Extending the gaia methodology for the design and development of agent-based software systems.

Wei Huang
Elia El-Darzi
Li Jin

Harrow School of Computer Science

Extending the Gaia Methodology for the Design and Development of Agent-based Software Systems

Wei Huang , Elia El-Darzi and Li Jin
School of Computer Science, University of Westminster
Watford Road, London HA1 3TP, United Kingdom.
E-mail:{ huangw, eldarze, jinl }@wmin.ac.uk

Abstract

Over the past decade, agent-based computing has emerged as a new and popular paradigm for design, implementation and analysis of distributed information systems. In this paper, the participant researchers in Health Care Computing Group at University of Westminster concentrate on the agent-oriented methodology for the analysis and design of agent-based systems and identify how methodology can support both the levels of "agent structure" and of "agent society" in the agent-oriented software design and development process. The research reported here takes one leading agent-oriented methodology-Gaia, and then extended it by the creation of innovative design tools which aimed at better supporting application to real-world domains. In discussion section, agent-oriented methodology and AUML approaches are compared and evaluated in great detail; the strengths and weaknesses of the current agent-oriented methodology are explored and discussed; the importance of effectively using methodology to improve agents and their productivity potential also is emphasized. Finally, we draw conclusions from the work presented and the experience gained in this research and look into the future possible improvements on agent-oriented software engineering in the agent technology research field.

1. Introduction: Gaia Methodology and its Concepts

"*A methodology is a recipe that enables an engineer to find a solution to a specified set of problems*". In the book [5], Hussmann has highlighted that a methodology should enable a good definition of the problem space to which methodology is applicable, and should provide a set of models, methods, and guidelines which allow developers to follow and to find solutions to the problem faced. Gaia methodology [9] is a practical approach to analysis and design, implementation and management of agent software systems. It aims to provide agents, services, resources, and share agent community/ experience, and to offer practical support to the tasks of system design, development and implementation. Gaia methodology suggests that the whole multi-agent system can be defined in team of agent roles and agent organizations, and an organization can be defined in term of a range of practical designable and reusable models. Later those models are used as the foundation stones for building the system architecture. There are two main phases in Gaia methodology to help system developers to define the agent structure (micro-level) and agent society and organization structure (macro-level). These concepts are presented in two basic stages in Gaia methodology: *analysis* and *design*[9].

- 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. It focuses on agent organizational aspects both in the team of concepts (such as roles, responsibilities, agent types, protocols) and in teams of organizational domain design (such as services, interactions, acquaintances, activities. In this phrase, the major tasks are to define a collection of roles of agents and define relationships between agents. So, roles can be seen as the basis of agent system design. Each role has four attributes, namely responsibilities, permissions, activities, and protocol, which are used to describe the feature and states of agents.
- In the *design* phase, the above two models will be further refined into three Gaia models: *agent model*, *services model* and *acquaintance model*. The *agent model* is a refinement of the role model. Although the mapping from roles to agents may not necessarily be a one-to-one correspondence in Gaia, a natural mapping is just to refine one role into one agent. The *services*

31st Annual International Computer Software and Applications Conference(COMPSAC 2007)
0-7695-2870-8/07 $25.00 © 2007 IEEE

model further specifies the functions and services provided by individual agent. The *acquaintance model* is a refinement of the interaction model, which further defines the communication links between agents.

2. Agent-oriented Design and Development Process in Gaia Methodology

The Gaia methodology describes a high level outline of design and development processes, identifies important steps to build an agent cooperation environment, and outlines models and plan for designing effective agent architecture. In this research, we divided the Gaia's two concepts, i.e. *Abstract Concepts* and *Concrete Concepts* into a progressive design process; it is illustrated in [Fig1].

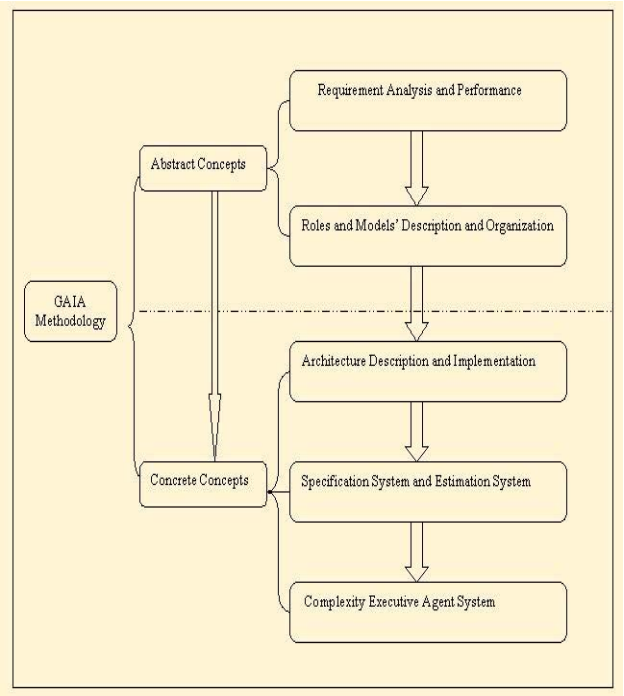

Figure1: Gaia-based Agent Environment Development and Processing Prototype Diagram

The diagram [Figure1] shows the Gaia model for agent-based system development, from conceptual requirement analysis to the creation of roles and models, then moving to practical architecture design and complex system implementation by using Gaia concrete concepts. In the *analysis* phase, i.e. *abstract concepts* level, the aim is to produce agent-oriented abstractions and specifications (roles, Interaction models) for the system under the development. This is an interpretation and decomposition of the complex problem to be solved. The role model is used to clarify function and formalize skills (responsibilities, activities) of agents. In order to share resources, skills, problems, and tasks and to facilitate the design of the solution, the interaction model is used to clarify the relationships between roles and to link the cooperative agents. In the *design* phase, i.e., *concrete concepts* level, the aim is to define a solution of the problem. The design process consists of identifying the three models (the *agent model*, the *services model* and the *acquaintance model*), specifying what they are to accomplish, and constructing and organizing all the components into suitable agent architecture. Those design models should provide a clear computational description of the system, which then promises the system's implementation as a relatively straight-forward refinement of adding details about the implementation platforms, programming language and computational algorithms, without reconsidering the structure of the agent system [4,7].

In this research, we also purpose a flow chart design diagram [Figure.2], which is able to decompose agent-based components and services from agent society level and to design and construct them into the three Concrete models, i.e. *agent model*, *service model* and *acquaintances model* in Gaia methodology.

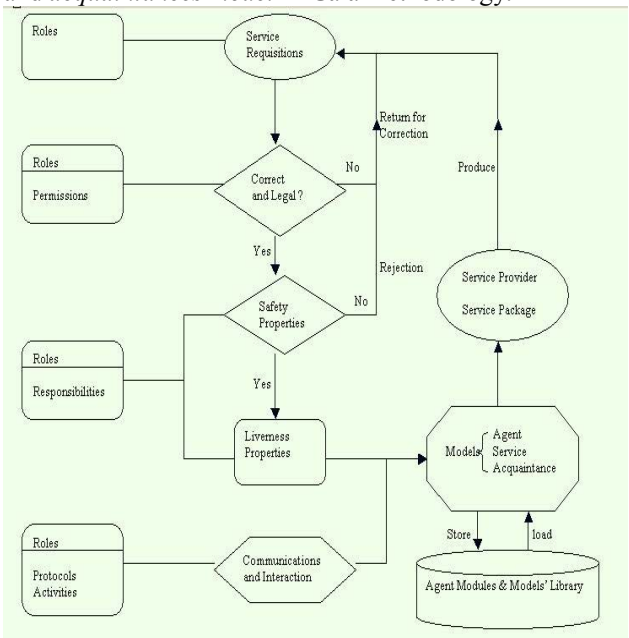

Figure2: The Flow Chart of Gaia's Design Processing

This design flow chart diagram[Figure2] also shows how a series of instructions guide system developers through the Gaia's design processing and pathway, i.e. from *Abstract concepts* to *Concrete concepts*. Each instruction may perform one of Gaia's basic functions or procedural abstractions. It begins with the initiation and definition of roles, and then comes a group of

roles' planning for organizing agents' behaviours, followed by loading particular models (agent type, service and acquaintance) from the agent models library. The use of agent-oriented techniques is to provide a structured way to deal with complexity, so the benefits of modelling and design compound as the application size grows large. Another benefit of structured models such as these is that they enable reuse of agent models and services. Eventually, it is hoped to build up a library of models of agent-based components, each one representing an implementation stored in a library; these could be reused and invoked by developers for multiple agent system design, implementation and maintenance. When another task needs the same functionality, the designer can quickly import its module from the library. In the implementation phase, the developer can just as easily import and organize the agent-based modules into the executable agent-oriented architecture.

3. Extension: Gaia-based practical design and modelling tools

Jennings[6] states that: "agent-oriented approaches are well suited to developing complex software systems in general and control system in particular". As yet, there are not many agent-oriented software and applications in the market. In part, this is due to the complexity of agent system development, the absence of mature techniques and the shortage of useful software tools. The commonly considered solutions are the development of the component-based software techniques for the integration of components like database software applications into agent systems and the development of the useful structures for collections of agents into teams and groups in agent architectures.. The Gaia methodology is an attempt to define a high-level methodology that is specifically tailored to the analysis and design of multi-agent systems.It offers a high-abstraction level specifications and a range of capable structural models for systems analysis and design. However, there is no given way to go from a Gaia analysis model to a design model. System implementations in most cases are still done through the extension of object-oriented techniques. Furthermore, the superstructure level, i.e. the level of agent domain modelling, is not considered in Gaia methodology; the role model has been emphasized in only textual schema. Therefore, the Gaia methodology provides a high level of abstraction for describing and organizing software agents rather than modelling and implementing them. Obviously, managing complexity issues and creating reliable agent-based software is a very challenging task facing developers of large-scale embedded software systems. In order to cope with the complexity of the models as a whole, it is valuable to have access to multiple graphical views of a system, each from a different perspective, each representing different models and their associated components or elements. The experience gained in the agent system design undertaken for this research is that the absence of such graphical modelling tools is a great obstacle to effective system design and implementation. Therefore, in this research, a Gaia-based graphic modelling tool was developed and exploited. The design philosophy for agent-based system is based on progressive development and decomposition of a system's intended behaviour, i.e., extending Gaia methodology and adapting some modelling techniques and ideas from object-oriented software engineering (since object-oriented modelling techniques are not directly applicable to agent systems , and agents are more complex than objects), providing a compositional modelling tool suitable for the verification of agent-based system structure and function. Here, in order to simplify and complete the Gaia methodology, a set of Gaia-based graphic modelling diagrams for addressing domain-level abstractions are introduced and prototyped in the *E.U-INCA* health care project [2,4]as follows:

• Agent and Role Schema Diagram: This defines the agent types by aggregating roles and it also defines role of each agent and its related attributes. It is a blueprint-style description on agent and their role schema.

Agent /Role Schema:	name of role
Description:	short description of the role
Protocols and Activities	Communication protocols and activities in which the roles plays a part
Permissions	The rights associated with the role
Responsibilities: Liveness Safety	Liveness responsibilities Safety responsibilities

Table .1: Template for defining agent and roles[9]

In Gaia analysis level, this schema helps to discover individual agent's Abstract concepts and agent's attributes e.g., responsibilities, permissions, activities, liveness and safety issues. This mainly describes individual agent's functions, states and abilities. The role model captures the attributes and properties: the Gaia-based Role can be described in the following template agent/role schema [Table.1].

For example in *E.U-INCA* (Intelligent Agent-based Health Care System) [2,4] a community care system consists of a set of professional bodies and organizations. Each of them plays a certain role in

their society. The formal schema for agent Care Service Coordinator could be outlined in [Figure3]

Role Schema: **Care Service Co-ordinator**

Description: The role of crisis management and organizing
The agency responsible for providing the range of services necessary to ensure that the service is arranged properly. The co-ordinator is responsible for preparing a plan and for monitoring its effectiveness in meeting the needs of the assistance services. Agent who can assist and schedule needed services in Health Care events handling.

Protocols and Activities:

Monitor Response& Activities, Get Requirement, Services Originations, Service Routing Management

Permissions:

Reads supplied: Services requirements
Services planning and scheduling
Services Processing procedure

Generates: Agents (organising)
Services (agents producing, mapping and routing)
Internal and external environments' data accessing.

Monitoring: Service agents' status
Service progress

Responsibilities:

Liveness:

Assistance co-ordinator = ManagementResponse

ManagementResponse= ((GetServices Requirements)|(Crisis discovery|| Identify Problem issues) | (Negotiation|| Conversation) |(Service Decision Making)| (Organize Service Agents)|(Monitoring Service Agents Activities)| (Feedback-Reports | Effective Evaluations))

Safety:

(Registration | network accessing | security service checking) = false => service =nil

Figure3: The Gaia role model: a formal schema for agent Care Service Coordinator

• Service Diagram: This provides a domain view of the services and tasks of individual agents. Those services and tasks all based on the agent's role which has described in Agent/Role Schema diagram, service diagrams also should include some additional information on practical events and particular conditions. The service may vary due to the different tasks and conditions applied to agent-based system. For example, in *E.U-INCA*[2] agent Care Coordinator in the event of the basic proposed services diagram for agent Care Coordinator can be seen in [Figure 4].

• Collaboration and Interaction Diagram: This describes how an agent responds to the system or to other agents, the communication/negotiation routing and service scheduling and planning pathway. This is normally achieved by using a commonly-accepted communication language and protocols.[Figure5] shows the communication paths and route between the patients and health care system. It illustrates that after having carefully analysed patients' requests, the agent Care Coordinator passes patient's case to an appropriated health care service system, e.g., a local clinic, dental office, or hospital and then this information is diverted to a responsible health care professional, such as a GP or a nurse.

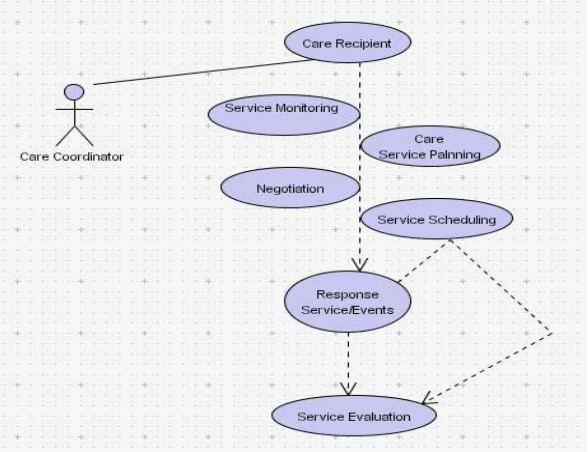

Figure 4: Gaia Service Diagram: Basic Services of Agent Care Coordinator

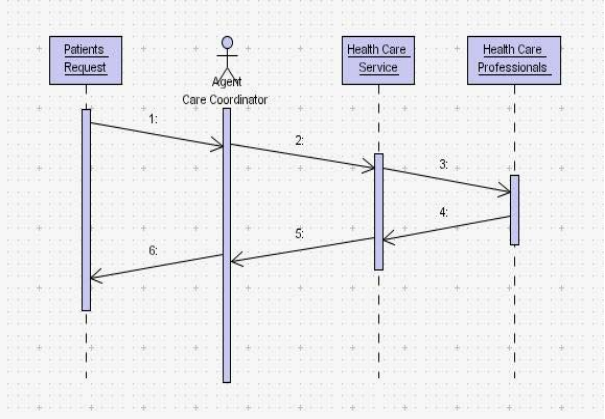

Figure5: Gaia interaction diagram: Agent Care Coordinator interacting with health care system and patients (simplified version)

• Acquaintance Diagram: This defines the desired agent domain behaviours of all the agents. It contains a set of use cases and the graphs corresponding to communication pathway among agent. For example, as showing in [Figure.6], Agent Care Coordinator is arranging an event for a patient, it could be to make an appointment with doctor or nurse or to receive a prescription. After doctor or nurses consolidation and all the health care professionals are all connected and carefully operated following the Acquaintance diagram [Figure.6].

0-7695-2870-8/07 $25.00

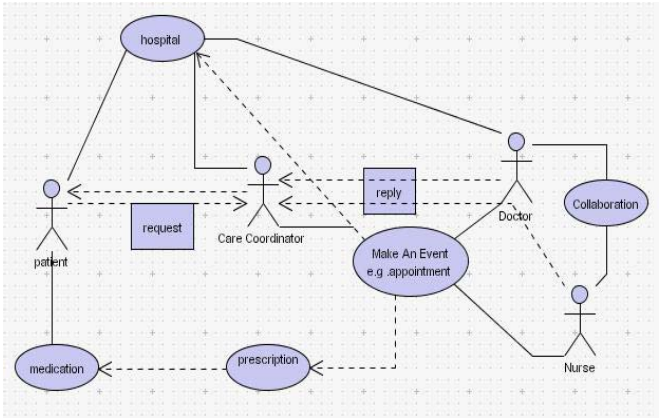

Figure6: Gaia Acquaintance diagram on event of appointment and a prescription request.

• Organization Diagram: This presents the overall structure of an agent-oriented society, i.e., the agent-based system architecture and working environment, which includes how the all the agents can be organized and managed into a single working environment while accessing databases and other shared resources. For example, here we designed a simplified version Gaia Organization diagram for a Health Care System which can be illustrated as following diagram [Figure.7]. The system can be divided into three sections, which are based on the role and functions of responsive agents: The Information Agents, i.e. Care Coordinator' Management with associated Social Care Service, are working together in the control of the system, commanding and scheduling the tasks and managing services. The User Task Interface Agents, such as GP, patients and carers, are the task providers or health care service receivers; this has a graphic interactive interface to communicate with Information Agents (Health Care Coordinator and Social Care Services). Some Task and Service Agents such as Hospital connect and cooperate with Doctors and Nurses and Care Coordinator Management Agents, and to support cooperative work.

4. The Benefits of Using Gaia Methodology and Extension

"*Agent-oriented methodologies provide the scene of new meaning, the scene of new design purposes and the scene of new ways of thinking and constructing of agents and their associated software environment*".

-W. Huang

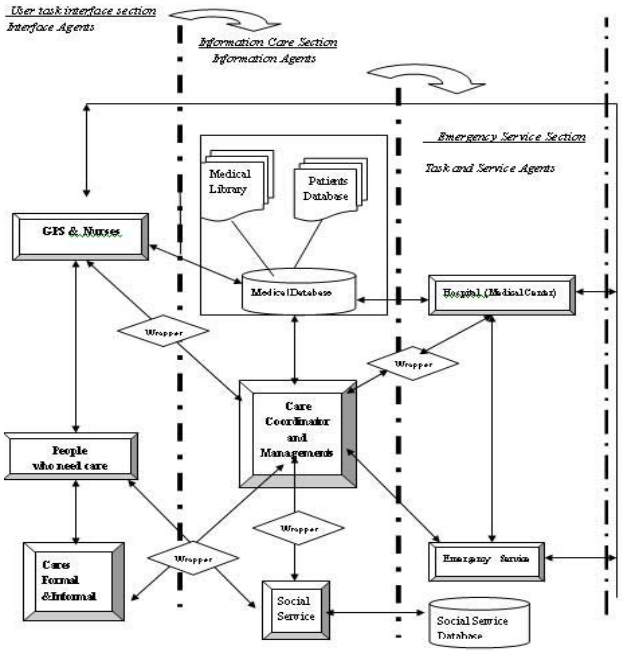

Figure7: Organizational Diagram for a Health Care System (simplified version)[4]

The Gaia methodology attempts to be a practical approach to the analysis, design, implementation, and management of agent software systems. It is an easy-to-use agent-oriented software development tool, which provides a range of model for addressing agents, services, resources, and its community. It offers practical support and guidance, which currently only covers the system analysis and design phases, while proving a range of guidelines that are directly relevant to agents in their coordinative working domain. With the proposed extension to Gaia methodology, i.e. Gaia-based modelling tools, the developer could easily put Gaia's analysis and design from text description into a live graphic design. Here, we suggest using Gaia methodology with supportive extensions: Gaia-based practical design and modelling tools to practical agents and their associated systems' design and development and system management. The numerous advantages and positive characteristics of applying methodology to agent design can be addressed as follows:

• **Inferential capability**: the ability to act on abstract roles and task specifications from perception model to action roles model. The derivation of macro-level descriptions from the problem specification, then mapping of the macro-level description into the micro level interaction specifications using adequate operation and formulisation;

• **Adaptable structure**:Gaia's modelling and structure are terse, the system is easy to operate and reconfigure to real–end user and developer requirements, to enable

re-usage of models, components and roles. Gaia offers a suitable description of the micro-level interactions providing a principled model and implementation of the targeted multi-agent system;

• **Affordable Communication**: Gaia provides built-in agent conversation role or models for agent inter-connections and interactions in common agent environments;

• **Scalable System design**: Gaia with extended graphic design tools offers a set of powerful and functional methods and roles to expand support capability and provide proactive agent services knowledge level system management with highly flexible agent architecture.

5. Discussions

Discussion1: Agent-oriented methodology v Agent-based UML(AUML)

In recent years, many researchers have been considering using Agent-based UML(AUML) and adapting UML tools and techniques for agent-based software system[3,7]. From a software development point of view, agent-based UML and agent-oriented organisational methodology both are based on the agent software development process and use their own syntax and notation to achieve the domain tasks and to create rational agent architectures. The most important of these limitations of using AUML are: firstly, AUML mismatches with agent concepts and incompatible with complex agent-based system design, it lacks of consideration for identifying agents and their associated flexible roles; and secondly it short of support and description for agents' social skills and interactive dynamic behaviours. Therefore, it is emphasized the importance of methodology as a critical initial step in developing and producing agent-based architectures and applications. It aims to present an effective schedule and plan for an agent software development process; for addressing complex agent-based environments, decomposition, abstraction, communication, organization software development process activities characteristics[1,8], whilst reducing the complexity of the complex agent systems' design and development.

Discussion2: Strengths and Weakness of the current agent-oriented methodology

a) Strengths of the agent-oriented methodology

There are many strengths of the agent-oriented approach for design software systems, which is why it has become so popular. Agent systems have great potential for complex applications such as community care and emergency services domain. The strengths of an applied methodology (such as Gaia) for agent-oriented software design and development are:

- It provides an engineering approach to the design and construction of collaborative agent systems.
- It provides a set of guidelines whereby designers and developers are involved in the whole planning process from information collection to development of action plans and management.
- It handles the complexity of the software development process increasing the quality of the resulting system.
- It attempts to be a practical approach to the analysis, design, implementation, management of agent software systems. Agent-oriented software development methodologies normally provide a range of models for addressing agents, services, resources, and their community and interaction
- It delineates between generic agent-level and domain-level problem-solving issues. It helps developers see the involvement of agents and understand the working of multi-level development phases and how decomposition of a complex domain problems into small sectors.
- It offers high-abstraction level specifications and a range of capable structural models for systems analysis and design. It offers practical support and guidance, which currently only covers system analysis and design phases, while proving a range of guidelines that are directly relevant to agents in their coordinative working domain[4].

Those efforts have resulted in a number of architectures and frameworks for agents, but there still has been relatively little systematic research, which addressed the issues of practical methods for the design of agent-based systems.

b) Weakness of the agent-oriented methodology

Agent-oriented software engineering is a technology still in its early stages of development. The research described here has involved the investigation and extension of the Gaia methodology for description, design and extension of software architectures based on the agent-oriented software engineering approach. It can be considered as a research foundation partly from the many aspects of intelligent agent technology. The agent-orient approach must overcome the challenge of system complexity and to provide vital support to agent-oriented software engineering. Currently, there still are gaps in the functionality of the design methodologies and toolkits. We need more powerful tools and better methods and methodologies to assist us to organize and control the agents and to manage resources, so that all the information and services that we need are accessible to us.

6. Conclusion

As we are already living in a world of cyber culture that is becoming increasing distributed and service oriented, and this is reflected in the systems we design, construct, build and use. Agents will continually change the way people live and work. Research and development on intelligent agents is dramatically increasing in importance. It is believed agent-based systems are new technologies that automate the process of linking constituents and their core competencies quickly and effectively on the Internet. Important aspects of this development include: the need to use agent-oriented architectures and methodologies to create flexible Internet-based applications and services more quickly and effectively than existing methods, to reach new platforms and to change relationships dynamically. It is also important to understand that agent technology is still in its infancy. Current agent-based design techniques are not sufficiently developed to enable design of fully functional, quality software in commercial or industrial software production environments. Before these techniques mature, it would be more practical and applicable to work on a combination of the established design methods e.g object-oriented approaches[1], to support and aid in the agent-oriented software development process; We believe this communication makes for a feasible approach to realistic software development and helps to exploit the productivity and potential of agents and agent-based systems.

The research "Extending the Gaia Methodology for the Design and Development of Agent-based Software Systems" is to enhance Gaia and improve modelling techniques to articulate agents' flexible behaviours, their availability, scalability, productivity, and potential and to complete and cover all the aspects of agent systems' analysis, design, organization, and management. The exploration of methodologies and system modelling for agent-based computing is not only allowing us to find effective ways of understanding and organizing agent environments, but it also gives us a good opportunity to discover in-depth some useful methods of designing collaborative agent-oriented architectures and practical distributed information system applications, so that we can face the challenges and pressures of ever greater complexity of complex software development. As a result of this, advanced agent-based systems and open environments will be developed and implemented very soon.

7. Acknowledgements

We would like to express our gratitude to all the participant researchers of Health Care project at The University of Liverpool and University of Westminster, for efforts and assistance. Special thanks to Dr. Peter McBurney for his advice, support and encouragement; thanks go also to Professor. Michael Wooldridge for the Gaia methodology and the permission of using and extending Gaia methodology for Agent-based Health Care System's creation and development.

8. References

[1] Booch, G., *Object-Oriented Analysis and Design with Applications*, second edition, Addison-Wesley, MA.,1994, pp.83

[2] Beer, M.D., Huang, W., Hill, R.and Sixsmith, A., *Using Agents to Promote Effective Co-ordination in a Community Care Environment.* Chapter 3, in Ye, Y.and Churchill, E (Eds):Agent Supported Collaborative Work, Kluwer Academic Publishers, USA,2003, pp.53-78.

[3] Garica-Ojeda,C.J and Arenas,.E.A *Extending the Gaia methodology with Agent-UML*, paper in conference proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3 (AAMAS'04) , New York, U.S.A, 2004. pp1456-1457

[4] Huang,W.,El-Darzi,E, Chountas,P& Liu.,P, "*Agent-based Models for Community Care Systems Analysis and Design*", paper in conference proceedings of the 3rd IEEE International Conference on Intelligent Systems (IEEE-IS2006),London,U.K, 2006. IEEE's E-proceedings Catalogue Number: 06EX1304C,pp273-283

[5] Hussmann, H. Formal Foundations for Software Engineering Methods. Springer-Verlag , 1997, pp67-69

[6] Jennings, N.R, *Agent-based Control System: why are they suited to engineering complex systems?* IEEE. Control Systems Magazine 0272-1708/03/. June 2003 pp.61-73

[7] Juan,T, Pearce.,A and Sterling,.L, *Roadmap: Extending the Gaia methodology for Complex Open Systems*, paper in conference proceedings of the conference Autonomous Agent and Multi-agent System(AAMAS 2002), Bologna ,Italy 2002.

[8] Sommerville,I., Software Engineering, seventh edition, Addison-Wesley Publisher Ltd, England, 2004.

[9] 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):285-312, 2000