Towards an agent-based framework for online after-sales services.

Lu Zhang[1]
Frans Coenen[2]
Wei Huang[3]
Paul Leng[2]

[1] Department of Computer Science and Technology, Peking University, Beijing
[2] Department of Computer Science, University of Liverpool
[3] Harrow School of Computer Science, University of Westminster

Towards an Agent-Based Framework for Online After-Sale Services

Lu Zhang [1], Frans Coenen [2], Wei Huang [3] and Paul Leng[2]

***Abstract*—The multi-agent paradigm for building intelligent systems has gradually been accepted by researchers and practitioners in the research field of artificial intelligence. There are also attempts of adapting agents and agent-based systems for creating industrial applications and providing e-services. In this paper, we present an attempt to use agents for constructing an online after-sale services system. The system is decomposed into four major cooperative agents, and in which each agent concentrates on particular aspects in the system and expresses intelligence by using various techniques. The proposed agent-based framework for the system is presented at both the micro-level and the macro-level according to the Gaia methodology. UML notations are also used to represent some software design models. As the result of this, agents are implemented into a framework for which exploits Case-Based Reasoning (CBR) technique to fulfil real life on-line services' diagnoses and tasks.**

***Index Terms*—Agent-based framework, after-sale services, agent-oriented software engineering**

I. INTRODUCTION

In recent years, intelligent agents and multi-agent systems have received much attention from various researchers. The agent technology originates from the research of distributed artificial intelligence, and is gradually becoming a new paradigm of software development (see e.g. [22], [14], and [6]). The multi-agent paradigm can provide both an intelligent solution to difficult problems and a natural way of decomposing systems into individual cooperative agents. Therefore, it is very promising that the agent technology can be widely adopted by the industry for developing complex software. Actually, several industrial applications using agent technology have already been reported in the literature (see e.g. [13], [2], and [11]).

In particular, this agent-based software engineering approach has been effectively adopted in constructing e-services systems in order to achieve the objectives as follows:

- The first objective is to improve the quality of e-services. As e-services are mainly aiming at replacing some previous human-based services, the lack of intelligence might considerably decrease the value of those services. It is natural to see some approaches investigated for artificial intelligence (such as the agent technology) appearing in e-services systems.
- The second objective is to reduce the complexity of the system' design and development. E-services systems usually integrate many various related services, and therefore are typically very difficult to organise. As the multi-agent paradigm is very promising for this kind of systems, using agents for e-services may be quite a natural choice. An example of an agent-based e-services system can be found in [2].

In this paper, we propose an agent-based framework for online after-sale services. This framework is actually motivated by the practice needs of a manufacturing company called Stoves PLC. In Stoves, there are 3000 versions of products to be maintained, and a large number of after-sale services personnel are employed for dealing with the process of product maintenance. It is clear that using I.T technology can reduce the labours involvement in this process and sequentially will be of great benefit to the company. In our framework, we propose to use software agents to replace humans to carry out some of their work. As Stoves is using a typical after-sale services procedure, we further argue that our framework is not only can be used to Stoves, but also can be extended to other consumer-based product manufacturers.

The organisation of the remaining of the paper is as follows. Section 2 presents the background information for the research. In section 3, we present the methodology we exploited in constructing the framework. Section 4 addresses our proposed framework. Section 5 discusses some attributes of the framework. Section 6 reports the undergoing implementation work and some planned future work. The conclusion is drawn in Section 7.

II. BACKGROUND

A. Domain

The problem we are facing originates from the needs of a manufacturer of domestic appliances in a flexible manufacturing context. The company concerned (i.e. Stoves) can deliver more than 3000 versions of its cookers to customers, aiming at satisfying a very wide range of different customer requirements. However, sometime it does create the problems of after-sale services and increase price of services, due to the high cost of the maintenance of its after-sale services personnel.

The existing system in use employs a large after-sale services department consisting of customer call receivers

This project was supported by the Department of Trade and Industry under the Foresight 'Link' programme in conjunction with partners *Stoves* PLC and *NA* Software Ltd.

[1]Department of Computer Science and Technology, Peking University, Beijing 100871, P.R.China, Email: zhanglu@sei.pku.edu.cn

[2]The Department of Computer Science, The University of Liverpool Liverpool L69 3BX, United Kingdom, Email: {frans,phl}@csc.liv.ac.uk

[3]Harrow School of Computer Science, University of Westminster Northwick Park, Watford Road, HA1 3TP, Greater London, United Kingdom, Email: huangw@wmin.ac.uk

and field engineers. When a customer calls to report a fault, the customer call receiver tries to solve that case through a telephone dialogue. If he/she fails to do so, he/she will record the case in an after-sale services information system as an unsolved case. He/She will also assign the recorded unsolved case to a field engineer with the help of browsing task load of each engineer. Every morning field engineers get their assigned cases and go to the corresponding customers to solve the cases one after another. After solving a case, the field engineer will feedback to the after-sale services department to report the solved cases with accomplished records. All the data about previous cases are stored in the system for quite a long period of time. The whole process is quite complicated; it can be depicted in Figure 1. These data are used to facilitate the upcoming diagnosis requirements and/or to adjust the assignment strategies.

Generally speaking, the current system does provide a practical framework for organising the company's after-sale services personnel. However, this system is lack of effectiveness and is not flexible enough, and there have to be many staffs involved to carry out those routeing tasks to complete the entire service process in daily basis. Ideally, we should introduce more intelligence to the system and to find more effective ways of organising and managing our work whilst reducing the costs of services. Some previous research in this domain has been discussed in [7], [8], [23] and [24].

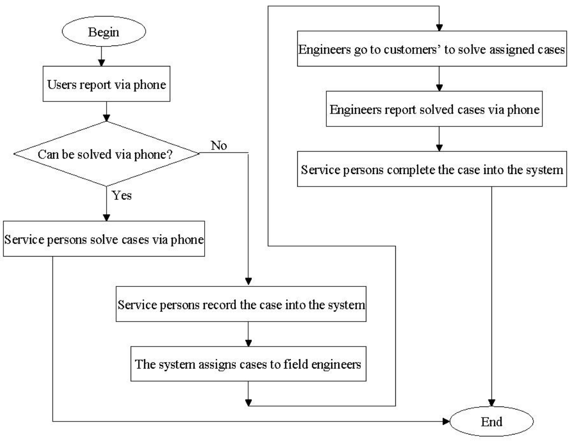

Fig. 1. Stoves's After-Sale Services Process

B. Requirements

To provide intelligent support for the process that is described in Figure. 1, the new proposal in such a system should provide intelligence in the following four parts.

- **Customer Requirements Collection**

Under the current process, there have to be customer call receivers to deal with customer requirements. Customers are usually not familiar with cookers, and they should be induced to report the real problems. As the current software system cannot provide intelligent interactions with customers, the customer call receivers should serve as mediators. It's a good solution to use software replacing part or all the responsibilities of the customer call receivers. For example, a interactive dialogue can be set up between the service website and a customer. The e-service providers can use the dialogue tool to acquire more information about the customers' requirements. The major challenge is how to automatically generate appropriate questionnaire on the individual cookers, which will lead to discover the real customers' requirements.

- **Engineer Job Planning**

The current situation is the call receiver's responsibility to assign the received job to an appropriate engineer. However, this solution is not obvious, neither intelligent nor flexible. In most cases, a call receiver concentrates on receiving customer requirements, and just browses the existing assignments briefly before deciding the final planning. Rationally speaking, job planning also requests intelligence, and itself is a topic in the AI research field. In practice, a good job planner should consider at least the following aspects when doing the planning: workload, speciality of jobs, and the availability. The workload of different field engineers should be balanced. Each engineer should better be assigned a job in his/her speciality. It is not likely to assign an engineer any job on his/her absent (due to illness or holiday) days. The planner should also consider each engineer's ability to gain experiences. This means that an engineer's speciality is changing all the time. The planner should try to always track the current speciality of each engineer and find out their needs and availability.

- **Intelligent Diagnosis Support**

An intelligent diagnosis support may substantially benefit either the customers or the call receivers or even the field engineers. A customer may log into the after-sale services website and diagnose his/her cooker. A call receiver may also consult this support to solve problems. A field engineer should be available, although he/she may encounter some unfamiliar products. In this case, consulting and support could be helpful for them. There are three main categories of techniques that have been proposed to support intelligent diagnosis.

  - Rule-Based Reasoning (RBR) is the traditional paradigm of intelligent systems, and also used in some diagnosis systems, such as [10], [18] and [16].
  - Model-Based and qualitative reasoning (MBR) [5] [12] uses qualitative models to describe the structure and/or relationships of components in the target for diagnosis. A qualitative model can be operated to simulate the procedure that results in the fault.
  - Case-Based Reasoning (CBR) [19] is also applied for diagnosis. Actually, we mainly adopt the CBR approach for diagnosis in our framework.

- **Knowledge Management and Data Analysis**

Another important issue that should be considered is the management of previous data and knowledge. First, maintaining previous data is essential to all the above three parts. In the customer requirements collection part, an effective dialogue may be based on tracking the profile of each individual customer. In the engineer job planning part, capturing the changes of each engineer's speciality should be based on each engineer's previous experiences. In the intelligent diagnosis support part, the CBR technique is totally based on comparing current case with previous cases. Secondly, key knowledge should be constantly discovered from the existing data. As the historical data will be growing all the time, this growth may affect the overall performance. A possible solution is to exploit an independent part to discover knowledge off-line constantly, and supply the knowledge to other parts.

III Methodology

A. Using Gaia

To construct the framework, we use the Gaia methodology for agent-oriented analysis and design. This methodology is originally proposed in [20], and further developed in [21]. It is a general methodology for various kinds of multi-agent systems. In Gaia, analysis and design issues are addressed distinctly at the micro-level (agent) and the macro-level (societal). In the analysis phase, the macro-level aspects are addressed by the interaction model, and the micro-level aspects are addressed by the role model. In the design phase, the above two models will be further refined into three models: the agent model, the services model and the acquaintance model. The acquaintance model is a refinement of the interaction model, which further defines the communication links between agents. The agent model is a refinement of the role model. Although the mapping from roles to agents may not be a one-to-one correspondence in Gaia, a natural mapping is just to refine one role into one agent. In this paper, we will also adopt this natural mapping. The services model further specifies the functions provided by the agents.

B. Using UML

UML [3] [17] is a comprehensive modelling language for various software systems. In recent years, UML has gradually become the dominant analysing and designing language for especially object-oriented systems. As multi-agent systems have many attributes similar to object-oriented systems, many attempts of using UML for analysing and designing multi-agent systems have been reported intensively in the literature (see e.g. [1], [4] and [15]). Therefore, although the original Gaia methodology does not support UML, we will have to instead present some Gaia models in UML diagrams.

In this paper, the UML diagrams are mainly used at the macro-level. We use the notation for objects in UML class diagrams to represent roles (agents) in the interaction model. UML class diagrams have been successfully used to represent object models in object-oriented systems. Due to the similarity between agents and objects, we think it is quite natural to use UML notation for objects to represent roles (agents) in the interaction model. We also adopt the approach proposed in [15] to use UML activity diagrams to represent the acquaintance model.

IV. The Proposed Framework

A. Micro Level

From the above requirements, we can naturally break the system into four roles/agents: the customer requirements collection role (agent), the engineer job planning role (agent), the intelligent diagnosis support role (agent), and the knowledge management role (agent). The above four roles/agents will interact with each other in the actual system. The specifications of the four roles/agents are as follows. Due to the limitation of space, we will not present the liveness and the safety properties in the agent models, and the pre-conditions and post-conditions in the services models.

- **Customer Requirements Collection Role/Agent (Customer Handler)**

The specification of the customer requirements collection (role /agent) is depicted in Fig. 2. The responsibility of this role (agent) is to interact with customers.

Role/Agent Schema: **CustomerHandler**
Description: Receives request from the customer and responses to the customer via consulting other agents.
Protocols and Activities: **WaitCall, ProduceDiagnosisCase, ProduceEngineerJob, RespondToCustomer**
Permissions: **reads** **supplied** customerDetails **supplied** customerRequest diagnosisResult **generates** diagnosisCase engineerJob customerResponse

Fig. 2. Role/Agent Model for Customer Handler

The specifications of the functions in Fig. 2 are presented in the services model for this role (agent), which is depicted in Table 1. The inputs and outputs of each service are presented in detail.

TABLE 1. Services Model for Customer Handler

Service	Inputs	Outputs
WaitCall	customerDetails customerRequest	customerRequirements
ProduceDiagnosisCase	customerRequirements	diagnosisCase
ProduceEngineerJob	customerRequirements	engineerJob
RespondToCustomer	diagnosisResult	customerResponse

- **Engineer Job Planning Role/Agent (Job Planner)**

The specification of the customer requirements collection role (agent) is depicted in Fig. 3. The responsibility of this role (agent) is to assign the job passed from the customer handler. It receives the job, and finds the most suitable engineer for the job. The specifications of the functions in Fig. 3 are presented in the services model for this role (agent), which is depicted in Table 2.

Role/Agent Schema: **JobPlanner**
Description: Receives an engineer job and assign the job to a field engineer according to his/her workload, speciality and availability etc.
Protocols and Activities: **GetEngineerJob, CheckEngineerWorkload, CheckEngineerSpeciality, CheckEngineerAvailability, CheckEngineerExpectation, ScheduleJob**
Permissions: **reads** engineerJob engineerWorkload engineerSpeciality engineerAvailability engineerExpectation **generates** jobAssignment

Fig. 3. Role/Agent Model for Job Planner

- **Intelligent Diagnosis Support Role/Agent (Fault Diagnoser)**

The specification of the customer requirements collection role (agent) is depicted in Fig. 4. The responsibility of this role (agent) is to diagnose the case passed from the customer handler. It receives the formatted case from the customer handler, tries to find similar cases in the previous cases obtained through the knowledge management agent. Then it uses the retrieved similar cases to synthesise the fault for this case. The diagnosed fault will be passed back to the customer handler as the result of diagnosis. This result may also be presented to field engineers.

TABLE 2. SERVICES MODEL FOR JOB PLANNE

Service	Inputs	Outputs
GetEngineerJob	-	engineerJob
GetEngineerWorkload	-	engineerWorkload
GetEngineerSpeciality	-	engineerSpeciality
GetEngineer Availability	-	engineerAvailability
GetEngineer Expectation	-	engineerExpectation
ScheduleJob	engineerJob engineerWorkload engineerSpeciality engineerAvailability engineerExpectation	jobAssignment

Role/Agent Schema: **FaultDiagnoser**
Description: Receives fault diagnosis request and produces one or more possible faults.
Protocols and Activities: **GetDiagnosisCase, GenerateDiagnosisResult, GetPreviousCases**
Permissions: **reads** diagnosisCase previousDiagnosedCases **generates** diagnosisResult

Fig. 4. Role/Agent Model for Fault Diagnoser

The specifications of the functions in Fig. 4 are presented in the services model for this role (agent), which is depicted in Table 3.

TABLE 3 SERVICES MODEL FOR FAULT DIAGNOSER

Service	Inputs	Outputs
GetDiagnosisCase	-	diagnosisCase
Generate DiagnosisResult	diagnosisCase previousDiagnosedCases	diagnosisResult
GetPreviousCases	-	previousDiagnosedCases

- **Knowledge Management Role/Agent (Knowledge Manager)**

The specification of the customer requirements collection role (agent) is depicted in Fig. 5. The responsibility of this role (agent) is to constantly discover useful knowledge from previously acquired data, and passes the knowledge to other agents whenever it is asked for.

The specifications of the functions in Fig. 5 are presented in the services model for this role (agent), which is depicted in Table 4.

B. Macro Level

At the macro level, we just present the interaction model and its refinement – the acquaintance model.

Interaction Model

From the specifications of the four roles/agents, we can represent the relationships between them as Fig. 6. In Fig. 6, the above-discussed roles (agents) are represented in the notation in UML for classes. The arrows represent the directions of data flow. For example, the arrowed line between the Knowledge Manager and the Customer Handler means that the Customer Handler will read information from the Knowledge Manager.

Role/Agent Schema: KnowledgeManager
Description: Maintains various kinds of data and discovers knowledge from the data.
Protocols and Activities: **GenerateEngineerWorkload, GenerateEngineerSpeciality, GenerateEngineerAvailability, GenerateEngineerExpectation, GeneratePreviousDiagnosedCases, DiscoverKnowledge**
Permissions: **reads** rawData **generates** engineerWorkload engineerSpeciality engineerAvailability engineerExpectation previousDiagnosedCases keyKnowledge

Fig. 5. Role/Agent Model for Knowledge Manager

TABLE 4. SERVICES MODEL FOR KNOWLEDGE MANAGER

Service	Inputs	Outputs
GenerateEngineerWorkload	keyKnowledge	engineerWorkload
GenerateEngineerSpeciality	keyKnowledge	engineerSpeciality
GenerateEngineerAvailability	keyKnowledge	engineerAvailability
GenerateEngineerExpectation	keyKnowledge	engineerExpectation
GeneratePreviousDiagnosedCases	keyKnowledge	previousDiagnosedCases
DiscoverKnowledge	rawData	keyKnowledge

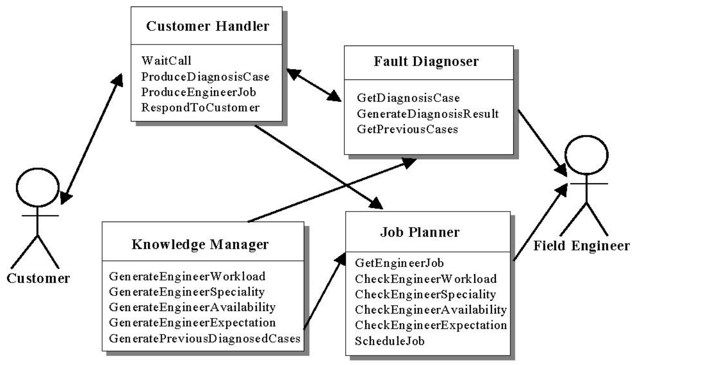

Fig. 6. Interaction Model

Acquaintance Model

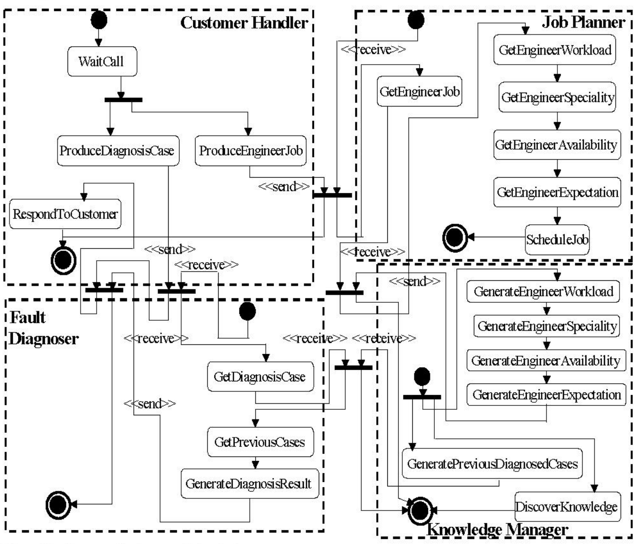

Fig. 7. Acquaintance Model

Based on the interaction model in Fig. 6, we can get the refined acquaintance model in Fig. 7, represented in UML activity diagrams. Communication channels between agents are represented as synchronisation points labelled with <<send>> and <<receive>>. These channels do not belong to any single agent, and therefore is depicted outside all the agents. In the Customer Handler, there is an internal synchronisation point. This means ProduceDiagnosisCase and ProduceEngineerJob can work parallel. The internal synchronisation point in the Knowledge Manger is also similar.

V. Implementation &Further Development

The research reported in this paper is still an ongoing one. In this section, we will briefly describe the implemented part, and the planned development.

A.. Fault Diagnoser

Our implementation is mainly focused on the fault diagnosis agent. The reason of this is that agents are comparatively independent in the framework. It can even work alone by attaching a human-friendly interface for entering symptom information and a database for storing relevant data.

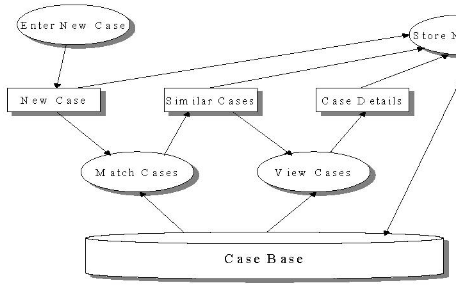

Fig. 8. Overview of the Fault Diagnosis Agent

In our implementation, we adopt the case based reasoning technique in this agent. The overview of this agent is depicted in Fig. 8. In the centre of the agent is the case base storing the history of the previously occurred cases of faults. When encountering a new case, the field engineer will provide a structured description of the new case. This description will be matched with the cases in the case base. Some most similar cases will be retrieved, which can be viewed in detail by the field engineer to help him/her to identify the fault of the new case. After the new case has been solved, the field engineer can store it into the case base for future diagnosis. Detailed reports of this agent can be found in [8] and [24].

In fact, as in Fig. 8 illustrated that it only depicts the case when this agent works alone. After the whole framework is implemented, a new case will not be entered directly by a person but by the customer requirements collection agent. This agent will not access previous cases directly but via the knowledge management agent.

The interface for entering a new case for diagnosis in the tool is depicted in Fig. 9, in which a field engineer can input the structured description of a case under diagnosis. This case will be matched with the cases in the case base, and the results will be displayed in the interface depicted in Fig. 10.

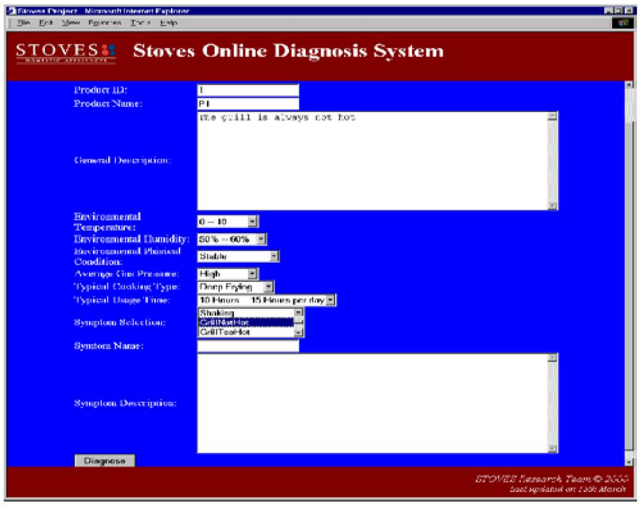

Fig. 9. Interface for Entering Cases for Diagnosis

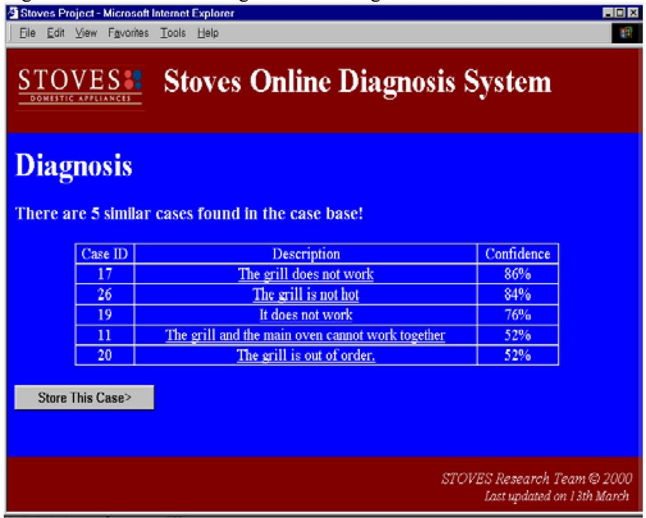

Fig. 10. Interface for Displaying Retrieved Cases

B. Planned Implementation

The development of the other three agents is planned. We think the key issues in the future implementation are as follows:

- ♦ For the customer requirements collection agent, we may apply natural language recognition techniques and/or human intention recognition techniques for our purpose.
- ♦ For the engineer job planning agent, the main difficulty is a job scheduling problem. Optimisation techniques have been used to tackle this kind of problems.
- ♦ For the Knowledge Management agent, we will use data mining techniques to discover relevant knowledge from the vast volume of data. Besides the three agents, we will also implement the communications between the three agents. This can be based on an existing agent communication language, such as KQML [9].

VI. Discussion

A. Property of the Framework

The main property of this framework is its generality. Although the framework is designed for the requirements of Stoves PLC, it also matches the needs and demands in other industrial organizations and business partners. The identified four agents are not unique in Stove's product maintenance process. They are typical and identical for any online after-sale services system. The system not only is able to deal with customer requests, but also provide instant help to customers, as the fault diagnosis agent does in this framework. Further more, it should be able to trigger the traditional human-based maintenance process in case the provided e-services are not satisfactory. The engineer job planning agent takes this responsibility in this framework. Finally, there should be an agent to maintain the vast volume of data, and provide relevant knowledge for other agents. Based on the above reasons, we think this framework will be adopted in other online after-sale services systems.

B. Benefits of Using Agents

According to our implementation, the benefits of using agents can be easily discovered as follows:

♦ First, the multi-agent paradigm can naturally reflect the nature of the problem. In our framework, the four agents come out naturally during the stage of problem analysis. Therefore, we can focus on how to develop each individual agent rather than on how to divide the system into agents.

♦ Secondly, this paradigm may reduce the complexity of developing the intelligent system. Providing intelligence in a system is usually a difficult task. This difficulty will be entangled with the complexity of software development. The case is even worse when there are several parts in the system that needs intelligence. The multi-agent paradigm can separate different parts requiring intelligence into different agents, and therefore can effectively deal with the complexity of developing intelligent systems.

♦ Thirdly, this paradigm can incorporate flexibility into development. The implementation of the entire framework may require quite a long time. However, the multi-agent paradigm makes it possible for us to apply the developed agents into operation with other agents still under development.

VII. Conclusion

In this paper, we have presented a general framework for agent-based online e-services. The requirements for this framework rise from the product maintenance process of our partner; the created framework itself also suits the needs of other manufacturers and can cope with diversity of changes in providing online services in industry. The framework is presented and designed in compliance with the Gaia methodology; The UML notations have contributed to produce the interaction model and the acquaintance model for the system's analysis and design. The fault diagnosis agent in the framework has been successfully implemented by using the Case-Based Reasoning technique and clearly shows their great usability and productive potential.

VIII. Acknowledgements

The research described in this paper is supported by the Department of Trade and Industry under the Foresight 'Link' programme. We wish to express our thanks to our partners in the project, *Stoves* PLC and *NA* Software Ltd, and in particular to Colin Johnson, Prof Mike Delves, and the Project Monitor, Stan Price.

References

[1] Bauer, B. UML Class Diagrams Revisited in the Context of Agent-Based Systems. In: M. Wooldridge, G. Weiss, and P. Ciancarini (eds.) Agent-Oriented Software Engineering II. Springer-Verlag Lecture Notes in Computer Science 2222, Canada,(2001) 101-110

[2] Beer, M.D, Huang, W., Using Agents to Build a Practical Implementation of the INCA (Intelligent Community Alarm) System. In: Proceedings of UKMAS2000, Oxford University, UK. December 2000.

[3] Booch G., Rumbaugh J., Jacobson I.: The Unified Modeling Language User Guide. Addison-Wesley. (1999)

[4] Caire G., Coulier W., Garijo F., Gomez J., Pavon J., Leal F., Chainho P., Kearney P., Stark J., Evans R., Massonet P.: Agent Oriented Analysis Using Message/UML. In: M. Wooldridge, G. Weiss, and P. Ciancarini (eds.) Agent-Oriented Software Engineering II. Springer-Verlag Lecture Notes in Computer Science 2222, Canada, May (2001) 119-135

[5] Chantler M. J., Coghill G. M., Shen Q., Leitch R. R.: Selecting Tools and Techniques for Model-Based Diagnosis. Artificial Intelligence in Engineering, 12, (1998) 81-98

[6] Ciancarini, P., Wooldridge, M.J. (eds.): Agent-Oriented Software Engineering. Springer-Verlag Lecture Notes in Computer Science 1957, January (2001)

[7] Coenen, F., Leng, P., Weaver, R., Zhang, W.: Integrated Online Support for Field Service Engineers in a Flexible Manufacturing Context. In: Applications and Innovations in Intelligent Systems VIII (Proc ES2000 Conference, Cambridge), eds A Macintosh, M Moulton and F Coenen, Springer, London, (2000) 141-152

[8] Coenen, F., Leng, P., Zhang. L.: Flexible Field Service Support using Multiple Diagnostic Tools. In Proceedings of 5th IEEE International Conference on Intelligent Engineering Systems, (2001) 225-229

[9] Finin T., Fritzson R.: KQML — A Language and Protocol for Knowledge and Information Exchange. In: Proceedings of the 13th International Distributed Artificial Intelligence Workshop, Seattle,WA, USA, (1994) 127–136

[10] Huang, W., Beer, M.D: Towards Intelligent Health Care System - An Introduction to Agent-based Community Care System INCA. In: Proceedings of PERP 2001, the University of Keele, UK April (2001) 95-99

[11] Huang.W, "Developing Advanced Agent Systems with UML and Gaia Methodology" invited speaker and paper in conference proceedings of the 6th World Multi-conference on Systemic, Cybernetics, and Informatics (SCI2002), Orlando, Florida, USA, July, 2002

[12] Hunt J. E., Lee M. H., Price C. J.: Progress in Applying Model Based and Qualitative Reasoning to industrial applications, The MONET Newletter 1(1) (1998)

[13] Jennings, N. R., Wooldridge, M.J., Applications of Agent Technology. In: N. R. Jennings and M. Wooldridge, editors, Agent Technology: Foundations, Applications, and Markets. Springer-Verlag, March (1998)

[14] Jennings N. R.: Building Complex, Distributed Systems: The Case for an Agent-Based Approach, Communications of the ACM, 44(4), April (2001) 35-41

[15] Moulton M.: A Rule-Based Incident Tracking System, In: A. Macintosh and R. Milne (eds), Applications and Innovations in Expert System V, SGES publications, London. (1997)

[16] Rumbaugh J., Jacobson I., Booch G.: The Unified Modeling Language Reference Manual. Addision-Wesley. (1999)

[17] Vazquez E., Chancon O. L., Altuve H. J.: An On-line Knowledge-Based System for Fault Section Diagnosis in Control Centers, In: Proceedings of 1996 International Conference on Intelligent Systems Applications to Power Systems, Orlando, FLA, (1996) 232-236

[18] Watson I.: Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kaufman, London, (1997)

[19] Wooldridge, M. J, Jennings. & Kinny, D.: The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems, 3(3). (2000) 285-312

[20] Wooldridge, M.J, Reasoning about Rational Agents. The MIT Press, July 2000.

[21] Zhang, L., Coenen, F., Leng, P.,: On-Line Support for Field Service Engineers in a Flexible Manufacturing Environment: the Stoves Project. In: Proceedings of IeC '2000 Conference, Manchester (2000) 31-40

[22] Zhang, L., Coenen, F., Leng, P.,: A Case Based Diagnostic Tool in a Flexible Manufacturing Context. In: Proceedings of the Sixth UK CBR Workshop, 10 December (2001) 61-69