
    <processing>decision.judgeDecision(thisTest)</processing>
  </processings>
  <decisionTree>
    <branch>
      <condition>decision.getRecommendation(processingResult).equals("do further investigation")</condition>
      <action>
        <receiver>RadiologistAgent</receiver>
        <content>decision.getDecisionDetails()</content>>>>
      </action>
    </branch>
    <branch>
      <condition>decision.getRecommendation(processingResult).equals("manage patient")</condition>
      <action>
        <content>decision.hold()</content>>>
      </action>
    </branch>
  </decisionTree>
```

Fig. 4. Specifition of a Behavioral Rule for the SurgeonAgent

Rule Interpretation Engine

This component is responsible for parsing BR, and can be able to get the XML content dynamically at runtime. It can be implemented by the combination of parsing XML with JDOM and java reflection mechanism.

(1) DOM and SAX are two common ways to handle XML documents [16, 17]. JDOM is an open source project, which is based on a tree structure, using java technology to achieve the parsing, generation, serialization, and a variety of operations

of XML document. The API of JDOM integrates the advantages of DOM and SAX, which makes DOM and SAX cooperate more naturally and coordinate.

(2) Reflection is a very important characteristic that makes Java be considered as a dynamic language, which allows dynamically discovery and bind classes, methods, fields, and all other element generated by the language. In other words, by using a mechanism such applications can achieve a description of their behavior and monitoring, and adjust or modify the state and related semantics describes behavior according to the state and result of their own behavior. Java reflection mechanism mainly provides the following functions: judging which class any object belongs to at runtime; construct an object of any class at runtime; judging what member variables and methods any class has at runtime; invoking the method of any object at runtime.

For example we want to exceute the portion of <processing> in the BR by the Rule Interpretation Engine; the defined form of < processing > is as follows:

```
<Processing>decision.judgeDecision (thisTest)
</processing>
```

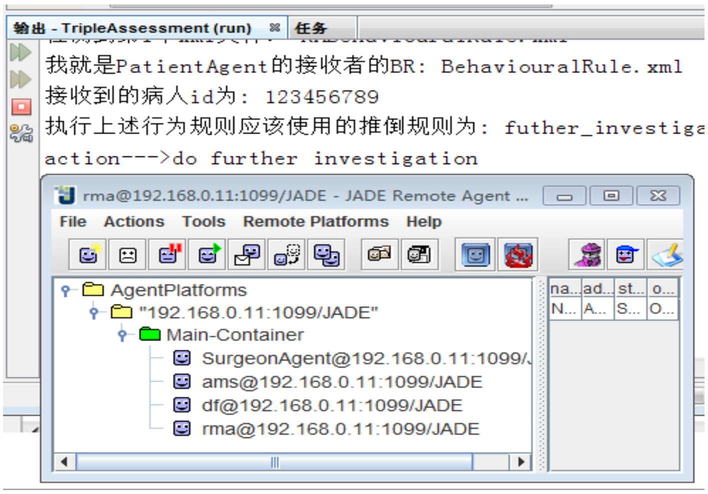
We can do a sample implementing:

```
/*Assume that there is only one processing, get the
processingcontent*/
String ProcessingContent=root.getChildText ("processing");
/*the processingcontent is the form of "object.method ()",
so we must use the java reflection mechanism.*/
/*before the first appeared point in the processingcontent
is the name of" object"*/
InstanceName=ProcessingContent.substring(0,processingCont
ent.indexOf (".")) 
int firstCircle=ProcessingContent.indexOf("(");
/*between the first point and the left bracket is the name
of method*/
methodName=ProcessingContent.substring(firstPoint+1,
firstCircle);
Method
method[]=instanceName.getClass().getDeclaredMethods();
If method[i].equals(methodName){

Obj=method[i].invoke(instanceName.getClass(),parameters[]);
}
```

In the case of Triple assessment for suspected breast cancer, when a patient reports symptoms, PatientAgent will send the Patient as an object to the SurgeonAgent, after SurgeonAgent receives an event, it knows the Sender of Event is PatientAgent; the content of the event is patient. It will find the satisfying Behavioural Rule according to these conditions in Rule Repository. And the Behavioural Rule found will be sent to Rule Interpretation Engine for parsing. If the conclusion of the production is do further investigation, the content of first branch in BR's DecisionTree will be executed, and then the result will be sent to the receiver RadiologistAgent. RadiologistAgent processes the message by the same Execution Engine and communicates with the next one, until the completion.

4.4 Expected Results

In the case of Triple assessment for suspected breast cancer, four agents are distributed on three computers. They separately execute Execution Engine, and the results are sent to the appropriate agent. PatientAgent is deployed in the computer with IP address 192.168.0.5, SurgeonAgent is deployed in the computer with IP address 192.168.0.11, RadiologistAgent is deployed in the computer with IP address 192.168.0.17. When PatientAgent sends a message to SurgeonAgent, the results are shown as Fig. 5. SurgeonAgent (192.168.0.11) receives a message, and matched the corresponding rules which can process current event,The selected rule is "BehaviouralRule.xml", the results deduced is do further investigation. The corresponding receiver should be RadiologistAgent according to the content of the BR decision tree.

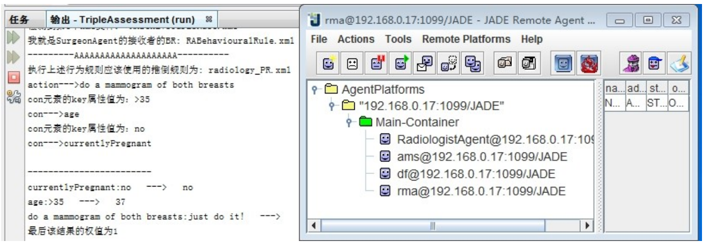


```
我就是PatientAgent的接收者的BR: BehaviouralRule.xml
接收到的病人id为: 123456789
执行上述行为规则应该使用的推倒规则为: futher_investiga
action--->do further investigation
```

输出 - TripleAssessment (run)
任务
rma@192.168.0.11:1099/JADE - JADE Remote Agent ...
File
Actions
Tools
Remote Platforms
Help
AgentPlatforms
"192.168.0.11:1099/JADE"
Main-Container
SurgeonAgent@192.168.0.11:1099/
ams@192.168.0.11:1099/JADE
df@192.168.0.11:1099/JADE
rma@192.168.0.11:1099/JADE
na...
ad...
st...
o...
N...
A...
S...
O...

Fig. 5. SurgeonAgent receives messages sent by PatientAgent.and deduced a result by performing Execution Engine

As is shown in Fig. 6, the RadiologistAgent (192.168.0.17) receives the message sent by SurgeonAgent and it executes Execution Engine, with the corresponding decision made described as follows.

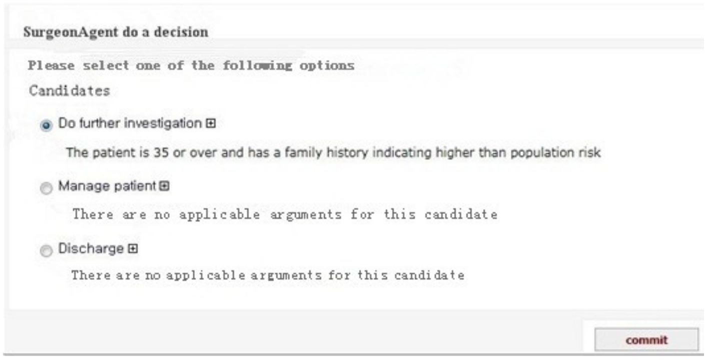

任务
输出 - TripleAssessment (run)
rma@192.168.0.17:1099/JADE - JADE Remote Agent ...
File Actions Tools Remote Platforms Help
na... ad... st... o...
N... A... ST... O...

```
我就是SurgeonAgent的接收者的BR: RABehaviouralRule.xml
---------AAAAAAAAAAAAAAAAAAAA----------
执行上述行为规则应该使用的推倒规则为: radiology_PR.xml
action--->do a mammogram of both breasts
con元素的key属性值为: >35
con--->age
con元素的key属性值为: no
con--->currentlyPregnant

-------------------------

currentlyPregnant:no   --->   no
age:>35   --->   37
do a mammogram of both breasts:just do it!   --->
最后该结果的权值为1
```

```
AgentPlatforms
  "192.168.0.17:1099/JADE"
    Main-Container
      RadiologistAgent@192.168.0.17:109
      ams@192.168.0.17:1099/JADE
      df@192.168.0.17:1099/JADE
      rma@192.168.0.17:1099/JADE
```

Fig. 6. RadiologistAgent receives messages sent by SurgeonAgent. and made a decision by performing Execution Engion

The results will be listed in a user friendly interface, and it is just a kind of suggest. Whether it is adopted or not is up to the doctors themselves. For example when SurgeonAgent do a decision, according to the production rules, there are three options that are listed below Fig. 7:



SurgeonAgent do a decision

Please select one of the following options

Candidates

[✓ Do further investigation ⊞]

The patient is 35 or over and has a family history indicating higher than population risk

Manage patient ⊞

There are no applicable arguments for this candidate

Discharge ⊞

There are no applicable arguments for this candidate

Fig. 7. SurgeonAgent do the option according to the decision made by system

5 Discussion and Conclusion and Future Work

Currently, lots of CDSSs have many disadvantages when used in the practical application.For example,It is difficult to update their knowledge and can't meet the new requirements so that it must be Re-development.This lead to increasing of the cost greatly.The last inconvenience is medical team often locate in different work environment but can't discuss each other.The system's design meets the aim to complete tasks and reach the final goals with cooperation and exchange in the way of communication between agents in the distributed environment. Each agent plays a different role and has a different function in the system. But in this MAS, any agent with its own role can complete their task by the same set of Execution Engine. It can also be used in diagnosis and treatment of many other diseases in addition to breast cancer. We can only modify the content of PR and BR. Therefore, the Execution Engine can be used commonly.

However, the design of the Execution Engine also has some limitations, such as the definition of requirements must be in a fixed XML format, if the format of XML need to be changed, so as to the Execution Engine. So there are some defection in the flexibility and this is what can be improved in future work.

Acknowledgments. This work is supported by the National Science Foundation of China under Grant Number 61202101 and 61151001.

References

1. Xiao, L., Cousins, G., Fahey, T., et al.: Developing a rule-driven clinical decision support system with an extensive and adaptative architecture. In: 2012 IEEE14th International Conference on e-Health Networking, Applications and Services (Healthcom), pp. 250–254. IEEE (2012)
2. Xiao, L., Greer, D.: Adaptive agent model: Software adaptivity using an agent-oriented model-driven architecture. Information and Software Technology 51(1), 109–137 (2009)
3. Patkar, V., Hurt, C., Steele, R., et al.: Evidence-based guidelines and decision support services: a discussion and evaluation in triple assessment of suspected breast cancer. British Journal of Cancer 95(11), 1490–1496 (2006)
4. Séroussi, B., Bouaud, J., Gligorov, J., et al.: Supporting multidisciplinary staff meetings for guideline-based breast cancer management: a study with OncoDoc2. In: AMIA Annual Symposium Proceedings of the American Medical Informatics Association, vol. 2007, p. 656 (2007)
5. Bellifemine, F., Poggi, A., Rimassa, G.: Developing multi-agent systems with JADE. In: Castelfranchi, C., Lespérance, Y. (eds.) Intelligent Agents VII. LNCS (LNAI), vol. 1986, pp. 89–103. Springer, Heidelberg (2001)
6. Ahmed, I., Nazir, R., Chaudhary, M.Y., Kundi, S.: Triple assessment of breast lumps. J. Coll. Physicians Surg. Pak. 17(9), 535 (2007)
7. Drew, P.J., Chatterjee, S., Turnbull, L.W., et al.: Dynamic contrast enhanced magnetic resonance imaging of the breast is superior to triple assessment for the pre-operative detection of multifocal breast cancer. Annals of surgical oncology 6(6), 599–603 (1999)
8. `http://jade.tilab.com/`
9. `http://www.openclinical.net/demos/triple-assessment-guideline-for-secondary-care.html`
10. Bellifemine, F., Poggi, A., Rimassa, G.: Developing multi-agent systems with a FIPA-compliant agent framework. Software-Practice and Experience 31(2), 103–128 (2001)
11. Xiao, L., Fox, J., Zhu, H.: Developing an Open and Adaptive Agent Architecture to Support Multidisciplinary Decision Making
12. Xiao, L., Greer, D.: Environment support for the configuration of adaptive agents. Multiagent and Grid Systems 5, 1–23 (2009)
13. Xiao, L., Fox, J., Zhu, H.: An Agent-oriented Approach to Support Multidisciplinary Care Decisions. In: 2013 3rd Eastern European Regional Conference on the Engineering of Computer Based Systems (ECBS-EERC), pp. 8–17. IEEE (2013)

14. Xiao, L., Hu, B., Croitoru, M., et al.: A knowledgeable security model for distributed health information systems. Computers & security 29(3), 331–349 (2010)
15. Musen, M.A., Shahar, Y., Shortliffe, E.H.: Clinical Decision-Support Systems Biomedical Informatics, pp. 698–736. Springer, New York (2006)
16. Harold, E.R.: Processing X M L. with Java: a Guide to SAX, DOM, JDOM, JAXP, and TrAX (2003)
17. Hunter, J.: JDOM makes XML easy. In: Java Developer Conference on Sun's 2002 Worldwide (2002)
18. Garg, A.X., Adhikari, N.K.J., McDonald, H., et al.: Effects of computerized clinical decision support systems on practitioner performance and patient outcomes. JAMA: The Journal of the American Medical Association 293(10), 1223–1238 (2005)
19. Pestotnik, S.L., Classen, D.C., Evans, R.S., et al.: Implementing antibiotic practice guidelines through computer-assisted decision support: clinical and financial outcomes. Annals of Internal Medicine 124(10), 884–890 (1996)
20. Bellifemine, F., Poggi, A., Rimassa, G.: JADE–A FIPA-compliant agent framework. In: Proceedings of PAAM 99(97-108): 33 (1999)