Evaluating Methodologies: A Requirements Engineering Approach Through the Use of an Exemplar

Luiz Marcio Cysneiros[1], Vera Werneck[2] and Eric Yu[3]
[1]*Dept. of Math. & Stat.- Information Technology Program*
York University – Toronto – Canada
cysneiro@.yorku.ca
[2]*Informatic and Computer Science Department*
Rio de Janeiro State University (UERJ) – Brazil
vera@ime.uerj.br
[3]*Faculty of Information Studies – University of Toronto - Canada*
yu@fis.utoronto.ca

Abstract. Systems development methodologies continue to be a central area of research in software engineering. As the nature of applications and systems usage move increasingly towards open networked environments, not only are new methodologies required, but new ways for evaluating methodologies for these new environments are also required. The agent-oriented approach to software engineering introduces concepts such as pro-activeness and autonomy to achieve more flexible and robust systems for complex applications environments. A number of AOSE methodologies have been proposed. In order to evaluate and compare these methods in depth, we proposed the use of a common exemplar – a detailed application setting within which each of the methodologies will be worked out. The evaluation method emphasizes a requirements engineering perspective. In this paper we show how to apply this exemplar to evaluate three agent-oriented methodologies.

Keywords: Agent-Oriented, Systems Development Methodology, Requirements Engineering, Evaluation

1. Introduction

As software becomes increasingly part of everyday life, traditional conceptions of software are being extended. In earlier conceptions, software information systems are conceived of as automating routine processes, as maintaining data in databases, or as reactive and interacting objects. The emerging agent-oriented paradigm conceives software as being proactive and exhibiting autonomy and sociality. This orientation parallels the shift in application and usage environments towards open networked environments, both in terms of technical systems and in the embedding human social organizations and institutions.
For example, healthcare quality and cost-effectiveness can potentially be greatly improved by effective use of information technology on a large scale. Agent-based systems have the potential to offer greater flexibility, enhanced functionalities, and better robustness, reliability, and security, compared to conventional information systems. Patients, family members, and healthcare professionals in hospitals, clinics, pharmacies, and so on, could be supported in their interactions and decision making by various kinds of software agents personalized to meet their information and communication needs. Agent oriented methodologies can offer the higher level of abstraction needed for this new conception of software.
In these more complex application settings, humans, hardware, and software interact in much more intricate ways then in conventional systems which automate routine work processes. A critical factor in the successful development of such systems is therefore the understanding of stakeholder needs and wants, how technologies might alter their relationships, facilitation of their negotiations, and communication of those needs to system developers. In assessing systems development methodologies for these more challenging types of environments, we need to raise new questions that were not considered in assessing conventional methodologies – for example: how well does the methodology support reasoning about autonomy and pro-activeness during the early stages of requirements elicitation?
One way to help clarify strengths and weaknesses of each methodology is to define a suitable example problem that can be used as a common example providing a stable and coherent base for discussion and exchange of ideas and results. This type of example is commonly referred as an "exemplar".
We have recently proposed the use of an "exemplar" [25] for evaluating methodologies. Differently from other work such as [21],[20],[9],[7],[10],[15], our main goal does not rely on measuring one methodology against others using pre-defined metrics. The exemplar primarily aims to be used by methodology developers to understand where their methodologies strengths and weaknesses lie. The exemplar also aims to help people to deeply evaluate different methodologies and therefore evaluate how well each would suit their needs. It may also help methodology developers to better contextualize their work towards other proposals.
Another distinction is that, differently from other exemplars such as [18],[12],[23] this exemplar is intended to be rich and complex enough to test the methodology to its limits. It focuses on a single problem from the health care domain embodying real-world issues and challenges. It was designed to be neutral regarding any methodology one might be testing. The exemplar can be found at [2]. By having such rich and complex example we expect to be able to deeply evaluate each methodology on complex properties that could be otherwise unfairly judged. For example:

- How is agent autonomy supported by each methodology? – Using a complex problem many times we may face challenges to cope with all the autonomy involving both human and software agents.
- How are Non-Functional requirements (or quality attributes) addressed in each methodology? – The exemplar presents real world challenges such as privacy, security and safety, which are critical in health care.

- How is sociality supported by each methodology? – A rich and complex exemplar may better expose problems regarding sociality that would have been missed if a simple example with few participants would be used.

Since many of the challenges introduced by new agent oriented concepts are directly related to requirements engineering, we think this exemplar could help on answering one important question: How effectively each methodologies helps to handle requirements elicitation, specially having aspects such as those mentioned above in light?

As mentioned before, in this work our primary objective is to show how an exemplar can be used to reveal most of the strengths and weaknesses of a methodology regarding requirements elicitation. We applied the exemplar to three different methodologies: Message [5], Gaia [26] and Tropos [3]. We chose to use these methodologies because all three aim to address all the phases of the software development and are well known. Based on our findings, we will present a summary of the strengths and weaknesses of each methodology regarding suitability to perform requirements elicitation. We will also present a glimpse of how some questions were answered and how they helped on the evaluation of each methodology. This paper highlights some interesting findings gathered during the application of the exemplar.

We start by detailing the research methodology we used for applying the exemplar to the three methodologies. Section 3 presents a sampling of answers to the evaluation questions of the exemplar together with some findings for each methodology. Section 4 presents some of the key findings common to the three methodologies. Section 5 concludes the work.

2. Research Methodology

In this work we applied the exemplar to three methodologies, MESSAGE [5], , [11], , , [4], [17], [19] , Gaia [26], [23], [24] and Tropos [3], [6], [16], [13], [14], [1]. Here, we have applied the exemplar using only the questions related to requirements elicitation.

The exemplar is based on the Guardian Angel Project [22]. A set of "guardian angel" software agents provide automated support to assess patients with chronic diseases such as diabetes or hypertension, integrating all health-related concerns, including medically-relevant legal and financial information, about an individual. The exemplar builds on software agents representing the hospital (GA_Hospital), the family members at home (GA_Home) and the patient being monitored (GA_PDA). This personal system will help track, manage, and interpret the subject's health history, and offer advice to both patient and provider. The system will maintain comprehensive, cumulative, correct, and coherent medical records, accessible in a timely manner as the subject moves through life, work assignments, and health care providers.

The exemplar is [22]expressed in terms of a set of numbered scenarios (EA0.0 until EA9.0) as the one below that illustrates those scenarios:

EA4.0-*Abby is uncertain what insulin dose to give this morning as she has a double session dance class at 10:00 and she remembers all too well that she has had mild hypoglycemic symptoms towards the end of even single session dance classes. She draws an exercise symbol spanning 10 to 11:30 on her daily schedule on the GA_PDA interface and then selects the Advise Dose icon. The GA_PDA informs her that she can either keep the dose unchanged if she thinks she can manage a double carbohydrate snack before the dance class or she can reduce her morning dose of insulin by two units of short acting (regular) insulin.*

The exemplar also provides a set of evaluation questions aimed to help evaluating how well the methodology supported the modelling of the set of scenarios. Table 1 shows the types of concerns addressed by each of these questions. An initial concern in applying the exemplar was the extent to which we could in fact evaluate the methodologies without being biased by either the authors' knowledge of the exemplar or our knowledge of some of the methodologies. To address this concern, the evaluation was performed by someone (the second author) with extensive experience in object-oriented methodologies, but no experience in agent-oriented methodologies and with no prior knowledge of the exemplar. The exemplar was applied to one methodology at a time, starting with MESSAGE [5], then Gaia [26], and finally Tropos [3].

For each methodology the existing documentation was searched and studied. The methodology was then applied to small examples and it was finally applied to the exemplar scenarios. All necessary models were developed according to directions within each methodology. Once requirements were elicited and defined she started answering the questions proposed in the "detailed questions" section of the exemplar [25]. Figure 1 illustrates the Process.

Note that although in this work we use only agent-oriented methodologies, the same process can be used to any methodology one wants to evaluate.

Each Question was marked using one of three possibilities: Strength, Weakness or Neutral. This aimed to facili-

QA1 – Pro-activeness
QA2- Human Autonomy vs software autonomy
QA3 - Autonomy reasoning
QA4 - Different levels of Abstraction
QA5 - Identifying participants in the domain
QA6 - Capturing, understanding and registering terminology
QA7 - Domain analysis
QA8 - Finding requirements
QA9 -Human-machine cooperation
QA13 - Reasoning about different non-functional aspects
QA15 - User interface design
QA19 - Eliciting and reasoning about Non-Functional aspects
QA28 - Formal Verification and Validation
QA31 - Tool support
QA32 - Learning curve
QA33 - Integration with other methodologies
QB7 - Lightweight versions of methodology for simpler problems

Table 1 – Issues addressed by Exemplar Evaluation Ques-

tate identify later the strengths and weakness of each methodology. Finally, the answers were complied in order to evaluate how well the exemplar helped us on evaluating different methodologies. Section 4 will summarize the findings.
The Guardian Angel Project was chosen as the basis for the exemplar for being a complex problem that encompasses many of the needs systems present today. It pushes methodologies to deal with problems such as: distribution, privacy, autonomy, pro-activiness and sociality. Furthermore, being a comprehensive problem, it enhances the chances for the exemplar to expose strengths and weaknesses of the methodologies.

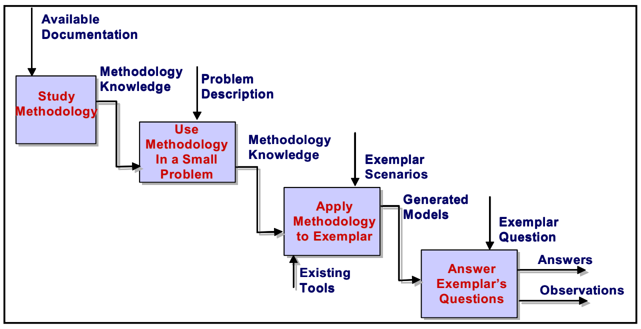

Figure 1. SADT for the Research Methodology

3. Applying the Exemplar

For each methodology, we applied the exemplar as described in Section 2. In this paper, due to space limitations, we show only a sampling of the evaluation questions to illustrate our approach. In this section, we describe, for each methodology, the models we constructed, answers to selected questions, and some observations from the experience in applying the exemplar and the evaluation questions to that methodology. Each question is shown in italics followed by the answer. The methodology documents will be available soon at the exemplar web site [2].

3.1. MESSAGE

The first methodology used was MESSAGE. We used the most recent definition that will appear at [5] and also the documentation found in MESSAGE website [24.For the analysis phase we were based in [11] describing in detail how to model the analysis level.
Message has different levels of abstraction. Level 0 is mainly concerned on showing the system as an unitary entity referring to its stakeholders and environment. Level 1 departs from the models used in level 0 and refines it into an organization of interacting agents.
In Level 0, four Organization Diagrams were defined: One showing the Structural Relationships and three acquaintance relationships and one for each GA subsystem (GA_PDA, GA_Home and GA_Hospital). Organization Diagrams show concrete entities in the systems and its environment

For the Goal/Task view the Goal/Task Implication diagram and some Task Workflow Diagrams were defined. Goal/Task View shows goals, tasks, states and their dependencies among them. Even temporal dependencies can be drawn in a UML Activity Diagram.
In level 1 the Organization view which is mainly geared towards representing the acquaintance relationships was developed. We also developed the Agent/Role view which focuses on the individual Agents and Roles showing goals, events and tasks related to each agent/role. The interaction view, highlighting which, why and when agent/roles need to communicate lead to the development of several Interaction Diagrams. Finally, the Domain view which is basically a UML Class Diagram was developed.

3.1.1 Selected Questions from the Exemplar, with Responses for MESSAGE

- *QA4 "Different levels of Abstraction" - How does the methodology support navigating from the abstract levels of reasoning to the concrete one and vice-versa*?

MESSAGE provides good support for navigating from abstract to the concrete. We began the Analysis at level 0 by describing in the Organization Diagram the structural relationships of Organizations (e.g. GA_PDA, GA_Home, GA_Hospital Hospital, and Laboratory), Agents (e.g. Instrument), Roles (e.g. Patient, Physician, and Parents), Resources (e.g. Patients' Information, Management Plans) and Structures (e.g. Drugs Information). Then, we described for each GA (e.g. GA_PDA,) an Organization Diagram showing the acquaintance relationships. Those diagrams and the Goal/Task Implication Diagram (e.g. Customize Therapy, Monitor Treatment Diagnosis Assisted,) gave us an overview of the global organisations and the goals. Switching to Level 1 (sometimes level 2 or 3) we focused on the system itself, identifying the functional requirements, specifying for each GA its goals, capabilities, knowledge, beliefs and agents requirements. We also detailed the information and relationship of the Domain and defined the Interactions between the roles, tasks and the information domain classes as well as the interface between agents (software and humans). The design model identifies the agents (e.g. GA_PDA Therapy Customized, GA_PDA Monitor Treatment) assigns their roles, describes the services with their tasks (e.g. get patient condition, show therapy solutions available), and refines the analysis interactions into protocols interactions and interactions protocols behaviours. The detailed design defines the system in terms of implementation and the agent architecture. MESSAGE does not make it clear how you can come back from detailed design to level 1 or level 0. The provided procedure serves only to navigate from a high level to a detail one.

- *QA5 'Identifying participants in the domain" - In scenarios with many participants (e.g., EA1.0, EA2.0 and EA2.1), how can the methodology help identify participants such as the physician and the GA in the patient's computer?*

  The methodology has a checklist for constructing level 0 and level 1. The first step in level 0 is to identify the stakeholders by listing the potential users, others stakeholders, agents and resources were obtained by analyzing the requirements or by discussing with the customer. In fact, although MESSAGE allows modelling participants in the domain, it does not strongly support the identifications of these participants. Figure 2 give an example of the Structural Relationship of Organization Diagram at the beginning of defining requirements (level 0).

3.1.2 Observations

In responding to question *QA1 "Pro-activeness"* and *QA3"Autonomy Reasoning"* we realized that although MESSAGE supports Pro-Activeness and Autonomy, it does not compel you to use it. It is totally upon the experience of the developer to apply the concepts. In fact, a developer who has little or no experience in developing agent-oriented software is more likely to follow a functional decomposition line of reasoning.

One strength of MESSAGE was noticed while answering question QA8 "Finding Requirements". MESSAGE acknowledges the need for gathering requirements and the existence of models such as the Organization and Task Workflow Diagram help on modelling requirements, although elicitation mechanisms could be added to MESSAGE.

In evaluating QA7 "Domain Analysis", we found out that although the Organization diagram shows part of the social relationship, it does not stress it. Further diagrams would be needed to demonstrate complex relationships. There is also a lack of tools to prevent inconsistencies among the existing models.

One very strong point of MESSAGE can be seen answering QA31 ´Tool support". MESSAGE is supported by a customisation of an existing commercial CASE tool called MetaEdit+.

3.2. Gaia

Gaia was the second methodology we modelled using the exemplar. We used the Gaia extension proposed in [26]. During the analysis, the system is subdivided into sub-organizations. The environment model, the preliminary roles model, the preliminary interaction model and the organizational rules are then defined. Gaia's offers also the organization division, the environment model and the organizational rules during the analysis. They continue to perform the agents roles and the interact models, but as a preliminary way that would be refined during the Design and after the definition of the Organizational structure.

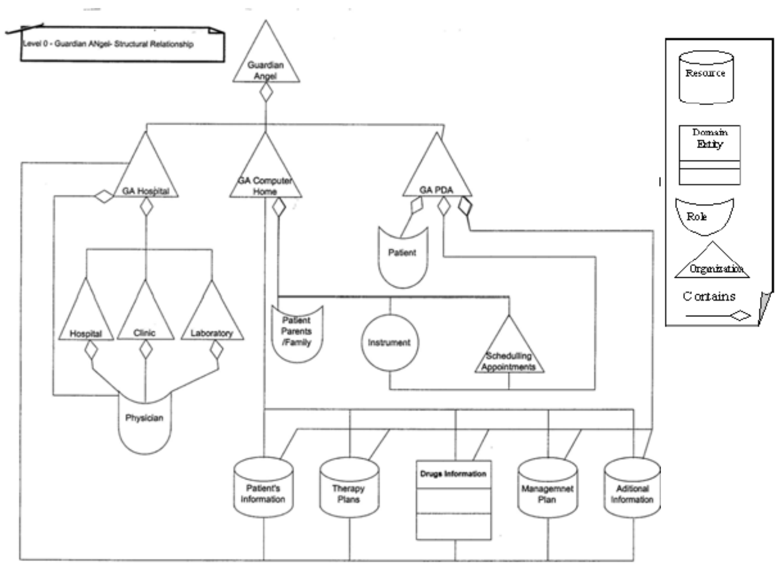

Figure 2 – GA Organization Model in MESSAGE Methodology

3.2.1 Selected Questions from the Exemplar, with Responses for Gaia

- *QA7 "Domain analysis" - GA involves complex social issues, how does the methodology support the modelling and reasoning about the social relationship involved in the above scenarios? How would they represent, for example, the fact that a patient expects to have a plan to monitor his progress established by the physician as in scenario EA2.0?*

In Gaia, to understand the social relationships one must analyze each role in both the Role Model and the Interact Model that shows the communications between the roles. The Gaia representation of scenario 2.0 was modelled by the Patient, Physician, GA PDA and GA Physician in the Role Schema Model and the Physician/GA Physician, GA Physician/GA PDA and Patient/GA PDA Communications in the Interact Model. We portray in Figure 3 the Environment Model. Although we could represent all the participants, social relationships and dependencies are difficult to model. Moreover, the lack of a graphical notation makes it harder to understand complex social environments. Thus, Gaia only partially supports this feature.

GA PDA	reads	Patient's condition,

		Patient's habits
	monitors	Management Plan
	changes	plan follow
		Communications
GA Physician	reads	Patient's information,
		Patient's condition
		Patient's exam results,
		Additional information
	changes	Plan follow
		Management plan

Figure 3. Example of Environment Model

- *QA9 "Human-machine cooperation" - The diet and exercise scenario (EA2.4) illustrates how the GA might explore alternatives to help the patient achieve therapy goals while respecting personal preferences and life styles. How does the methodology help identify and analyse cooperative problem solving scenarios?*

Protocol Name: Communication Patient-GA PDA		
Initiator: **GA Patient**	Partner: **GA PDA**	Input: **Patient informs habits and preferences and GA PDA explores alternatives to customize plan**
Description: **Patient informs the GA PDA habits and preferences and GA PDA ask more information, suggest some changes, make notes,...**		Output: **Patient Management Therapy Plan Options**

Figure 4. Example of Communication Patient and GA PDA Protocol Schema

Gaia helps by defining responsibilities (liveness) in the Role Model for each role (Patient and GA Patient roles) as well as the protocols defined in the Role Model and detailed in the Interact Model (Figure 4 gives an example).

3.2.2 Observations

One strength of Gaia comes from the definition of agents' responsibilities and permission in the Role Model. Gaia has the responsibilities for determining the expected behaviour of a role especially when describing the states of affairs that an agent must bring about in certain conditions. Using this feature facilitated modelling autonomy and architectural aspects as we realized from QA3 "Autonomy Reasoning".

The definition and discussion to construct the Organizational structure in Architecture design is another point that helped to understand agents characteristics ("Human Autonomy vs. software autonomy, QA3 "Autonomy Reasoning" and QA21 "Architectural design and reasoning".

One interesting aspect we observed about Gaia was the lack of a formal model check between the different views of the system and diagrams (QA28 "Formal Verification and Validation"). Being based on formal notations we would expect Gaia to have it available.

Another weakness of Gaia could be seen while answering QA33 "Integration with other methodologies**".** No guidance was found on how to integrate Gaia with other methodologies and/or frameworks. For example, it is not clear how one could implement part of the exemplar using object-oriented approach since there is no guideline for translating Gaia models into UML models. Moreover, Gaia does not consider implementation issues.

Another point to discuss in Gaia is the model specification. The role model expresses the agents' characteristics very well, but a graphical notation is needed to show the relationship among the agents, resources and tasks (QA31 "Tool support"). Because the exemplar deals with a complex and large subject, we had difficulties for fully grasping the ideas and concepts using only the descriptive models from Gaia. A graphical notation of some kind may improve the ability to view interactions and dependencies among different model constructs. For example, one can view the agent communications in the Interact Model, but the resources and tasks used by more than one agent can only be seen by revising the whole Role Model.

3.3. Tropos

Tropos [3] was the third methodology we evaluated. Although we mainly followed the methodology defined in the most recent work [3], we also used other sources for modelling in Tropos [6], [16], [13], [6], , [14], and [1]. In the Early Requirements we built one Actor Diagram and one Goal Diagram. Then in the Late Requirements we focused on some agents (humans and software) and we detailed the Actor Diagram and the Goals Diagrams.

3.3.1 Questions

- *QA2 "Human Autonomy vs. software autonomy" - In scenario 4.1 Abby (a human being) has the autonomy to follow or ignore advices from the GA and to modify the GA-PDA authorization to communicate with her parent's desktop computer. How would the software engineer handle this autonomy using this methodology? How does one decide which decisions are to be made at design-time and which at run-time?*

  The software engineer can handle this scenario using the Actor diagram. Showing the interactions between the patient and the GA-PDA the Actor Diagram allows showing the social relationship and the dependencies between them (Figure 5). The software engineer has to choose which decisions will be made at run-time. This can be detailed by using goals, softgoals and plans in the Goal Diagram. In the architectural design, the agents and sub-agents are defined and for each actor and agent we have to identify the capabilities. In detailed design, the plan diagram can be defined in the capability diagram and plan diagrams.

- *QA7 "Domain analysis" - GA involves complex social issues, how does the methodology support the modelling and reasoning about the social relationship involved in the above scenarios? How would they represent for example the fact that patient expects to have a plan to monitor his progress established by the physician as in scenario EA2.0?*
  This is a strong point in Tropos. By allowing one to model the dependencies among different agents and to evaluate how this dependencies are satisfied or not (using goals, softgoals, resources, agents, roles and positions) Tropos strongly help to clarify the complex social relationships intrinsic to the exemplar and Figure 5 shows how the actor Diagram can model this situation.

3.2.2 Observations

By answering question QA4 "Levels of Abstraction" we realized that Tropos support the navigation from different levels of abstraction using the same diagrams and elements of the diagrams. This feature is very powerful since one can use the same notation to express different levels of specification from early requirements to design. Working with the same kind of models throughout the whole software development life cycle facilitates the allocation of developers to different activities while facilitating the ability to partition the software into models and assign them to different teams without having to be worried about compatibility and understandability due to the use of different modelling techniques.

Question QA7 "Domain Analysis" raises questions about modelling complex social relationships, a common need for multi-agent systems. This is an important strength of Tropos. The methodology helps on reasoning about social relationships through the use of constructs such as actors, goals and dependencies.

Being a requirements driven methodology, Tropos has one of its strengths in helping developers to elicit requirements, QA8 "Finding Requirements". However, answering QA31 "Learning Curve" we see that there is a price to be paid for some of the Tropos strengths. Although it can be considered a lightweight methodology with not too many different constructs, the syntax and semantics behind the constructs reveals not to be as easy to follow as it appears to be at first glance. Therefore, one can spend more time learning how to use the methodology than one may expect.

4. Key Findings

Applying the same example to different methodologies allowed us to better visualize shortcomings and strengths in the methodologies. It also helped us to determine that some methodologies have better solutions for modelling the same concept than others.

The exemplar was also very successful in stimulating the methodologies to their limits. Being a realistic and complex problem, it allowed us to verify how important it is to use some of the constructs used by agent orientation to cope with complex system. Notably, sociality, pro-activeness, human and software autonomy were very important to completely model the exemplar.

In applying the exemplar, it became clear that all three methodologies still have some work to do in order to achieve a mature state where they could be easily used in large, real life projects. For example, the lack of tool support revealed to be an important weakness for Gaia.

We also realized that MESSAGE and Tropos have a broader coverage of the whole software development life cycle. Tropos is stronger in the early stages of the software development, while MESSAGE is stronger in later stages of the software development.

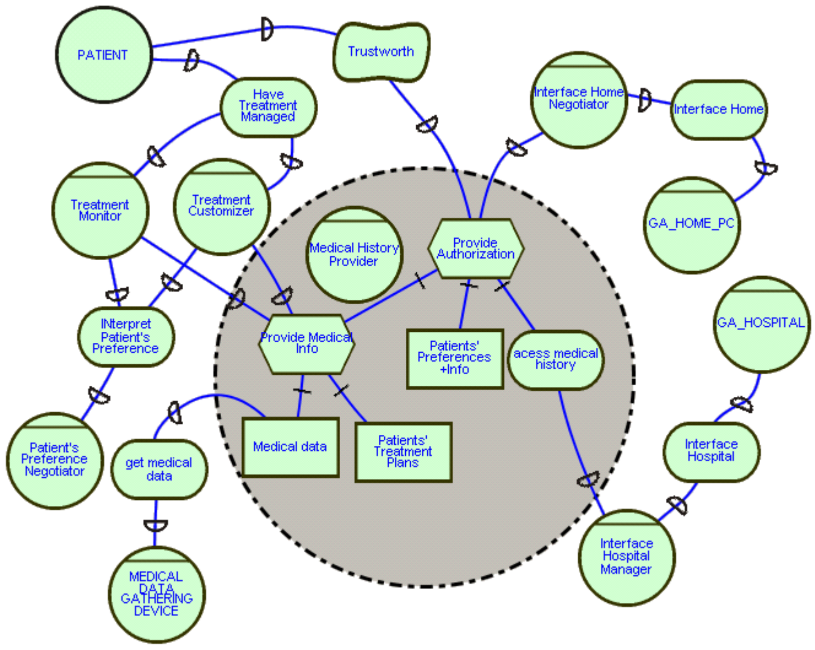

Figure 5. Extended GA PDA Actor Diagram in the Late Requirements

Modelling some non-functional requirements such as privacy, security, and usability, we realized that those requirements were essential to be satisfied in the GA domain. The patient and the physician will only use the system if they know their expectation of privacy and security can be fulfilled. Another point that is fundamental for the patient is usability. The patient has to use the GA frequently so the system can help him/her to monitoring his/her treatment and medical conditions. Thus, usability is a must for the GA_PDA. Considering these expectations, modelling and answering question

QA19 “Eliciting and reasoning about Non-Functional aspects” for the three methodologies, we realized that only Tropos offers support to systematically deal with the non-functional requirements elicitation and reasoning. Despite the fact that MESSAGE defines an Interaction Diagrams and details it using AUML sequence diagrams, it does not support, compared to Tropos, to an early *reasoning* about usability. It also does not support the modelling of different alternatives for usability and other non-functional requirements together with an evaluation of how each alternative would contribute not only to usability but also to correlated requirements such as security. In its turn, Gaia defines the protocols in the Interaction Model which is a pattern specification. Here, the lack of graphical notation was an important fact that led us to a weak evaluation of Gaia’s User Design Interface. Since non-functional requirements are among the most difficult and expensive type of requirements to deal with [8], the lack of support to deal with them could jeopardize the success of software such as the one proposed in the exemplar.

Another interesting point we observed is that both Gaia and Tropos drives the developer to use agent concepts. In contrast, a developer with no or little acquaintance with agent orientation, if using MESSAGE, may not explore the whole potential of the methodology. In fact, it was only when we started answering the questions for MESSAGE that we realized we were not exploring all the strengths that agent concepts bring to software development. We had to revisit our models to obtain more accurate models.

Scenario EA4.3 was very important to some of our findings. The GA_Home and the GA_Hospital agents may not come to an agreement about rescheduling the consultation. Modelling this aspect and answering question QA3 “Autonomy Reasoning” we realized that in Gaia we can model and reason about autonomy in the Organizational structure. Tropos also support this reasoning with the Actor and Goal Diagram by analysing Goal and Softgoal satisfaction. Although MESSAGE has the Organization and the Agent Diagrams, this feature can also be detailed in the Task Workflow Diagram and the State chart. Here, the possible overlap of representations and the lack of orientation on how to proceed in this kind of situation diminish the ability of MESSAGE to cope with autonomy reasoning.

Gaia does not strongly support requirements elicitation, but MESSAGE and Tropos are strong methodologies in QA8 “Finding Requirements”. However, Tropos has a better support for early phases of requirements modelling, since it guides the requirements engineer on findings about the different actors involved and their relationship. Showing these dependencies and stimulating to evaluate how properly these dependencies are established helps one to deal with vulnerabilities and opportunities which is a strong point for supporting sociability properties for agents, QA7 “Domain Analysis”

QA31 “Tools support” tackles the ability of each methodology to support the software engineering in modelling the system. We could not find any tool support for Gaia. The MESSAGE tool support is a commercial tool that was defined a specific meta-model to MESSAGE and proved to be very helpful. Tropos offer the OME3 tool, but it does not support the modelling during design phase.

In the QA32 “Learning Curve”, we recognize that a methodology evolves, and therefore new constructs may be proposed and existing ones could be abandoned. We also recognize that, understandably, as academic work those methodologies do not place documentation as top priority. However, MESSAGE and Tropos does not provide a consistent documentation. One can see different artefacts being used in previous papers that are not mentioned in more recent documentation. Nonetheless, there is no guidance on whether these artefacts are still used or not, and if they are, how they relate to the new ones. In its turn, Gaia defines the concepts in a consistent way referring others documents and also the extensions proposed justifying significant changes Furthermore, the examples used to illustrate the methodologies are too simple and it is not very helpful for someone aiming to use it on complex problems. Many doubts were left unanswered because of the lack of complexity in the examples and thus the lack of guidance to more complex situations.

5. Conclusion

In this work we applied an exemplar [25] to three different methodologies, MESSAGE [5], Gaia [26] and Tropos [3]. Our primary goal was to evaluate how well an exemplar can be used to evaluate to what extent a methodology supports requirements elicitation. We also wanted to evaluate to what extent this exemplar [25] was really stressing and evaluating the methodologies; revealing their strengths and weaknesses. We presented a set of findings indicating strengths and weaknesses of each methodology to handle requirements elicitation.

One common problem of all methodologies was the lack of good examples to illustrate the methodology. All of them use simple examples. While this may help first time readers, it does not provide clear guidance on more complex situations to those trying to use the methodology for more complex problems. This is where the exemplar proved to be of a great help. We believe that if those developing the methodologies use the exemplar to illustrate the methodology, it may provide future users with strong documentation on how to use the methodology.

We also used this experiment to evaluate the exemplar. We noticed for example that there were methodologies with modelling constructs that were not promoting additional clarity or understandability. They were in fact only increasing the complexity of the methodology. Thus we introduced a new question to clearly tackle this problem. However, due to the lack of space we did not covered this aspect in this paper.

As for future work, we envision applying the exemplar to all the phases of the software development life cycle, i.e., coding, testing, deployment and evolutionary changes. We also expect to apply it to RUP to contrast object oriented development.

We also intend to send to developers of MESSAGE, Gaia and Tropos our findings. We hope that our findings will stimulate the developers of each methodology to apply the exemplar themselves. Their observations and findings could help us to evaluate if any change is needed either to the exemplar or to how the exemplar is applied.

The exemplar is currently available at [2] and we expect the community would contribute with ideas for improving it. We believe this exemplar could become a standard to be used by most of the methodologies allowing not only for them to benefit from a strong test case, but also by providing a common base for evaluation developers may easily position their work towards others.

References

[1] Alencar, F., Castro, J., Cysneiros, G. and Mylopoulos,. J. , From Early Requirements Modeled by the i* Technique to Later Requirements Modeled in Precise UML, In Anais do III Workshop em Engenharia de Requisitos, Rio de Janeiro, Brazil, 2000, pp 92-109.

[2] http://www.cs.toronto.edu/km/aometh/.

[3] Bresciani, P. Giorgini, P., Giunchiglia, F. Mylopoulos J. and Perini. A., TROPOS: An Agent-Oriented Software Development Methodology. Journal of Autonomous Agents and Multi-Agent Systems. Kluwer Academic Publishers, 2003.

[4] Caire, G., Leal. F., Chainho, P., Evans, R., Gómez-Sanz, J., Pavón, J., Kearney, P., Stark, J. and Massonet. P., Project p907, deliverable 1: Initial methodology. Technical Information Final version, European Institute for Research and Strategic Studies in Telecommunications (EURESCOM), July 2000. Available from http://www.eurescom.de/public/projectresults/P900-series/907d1.asp

[5] Caire, G., Coulier, W., Garijo, F., Gómez-Sanz, J., Pavón, J., Kearney, P.and Massonet. P., MESSAGE: A Methodology for Development of Agent-Based Applications, To appear at Methodologies And Software Engineering For Agent Systems, edited by Federico Bergenti, Marie-Pierre Gleizes and Franco Zambonelli, to be published by Kluwer Academic Publishing (New York), 2004.

[6] Castro, J., Kolp M. and Mylopoulos. J., Towards Requirements-Driven Information Systems Engineering: The Tropos Project. In Information Systems, Elsevier, Amsterdam, The Netherlands, 2002.

[7] Ceruzzi, L. and Rossi, G., “On the Evaluation of Agent-Oriented Methods”, Agent Oriented Methodology Workshop, November 2002.

[8] Cysneiros, L.M. and Leite, J.C.S.P. “Non-Functional Requirements: From Elicitation to Conceptual Models” To Appear in IEEE Trans. On Soft. Eng., 2004.

[9] Dam K. H. and Winikoff, M., “Comparing Agent-Oriented Methodologies”, 5th International Workshop on Agent-Oriented Information Systems (AOIS’03), July 2003, pp 52-59.

[10] Dam K. H.., “Evaluating and Comparing Agent-Oriented Software Engineering Methodologies”, Master thesis of Applied Science in Information Technology. School of Computer Science and Information Technology of RMIT University, Australia, June 27, 2003.

[11] Evans, R., Kearney, P., Stark, J., Caire, Garijo. F. J., Gómez-Sanz, J., Pavón, J., Leal, F., Chainho, P. and Massonet. P., Project p907, deliverable 3: Methodology for agent-oriented software engineering. Technical Information Final version, European Institute for Research and Strategic Studies in Telecommunications (EURESCOM), September 2001, Available from http://www.eurescom.de/public/projectresults/P900-series/907ti1.asp

[12] Feather, M., Fickas, S., Finkelstein, A., van Lamsweerde, A. “Requirements and Specification Exemplars”. *Automated Software Engineering.* 4(4) 1997.

[13] Garzetti, M., Giorgini, P., Mylopoulos J. and Sannicolò. F., Applying Tropos Methodology to a real case study: Complexity and Criticality analysis. In Proceeding of the Italian workshop on "Dagli OGGETTI agli AGENTI - Dall'informazione alla Conoscenza (WOA02)", Milano, November, 2002.

[14] Giorgini, P., Perini. A., Mylopoulos J., Giunchiglia, F and Bresciani. P., Agent-Oriented Software Development: A Case Study. Proceedings of the Thirteenth International Conference on Software Engineering & Knowledge Engineering (SEKE01), Buenos Aires, Argentina, June 13-15 2001.

[15] Iglesias, C.A. and González, J.C. “A Survey of Agent-Oriented Methodologies” In *Proceedings of the 5th International Workshop on Agent Theories, Architectures and Languages* (ATAL'98), LNAI n1555 - Springer Verlag, Paris, France, July 1998, pp:317-330.

[16] Kolp M., P. Giorgini, P. and Mylopoulos J.,.A Goal-Based Organizational Perspective on Multi-Agents Architectures, In Proceedings of the Eighth International Workshop on Agent Theories, architectures, and languages (ATAL-2001),Seattle, USA, August 1-3, 2001.

[17] MESSAGE, http://www.eurescom.de/~public-webspace/P900-series/P907/index.htm, May 23, 2000.

[18] Parnas, D.L. “On the Criteria To Be Used in Decomposing Systems into Modules” Reprinted from *Communications of the ACM*, Vol. 15, No. 12, December 1972, pp. 1053-1058

[19] Pavóon, J., Gómez-Sanz, J., "Agent Oriented Software Engineering with INGENIAS". In: Multi-Agent Systems and Applications III, Proc. of CEEMAS 2003, LNCS 2691, Springer-Verlag, 2003, pp 394-403.

[20] Shehory, O. and.. Sturm, A, “Evaluation of Modelling Techniques for Agent-Based Systems”, 5th International Conference on Autonomous Agents (Agents 01), ACM Press, May 2001, pp 624-631.

[21] Sturm, A. and. Shehory, O., “A Framework for Evaluating Agent-Oriented Methodologies”, 5th International Workshop on Agent-Oriented Information Systems (AOIS’03), July 2003, pp 60-67.

[22] Szolovits, P., Doyle, J., Long, W.J. “Guardian Angel: Patient-Centered Health Information Systems” Technical Report MIT/LCS/TR-604, http://www.ga.org/ga/manifesto/GAtr.html

[23] Wooldridge, M., Jennings, N.R. and Kinny, D., The Gaia methodology for agent-oriented analysis and design. Autonomous Agents and Multi-Agent Systems, 3(3), 2000, pp 285-312.

[24] Wooldridge, M., Jennings, N. R. and D. Kinny. A methodology for agent-oriented analysis and design. In Proceedings of the Third International Conference on Autonomous Agents (Agents 99), Seattle, WA, May 1999, pp 69–76.

[25] Yu,E., Cysneiros.L.M., "Agent-Oriented Methodologies-Towards a Challenge Exemplar" in Proc of the 4th Intl. Bi-Conference Workshop on Agent-Oriented Information Systems (AOIS 2002) Toronto May 2002

[26] Zambonelli, F., Jennings, N. R. and Wooldridge, M., Developing Multiagent Systems: The Gaia Methodology, In ACM Transaction on Software Engineering and Methodology, Vol. 12, No. 3, July 2003, pp 317-370.

[27] First International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design http://www.ait.unl.edu/siau/conference/emmsad03-CFP.htm