Design and Implementation of Service Discovery Architecture Based on Multi-Agent Systems in an Ad-Hoc Environment - For the Observing and Recording System

Feng-Chao Yang*, Chia-Hao Chang, and Cheng-Li Chang

Department of Information Management, Da-Yeh University

Changhua 515, Taiwan, ROC

yfc@mail.dyu.edu.tw

Received 29 November 2006; Revised 16 March 2007; Accepted 24 March 2007

Abstract. Since the mobile device characters features in motion while traditional information services are unable to meet demands of the mobile environment, this study therefore proposes a "multi-agent architecture – a DASS (Distributed Agent-based Service Sharing) – with the purpose to design and implement a service sharing architecture using the autonomy and communication abilities of agents, to actively discover services to mediate among shared services, so as to allocate and manage the distributed service. The system development process follows PASSI (a Process for Agent Societies Specification and Implementation) methodology to form "multi-agent society" models. System development of this study is carried out on the agent middleware platform, JADE (Java Agent DEvelopment Framework); substantial contributions are: (1) Establishment of service-sharing System by using the agent technology that uses a single window for communicating with other agents; (2) the system framework and the service discovery template accomplished in this study enable the developers to promptly configure service-sharing applications; (3) inheriting the abstractive service search mechanism of FIPA (The Foundation for Intelligent Physical Agents), this System has increased compatibilities with other systems; and using the existing JXTA based service discovery architecture to implement service distribution and search on the agent platform resolves difficulties in accessing peer to peer services in the Ad-Hoc network; (4) the System sets up mediation models for service-sharing according to agent communication language set up with (defined by) FIPA standards, and sets up communication ontology for service-sharing, therefore in the communication process, it is easy to understand intentions of each other without the need of particular understanding of the vocabulary and syntax used by each other; this substantially simplifies communication problems between heterogeneous agents.

Keywords: multi-agent system, Agent Society, FIPA Specifications, PASSI Methodology, JXTA

1 Introduction

Following the gradually matured development and popularization of mobile devices, their calculation functions are mostly capable of meeting with majority of personal requirements; however, comparing with the popularity and functionality provided to users by the calculating and storing facilities of PCs, there still exists rooms for further development in mobile devices. Architecture of application programs can be divided into individual and collaborative ones. For supplementing functional insufficiencies, many have proposed agent technology, together with decentralized architecture, to distribute resources and functions onto various devices, so as to strengthen the insufficient capabilities of personal mobile devices [1][2]. The agent issue, however, exists problems in communication and discovery aspects; this study aims to incorporate collaborative concepts to implement service sharing on mobile devices; incorporating agent technology enables to increase system flexibility, such as in system configuration, service deployment, service discovery, and so on; agents are capable of solving the unstable service accesses problems in mobile devices owing to the highly mobile nature.

This study aims to build up a multi-agent system [3] in the mobile environment and to extent the exiting service discovery mechanism [4] so as to solve the problem of mutual exchange of services under different protocols, and to manage resources and information services [5] of the physical mobile devices by using the autonomy and decision making capability of agents, as well as to propose flexible transfer of different service discovery protocols based on FIPA (The Foundation for Intelligent Physical Agents) organization [6] agent discovery mechanism, thus to improve flexibility and stability of the service sharing system.

* Correspondence author

Mobile devices can classify as many groups of small LAN in the Ad-Hoc environment. However, services can't communicate each other, because mobile devices use different communication protocols. Therefore, this research extends agent discovery mechanism to implement a service sharing platform. To accomplish the purpose, we must solve some problems are as follows:

1. Communication among the heterogeneous agents: These agents must use the same agent communication language to reduce the gap among them when they want to do some behaviors like communication, negotiation or collaboration. However, many system developers build their agent systems by using the traditional object-oriented method, they must rewrite or modify some source codes for sharing their resources or running tasks when these agents follow different communication standard. When the system developers build the agent systems which follow the same standard, therefore it makes the heterogeneous agents can communication among them.
2. Translation of service descriptions among service-discovery mechanisms: The traditional agent platforms have no the standard service descriptions format, but nowadays the agent middleware follows the FIPA standard which uses XML. FIPA standard also proposes the bit-efficient format in the low bandwidth network environment. To think about the existing Service Discovery Protocol (such as JXTA, Bluetooth) and the characteristic of low bandwidth network in Ad-Hoc environment, the agents' service descriptions incur the format and version control problems. This study introduces the Template Translating Agent for the middleware translation mechanism, and this agent searches the suitable agent platform and registers the physical location temporarily for other platform's agents by integrating these different service descriptions.
3. Self-Configuration and Detection of Agent for their Environment: The end users don't know more about the related knowledge of protocol architecture. The agents need the self-configuration ability for deployment, and have the ability of evaluating the performance and ability of mobile devices. Because of the limited resources of mobile devices, the agents must manage and allocate them in a well-defined manner.
4. Friendly User Interface and Configuration: From the users' viewpoint, they need not understand how to use which kind of the connection protocol and architecture of the service of the Discovery Agent. The Discovery Agent detects the existing devices and the supplied services of agent platform under some pre-defined conditions (such as the user preferences, the connection method and cost, the service function and quality) constantly. Besides, thinking about the relationship within the agent society, the pre-defined cooperation or the frequent access point can be registered in the static cache for more fast access in the future.
5. Frequent Connection in Ad-Hoc [7][8]: The dynamic and the static peer to peer connection methods are different; the mobile device lacks of resources (such as CPU and storage), the agent on the mobile device cannot provide the same strength and continuous services just like directory servers in the agent platform on the PC. It is possible to incur the service link error, because the mobile devices are not within the connection scope, or the poor quality, or the bad routing protocol. This study applies the concept of peer to peer storage, accesses data among the different agent platforms, and update service information on the regular time schedule in the distributed manner.

2 Survey of Related Researches

PASSI, a Process for Agent Societies Specification and Implementation, is the methodology of the multi-agent system [9][10] where detailed definition and specifications are provided for from requirement analysis to program compilation and deployment. PASSI integrates design models and design concepts in both object oriented software engineering and multi-agent, and completes analysis of multi-agent system by way of 5 models and 12 steps using the expression of Unified Modeling Language, UML [11] to describe the system architecture.

The main reason that this study uses PASSI methodology lies on the more flexibility of looping development in analysis and design phases; this is because the PAASI methodology allows arbitrary addition of new requirements to remodel system requirements during the design phase. PASSI methodology systematically and clearly depicts the interactive relationship, communication models and the agent relationship. General UML descriptions used in system model analysis and design allow other analyzers to quickly understand the system architecture by way of graphics; PASSI provides convenient analysis steps that allow system analysts easily and quickly enter into system implementation phase, to shorten the developing time of the Multi Agent System [10][12].

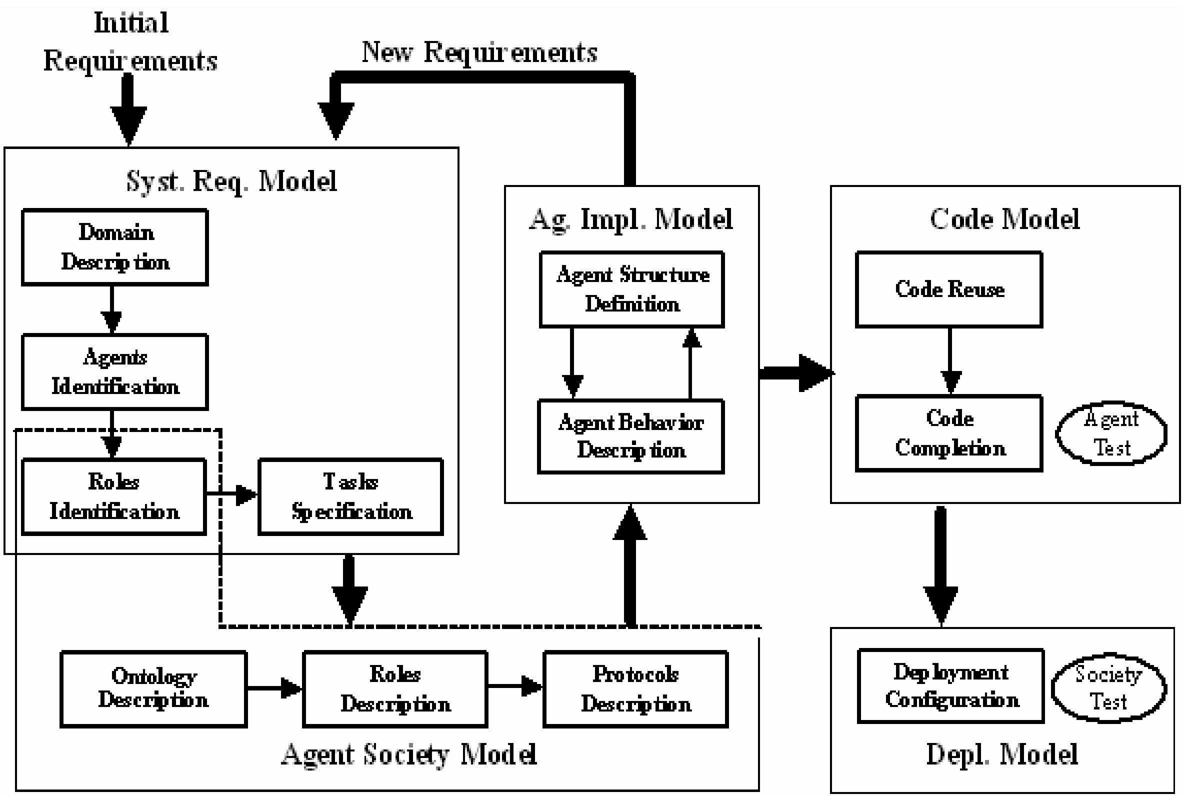


Fig. 1. PASSI Methodology Flowchart.

Fig. 1 depicts agent modeling process of the PASSI methodology, from the initial system requirement to the System Requirements Model of the methodology, Agent Society Model, Agent Implementation Model, Code Model, Deployment Model, etc. New requirements are allowed to add in to the Agent Implementation Model at any time; the serial repetition makes the architectural analysis and design of the multi-agent system more precise, facilitating system developers to perform system slicing and implementation [13].

3 System Requirement Analysis

3.1 User Requirement Analysis

Traditional service-sharing architecture has various disadvantageous and inconveniences, the study induced several reasons for them from the view points of both the users and the system developers. From that of the users', although the user owns mobile devices that possess on-line capabilities, not every user knows the networking architecture nor is he/she capable of configuring the network; further, services of different protocols are not capable of discovering each other, rendering the users have to use devices that support the same protocol so as to be able to share the services.

For the system developers, new subsystems must be developed from scratch to cope with "user requirements" for services that follow different protocols; moreover, when supporting different discovery service protocols, system developers generally do not consider the flexibility of the system discovery mechanism because revision of the system is required to accommodate the new protocol. [14]

Fig. 2 illustrates independent operation of each agent within the platform. When a user files a service requirement, the agents accomplish the task in the form of collaborative cooperation. [15] The agent platform proposed in this study proposed following roles base on requirement analysis of user functions; they are respectively named Discovery Agent, Template Translating Agent, Config Agent, Recording Agent, Resource Management Agent, and Invoking Agent.

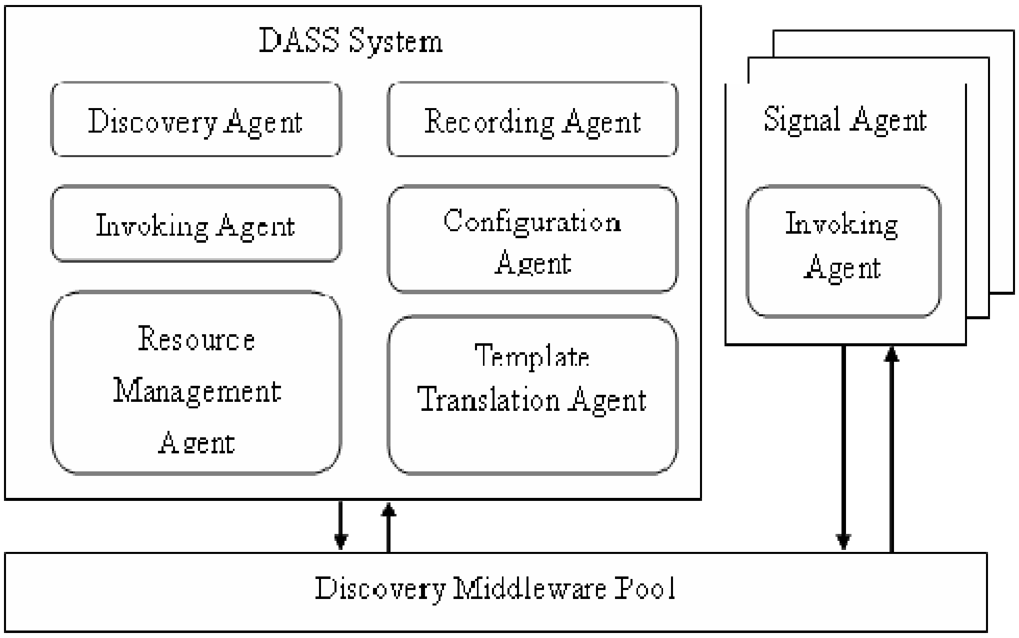


Fig. 2. The DASS (Distributed Agent-based Service Sharing) Structure of the Study.

3.2 Model of System Requirement

Use Case Diagrams are used to depict system functions in the Domain Description phase, with detailed scenario given for use at the role identification phase. The domain description diagram is for assisting the system analyst to carry out preliminary analysis of the entire system, and to split system functional requirements. Fig. 3 is the domain description of DASS (Distributed Agent-based Service Sharing). The scenario of the system triggering the discovery of matters is described as follows:

1. A system is triggered by the requisition function of the service request, which enables the service searcher mechanism to autonomously search for appropriate services and transfer the service into a service file format.
2. Search Program, a process that is responsible for searching the services, will firstly attain current device resources; connection and user configure information from Basic Configuration. Search Program then transfers defined service requirements to Template Translating according to protocols used by different configurations for broadcasting or routing to device locations. After this action, a record of devices and services will be received, Search Program then requests Recording Service Description of the local platform for updating the service log. The Recording Service Description will periodically submit search requisition to Discovery Service to ensure effectiveness of the recorded data.
3. After attaining service log, Invoking Service directly calls the remote agent platform or indirectly access the remote service via a third party agent gateway. Prior to building up the link, it is required to attain the authorization. The service request will search for service providers that have already authorized the service request, among groups, to proceed with the service connection. Besides that, when calling for service, it requires to consider the calling mode so that Invoking Service is capable to coordinate with different transferring modes.
4. Resource Service Management comprises deploy resource, control security, monitor resource of system, and maintain lease activities which can provide for resource configuration of the local agent platform [16].

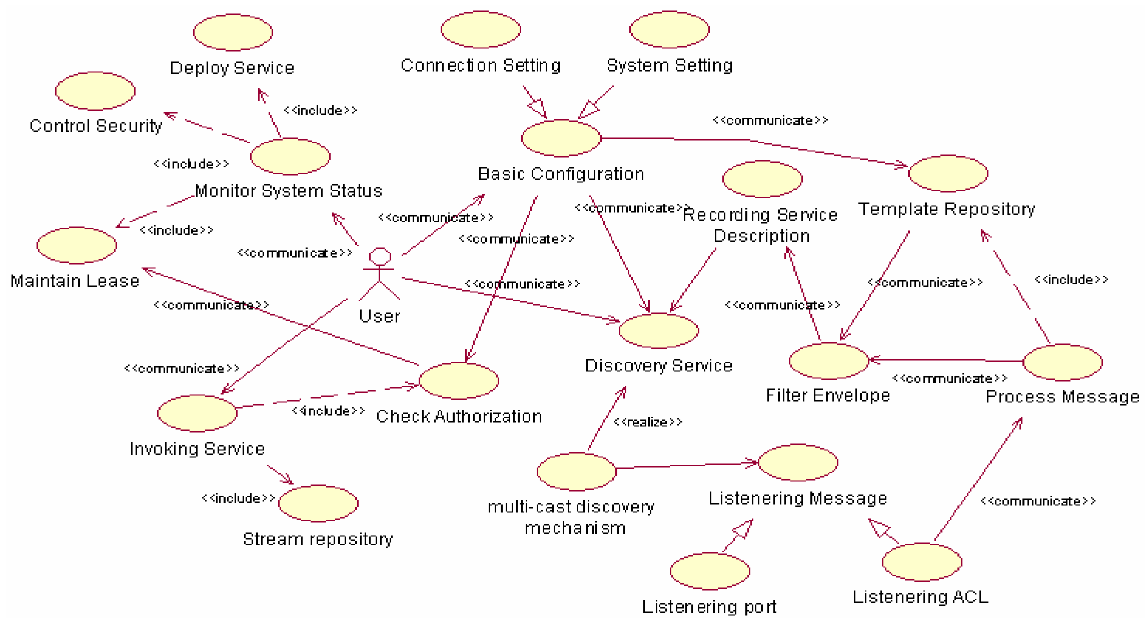


Fig. 3. System Domain Description Diagram.

Agent Identification Phase. The agent identification diagram depicting all the agents of the service sharing mechanism helps in understanding the role played by the agent in the system and the interacting relationship with other roles.

After proceeding multi-agent domain description, we proceed with the agent identification phase as shown in Fig. 4; each package represents an agent role; the package includes the action capability of the agent. By categorizing the domain descriptions of Fig. 3 in term of the tightly coupled functionalities, they are respectively:

1. Resource Management Agent: system operation is initiated on receipt of user requisition; Resource Management Agent deploys for the service of devices, the system also automatically detects device resources and select if service is released according to user preferences and presented in a unique style.
2. Discovery Agent: a Discovery Agent autonomously queries for services opened on a remote agent platform via third party protocols (such as JXTA, Bluetooth) deployed on the device. Searcher controller firstly attains resource information of the current device from Config Agent and then transferred service requirements defined by the user to Template Translating Agent, and then broadcasts with protocol used by different architecture or searches for each device node according to third party protocols. After that, response from agent platform of each device will be received, with the reply of services that are deployed on the remote platform. The local end searcher controller will then request Recording Agent to update the current service log.
3. Invoking Agent: when the local end Invoking Agent carries out the service call to a remote agent platform, two different situations may be seen: (1) direct access to the remote service; (2) accomplish service sharing via a third party gateway as a relay. Whether or not of one of the above situations, the service request must attain authorization from the service provider prior to building up the connection.
4. Config Agent: the Config Agent, being the window for the user to access the system, communicates with the user to attain user preferences for configuring the system. It requires action abilities to detect linking capability, operation ability, and storing capacity of the mobile device; it also needs to remind other agents of the current status of the mobile device and that of the user.
5. Recording Agent: the Recording Agent attains a descriptive file on the services deployed for the machine; it also stores service descriptions searched from other platforms. The Recording Agent periodically issues re-search requisitions to Discovery Agent to update service descriptions of its own platform. When the remote agent platform performs a search against the local machine, the Recording Agent will compare the query issued by the remote Discovery Agent and reply appropriate linking references to the remote platform via the Discovery Agent.
6. Template Translation Agent: the Template Translation Agent is capable of package parsing for Bluetooth and JXTA service discovery mechanisms. Combining with the agent communication language FIPA ACL, it packages the service descriptions on the system. Currently package format of the DASS agent communication language includes mainly XML, string, and byte.

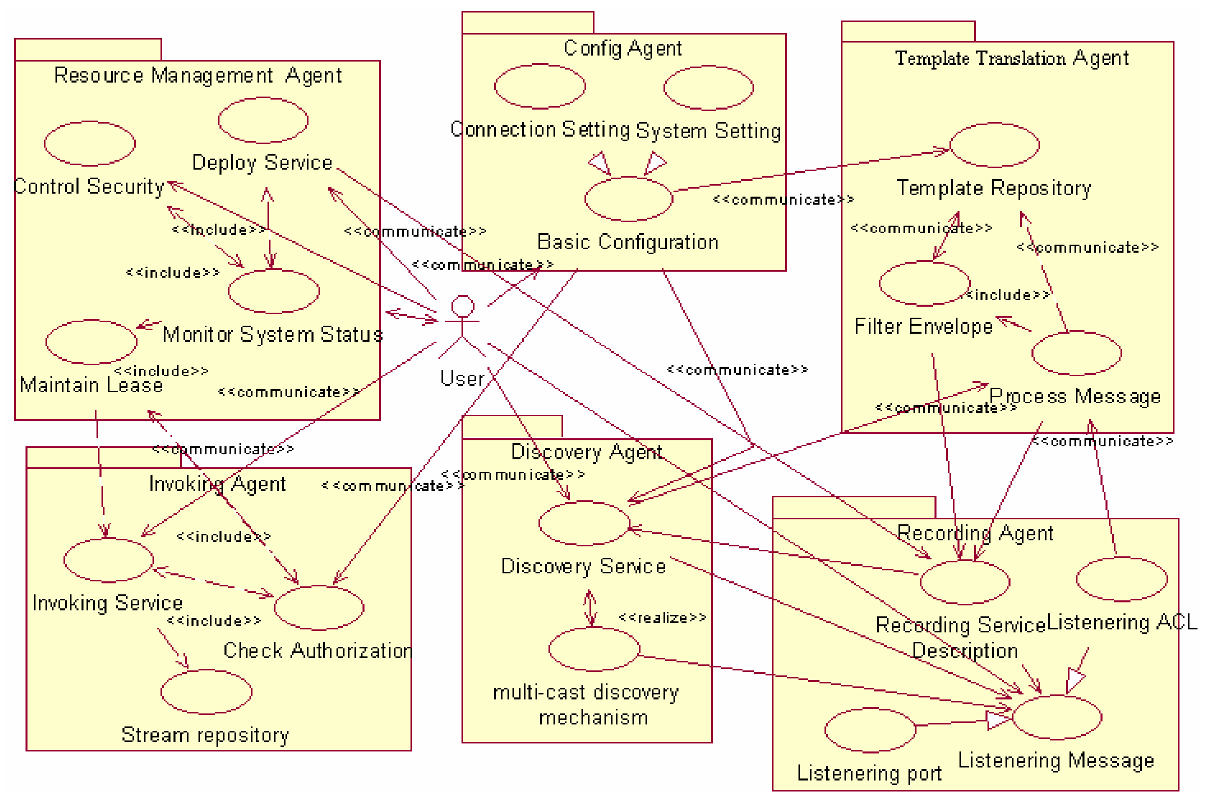


Fig. 4. System Agent Identification Phase.

Role Identification Phase. In the role identification phase, the scenario generated in the agent identification phase is described using a sequence diagram for the sequential order, with detailed interactive relationships between the agents thoroughly depicted; the study will illustrate this for the Discovery Agent as an example. Other agents are of the same manner and therefore not repeated.

Role identification of Discovery Agent is illustrated in Fig. 5. After service requester issues the request against Discovery Agent, the Discovery Agent needs to attain from Config Agent of the current platform environment and user preferences, and to transfer the attained user service requirements and discovery mechanism to be used to Template Translating Agent for converting the packet, with the result so attained replied to Discovery Agent, for Discovery Agent to proceed with platform search via a third party service discovery protocol. Discovery Agent s of remote platforms will then, according to the attained service request, submit the request to Recording Agent for proceeding with data filtering. Lastly, the result will be sent back to the Discovery Agent on the platform that the request was issued from and recorded.

Task Specification Phase. The task specification phase is depicted with UML [17] activity diagram for the interactive capability between a specific agent and other agents, it is divided into 2 parts: the right half of the figure is the internal task flow of the agent; the left half is the reactive relationship between the specific agent and other agents. This paper discusses the Discovery Agent as an example. Other agents are of the same manner and therefore not repeated.

Fig. 6 depicts task scope of the Discovery Agent and the mode of communication with other agents. At the beginning the service user submits a service request to Discovery Agent; packet conversion is made by Template Translating Agent addressing the protocol; and then, via a third party protocol, search of service is carried out against the remote agent; lastly the information sent back is passed on to Recording Agent to proceed with compilation of service description files.

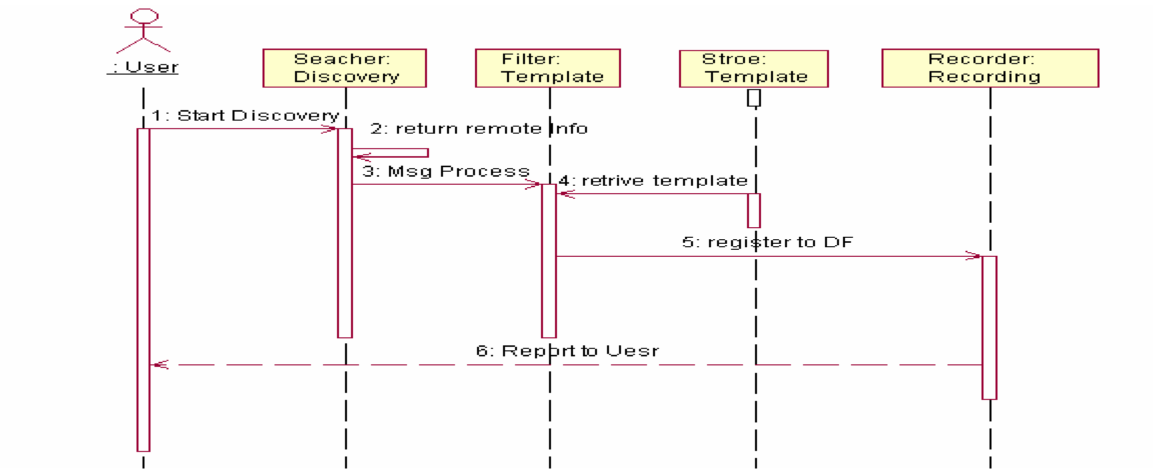
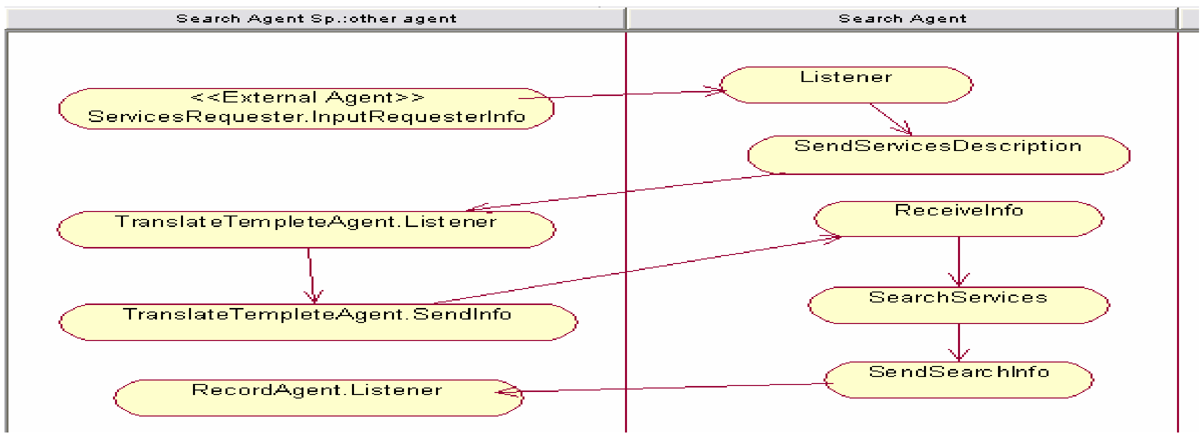


Fig. 5. Discovery Agent Role Identification.



Search Agent Sp.:other agent
Search Agent
<<External Agent>> ServicesRequester.InputRequesterInfo
Listener
SendServicesDescription
TranslateTempleteAgent.Listener
ReceiveInfo
TranslateTempleteAgent.SendInfo
SearchServices
RecordAgent.Listener
SendSearchInfo

Fig. 6. Job Specification of the Discovery Agent.

4 Analysis and Design of System Architecture

4.1 Agent Society Model

There are 4 phases of the Agent Society Model, namely role identification phase, ontology description phase, role description phase and protocol description phase. We will only discuss ontology description phase and role identification phase which are of essential importance.

Following the PASSI methodology, the Agent Society Model has four phases, including Role Identification Phase, Ontology Description Phase, Role Description Phase and Protocol Description Phase. Role Identification Phase describes the relationship among the agents and the actors, and Agent Society Model extends on Role Description Phase. To reduce the duplicate content, we don’t discuss these issues again. However, the agents could know other agents’ attention by sharing the same knowledge; this is the reason why Ontology Description Phase is existed. Besides, each agent play the different role in the different time, and each role provides the different services, so we have to descript agents’ role in detail during the communication. This is the purpose of Role Description Phase. To avoid they don’t know how to response the message among the agents, they must know the message performative each other, and how to handle the messages from other agents etc, they can communicate efficiently, so this is the reason why Protocol Description Phase has to be proposed.

Ontology Description Phase. In this phase PASSI methodology defines two ontology descriptions namely Domain Ontology Description (DOD) and Communication Ontology Description (COD). DOD uses three "relationship elements" of the class diagram – generalization, association and aggregation – to configure the ontology.

Fig. 7 is the Ontology Description of the Discovery Agent Domain describing the agent domain ontology used by the Discovery Agent. Discovery service of JXTA is based on the Advertisement file, while each of other service discovery architectures [18] owns an individual Advertisement description. The header of the internal data of these service discovery description files includes simplified text attributes such as station name, physical location, web address, name of agent platform, service description on the agent platform, etc.

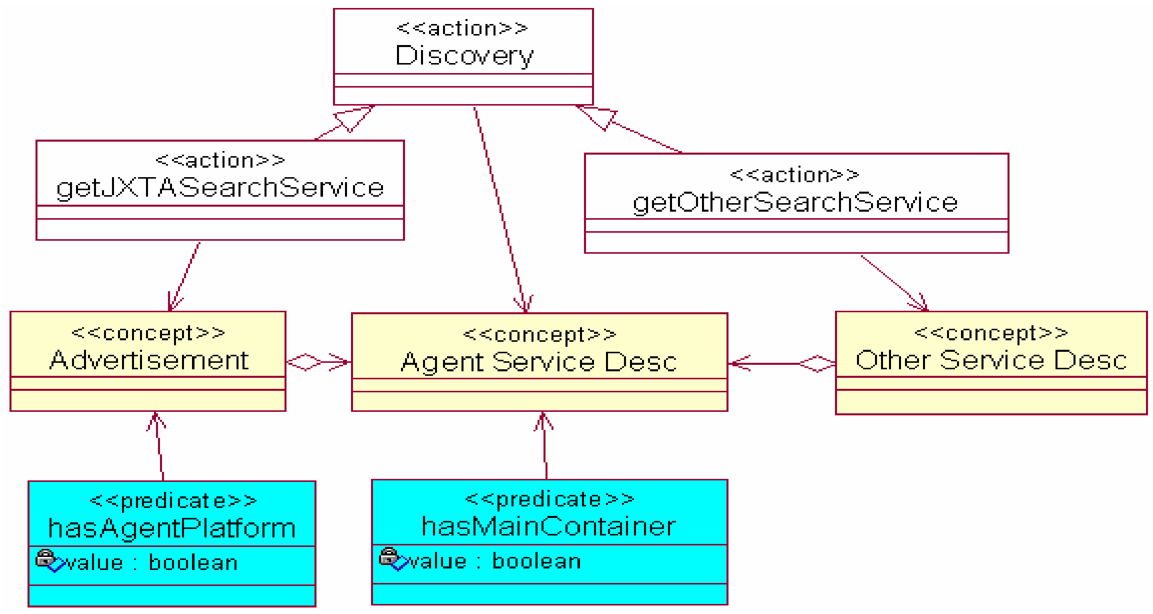

<<action>> Discovery
<<action>> getJXTASearchService
<<action>> getOtherSearchService
<<concept>> Advertisement
<<concept>> Agent Service Desc
<<concept>> Other Service Desc
<<predicate>> hasAgentPlatform
value : boolean
<<predicate>> hasMainContainer
value : boolean

Fig. 7. Ontology Description of the Discovery Agent Domain.

Fig. 8 describes that Communication Ontology Description diagram describes the common specifications used for the communication between the two agents for ensuring an identical object is described by the both agents during a communication. The modeling diagram includes two elements: one is agent stereotype, which is marked with yellow notes and describes its own knowledge facilities of the agent; the other is Communication Stereotype, which is marked with white notes.

The agents communicate by using the FIPA-ACL (FIPA-Agent Communication Language), the parameter "ontology" in the FIPA-ACL statement can assign the "specific domain Ontology" during their communication. In Fig. 8, when Discovery Agent communicates with Template Translation Agent, they use Ontology "Template" during their communication, and the content of FIPA-ACL statement uses the vocabulary defined in Template Ontology. By using the same knowledge, therefore they can communicate efficiently each other.

Respect to the reasoning rules, and let's take communication between Discovery Agent and Template Translation Agent for example, the reason rule is as following:

(?agent yfc:request ?format) (?template yfc:hasFormat ?format) -> (?agent yfc:hasTemplate ?template)

This reasoning rules mean that if one agent requests a special template format, and this template just has the format they want, therefore, the other agents can own the same template. By following the reasoning rule, Template Translation Agent can transform the message he receives to the XML format, and sends the result to the Discovery Agent.

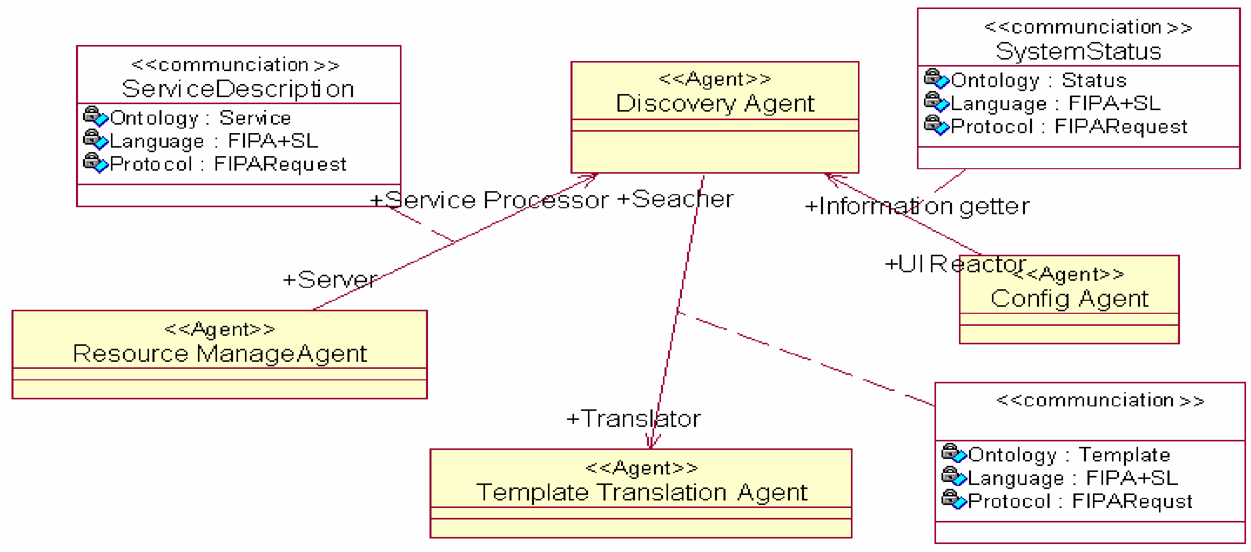


Fig. 8. Ontology Description Diagram of System Communication.

Role Description Phase. In the role description phase describes the role each agent plays in its life cycle and the communication status in the collaborative cooperation. The role description diagram uses a package to represent an agent; a category stands for the role that the agent plays; every role comprises several tasks. Therefore a role description diagram can precisely describe all roles and rules in the agent society as well as interactions between agents.

Protocol Description Phase. As we have seen in the Ontology Description phase and as specified by the FIPA architecture, a protocol has been used for each communication. By following FIPA's standard, hence we don't need to specify protocols by our own. We will discuss the protocols using in our study as follows:

Fig. 9 describes that the FIPA Request Interaction Protocol (IP) allows one agent to request another to perform some actions. The Participant processes the request and makes a decision whether to accept or refuse the request. If a refuse decision is made, then "refused" becomes true and the Participant communicates a refuse. Otherwise, "agreed" becomes true.

If the conditions indicate that an explicit agreement is required (that is, "notification necessary" is true), then the Participant communicates an "agree". The agree may be optional depending on circumstances, for example, if the requested action is very quick and can happen before a time specified in the reply-by parameter. Once the request has been agreed upon, then the Participant must communicate either:

- A failure if it fails in its attempt to fill the request,
- An inform-done if it successfully completes the request and only wishes to indicate that it is done, or,
- An inform-result if it wishes to indicate both that it is done and notify the initiator of the results.

Any interaction using this interaction protocol is identified by a globally unique, non-null conversation-id parameter, assigned by the Initiator. The agents involved in the interaction must tag all of its ACL messages with this conversation identifier. This enables each agent to manage its communication strategies and activities, for example, it allows an agent to identify individual conversations and to reason across historical records of conversations.

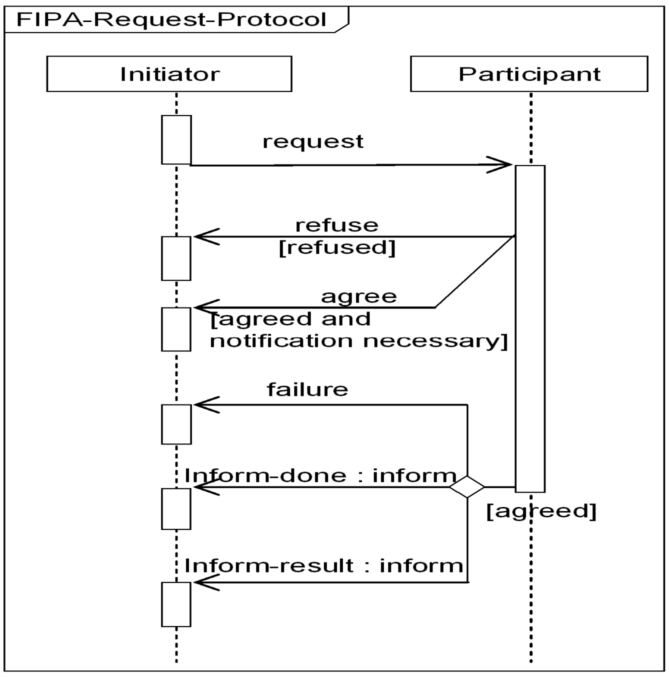


Fig. 9. FIPA Request Interaction Protocol.

4.2 Agent Implementation Model

There are two phases of the Agent Implementation Model namely Agent Structure Definition and Agent Behavior Description phase as defined as follows:

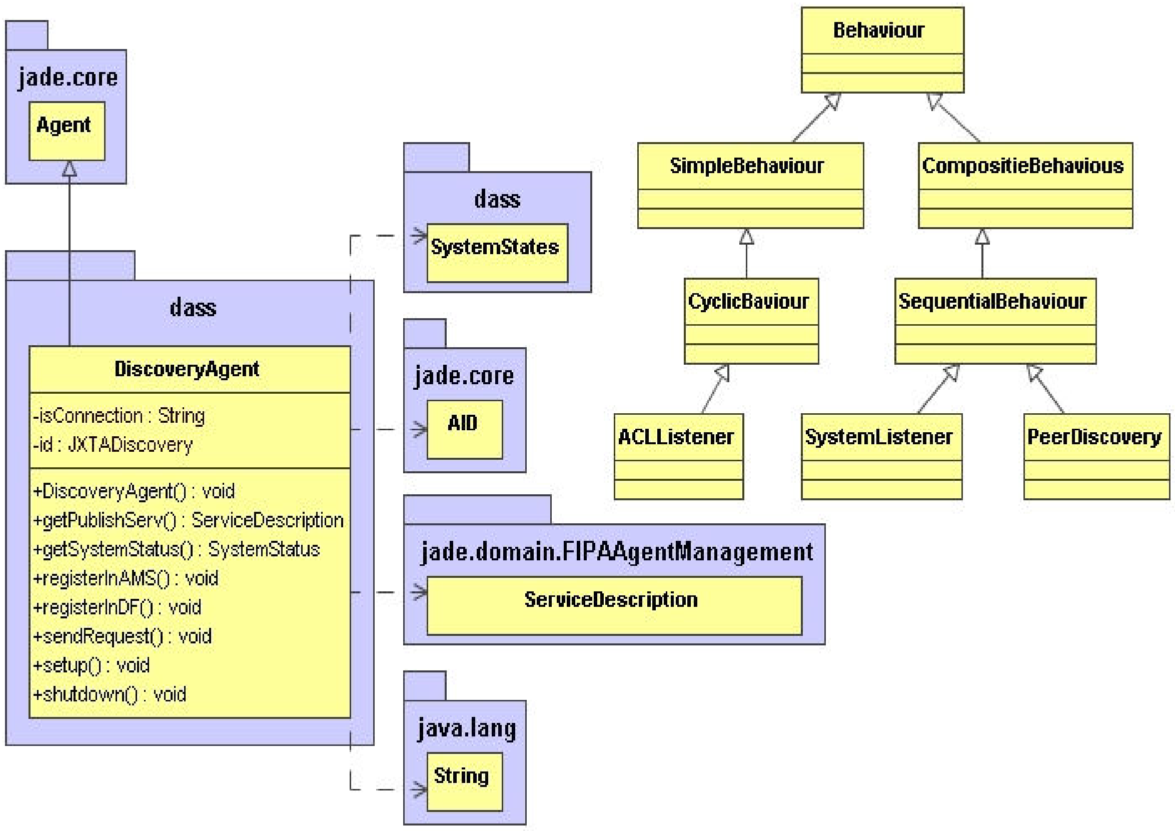


Fig. 10. Diagram of Single Agent Structure Definition.

Agent Structure Definition Phase. A single-agent view refers to descriptions of details of the internal structure, addressing each agent, including the arbitration and method. Fig. 10 is the structure definition diagram of the Discovery Agent. On the left of Fig. 10 shows the internal structure of the Discovery Agent, one class stands for one agent: the Discovery Agent comprises function formulas including setup, shutdown, registerInAMS, registerInDF, searchAP, getSystemStatus, etc. When the Discovery Agent is created, the system will register Agent Management System (AMS) by calling the registerInAMS() function, then the Discovery Agent initiates by calling setup() function. If the Discovery Agent needs to call other agent's service, he will call the registerInDF() function to register the Directory Facilitator (DF). Finally, if the Discovery Agent finish his job, the other agents will stop it by calling the shutdown() function. On the right part of Fig. 10 are facilities that the agent is capable of during the communication, they are respectively ACLListener and SystemListener & PeerDiscovery, being the inheritances of SimpleBehaviour and CompositionBehavious respectively.

Fig. 11 is the Definition of Multi-Agent Structure. After system initiation, Config Agent then query for service provider and proceed with system configure; then Resource Management Agent will access the current path of files and directory, with automatic release carried out for the file type specified by the user. Resource Management Agent will carry out tasks such as collecting names of file extensions. Every now and then, the Discovery Agent in the system will carry out search of remote agent platform. Discovery Agent will collect system information from Config Agent and Resource Management Agent before carrying out search actions; in case it is required to perform the another service discovery architecture, Config Agent will request Template Translating Agent to transform the format of the search request or translate the replied data into the format save in its own platform. Except the recording service descriptions of its own and of other agent platforms, the Recording Agent also keeps record of preferred services for quick access by the user. Invoking Agent is responsible for contracting and certifying maintain leases; it is also responsible for meeting with service access needs of the Config Agent and Recording Agent.

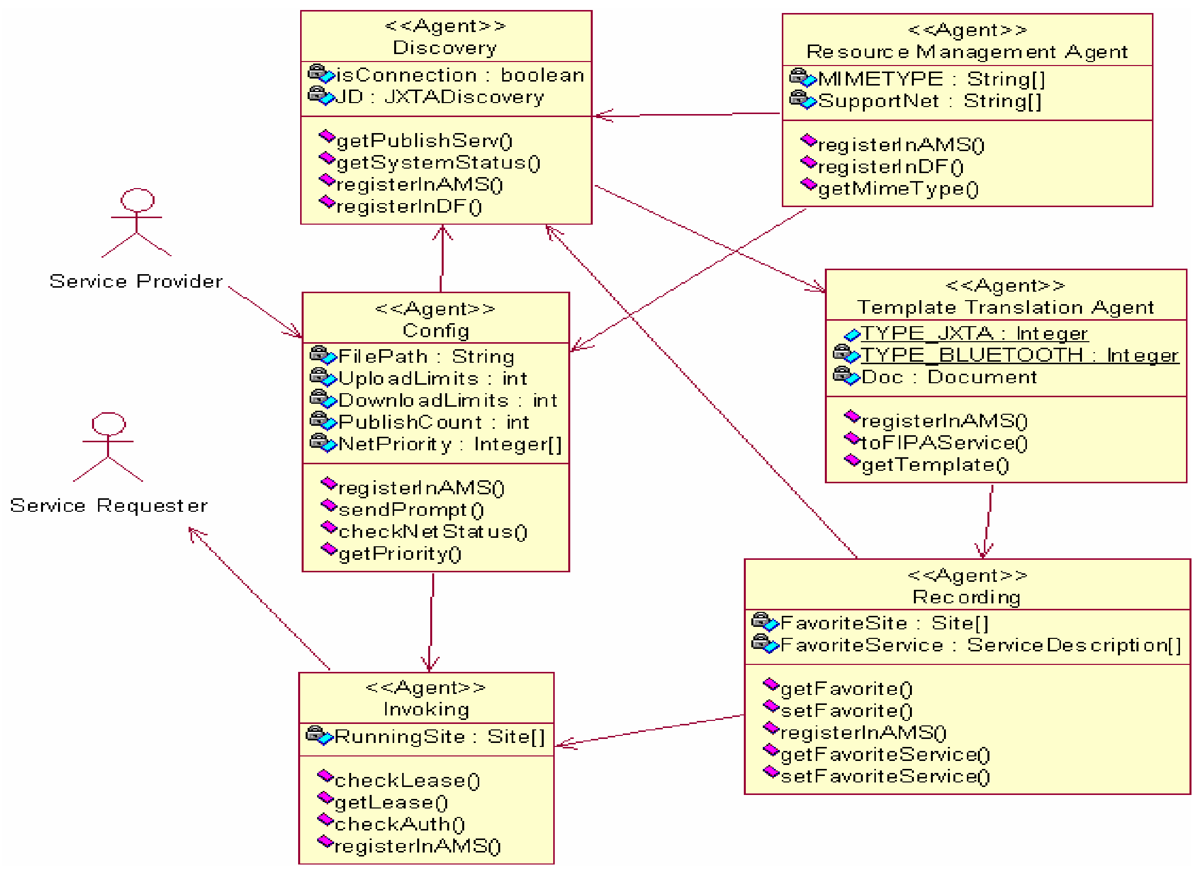


Fig. 11. Definition of Multi-Agent Structure.

Agent Behavior Description Phase. In the Agent Behavior Description phase, further describes role identification diagram, COD diagram and DOD diagram with sequence diagram as well as comprehends the internal calling method of the agents and the massage interchanges between agents. Fig. 12 is the agent behavior diagram with preference information provided by service request where the different water lines differentiate the different agents, and each channel's title uses the format of "Agent.Task". "Agent" is the name of the agent, and "Task" is the task of the agent. First of all, interface agent calls for internal method: newTask() passes service information to the consultant via agent communication language. After informing Discovery Agent about user preferences, of which critical information is passed on to the requester and Discovery Agent for performing respective actions. On completion of the task, method: done() is called to end the task.

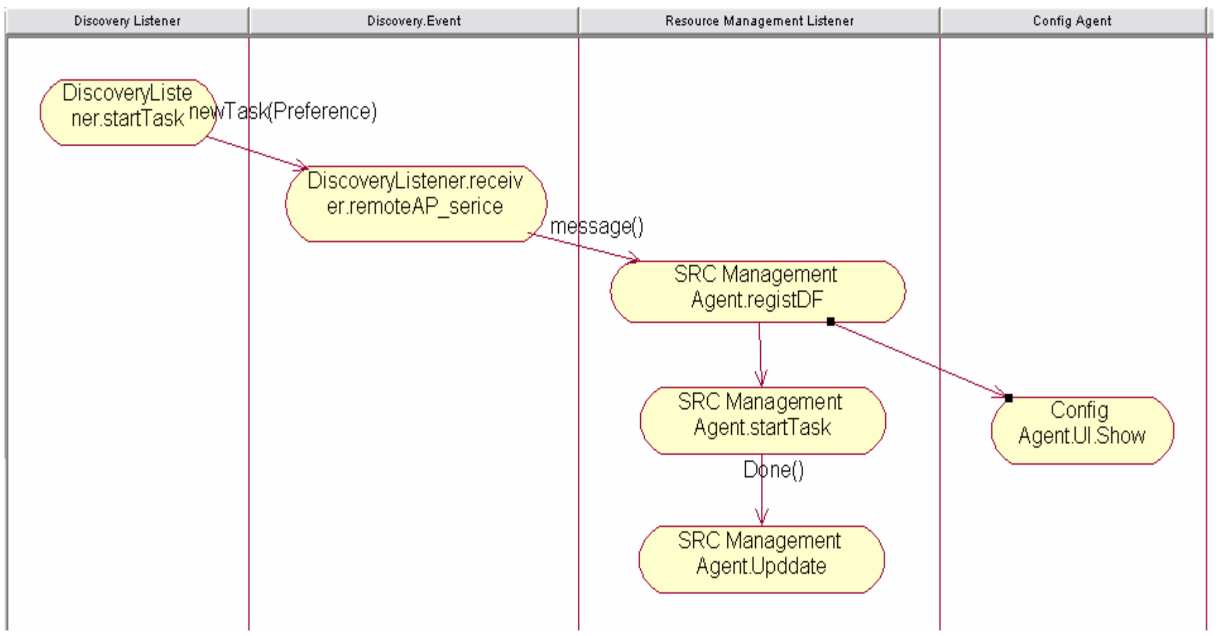
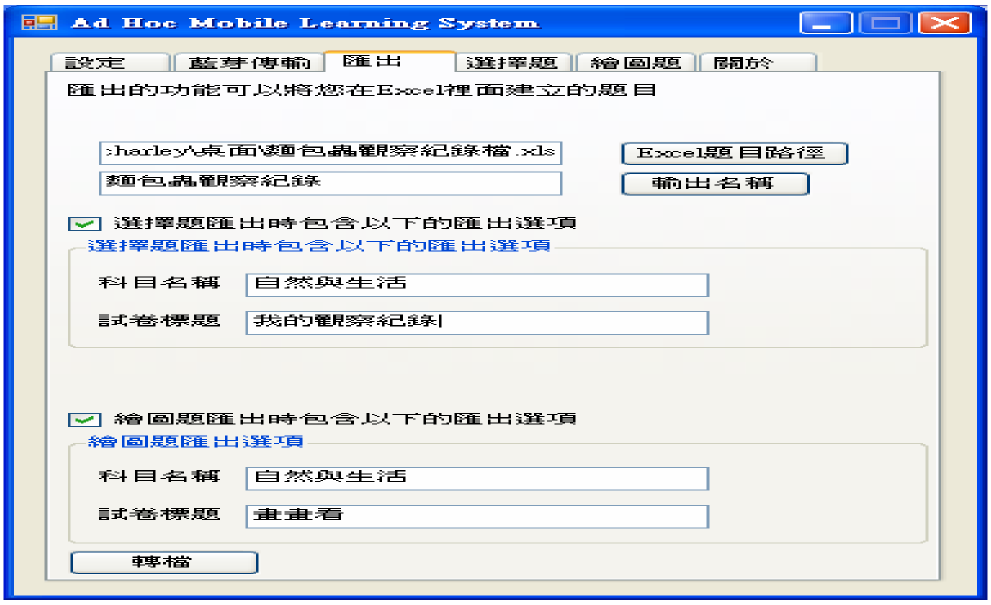

Discovery Listener
Discovery.Event
Resource Management Listener
Config Agent
DiscoveryListener.startTask
newTask(Preference)
DiscoveryListener.receiver.remoteAP_serice
message()
SRC Management Agent.registDF
SRC Management Agent.startTask
Done()
SRC Management Agent.Upddate
Config Agent.UI.Show

Fig. 12. Multi-agent Behavior Descriptions.

5 System Implementation

In the versatile modern teaching, teachers and students have more opportunities to perform field teachings outside the school; however, while doing so, traditionally students make notes with pen and paper which is inadequate to record the audio and colorful world in full. Therefore we designed an Ad-Hoc observing and recording system for the students to record and compile the learning in real-time in the Ad-Hoc environment. The teacher, via cooperation between the agents, is able to monitor and control the learning status of the students, with the capability to perform individual discussions with the student as well. The interaction between the students and the teachers is in real time, and the multiplicity between the students and the teachers is multiple to multiple.

The study takes example of observation of bread worms performed by the primary school students in the course of Nature and Life. The teacher, priors to field teaching, may compile in advance the pre-designed observation process and questions in Excel, and then convert and export the file, as shown in Fig. 13. In view that the screen of the mobile device is rather small and inconvenient for text input, the questions are mostly in the form of selective and graphic ones; the exporting format is XML, as shown in Fig. 14.



Ad Hoc Mobile Learning System

匯出的功能可以將您在Excel裡面建立的題目

:harley\桌面\麵包蟲觀察紀錄檔.xls

麵包蟲觀察紀錄

✓ 選擇題匯出時包含以下的匯出選項

選擇題匯出時包含以下的匯出選項

科目名稱: 自然與生活

試卷標題: 我的觀察紀錄

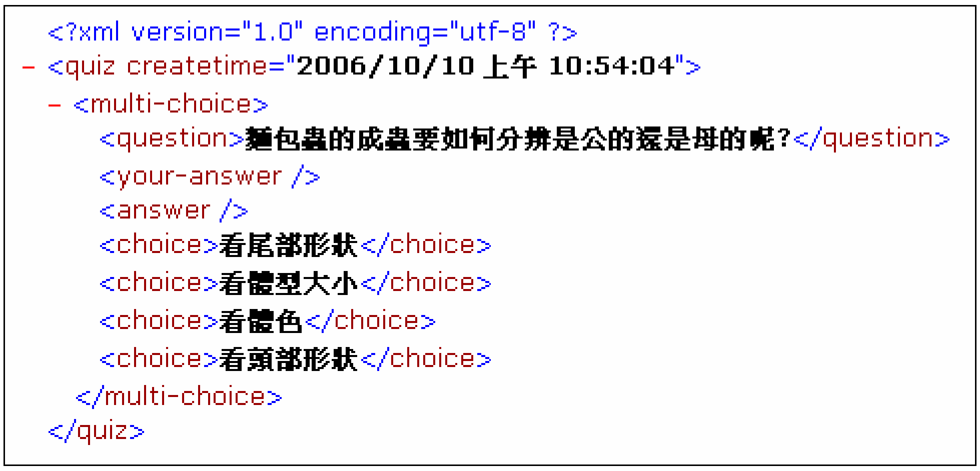
✓ 繪圖題匯出時包含以下的匯出選項

繪圖題匯出選項

科目名稱: 自然與生活

試卷標題: 畫畫看

Fig. 13. The Exporting Interface.



```
  <?xml version="1.0" encoding="utf-8" ?>
- <quiz createtime="2006/10/10 上午 10:54:04">
  - <multi-choice>
      <question>麵包蟲的成蟲要如何分辨是公的還是母的呢?</question>
      <your-answer />
      <answer />
      <choice>看尾部形狀</choice>
      <choice>看體型大小</choice>
      <choice>看體色</choice>
      <choice>看頭部形狀</choice>
    </multi-choice>
  </quiz>
```

Fig. 14. The Exported Question.

Before proceeding with observation activities of the course, the teacher may first explain precautions during the lesson activities, and then deploy the internal Bluetooth function of the system for file transfer service, so that the students use the observing and recording system of the PDA to search for the service deployed by the teacher, as shown in Fig. 15.

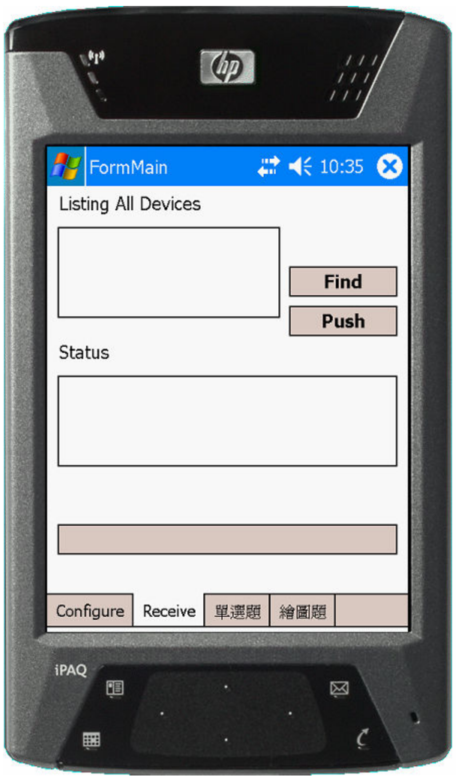


Fig. 15. Bluetooth Discovery Service.

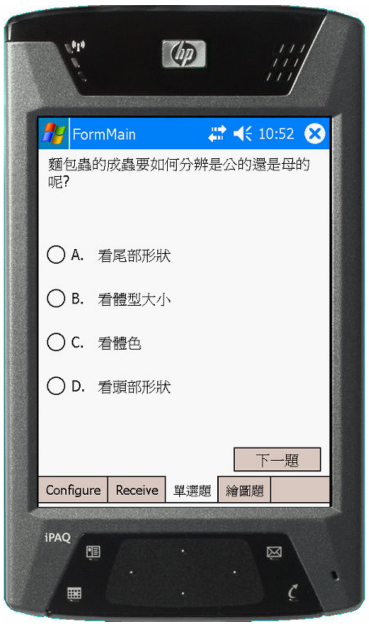
Now we discuss how the students know teacher's service. Teacher's devices will broadcast service advertisements, and student's devices will receive them. If students' devices receive the advertisement of a service it is providing, it will silently discard the message. If the received service is not local but it is stored in service cache, the devices will compare lifetimes of both services (stored and received). It will, definitely, store the one with higher lifetime. Finally, when students' devices receive an advertisement of a service which is not stored, it will directly extract the necessary information and store it. Fig. 16 shows the algorithm of Service Advertisements. [19]

```
if (received service is stored)
    if (stored service is local)
        Discard received message
    else
        if  (stored service bigger lifetime)
            Discard received message
        else
            Store received service
else
     received service
```

Fig. 16. The Algorithm of Service Advertisements.

File transfer can be performed when the service has been found; the default setting will send the question to My Documents under the root directory; observation can be carried out when the questions are received. Students can record observation result by answering two type questions. A select question is as shown in Fig. 17; students may answer the question by selecting the items observed.

The PDA is equipped with touch panel that allows the student to record the observed results with the touch pen, with different colors as desired, as that shown in Fig. 18.



麵包蟲的成蟲要如何分辨是公的還是母的呢?

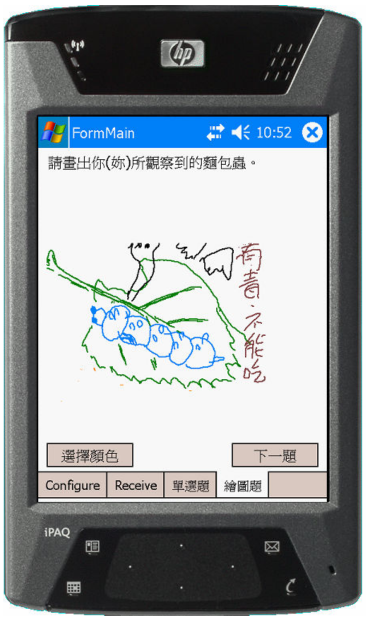
A. 看尾部形狀

B. 看體型大小

C. 看體色

D. 看頭部形狀

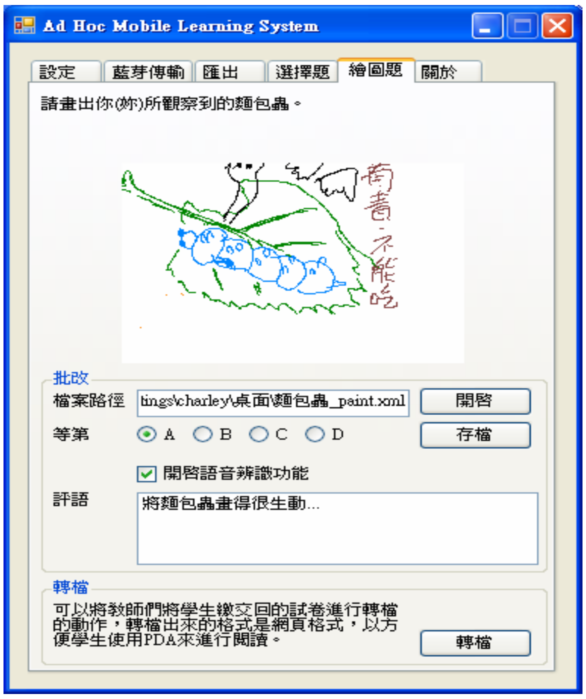
Fig. 17. A Multi-Choice Question.



請畫出你(妳)所觀察到的麵包蟲。

Fig. 18. A Graphic Question.

When the students complete the observation activities, file transfer service may be performed; the teacher may use Bluetooth to perform service discovery and service use. As indicated in Fig. 19. When the teacher receives the file transfer service, correction and rating can be carried out addressing selection and graphic questions; since each selection question has a fixed answer, and it only requires importing the answers made by the students. As for the Graphic questions, the teacher can also make corrections by using this system while received the graphs, as shown in the figure, and to give rating or comment while browsing them; it also allows to check-select the speech recognition function to convert oral (voice) ratings or comments into texts for recording the comment.



請畫出你(妳)所觀察到的麵包蟲。

批改

檔案路徑 tings\charley\桌面\麵包蟲_paint.xml

等第 [✓ A] B C D

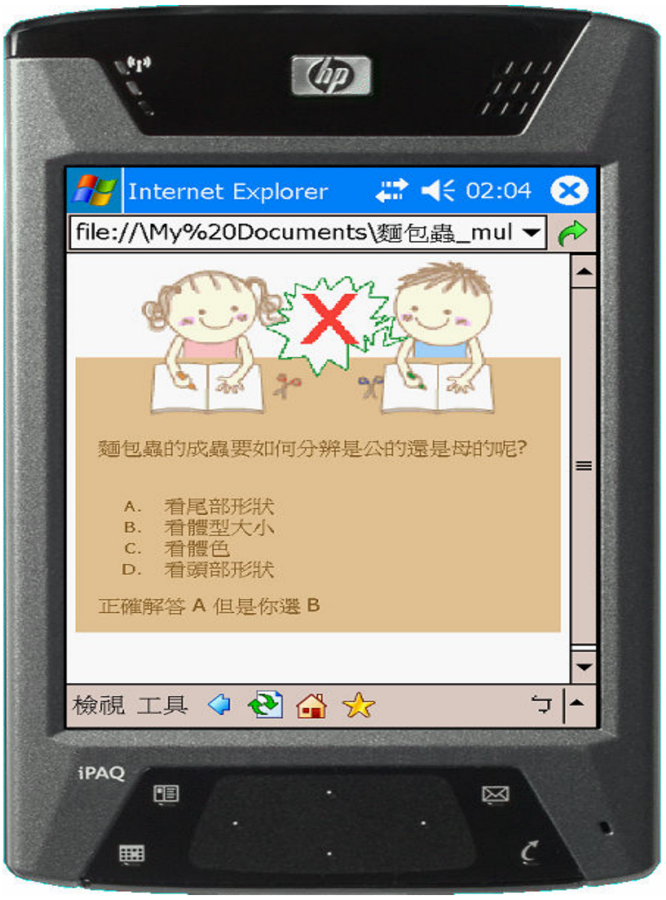
[✓] 開啓語音辨識功能

評語 將麵包蟲畫得很生動...

轉檔

可以將教師們將學生繳交回的試卷進行轉檔的動作，轉檔出來的格式是網頁格式，以方便學生使用PDA來進行閱讀。

Fig. 19. Interface for Correcting Graphic Questions.



麵包蟲的成蟲要如何分辨是公的還是母的呢?

A. 看尾部形狀
B. 看體型大小
C. 看體色
D. 看頭部形狀

正確解答 A 但是你選 B

Fig. 20. Browse Screen on PDA of a Multi-choice Question after File Conversion.

Besides that, after the teacher completes correcting the answers, file conversion function may be used to transform XML files into HTML format and return them to the student, facilitating the student to directly browse the observed records using the PDA, or place the observed records into a Blog site, as shown in Fig. 20.

The service of the system is divided in the observing and recording system and the correcting system. When these students execute the observation activities, the observing and recording system will record the records in the temporary file. If the service is terminated, the students just need restart their PDA, the system will read the temporary file, and the students could continue to execute the observation activities. When the students complete the observation activities, the file transfer service may be requested and using transaction as the transaction of database system to handle this error situation. If the file transfer service is terminated during the upload, the system will restore the record to the previous state. On the other words, the transaction must be either entirely completed or aborted; no intermediate states are accepted. The observing and recording system is based on the multi-agent system, so the life cycle of the service and agent is the same. In this study, the six agents don't in the active state. They will be in the pause state to release these system resources if and only if the system doesn't need them to work in the near future.

6 Conclusions and Future Prospects

The "Multi-agent System－Distributed Agent-based Service Sharing" proposed by our research deploys various resources by using agent technology as the single window, to simplify service sharing on mobile devices and to strengthen management of resource sharing. In the service query aspect, implementation of abstract discovery specifications established by FIPA [20] strengthens their compatibility with other systems; and through agent communication language and the ontology that specifies service sharing, communications between agents on heterogeneous platforms are performed to achieve service finding and request; lastly, example is made on using Ad-Hoc observing and recording system to develop the system.

The DASS structure proposed by the research is capable of resolving service sharing problems under the decentralized environment and proposing substantial contributions as depicted as follows:

- ▷ Communication among the heterogeneous agents: according to FIPA standard, the definition of agent communication language (of the standard) is capable of simplifying communication among the heterogeneous agents; no matter what program language or structure is used for implementation, agents will be able to understand each other as long as FIPA standard is observed by both parties.
- ▷ Setting up agent system on mobile devices: use JADE (Java Agent DEvelopment Framework)-LEAP to build light-weight multi-agent on the mobile device; operation platforms include PDAs and Smart Phones. By adding service sharing negotiation mechanism into the communication among the agents, whether the supply/demand relationship between devices is met can be deduced via this functionality, so as to ensure correctness and quality of the service requested by the user.
- ▷ To succeed to the abstract FIPA service registration and query specification. Doing so can increase system compatibility. And for expanding system discovery, Template Translating Agent model is proposed to perform Template Translation between service discovery architectures. Lastly, use the existing service sharing architecture – JXTA implementation to enable inter-agent services to communicate across different agent platforms and devices.

Mobile device and agent program design are the most popular research subjects. Our study combines these subjects to propose the DASS architecture to solve the problems encountered when performing service sharing under Ad-Hoc environments; the study also implements information sharing systems with the same architecture. Directions for succeeding researches based on this study are as follows:

- ▷ Currently discovery of the agent platform is mainly JXTA-based, more distributed service architectures that support mobile devices such as UPnP and Bluetooth can be included to expand discovery effectiveness of the system.
- ▷ This study focuses on agent-oriented programming. Only fundamental functions are provided for service sharing under the distributed environments. Further extension and development can be addressed on different requirements in the future.
- ▷ Further reinforcement can be made on the deduction and decision making capabilities of the agent itself so that agents are more intelligent. With reinforced autonomy in the agents, user operation steps can be reduced.
- ▷ Architecture used in this study may be combined with other component-based service architectures like web-services, enabling service providers to be more diversified.

7 Acknowledgement

This work described in this paper has been partially supported by the Da-Yeh University under the grant ORD-9416.

References

[1] S. Berger, S. McFaddin, and C. Binding, "Towards Pluggable Discovery Frameworks for Mobile and Pervasive Applications," in *Proceedings of 2004 IEEE International Conference*, pp.308-319, 2004.

[2] M. Panti, L. Penserini, L. Spalazzi and S. Valenti, "A FIPA Compliant Agent Platform for Federated Information Systems," *International Journal of Computer & Information Science*, May 2000.

[3] B.K. Langley, M. Paolucci, and K. Sycara, "Discovery of Infrastructure in Multi-Agent Systems," in *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, Australia, pp.1046-1047, July 2003.

[4] C.R. Dunne, "Using Mobile Agents for Network Resource Discovery in Peer-to-Peer Networks," *ACM SIGecom Exchanges*, Vol. 2, No. 3, pp.1-9, 2001.

[5] M. Storey, G. Blair, and A. Friday, "MARE: Resource Discovery and Configuration in Ad Hoc Networks," *Mobile Networks and Applications*, Vol. 7, No. 5, pp. 377-387, Oct 2002.

[6] L. Chunlin, and L. Layuan, "Combine Concept of Agent and Service to Build Distributed Object-Oriented System," *Future Generation Computer Systems*, Vol. 19, No. 2, pp. 161-171, Feb. 2003.

[7] C.-F. Chiasserini and V. Srinivasan, "Quality of Service in Ad Hoc and Sensor Networks," *Performance Evaluation*, Vol.64, No.5, pp. 377-378, June 2007.

[8] C.-S. Hsu, Y.-C. Tseng and J.-P. Sheu, "An Efficient Reliable Broadcasting Protocol for Wireless Mobile Ad Hoc Networks," *Ad Hoc Networks*, Vol.5, No.3, pp. 299-312, April 2007.

[9] M. Wooldridge, N.R. Jennings, and D. Kinny, "The Gaia Methodology for Agent-Oriented Analysis and Design," *Autonomous Agents and Multi-Agent Systems*, Vol. 3, No. 3, pp. 285-312, 2000.

[10] P. Charlton, E. Mamdani, and R. Cattoni, "Evaluating the FIPA Standards and Its Role in Achieving Cooperation in Multi-Agent Systems," in *Proceedings of the 33rd Hawaii International Conference on System Sciences*, pp.8034-8041, 2000.

[11] G. Booch, J. Rumbaugh, and I. Jacobson, *The Unified Modeling Language User Guide*, Addison Wesley, 1999.

[12] P. Vrba, and V. Hrdonka, "Material Handling Problem: FIPA Compliant Agent Implementation," in *Proceedings of the 12th International Workshop on Database and Expert Systems Applications*, pp.635-639, 2001.

[13] K. Jun, L. Boloni, K. Palacz, and D.C. Marinescu, "Agent-Based Resource Discovery," in *Proceedings of the 9th Heterogeneous Computing Workshop*, pp.43-49, 2000.

[14] O. Ratsimor, D. Chakraborty, and A. Joshi, "Service Discovery in Agent-Based Pervasive Computing Environments," *Mobile Networks and Applications*, Vol. 9, No.6, pp.679-692, 2004.

[15] M. Berger, M. Bouzid, and M. Buckland, "An Approach to Agent-Based Service Composition and Its Application to Mobile Business Processes," *IEEE Transactions on Mobile Computing*, Vol 2, No 3, pp. 197-206, July 2003.

[16] H. Tian, and H. Shen, "Mobile Agents Based Topology Discovery Algorithms and Modeling," in *Proceedings of 7th International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN'04)*, pp.502-507, May 2004.

[17] AUML, The FIPA Agent UML, *http://www.auml.org*, 2004.

[18] M. Barbeau, "Service Discovery in a Mobile Agent API Using SLP," in *Proceedings of the Global Telecommunications Conference*, pp.391-395, 1999.

[19] Y. Yang, H. Hassanein, A. Mawji, "Efficient Service Discovery for Wireless Mobile Ad Hoc Networks," *IEEE International Conference on Computer Systems and Applications*, pp. 571-578, March 2006.

[20] FIPA, Services Work Plan, Foundation for Intelligent Physical Agents, *http://www.fipa.org/docs/wps/f-wp-00019/f-wp-00019A.html*, 2003.

[21] G. Hattori, S. Nishiyama, and C. Ono, "Making Java-Enabled Mobile Phone as Ubiquitous Terminal by Lightweight FIPA Compliant Agent Platform," in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, pp.553-561, 2003.