Application of Agent Methodology in Healthcare Information Systems

Reem Abdalla [1], Alok Mishra [2]

[1] *Department of Modeling & Design of Engineering Systems, Atilim University, Ankara, Turkey*
[2] *Department of Software Engineering, Atilim University, Ankara, Turkey*

Abstract: **This paper presents a case study to describe the features and the phases of the two agent methodologies.**

The Gaia methodology for agent oriented analysis and design, Tropos is a detailed agent oriented software engineering methodology to explore each methodology's ability to present solutions for small problems. Also we provide an attempt to discover whether the methodology is in fact understandable and usable. In addition we were collecting and taking notes of the advantages and weaknesses of these methodologies during the study analysis for each methodology and the relationships among their models.

The Guardian Angle: Patient-Centered Health Information System (GA: PCHIS) is the personal system to help track, manage, and interpret the subject's health history, and give advice to both patient and provider is used as the case study throughout the paper.

Keywords: **Agent Methodlogies,GA:PCHIS, Gaia, Tropos**

1. Introduction

Agent technology has appeared with several new basic entities such as agents, roles, behaviors, ontology, etc. Assembled in a correlated models and stages approach [5], agent-based systems have the potential to offer flexibility, improved functionalities, robustness, reliability, and security compared to traditional information systems[10][11][12][13]. Because there is no unified approach that can be used to design and implement agent applications as Unified Molding Language - UML in the area of Object Oriented - OO analysis and design, Agent Oriented Software Engineering - AOSE methodologies can further provide methods for evaluating and comparing agent-oriented approaches to assist practitioners in selecting among the available alternatives [14].

Health care information systems have become more and more computerized. A large amount of data in this sector needs to be stored and analyzed, and with the help of computer systems, this task can be done faster and more efficiently. In this paper, a design has been developed for the Guardian Angel (a patient-centered health information system) [1], using two agent methodologies, Gaia [3] to provide the designers with a modeling framework and several associated techniques to design agent-oriented systems. And, Tropos [4] which is an agent-oriented methodology that guides the development of collaborative agent systems. It covers a wider range of software development lifecycle activities by applying theses methodologies to the same application. This type of methodology is based on the concept of agent and Multi Agent System MAS which are said to be better than the other types. It produces social level abstractions, which are suitable for both the agent and the MAS [7, 8, 9].

2. Selecting methodologies

There are numerous methodologies for developing agent-based systems in the literature. As such, a multi-stage selection approach needs to be adopted [15]; the initial stage reduces the set of methodologies by using the following criteria:

Documentation: The document of methodology needs to be clear and described in more detail, it should be presented in books, journal papers, or detailed technical reports rather than merely as a conference paper.

DOI: 10.18421/TEM61-22
https://dx.doi.org/10.18421/TEM61-22

Corresponding author: Alok Mishra, Department of Software Engineering, Atilim University, Ankara, Turkey
Email: profalokmishra@gmail.com

Maturity: The selected methodology needs to have a wide area of usage and refinement over time, including application by people other than its authors and their colleagues/students.

Tool support: Methodologies that have supporting tools rather than those that do not are preferred. When applying a selected methodology to design a system [16], the availability of tool support is a practical advantage. Two methodologies are selected- Gaia [3] and Tropos [4] - designed specifically for agent-based system development and which stand out as meeting the criteria stated.

3. Motivation and Objectives

The main aim of this paper is to achieve greater functionality and flexibility in such type of systems. It is an attempt to build an agent information systems center by applying the Guardian Angle: a patient-centered health information system based on the two agent oriented methodologies shown in this paper. The system is constructed as software agents representing the hospital (GA-Hospital), the family members at home (GA-Home), the patient being monitored (GA-PDA), and the health care providers (GA-Health care provider).

4. Guardian Angel: A Patient-Centered Health Information System

There is a number of examples of software agent applications and each of the two selected methodologies has been used to analyze and design the Guardian Angel: Patient-Centered Health Information System (GA: PCHIS) as another agent-based application designed and also what forms the basis for the case study component of our research. The patient receives monitoring and health care services from different providers available in the form of hospitals, physicians, nurses, pharmacies, laboratories, clinics, emergency centers, consultants, etc.

Today's health care systems are highly inefficient, slow and expensive. There are many problems that the patients face, beginning with health care records to specialists' referrals and prescriptions. These problems are related to the control of the expenses as well as the low quality of the health care provider systems [1]. The goals of the Guardian Angel System Project are building information systems to focus on the patient instead of the provider. The set integrates all the concerns of health-related clients, including legal, financial, and medical information relevant to the individual. The system is capable of maintaining comprehensive medical records as cumulative, correct , and coherent data which can be reached in a timely manner [6].

5. Gaia Methodology

The five models generated by Gaia [3] methodology are examined in two stages: the first is the analysis stage, and the second is represented in the design phase. Below is a detailed description of each development stage.

5.1. Analysis Stage

This stage covers the roles and the interactions to form a conceptual model of the GA: PCHIS. Different roles are available in any MAS, the main aim of this model is to distinguish the system roles, to define these roles and clarify their functions. Based on a number of protocols, these roles interact with each other to achieve the main goals of the system. Each protocol defines the purpose, the initiator, the responder, the inputs, the outputs and the processing during the interaction. Figure 1. shows an example of two protocol definitions: one for 'querypatientpreference' and the other for 'replypatientoptions'. The figure illustrates that the protocol querypatientpreference is initiated by the role PDA and includes finding out about the patient's preference. The second protocol, on the other hand, includes the protocol replypatientoptions and the role PDA is now the responder.

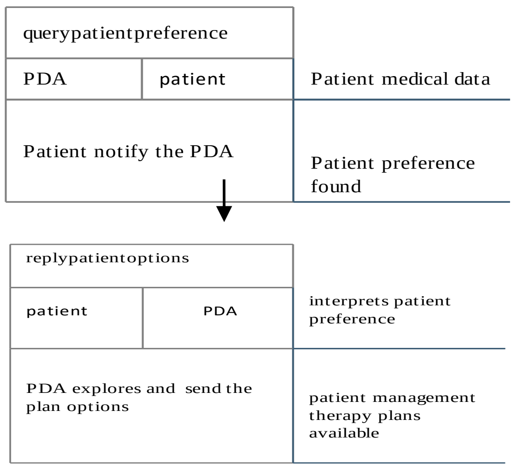

Figure 1. The querypatientpreference and replypatientoptions Protocol Definitions

In the last step of the Gaia analysis phase, one can define the main roles identified in the first step. This process involves the identification of permissions of roles and their responsibilities in addition to the protocols and activities in which they participate. This detailed description of a role is depicted by a Role Schemata. A set of Role Schemata forms the Role Model, which is considered as the key artifact of this analysis phase.

There are two forms of functionality when it comes to the issue of responsibility: liveness properties and safety properties [17]. Permissions define which resources the agents playing that role can and cannot use when performing a particular action. Activities are "private" actions which do not involve interactions with other roles. Protocols are actions dealing with interactions related to other roles. They are taken from the protocol model which is constructed in the step before. Figure 2. shows a role schema for the role PDA.

ROLE SCHEMA: PDA
Description: This role presents management therapy plan options, monitor and customize treatment
Protocols: get patient data **involves** Store patient's information. Data sorting. Data schedule Monitor treatment **involves** monitoring tests results. Interface with devices. Get results from patient &Get information about meals and exercise & Assessment treatment compliance. Suggest changes in diet or exercise. Suggest changes in treatment plan Data Update **involves** update results. Update changes in medicines
Permission Reads Patient's condition Reads Patient's preference Monitors progress Changes plan follow Communications Allows patient to customize therapy plan
RESPONSIBILITIES Responsibilities: Liveness: PDA = $(All)^{\omega}$ PDA = get patient data, Monitors progress of patient condition, Data Update Safety: Consistent (patient's medical information) Consistent (patient's profile)

Figure 2. Role schema for role PDA

This role involves detecting and interpreting the facts and medically-related plans and options depending on the patient preference and available activities. There are several protocols associated with this role, such as querypatientpreference, replypatientoptions, etc. This role has the permission to read the patient's condition, monitor the progress, and allow the patient to customize the therapy plan. The liveness property of the role PDA indicates the sequential execution of its associated protocols and activities in patient treatment.

5.2. Design Stage

All analysis models are carried into the design phase. During this process, they are updated to reflect the design decisions. Three design models - the agent model, the service model and the acquaintance model - are created from the updated analysis models. These models are refined iteratively until sufficient design information is finally attained. The agent model is the first step in the design process when map roles are identified within the analysis process. In the design phase, this model is created by assigning roles to agent types, a role can be mapped to one or more agent types. For every agent role, a number of related services can be specified as to properties such as inputs, outputs, pre-conditions, post-conditions. The inputs and outputs are derived from the protocol model. The pre- and post-conditions which define the constraints on the services are derived from the safety properties of a role. However, to assign the relationship between multiple roles and one agent is not trivial. Many design quality factors need to be considered [18]. Software quality has long been a critical issue for software developers [19]. An example of the Service Model is shown in Table 1. The GA-PDA agent playing the PDA role has three main services: receive patient data, monitor treatment, and update data. Each of these services is described with its inputs, outputs, pre-conditions, and post-conditions.

Table 1.Service Model for GA-PDA agent

SERVICE	INPUT	OUTPUT	PRE-CONDITION	POST-CONDITION
Get patient data	patient-data	patient-preference	patient-preference=nil	(patient-preference=nil) V (patient-preference!=nil)
Monitor treatment	medical-data	alternative treatment	alternative treatment=nil	(alternative treatment=nil) V(alternative treatment!=nil)
Data Update	recent data available	data-processing	data-processing=nil	(data-processing=nil)v (data-processing!=nil)

6. Tropos methodology

Tropos methodology [4], agent-related concepts such as goals, plans, tasks, etc. are included in all of the development phases with the purpose to cover a wider range of software development life-cycle activities. The work stages in this methodology are divided to five stages: early requirements, late requirements, architectural design, detailed design and implementation phase. A main difference between Tropos and others is the emphasis put upon analysis of early requirements.

6.1. Early Requirements

The main objective of early requirements analysis is to determine the stakeholders in the target system and their intentions. To model stakeholders and intentions respectively Tropos uses the concepts of actors and goals. The goals are divided into two different groups. In the end, hardgoals result in functional requirements, whereas softgoals deal with non-functional requirements. There are two models that represent them at this point in the Tropos. Firstly, the actor diagram describes the stakeholders and their relationships in the domain. Secondly, the social dependencies that depict how actors depend on each other for goals to be accomplished, plans to be executed, and resources to be provided. Next, the goal diagram displays goals and plans analyzed in terms of the specific actor in charge of realizing them.

Figure 3. shows the dependency of GA: PCHIS to provide information (hard goal). It also needs a usable GA: PCHIS (soft goal). These goals are then decomposed into sub-goals. For example, the goal 'provide information' is fulfilled by the composite achievement of two sub-goals 'search information' and 'display output'. The sub-goal 'search information', in turn, has several sub-goals such as 'access to patient data', 'access to physician data' and 'access to pharmacy data'. The soft goal 'easy to use' is also shown. A user-friendly interface that offers simplicity and provides guidelines also realizes the goal 'easy to use'.

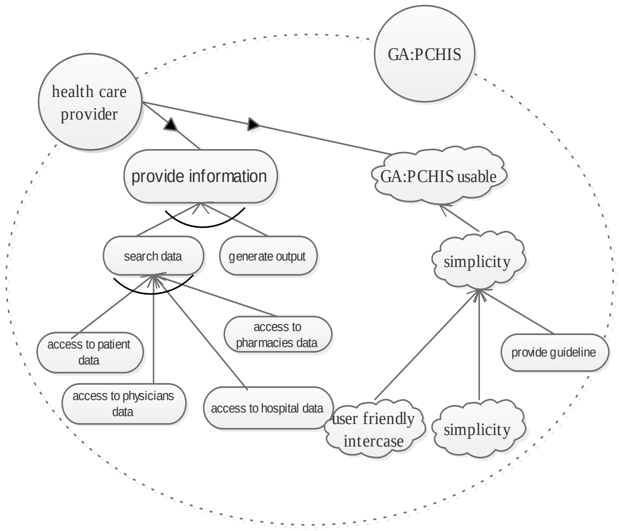

Figure 3. GA:PCHIS Goal Diagram

6.2. Late Requirements

Late requirements stage modeling the target system in its environment. The system intended for use is modeled in the form of one or more actors. The dependencies of these special actors are also identified by following a similar process to that used in the Early Requirements phase. In reality, such dependencies clarify both the requirements regarding the functional and non-functional aspects of the system.

6.3 Architectural Design

During the architectural design, Tropos methodology determines three steps which system developers can apply in this phase. The purpose of the first step is to define the comprehensive architectural organization of the target system; whereas the second step includes identifying the capabilities required by the actors to achieve their goals and plans. In the final step, agent types are defined together with the allocation of functionality to be assigned to them. Figure 4. shows the decomposition in the sub-actors of the target system and the delegation of some of the goals from the GA: PCHIS to them.

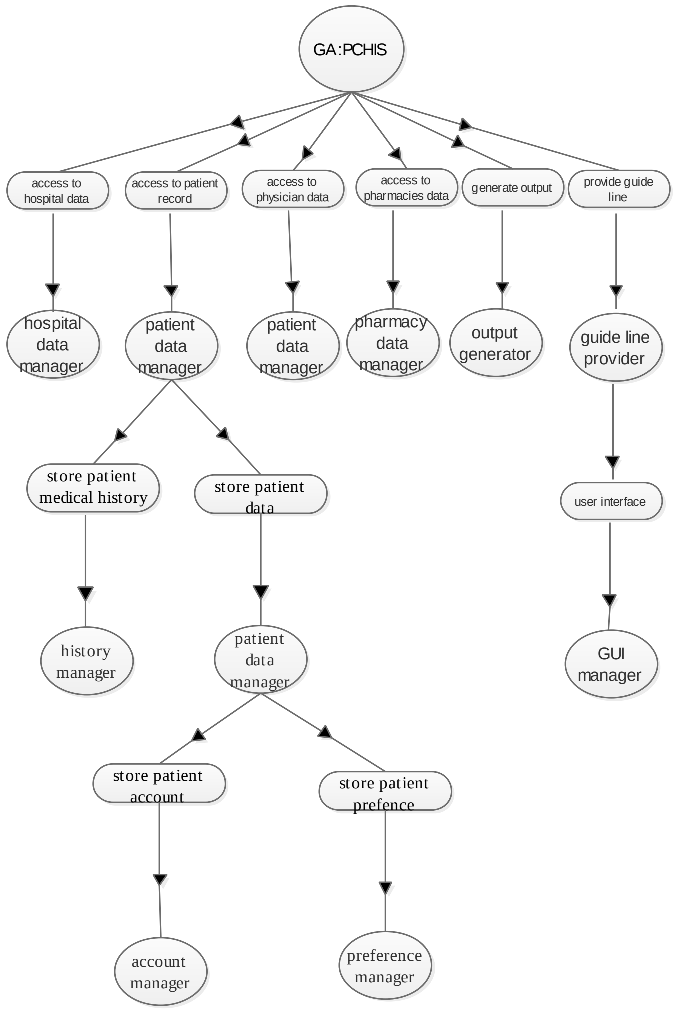

Figure 4. Actor diagram for the GA:PCHIS architecture

The system depends on the patient database manager to have access to the patient database, the hospitals database manager to have access to hospitals data, the output generator to generate output, and so on. In addition, each sub-actor (e.g., patient Info Manager) can itself be decomposed into sub-actors (e.g., account manager and preference manager) responsible for the achievement of one or more goals (e.g., store patient account and store patient preference).

7. Results and Discussion

7.1 Gaia

In the present work, the first methodology used was Gaia in the form of its most recent and updated definition as appears in [3]. It focuses on building agent systems and emphasizes the societal aspects of an agent system.

The steps in Gaia were found to be easy to follow. Nevertheless, some of these steps are not clear as to the motivation. In the analysis phase, the two models are created to allow conceptualization of the system as an organization.

The role model is also applied where the one in Gaia was found to be fairly well-described as it consists of a set of role schemes for each role in the system. The roles define expected behaviors of the agents and provide verification services to any authorized agent at run-time. The role model allows us to construct an abstract view of the organization according to which its member agents behave. During the design phase, the initial conceptualization of the system is refined for the chosen quality goals, namely performance. The original models are improved and modified to reflect the design decisions. In addition, three design models (agent model, service model and acquaintance model) are added to populate the updated organization with member agents.

Later, all the analysis models are moved into the design stage. For instance, the role model is re-factored for the chosen quality goals where the agent model is built along with assigning roles to agent types. The target system agent type is described which takes the GA-PDA role and implements the protocols and activities of the role.

Gaia does not consider implementations and there is no tool support, implying that there are some notations necessary to be described clearly. The analysis process needs many iterations in order to further improve model consistency, links between models and keeping track of the changes made.

7.2 Tropos

The Second methodology used in this study was Tropos. It generates early requirements to cover many aspects of agent-oriented requirements not previously provided by other approaches. The developing process consists of four stages: the early requirements analysis phase is the first step for requirements engineering techniques in order to identify the goals and resources, plans as well as tasks to accomplish these goals. However, there are some difficulties at this phase. For example, in some situations it was not possible to distinguish between the goal of mean-end analysis AND/OR decomposition.

This confusion resulted from the fact that Tropos usually considers AND/OR goal decomposition as a special case of mean-end analysis. In the late requirement analysis stage, a system actor is provided whose purpose is to provide system operational services to actors depending on services from the last analysis stage. However, when a new sub-goal is generated from the goal discipline, dependencies among the new goal and every actor in the GA: PCHIS have to be recalculated.

Using the Tropos to complete the design gave the authors a positive impression of the methodology as more clear scenarios of agents are depicted. The notations used throughout the analysis and design phase help preserve semantic mapping.

8. Conclusion

In this case study, the target system is applied on two agent-oriented methodologies Gaia and Tropos in the analysis and design of the Patient-Centered Health Information Systems to deliver an integrated health care evaluation model for patients' needs. In our view, both agent-oriented methodologies can cope well with the challenge of analyzing and designing the GA: PCHIS system, and they are clearly agent-based, the designs are widely regarded as agent-oriented methodologies, and both consider the social aspects involved in the matter. The most commonly addressed social features in the two agent methodologies provide strong support for most social criteria as communication, language and relationship. Furthermore, distinguishing features exist such as an early requirements stage in Tropos which guides practitioners to take advantage of richer requirements enabled by agent technologies and their implementations. However, testing and maintenance are not clearly supported by either one of the two methodologies.

References

[1] Ericsson Nikola Tesla d.d. Technical Description, Integrated Health Care Information System. Croatia, April 2004.

[2] Juan, T., Pearce, A. and Sterling, L. Extending the Gaia Methodology for Complex Open Systems, Proceedings of the 2002 Autonomous Agent andMulti Agent

[3] M. Wooldridge, N.R. Jennings, and D. Kinny. The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems, 3(3), 2000.

[4] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, A. Perini, Tropos: an agent-oriented software development methodology, Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS) 8 (2004) 203–236.

[5] Michael Winikoff. Future Direction for Agent-Based Software Engineering RMIT university, Australia and university of Otago,New Zealand.2013.

[6] Szolovits, P., Doyle, J. and Long, W.J. Guardian Angel: Patient-Centered Health Information Systems , Tech Report MIT/LCS/TR-604.2004.

[7] Iglesias, C.A., Garijo, M. and Gonzalez, J.C. A survey of Agent-oriented Methodologies.In: Muller, J.P., Singh, M.P., Rao, A. (eds.): Intelligent Agents V (Atal'98), LNAI 1555. Springer-Verlag, Berlin ,1999.

[8] Tveit, A. A survey of Agent-oriented software engineering. First NTNU CSGS, 2001.

[9] Weiss, G. Agent oriented Software engineering. Knowledge Engineering Review, Vol. 16(4) ,2002.

[10]Faezeh,P. Evaluating Agent-Oriented Software Engineering Methodologies,IEEE Internatinal Workshop on Soft Computing Application.21-23 August 2007.

[11] BradShaw, J. Introduction to Software Agents. In J. BradShaw, editor, Software Agents, pages 3-46. AAAI Press, Menlo Park, California, 1997.

[12] Wooldridge, M., Jennings, N. R. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115-152,1995.

[13] Hoa,K.D.,Michael, W.Towards a Next-Generation AOSE Methodology,*Science of Computer Programming* 78(2013) 684-694.2013.

[14] Mansura,H.,Metrics for Evaluating Agent Oriented Software Engineering Model,IEEE/OSA/IAPR International conferenve on Informatics,Electronic &Vision,2012.

[15] D. Law, Methods for Comparing Methods: Techniques in Software Development, NCC Publications, 1988.

[16] K.H. Dam, M. Winikoff, Comparing agent-oriented methodologies, in: P. Giorgini, B.Henderson-Sellers, M. Winikoff (Eds.), Agent-Oriented Information Systems, AOIS, in: Lecture Notes in Computer Science, vol. 3030, Springer, 2004, pp. 78–93.

[17] M. Wooldridge, N.R. Jennings, and D. Kinny. The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems, 3(3), 2000.

[18] Yu, L., Mishra, A. Multiple-Role Agent Based Distributed Computing, *Technics Technologies Education Management**, Vol. 8, No. 1, 2013, 238-243.

[19] Mishra, D. and Mishra, A. *Simplified software inspection process in compliance with international standards*. *Computer Standards & Interfaces*, 2009, 31(4): 763-771.