Enterprise Information Systems, Vol. 00, No. 00, Month 200x, 1–17

Flood Decision Support System on Agent Grid:Method and Implementation

Jiewen Luo*†‡ Lida Xu §† Zhongzhi Shi † Jean-Paul Jamont ¶ and Li Zeng †‡

† Institute of Computing Technology,Chinese Academy of Sciences,Beijing,China
‡ Graduate University of Chinese Academy of Sciences,Beijing,China
§ Old Dominion University, USA
¶ Laboratoire de Conception et d'Intégration de Systèmes, LCIS/INPG, 26000 Valence, France

(v1.1 released June 2006)

In this paper, we introduce the concept and architecture of agent grid. Agent grid is an intelligent platform that enables the independent operating entities (agents) to interact with one another to form dynamic services on the Grid. Under this view, we built an agent grid platform named AGrIP that includes four layers and several useful toolkits.With the platform support, we implemented the flood decision support system which combines the wireless sensor network for data acquisition and software agent technology for legacy system integration. Additionally, we developed a toolkit for programmers to visually develop software agents which makes the development process easier.Besides, the MWAC model proposed is for sensor network to save power which can transit the information for long distance.This system is now applied as a module in the City Emergency Interact Project.

Keywords: Agent,Application Integration,Data Mining,Case studies,Data Integration,DAI,Grid,Legacy Systems

1 Introduction

China has been frequently hit by big floods and suffered from flood disasters. The critical issue is that about 8% of the middle and down stream of the seven major rivers are prone to floods, where is inhabited 50% of total population and contributes over 2/3 of total agricultural and industrial product value . When there is a risk of flooding, decision makers have to decide the most appropriate actions to take: evacuation of the population, reinforcement of dikes, intentional breaking of dikes, etc. This motivates us to develop an effective flood decision support system that can help decision-makers take scientific decisions to limit flood damage.

In general,decision support system is a comprehensive system that often includes data acquisition,information transmission,data mining,knowledge reasoning and system integration etc. In this paper, we mainly discuss how to efficiently use wireless sensors for data acquisition and agent grid platform for system integration since these two issues are critical for whole system implementation. Someone may argue that why need wireless sensors for data acquisition in the flood decision support system. The answer is, in some cases, especially for aggressive environment applications, data acquisition nodes cannot be interconnected through classical field buses where wireless technology is superior than traditional wire transmission not only in economic side but also in deployment side.

In this paper, we first propose the agent grid concept and architecture since the flood decision support system built on it. The agent grid methods aim at decreasing the complexity of system design by a decentralized analysis. The distributed and open nature of entire decision support system means that the agent grid approach is an adapted answer. Another advantage of this approach is software agents are good choice for legacy systems integration.

The rest paper is organized as follows. Section 2 introduces the research background which includes the grid computing and agent grid. Section 3 introduces agent grid platform and its toolkits. Section 4

*Corresponding author. Email: luojw@ics.ict.ac.cn

Enterprise Information Systems
ISSN 1751-7575 print / ISSN 1751-7583 online © 200x Taylor & Francis
http://www.tandf.co.uk/journals
DOI: 10.1080/1751757YYxxxxxxxx

introduces the agent service interface.Section 5 illustrates the decision process of flood decision support system and sensor network for data acquisition. Section 6 summarizes the paper and gives a conclusion.

2 Research Background

2.1 *Grid Computing*

A Grid is a geographically distributed computation platform composed of a set of heterogeneous machines that users can access via a single interface. Grids therefore provide common resource-access technology and operational services across widely distributed virtual organizations composed of institutions or individuals that share resources.

Grid computing differs from conventional distributed computing in that it focuses on large-scale resource sharing, offers innovative applications, and, in some cases, is geared toward high-performance systems. Although originally intended for advanced science and engineering applications, Grid computing has emerged as a paradigm for coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations in industry and business. Thus, today Grids can be used as effective infrastructures for distributed high-performance computing and data processing (Ian Foster *et al.* 2004).

Grid applications include:

- Intensive simulations on remote supercomputers,
- Cooperative visualization of very large scientific data sets,
- Distributed processing for computationally demanding data analysis,
- Coupling of scientific instruments with remote computers and data archives

Although originally intended for advanced science and engineering applications, grid computing has emerged as a paradigm for coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations in industry and business (see GridForum Website). In the last five years, toolkits and software environments for implementing grid applications have become available. These include Legion (see Legion Website), Condor (see Condor Website), and Unicore (see Unicore Website). In particular, Foster and Kesselman's Globus Toolkit (see Globus Website) is the most widely used middleware in scientific and data-intensive grid applications, and is becoming a de facto standard for implementing grid systems. The toolkit addresses security, information discovery, resource and data management, communication, fault detection, and portability issues. It does so through mechanisms, composed as bags of services that execute operations in grid applications. Today, Globus and the other grid tools are used in many projects worldwide.

2.2 *Agent Grid*

As Ian Foster says, the Grid and agent communities both develop concepts and mechanisms for open distributed systems, albeit from different perspectives. The Grid community has historically focused on "brawn": infrastructure, tools, and applications for reliable and secure resource sharing within dynamic and geographically distributed virtual organizations. In contrast, the agent community has focused on "brain": autonomous problem solvers that can act flexibly in uncertain and dynamic environments (Ian Foster *et al.* 2004). While Grid infrastructure has focused on such things as the means for discovering and monitoring dynamic services, managing faults and failures, creating and managing service level agreements, creating and enforcing dynamic policy.To date, only limited progress has been made on creating the higher level reactive behaviors that would enable truly dynamic formation of service composition. Hence, it is necessary to build an intelligent platform that enables the independent operating entities (agents) to interact with one another with partial knowledge and emerge a robust desirable behavior to form dynamic services on the Grid.

Specifically, we try to apply the Grid infrastructure to integrate the distributed data resource on different sites, as figure 1 shows. It connects heterogeneous resource and provides a uniform service interface for agents to access and process data. Upper the infrastructure,the middle-ware provides a run-time environ-

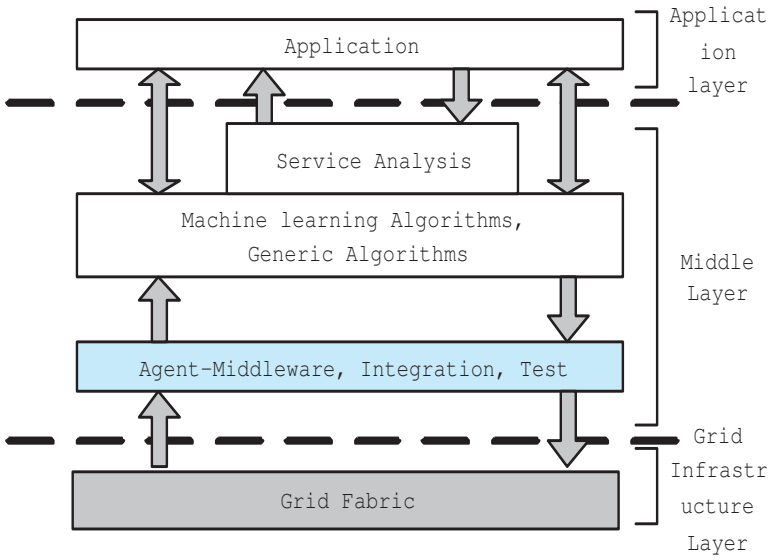

Figure 1. The Agent Grid Hierarchical Structure

ment for agents where multi-agents could communicate and interact to exchange the partial knowledge. We call this hierarchical system Agent Grid, which is a parallel and distributed software that integrates flexible agent services (e.g. data mining service), agents' run-time environment and development tool-set on the grid infrastructure. In the Agent Grid, several useful tools are integrated with generic and data grid mechanisms and services.

3 Agent Grid Platform

3.1 *Platform Architecture*

From conceptual point of view, Jeffery proposed a three-layer model for the Grid infrastructure in a strategy document (Keith Jeffry 2000). In this model the computing infrastructure consists of three conceptual layers: data/computation, information and knowledge. The data/computation layer is a lower layer, which primarily concerned with computational and data resources. It is characterized as being able to deal with large-scale data, providing fast network and diverse resources as a single meta-computer. This layer builds on the physical grid fabric concerned with a distributed collection of files, databases, computers and devices.

The information layer means the information is represented, stored, accessed, shared and maintained. Here information is understood as data equipped with mean-ing. Uniform access to information sources relies on metadata to describe information and integrate heterogeneous sources. There are following functions in the information layer:

- Connecting together the major information sources
- Interfaces: Uniform access to heterogeneous distributed information
- Sophisticated statistical analysis/reduction techniques for floating point numbers, textual information and multimedia information
- Special facilities for visualization and virtual reality

The knowledge layer is concerned with the way that knowledge is acquired, used, retrieved and maintained. Here knowledge is understood as information applied to solve a problem and achieve a goal or decision-making. We can see that following functions will be contained in the knowledge layer:

- Acquire knowledge through KDD (knowledge discovery in database) technology of which a well-known component is data mining;
- Apply knowledge to support intelligent assists to decision makers;

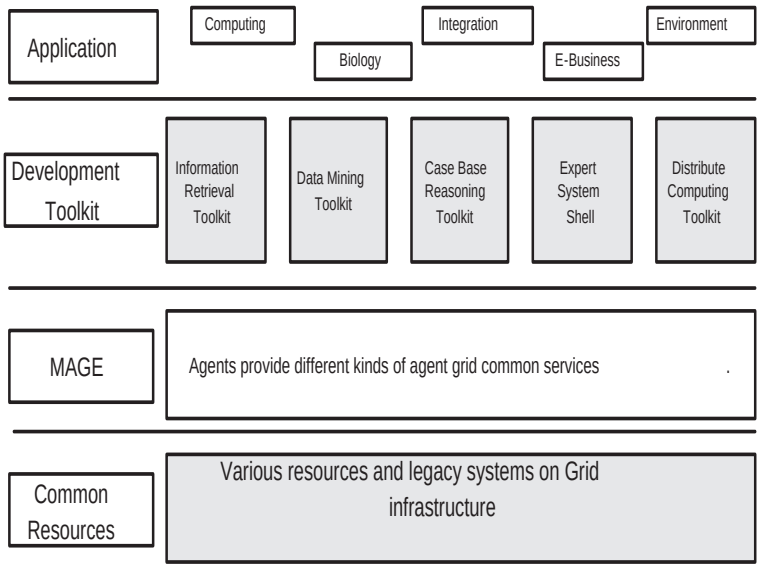

Figure 2. Architecture of AGrIP Platform

- Provide interpretational semantics on the information.

Different with the method above,we proposed a four-layer model for AGrIP (Jiewen Luo, Zhongzhi Shi *et al.* 2006) platform from the implementation point of view, as illustrated in figure 2:

- Common resources: consisting of various resources distributed in Grid environment, such as workstation, personal computer, computer cluster, storage equipment, databases or datasets, or others, which run on Unix, NT and other operating systems.
- Agent environment: it is the kernel of Grid computing which is responsible to resources location and allocation, authentication, unified information access, communication, task assignment, agent library and others.
- Developing toolkit: providing development environment, containing agent creation, information retrieval, distributed data mining, to let users effectively use grid resources.
- Application service: organizing certain agents automatically for specific purpose application, such as e-science, e-business, decision support and bio-information.

3.2 *Multi Agent Environment MAGE*

MAGE is located at the second layer,which is a multi-agent environment with a collection of tools supporting the entire process of agent-oriented software engineering and programming. It is designed to facilitate the rapid design and development of new multi-agent applications by abstracting into a toolkit the common principles and components underlying many multi-agent systems. The idea was to create a relatively general and customizable toolkit that could be used by software users with only basic competence in agent technology to analyze, design, implement and deploy multi-agent systems (Zhongzhi Shi *et al.* 2004).

Figure 3 illustrates the architecture of MAGE. It mainly consists of four subsystems: Agent Management System, Directory Facilitator, Agent, and Message Transport System.

- **Agent Management System** is a mandatory component of MAGE. It maintains a directory of AIDs (Agent Identifiers), which contain transport addresses for agents registered in MAGE and offer white pages services to other agents.
- **Directory Facilitator** (DF) is an indispensable component of MAGE. It provides yellow page services to other agents. Yellow page service allows agents to publish one or more services they provide so that other agents can find and successively exploit them. Agents may register their services with the DF or query the DF to find out which services are offered by other agents.

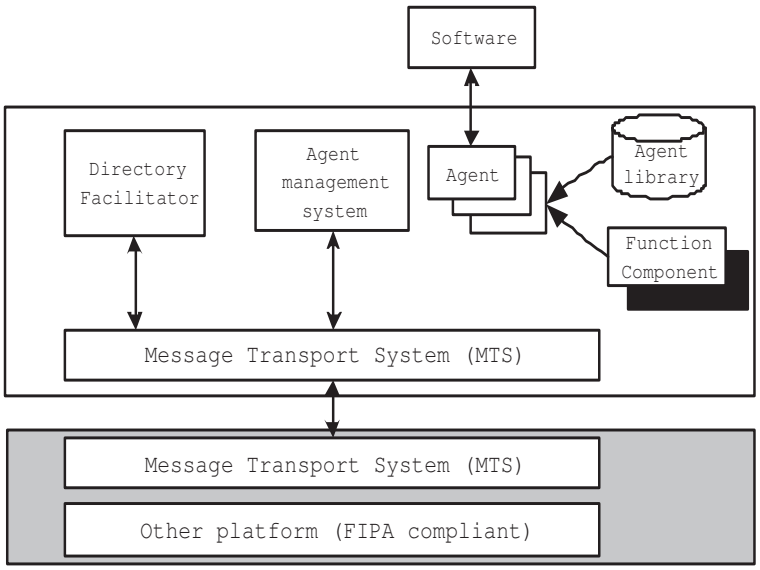

Figure 3. The Architecture of MAGE

Table 1. Legacy component descriptions

<Legacy component>::=<Entity Component> | <Link Component>
<Entity Component>::=<Entity Name><Entity Type><Entity Description><Entity Resource><Operate Procedure><Operate Environment>
<Entity Type>::= Procedure | Function | Java Bean | DLL
<Entity Resource>::=Variable | Class | <Entity Component>
< Link Component >::=< Link Name >< Link Type >< Link Description >
<Connectors><Precondition><Post conditio>
<Link Type>::=Pipeline |Procedure Call | RPC | Message Passing |

- **Message Transport Service** (MTS) is the default communication approach between agents on different FIPA-Compliant agent platforms. It uses FIPA ACL [1] as the standard communication language.
- **Agent** is the fundamental actor in MAGE, which combines one or more service capabilities into a unified and integrated execution model that may include access to external software, human users and communications facilities.
- **Software** in figure 3, is not a internal part of MAGE.It represents all non-agent, external components accessible to an agent. For example, Agents may add new services or acquire new communication/negotiation protocols, etc.

3.3 *Agent Development Tool VAStudio*

Although having grid platform AGrIP and multi agent environment MAGE, it is useless if we have not corresponding agent development environment. Hence, we built a visual agent studio named VAStudio to make up this gap. VAStudio is designed to provide the agent software developers with an integrated environment for quickly constructing intelligent agents and agent-based software (Jiewen Luo *et al.* 2005).

In flood decision support system, we need multi-agent for information collection and system integration. How to construct these agents become an important issue.

From software engineering perspective, the reusable classes are main components of new software and

[1] http://www.fipa.org/specs/fipa00061/

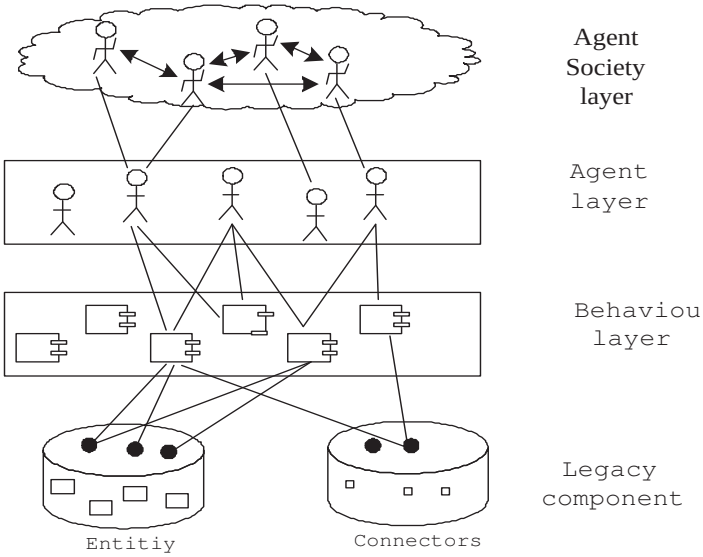

Figure 4. Hierarchical Structure of VAStudio

programmers prefer to integrate the existing modules for rapid development. Hence, VAStudio adopts a hierarchical design pattern in development phase. Figure 4 shows VAStudio's structure, which is composed with four different layers: the legacy component layer, behavior layer, agent layer and the society layer. The lower layers supply the fundamental materials for upper ones to construct more abstract and complex instances. Legacy component layer includes connectors and entity components. The entity components are independent existence and can be applied directly. For example: procedures, functions, DLL and java Beans etc. Connectors are used for guaranteeing connection, which includes pipe, cache, and message transmission etc. We can describe them as table 1.

Behaviors are the basic components to build agents. In VAStudio, every behavior has at least one action. We provide the definition of the base Behavior Class and every child Behavior is generated as a subclass of it, which implements common interfaces for subclasses to inherit.

Although behaviors are components having concrete actions, they cannot run independently. Behaviors have to be added into Agents.When a Agent adds a behavior, it can execute the actions implemented in the Behavior Class.

The highest layer of the structure is agent society that provides a platform for simulating multi-agent interaction in distributed data mining. In this layer, we can monitor the message transmission, collaboration process and move route of mobile agents.

3.4 *Information Retrieval Tool GHunt*

Information retrieval is very important for the whole system. We have developed an intelligent information retrieval toolkit Ghunt [1]to facilitate this work. From the function aspect, it is an all-sided solution for information retrieval on the Internet. When it runs on the internet, a parallel, distributed and configurable Spider is used for information gather; a multi-hierarchy document classification approach combining the information gain initially processes gathered web documents; a swarm intelligence based document clustering method is used for information organization; a concept-based retrieval interface is applied for user interactive retrieval. Since huge original information is crucial in a decision support system, it was integrated as a module of the AGrIP platform, which provides a powerful information retrieval function (see fig.5)(Zhongzhi Shi *et al.* 2003).

[1]http://www.intsci.ac.cn/GHuntWeb/

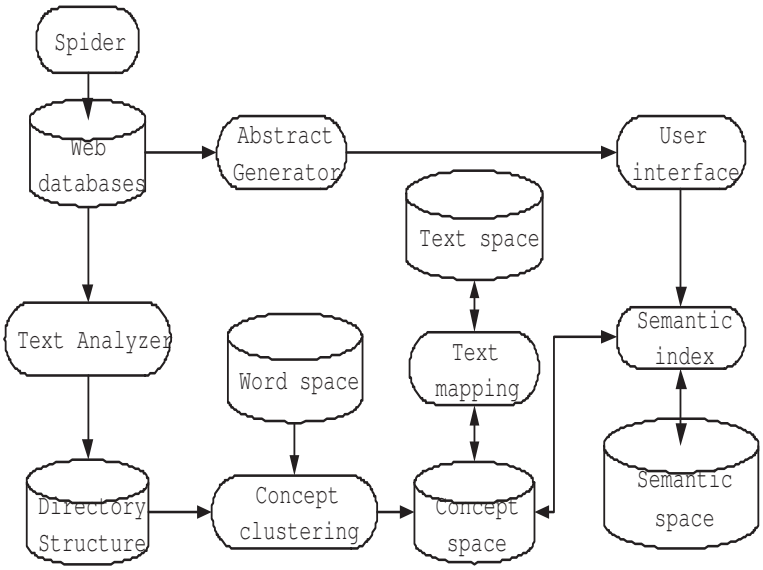

Figure 5. Information Retrieval Toolkit GHunt Work Flow

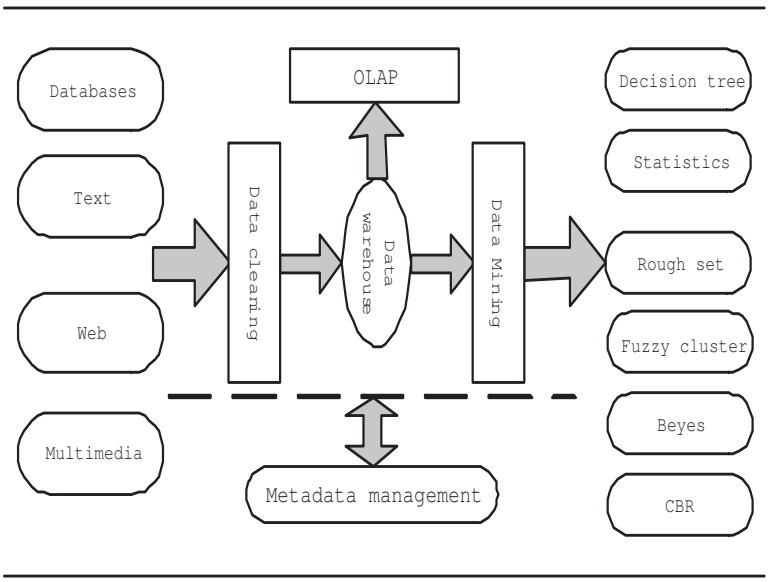

Figure 6. The work flow of MSMiner

3.5 *Data Mining Tool MSMiner*

MSMiner is a generic multi-strategy data-mining tool. Figure 6 demonstrates the data process of MSMiner. The functions of the system include database access, data modelling, data preprocessing, data mining, and data visualization. In addition to this, emphasis has been put on the extensibility characteristics of the system. As multi-strategy data mining software, MSMiner not only provides convenient tools to develop new data mining algorithms, but also includes many build-in algorithms such as SOM and C4.5. MSMiner has an open interface for adding data preprocessing function and can access a variety of databases such as SQL Server, Oracle, and Informix. It can collect information from web, text, database and multimedia database. After data cleaning, they are stored in the data warehouse for data mining. Since it is a multi-strategy data mining tools, we integrated machine learning, rough set, CBR and Statistics techniques in the algorithms library which provides a strong data mining for decision support (Zhongzhi Shi *et al.* 2005). Figure 7 shows the MSMiner Frame.

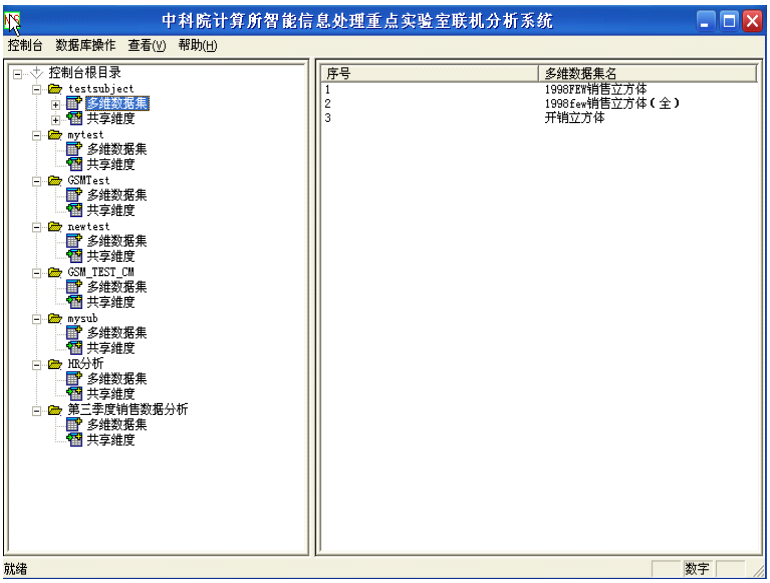

Figure 7. The Sceenshot of MSMiner

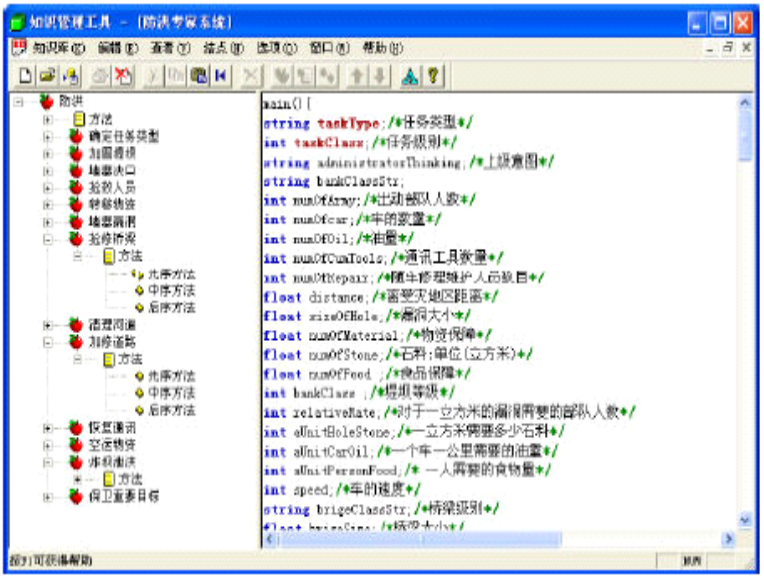

Figure 8. Knowledge base editor in OKPS

3.6 *Expert System OKPS*

Object-oriented Knowledge Processing System (OKPS [1])is an expert system tool. In OKPS, object-oriented style is used for the knowledge representation. This method not only has the features included in an object-oriented mechanism, such as encapsulation, inheritance and message processing, but also adopt and expand some features in Frame Theory, and Semantic Network.

A database-based knowledge storage technology is also implemented . Based on those aspects, the tool explores how to implement a visualized knowledge acquisition and management tool and construct an expert system efficiently, by a mean of separated knowledge base and inference engine.

In order to provide a powerful inference mechanism, as well as the maximum flexibility and convenience, the system also proposed a high-level language, the Inference Control Language (ICL), which can be used to express experts knowledge and rules, and control the inference process.

[1]http://www.intsci.ac.cn/en/research/okps.html

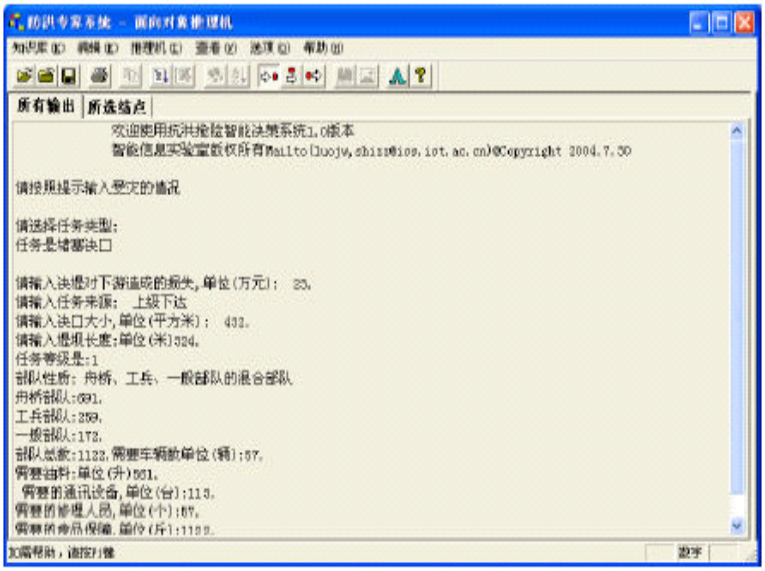

Figure 9. The reasoning engine of OKPS

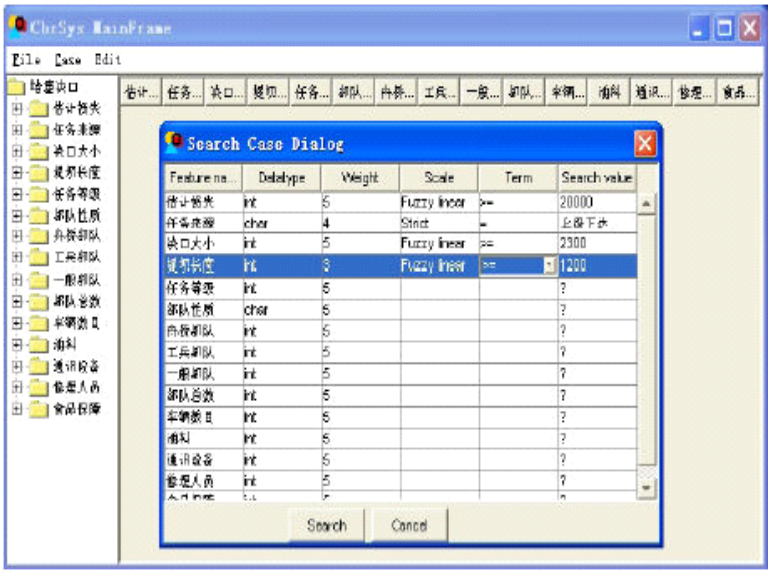

Figure 10. Case base editor: including insert,delete,revise the cases

The figure 8 illustrates knowledge base editor in OKPS.With the help of ICL and visual editor, it is easy to convert the expert knowledge to rule program. After building the rules with the editor, we can use the OKPS reasoning engine for rule reasoning (see fig.9).

3.7 *Case Base Reasoning*

In order to quicken the response,we integrate the case base reasoning (CBR) technique in the flood decision support system.We first adopt the CBR when a flood emergency happens and then using the rule engine-OKPS.

In case-based reasoning (CBR) systems expertise is embodied in a library of past cases, rather than being encoded in classical rules. Each case typically contains a description of the problem, plus a solution and/or the outcome. The knowledge and reasoning process used by an expert to solve the problem is not recorded, but is implicit in the solution.

To solve a current problem: the problem is matched against the cases in the case base, and similar cases

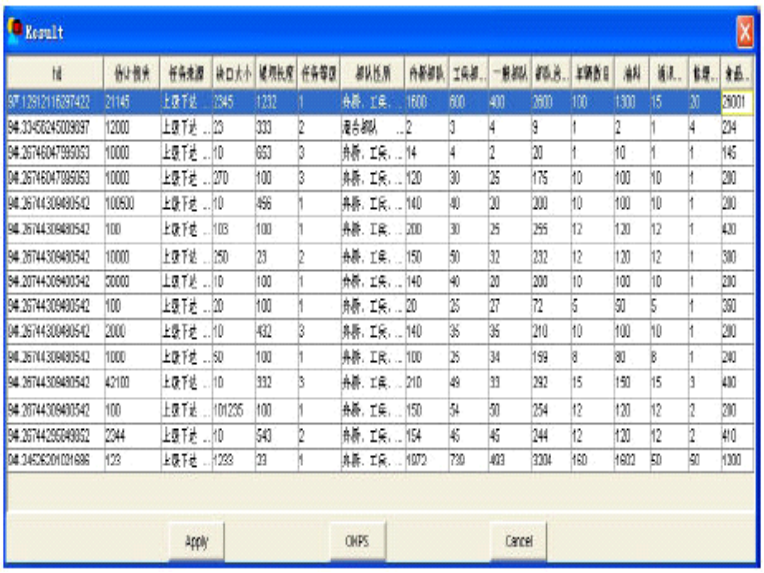

Figure 11. Case similarity retrieval

are retrieved. The retrieved cases are used to suggest a solution which is reused and tested for success. If necessary, the solution is then revised. Finally the current problem and the final solution are retained as part of a new case.

Case-based reasoning is liked by many people because they feel happier with examples rather than conclusions separated from their context. A case library can also be a powerful corporate resource, allowing everyone in an organisation to tap into the corporate case library when handling a new problem (see CBR Website for more information).

Figure 10 illustrates the case base editor which we can insert,delete and revise the cases.We first define the key parameters of the cases and then determine the schema. So, different applications have different schema in the case database. After building the case base, we can retrieval the cases according to the similarity when a new flood emergency happened,as showed in figure 11.This method is better than the RBR (rule base reasoning,or expert system)because it need not to reasoning so as to save the response time.However, it is often difficult to build the case bases.If we can not find the similar cases in the base, we have to use the expert system to reasoning according to the general rules.

4 Agent Interface Services

From the platform architecture discussed in Section 3, we can see that how to provide an Agent Environment in the second layer is the key to implement multiple applications because of following two reasons:

- Agent Environment integrates the components of Common Resources and makes these resources thus available for data mining agents to access and process.
- Agent Environment provides different kinds of agent grid common services or composition for the upper layer to complete complex tasks.

Hence, we need a powerful agent environment to provider interfaces for the hierarchical architecture of the AGrIP platform. As referred in Section 3.2, MAGE is a distributed agent environment and has many advantageous features. Accordingly,we built the agent interface services based on the MAGE environment.

4.1 *Directory Service*

Grid applications often involve large amounts of data and computing and are not easily handled by today's Internet and web infrastructures. Grid technologies enable large-scale sharing of resources within

groups of individuals and institutions(Viktors 2000). In these settings, the discovery, characterization, and monitoring of resources, services, and computations are challenging problems due to the considerable diversity, large numbers, dynamic behavior, and geographical distribution on the entities in which a user might be interested.

Consequently, directory services are a vital part of any grid software or infrastructure, providing fundamental mechanisms for discovery and monitoring, and hence for planning and adapting application behavior. In AGrIP, there are two types of directory service. Correspondingly, there are two types of agents: DF (Directory Facilitator)agent and GISA (Grid Information Service Agent)agent.

DF is a mandatory component that provides a yellow pages directory service to agents. It is the trusted, benign custodian of the agent directory. MAGE may support any number of DFs and DFs may register with each other to form federations. Every agent that wishes to publicize its services to other agents should register its service description with DF. An agent can also deregister itself form DF, which has the consequence that there is no longer a commitment on behalf of the DF to broker information relating to this agent. At any time, and for any reason, the agent may request the DF to modify its service description. An agent may search in order to request information from a DF.

We extended DF with a more abstract interface so that one can query: which agent can do classifying/cluster work? which agent is available at hand? Since we added a reasoning machine embedded in DF, it can "think" by itself. If it thinks that a single agent can not do a specific job, it maybe return more than one agent whom together can do that job. Moreover, if it cannot find agents that are capable of finishing the work, it may resort to other DFs for help.

GISA contains static and dynamic information about compute resources, as well as static and dynamic information about the network performance between compute resources. It provides information directory service. One can query GISA to discover the properties of the machines, computers and networks or other common resources that one wants to use: What is the state of the computational grid? Which resources are available? How many processors are available at this moment? How much bandwidth is provided? Is the storage on tape or disk? GISA provides middleware information in a common interface to put a unifying picture on top of disparate equipment.Additionally,the GISA uses the LDAP (Lightweight Directory Access Protocol) as a uniform interface to such information.

4.2 *Resources Management*

We also built the GRMA (Grid Resource Management Agent) in MAGE platform, which is an important agent that provides capabilities to do remote-submission job start up. GRMA unites common resources and services, providing a common user interface so that one can finish a job with any common resource or service. GRMA is a general, ubiquitous service, with specific application toolkit commands built on top of it.

The GRMA can process the requests for resources for remote application execution, allocate the required resources, and manage the active jobs. It can also return updated information regarding the capability and availability of the computing resources to GISA and DF.

Furthermore, GRMA provides an API for submitting and cancelling a job request, as well as checking the status of a submitted job. We extended Globus Resource Specification Language (RSL) [1] to describe requests. Users can write request in extended RSL and then the request is processed by GRMA as part of the job request.

For example, suppose in the *ics-domain* (ics.ict.ac.cn), which connects hundreds of machines to form a Grid. If one wants to start a MAGE platform in the domain.However,his machine is busy to the moment. He could first query DF to ask which agents have the capabilities to provide information services. After DF tells him GISA agent can do it, he then queries GISA which machine is free in *ics-domain* with a command: "GISA-query-machine ics-domain". Once he gets an answer that "*machine-name.ics.ict.ac.cn*"is free, he can start a MAGE platform on host "*machine-name.ics.ict.ac.cn*" with a request AGrIP to run "*java machine-name.ics.ict.ac.cn mage.Boot -gui*".

[1] http://www.hep.anl.gov/globus/RSL.html

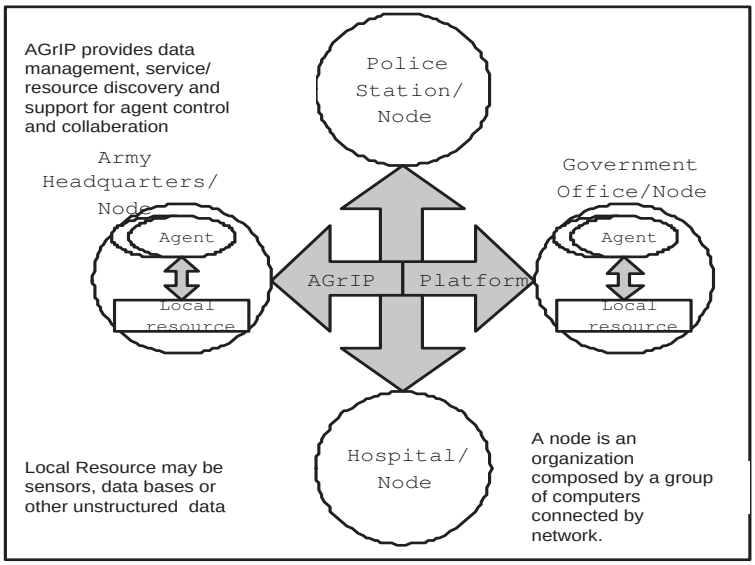

Figure 12. Interconnect environment in the Flood Decision Support System constructed on AGrIP

4.3 *Data Management*

In an increasing number of scientific disciplines, large data collections are emerging as important community resources. In domains as diverse as global climate change, high-energy physics, and computational genomics, the volume of interesting data is already measured in terabytes and will soon total petabytes. The communities of researchers that need to access and mining this data (often using sophisticated and computationally expensive techniques) are often large and are almost always geographically distributed, as are the computing and storage resources that these communities rely upon to store and analyze their data.

DMA (Data Management Agent) is built in MAGE as an agent mainly aiming at accessing remote data and avoiding useless data transfer. Specifically, when a client submits only one distributed data mining task, it is possible to optimize the overall computation time when several servers can perform the computation or when data needed are already stored on storage elements. However, considering from the data side, if data needed by computation are not already present, there is no possibility to optimize. Now, if the client submits a sequence of DDM tasks, which share data, then some data may be transferred more than once. If the result of a task is used by the next task in the sequence, it is useless to transfer this result to the client and back to the platform. A data management service, added to the platform, will register and store this middle results to avoid these useless transfers. Using this service, clients will conduct their data mining more efficiently.

5 Flood Decision Support System

Flood Decision Support system is completely constructed on AGrIP, which provides a high-level infrastructure for distributed system integration. As showed in figure 12, it enables interoperability between distributed heterogeneous systems (organizations), expedite dissemination and retrieval of data, automate operator tasks, support the decision-making process, and facilitate dynamic network disconnection and reconnection of mobile systems emergency environment.

Firstly, GHunt collects the raw information from the Internet and stores it for data mining toolkit MSMiner . Secondly, MSMiner automatically classifies the information and converts them to useful knowledge, which is stored in the case base for CBR (case base reasoning) toolkit to retrieval according to similarity. When a new emergency happens, system first checks the case base. If the case base has not the similar cases for current emergency situation, it applies the expert system to induce the decisions according to rules. Once acquiring the preliminary decisions, AGrIP transmits the result to the front executive

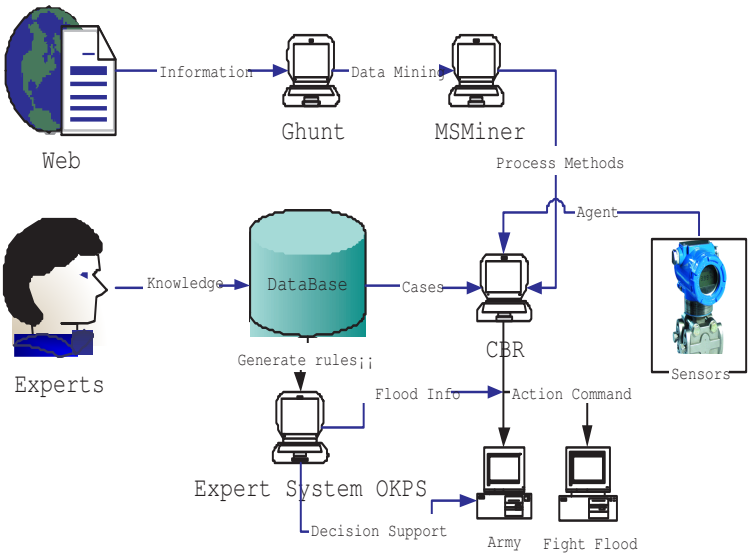

Figure 13. Information Flow of Flood Decision Support System

department for dispatching troops to fight flood. At the same time, current decision is stored the case bases for future emergency process. After the Army and/or police stations acquire the decision commands from the government officers and/or headers, they promptly dispatch the multi-troops to the flood scene according to the Geographic information system, as figure 13 shows.

We use wireless sensor network to monitor rivers, which transmit the real time data to mobile agents. After collecting flood information, the agents sent the message to the central control office where has flood information analysis software wrapped by agents.

5.1 *Wireless sensors for Data Acquisition*

Wireless sensors (Kris *et al.* 2002)provide exciting new opportunities for monitoring the natural environment, such as measurement of water levels. Traditional solutions involve dataloggers from which data is collected periodically in person or via telemetry. With wireless communications and energy drawn from local sources such as solar cells, wireless sensors can be deployed without the constraints of having to wire them up, and data can be conveyed when needed. Significantly, these technologies make it possible to deploy more devices in order to obtain more data more often, and this greater richness of data is set to create a powerful impact on environmental monitoring and decision-making.

Deploying wireless sensors in the natural environment brings a number of challenges. Devices need to withstand harsh environmental conditions. It is often also the case that deployment is expensive and it may be difficult or impossible to access the devices again later, in contrast to working with handheld pervasive devices such as mobile phones. Devices need to run for considerable periods of time making use of available power sources, so conserving energy is hugely important. Taking sensor readings consumes some power but transmitting the data typically takes significantly more energy, so the more often data is sent, the more likely it is that the device will then not have sufficient power to continue its function.

5.1.1 ***Middleware for Sensors.*** The sensor nodes usually end their data to a specific node : the collecting workstation. The node cannot communicate directly with this workstation : it is a *multihop communication*. To communicate, entities require help from other hosts (multihop communication). So, in wireless sensor networks, the nodes must self-organize to monitor the selected area as long as possible according energy consumption optimization. In fact,hosts have limited power resources. One of the whole system aims is so to reduce as much as possible the energy expense. When they have nothing to do generally for sparing energy they enter in a sleep mode. When they communicate they must use good routing protocols and

optimal ways (generally the criteria are the number of hops). But they must decrease as much as possible the flooding scheme because the associated power cost is very high. The communication infrastructure must be very adaptive, fault tolerant and self-stabilized: an agent failure must not have an important impact on the system. This system must provide reliable communications and must adapt to "real-time" constraints.

We need to design a mobile communication management layer to manage the wireless communications between the different agents of the system. This layer must increase interoperability, portability and flexibility of an application by allowing the application to be distributed over heterogenous multiple agents. It must reduce the complexity of development of the agents. This layer is a Message Oriented Middleware. The middleware is based on the MWAC model(J.-P. Jamont *et al.* 2006).

5.1.2 ***MWAC model.*** In wireless sensor network no one can control the organization a priori. Relations between agents are going to emerge from the evolution of the agents'states and from their interactions. We are going to be content with fixing the organization parameters, i.e. agents'tasks, agents'roles.

Our organizational basic structures are constituted by (see fig 14) : one and only one *group representative agent* (r) managing the communication in its group, some *connection agents* (c) which know the different representative agents and can belong to several groups, some *simple members* (s) which are active in the communication process only for their own tasks.

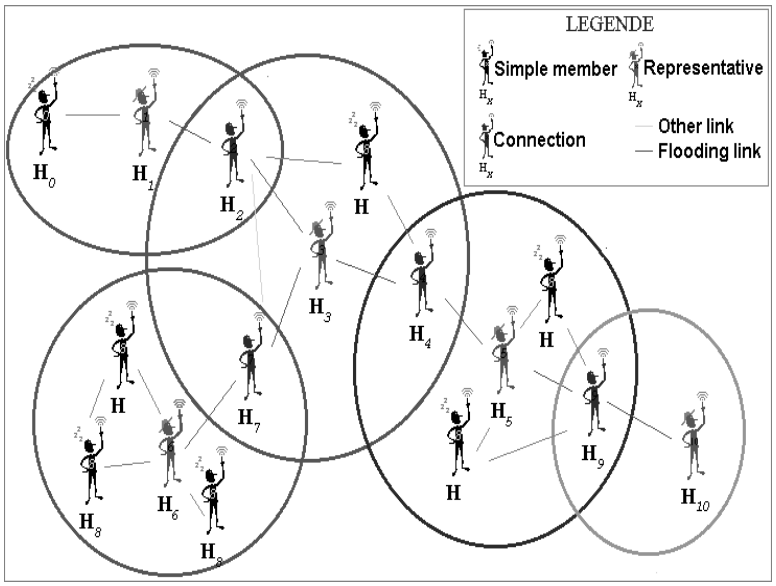

Figure 14. Group organization

With this type of organizational structure, the message path between the source (a) and the receiver (b) is $((a, r),^* [(r, c), (c, r)], (r, b))$. If the source is a representative agent the first term doesn't exist. If the receiver is a representative agent the last term doesn't exist[6].

Because a representative agent is the most solicited agent in a group, the best one is the one having the most important level of energy and the most important number of neighbors. We use a role allocation based self-organization mechanism involving the election of a representative agent based on a function which estimates the adequation between its desire to be the boss and its capacity to be: so the organization is modified only when a problem occurs. We don't try to maintain it if we have no communication.

The energy saving is obtained owing to the fact that the flooding is only directed to the representative agents of the groups and to some connection agents. To give an order of idea, a receiver path research with flooding techniques will cost, in the case of a traditional wireless network, a number of emissions equal to the number of stations. In the case of a clustered wireless network, the number of transmitted messages are about twice the numbers of representative agents (all the representative agents are contacted via one

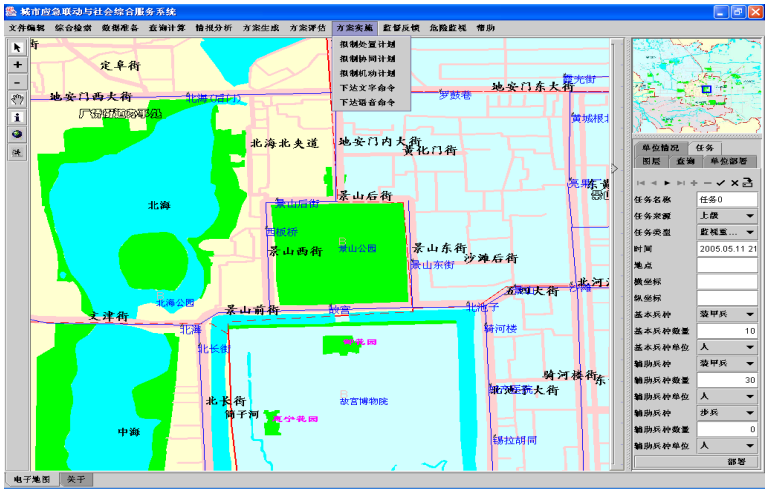

Figure 15. The GUI of Flood Decision Support System

connection agent).

However, the networks with an organizational structure must take care of the maintenance of their routing table. Generally, the adaptive features of these tables come from periodical exchanges between the different nodes. In our approach we do not wish to use this technique to ensure the maintenance of coherence. Indeed, our principle will be "if we do not need to communicate, it is useless to spend energy to ensure the coherence maintenance". However, we will thus use eavesdropping of surrounding agent communications. We extract knowledge from these messages exchange to update our beliefs about our neighbors. Moreover, our self-organization mechanism will integrate an energy management policy. These structures will thus emerge.

5.2 *GIS*

GIS (Geographic Information Systems) are the combination of data management with content and locational indexing, spatial representation and analysis techniques to facilitate the understanding of real-world entities in their context and interactions. A GIS, therefore, records objects in terms of their location and characteristics, and provides analysis and display tools to permit the modeling of the objects and the further modeling of the effects of changes to the characteristics and / or location of the objects upon both themselves and the surrounding environment. For flood decision support system, the decision makers need GIS for decision dispatch.

We implements the GIS (see fig. 15) which facilitates the decision-makers to visually dispatch army to fight flood.It have the basic function on map operation.Besides, the GIS GUI contains menus which allow the users to select emergency places,task parameters, etc.The GIS also integrates algorithms to compute the optimized paths from the policy (army) station (hospital) to the flood emergency.

6 Conclusion and future work

Establishing grids is an important undertaking in developing scalable infrastructures. Although Grid and agent communities both develop concepts and mechanisms for open distributed systems, they focus different perspectives historically. In this paper, based on our past work on multi-agent system, we have presented an agent grid intelligent platform AGrIP from the implementation point of view. AGrIP focuses on service-oriented layer in terms of current existing running environment and applies the agent interface services for interaction between different layers, which make it convenient to access data resource and

integrate heterogeneous legacy systems.

With the AGrIP platform and its toolkits support, we combine the wireless sensor network and software agents to construct the flood decision support system.Through the help of VAStudio, we can conveniently develop software agents to integrate legacy systems.With the wireless sensors, real time data can be collected and transmitted to mobile agents which is analyzed by expert system tool OKPS and case base reasoning system. GIS system embedded in the GUI is to help the decision-makers dispatch command with map to fight floods.

In future work, we plan to investigate the use of policy-based means to establish agent service self-reconfiguration mechanism. Though the policy control, the platform can automatically adjust its way and quantity of services according to the changeable requirements of external nodes in the Grid environment which will improve the system's stability and reliability .

Acknowledgements

This work is supported by the National Science Foundation of China (No. 60435010, 90604017, 60675010). National Basic Research Priorities Programme No. 2003CB317004 and the Nature Science Foundation of Beijing No. 4052025.

References

Jiewen Luo, Zhongzhi Shi,Maoguang Wang, Fen Lin. VAStudio: A Generic multi-agent development toolkit IFIP AIAI 2005 Page 451-458

L.D.Xu, Ning Liang, Qiong Gao: An integrated knowledge-based system for grasslands ecosystems. Knowl.-Based Syst. 14(5-6): 271-280 (2001)

H.X. Li, L.D. Xu. Feature space theroy - a mathematical foundation for data mining. Knowledge-based Systems, 2003. 35:p.45-87

L.Zeng, L.D. Xu, Z.Z. Shi, etc,. Techniques, Process, and Enterprise Solutions of Business Intelligence. In Proc. 2006 IEEE International Conference on Systems, Man and Cybernetics. Taipei, 2006

Zhongzhi Shi, Youping Huang, Qing He, Lida Xu, Shaohui Liu, Liangxi Qin, Ziyan Jia, Jiayou Li. MSMiner-A Developing Platform for OLAP. Decision Support Systems, 2005

Zhongzhi Shi, Qing He, Ziyan Jia and Jiayou Li. Intelligence Chinese Document Semantic Indexing System. International Journal of Information Technology and Decision MakingVol.2, No.3, 2003407-424.

J.-P. Jamont and Occello, M. and A. Lagreze. A Multiagent System for the instrumentation of an underground hydrographic system.Proceedings of IEEE International Symposium on Virtual and Intelligent Measurement Systems 2002

J.-P. Jamont and M. Occello.A self-organized energetic constraints based approach for modelling communication in wireless systems.Proceedings of the 19th International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE'06)2006

Jiewen Luo, Zhongzhi Shi, Maoguang Wang, Jun Hu: AGrIP: An Agent Grid Intelligent Platform for Distributed System Integration. APWeb Workshops 2006: 590-594

Krishanamachari, B., Estrin, D., and Wicker, S. The impact of data aggregation in wireless sensor networks. In Proceedings of International Workshop of Distributed Event Based Systems ,2002

Monique Calisti, Thomas Lozza, Dominic Greenwood:An Agent-Based Middleware for Adaptive Roaming in Wireless Networks.AAMAS Workshop on Agents for Ubiquitous Computing,2004

Jisun Park, K. Suzanne Barber. Finding Information Sources by Model Sharing in Open Multi-Agent Systems AAMAS Workshop on Agents for Ubiquitous Computing,2004

Yoav Shoham. Agent-oriented programming.Artificial Intelligence 60 (1993) 51-92

Zhongzhi Shi, Haijun Zhang, Yong Cheng, Yuncheng Jiang, Qiujian Sheng, Zhikung Zhao: MAGE: An Agent-Oriented Programming Environment. IEEE ICCI 2004: 250-257

Ian Foster. and Kesselman. The Grid: Blueprint for a New Computing Infrastruc- ture (2nd Edition). Morgan Kaufmann, 2004.

Ian Foster, Carl Kesselman, Nicholas Jennings. Brain Meets Brawn: Why Grid and Agents Need Each Other. AAMAS2004, New York

Keith G. Jeffry, Knowledge, Information and Data, A briefing to the Office of Science and Technology, UK, February 2000.

Viktors Berstis. Fundamentals of Grid Computing. IBM Redbooks Paper 2002

Grid Forum http://www.gridforum.org/building-the-grid.htm

Legion http://legion.virginia.edu

Condor http://www.cs.wisc.edu/condor

Unicore http://www.unicore.org

The Globus Toolkit http://www.globus.org/toolkit

CBR website http://www.aiai.ed.ac.uk/links/cbr.html