PUC

ISSN 0103-9741

Monografias em Ciência da Computação
n° 35/08

Applying the Concept of Agent Reputation to the Context of Diagnoses and Recommendations for Agent's Execution Plans

Andrew Diniz da Costa
Carlos José Pereira de Lucena
Viviane Torres da Silva
Baldoino Fonseca dos Santos Neto

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO
RUA MARQUÊS DE SÃO VICENTE, 225 - CEP 22451-900
RIO DE JANEIRO - BRASIL

Monografias em Ciência da Computação, No. 35/08 ISSN: 0103-9741
Editor: Prof. Carlos José Pereira de Lucena August, 2008

Applying the Concept of Agent Reputation to the Context of Diagnoses and Recommendations for Agent's Execution Plans

Andrew Diniz da Costa, Carlos José Pereira de Lucena,
Viviane Torres da Silva[1], Baildoino Fonseca dos Santos Neto

[1]Departamento de Sistemas Informáticos y Computación, Universidad Complutense de Madrid, Spain

{acosta, lucena, bneto}@inf.puc-rio.br, viviane@fdi.ucm.es

Abstract. In Multi-Agent Systems, autonomous and heterogeneous agents can work together to achieve the same or different goals. When an agent is not able to attain its goals, a big challenge is to understand the reason why this happens, and what can be done to remedy the problem. In this paper, we propose a hybrid diagnostic-recommendation framework that provides support for different challenges in performing diagnoses, namely determining what caused the failure of a plan. We also offer recommendations for alternate plans so that the collection of agents may repeat the attempt to achieve its goal. The example used to demonstrate the proposed framework is based on ubiquitous computing.

Keywords: Multi-Agent Systems, Diagnosis, Recommendation, Reputation, Ubiquitous Computing

In charge of publications:

Rosane Teles Lins Castilho
Assessoria de Biblioteca, Documentação e Informação
PUC-Rio Departamento de Informática
Rua Marquês de São Vicente, 225 - Gávea
22451-900 Rio de Janeiro RJ Brasil
Tel. +55 21 3527-1516 Fax: +55 21 3527-1530
E-mail: bib-di@inf.puc-rio.br
Web site: http://bib-di.inf.puc-rio.br/techreports/

Table of Contents

1 Introduction

Ubiquitous computing assumes a world in which people are surrounded by mobile or fixed devices with a computing environment that supports them in almost all tasks. Since these services can be provided by heterogeneous agents designed and implemented by different developers, it is reasonable to consider that failures may occur. Therefore, a big challenge in Multi-Agent Systems [Jennings and Wooldridge, 1999] [Wooldridge and Ciancarini, 2000], which is a new paradigm for the software engineering of complex and distributed systems [Boella and Torre, 2004] where collections of autonomous and heterogeneous agents can wok together to achieve the same or different goals, is to diagnose some execution that was not able to attain some desired goal and to provide recommendations of execution's plans, which allow remedying some problem that happened.

Different challenges are related to performing diagnoses and providing recommendations to achieve a goal that was not previously achieved by an agent. As follows, such challenges are mentioned.

1. How to analyze the agent execution aiming to understand the reason that prevented the achievement of a desired goal.
2. Selecting appropriate data to produce a diagnosis related to the execution of the failed agent.
3. Different application areas use domain-specific information to make diagnoses. Therefore, the problem is to define a flexible approach that can be adapted to those different domains.
4. Each agent has particular properties and characteristics that can be represented in a profile. The challenge is to define a component that could be used to represent generic agent profiles and to provide services that help diagnosis and recommendation strategies use these profiles.
5. The strategies for generating recommendations may be domain-dependent, since domain-dependent data can determine the recommendation that matches the diagnosis. Therefore, it is important to provide a flexible approach that could be used in several domains.

In this paper we describe a hybrid diagnostic-recommendation framework called DRP-MAS (Diagnosing and Recommending Plans in open Multi-Agent Systems), which allows using reputations to diagnose and recommend execution plans to software agents. The framework is able to analyze different sets of information related to the agent's execution and to provide proper diagnosis by assembling all the facts and indicating the (main) problem that occurred. A failure may have many causes such as: the resource that an agent (provider of a service) is to use is not available or is damaged; an agent that normally collaborates in the execution of a given task chooses not to collaborate in a specific situation; or the information provided by an agent is inadequate.

Recommendations based on the diagnosis should provide alternative strategies to support an agent achieving the same goal. The recommendation system can recommend actions such as the use of another resource, the execution of another plan, or interaction with other agents. While recommending other agents for interaction, the recommendation system bases its choice on the new partner's reputation. Reputation is a social notion associated with how trustworthy an individual is observed to be within a

society [Koogan and Houaiss, 1995]. Several reputation models have been proposed to collect, aggregate and distribute feedback about an agent's past behavior.

The DRP-MAS framework assists an agent in diagnosing its failures and to recommend alternative plans to support an agent in achieving its goals. The paper is organized as follows. Section 2 we provide an overview of the DRP-MAS framework. Section 3 describes a complex problem associated with ubiquitous computing that is supported by our approach. Section 4 presents a simple case study involving ubiquitous computing. Section 5 contains a description of some related work and Section 6 concludes and indicates possible future research directions.

2 The DRP-MAS Framework

In this section, the DRP-MAS framework (Diagnosing and Recommending Plans in Multi-Agent Systems) is presented. Initially its main idea is explained, followed by the architecture proposed and its main concepts.

2.1 The Main Idea

When an agent of software does not achieve one of its goals after the execution of one of its plans, the DRP-MAS framework can be used. Figure 1 illustrates the general idea of an agent requesting recommendations to the DRP-MAS.

Initially, the agent (*Requester agent*), which did not achieve the desired goal, requests to a *Mediator agent* the creation of a *Diagnostic agent* (responsible for providing diagnoses) and a *Recommendation agent* (responsible for providing recommendations). After that, the *Mediator agent* informs to the *Requester agent* the *Diagnostic agent* created (Figure 1).

After the *Diagnostic agent* to be created and identified, the *Requester* sends a message to the *Diagnostic agent* with the set of information that can help in meeting the reason of the agent does not achieve the desired goal, such as: plan executed, goal not achieved, the agents with whom it has interacted, its profile, etc (more details in subsection 2.3).

When the *Diagnostic agent* receives the message, it tries to discover what has caused the failure in a *Requester agent* execution. At the end of the analysis, the *Requester agent* provides the diagnosis to the *Recommendation agent*, which then provides a recommendation. Even if a diagnosis could not be provided, the *Diagnostic agent* sends a message to the *Recommendation agent* stating that it was not possible to determine why the *Requester agent* did not achieve the desired goal. In this case, when the *Recommendation agent* receives the message indicating it was not possible to provide a diagnosis; it still tries to select plans that can be used to achieve the desired goal.

In the case in which a diagnosis is provided, the *Recommendation agent* searches for alternative plans to achieve the goal (details shown in subsection 2.5) using the diagnosis information. When the diagnosis indicates a problem in the interaction with a specific agent, an analysis is made to decide which other agents can be used to perform the interactions or services identified in the recommended plans.

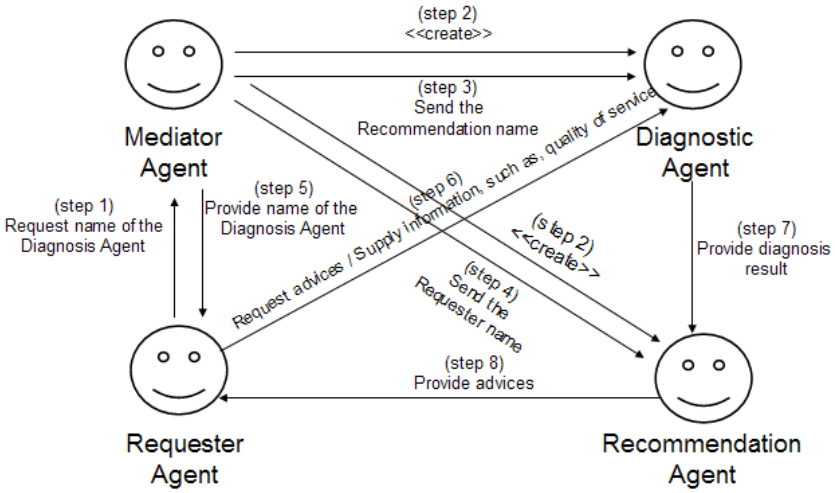

Figure 1. Conceptual Model for requesting the name of the diagnostic agent

When the set of agents, that can perform the same services, is met, the *Recommendation agent* selects the agents with highest reputations. The profile of the *Requester agent* can be useful to select plans and to choose agents. When the *Recommendation agent* finishes, a message is sent to the *Requester agent* with the recommendations, namely the alternative plans to achieve the same goal.

In order to be able to provide recommendations, the *Recommendation agent* must receive the list of plans the *Requester agent* can execute and their related information, such as: the goals of each plan, the services requested by the plan, and the resources used. These plans are stored in a plan base, which any *Recommendation agent* can access in order to provide the advices.

2.2 Architecture

As illustrated in Figure 2 the DRP-MAS is composed of five modules: Mediation, Diagnosis, Recommendation, Artificial Intelligence Toolset and Reputation.

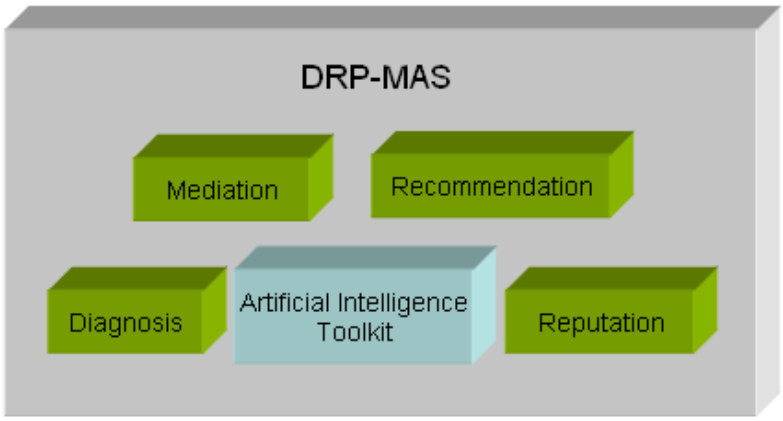

Figure 2. Architecture

Mediation module defines the *Mediator* agent (see subsection 2.1), which creates a *Diagnostic agent* and a *Recommendation agent* to interact with the *Requester agent* (represented for some application). The *Diagnosis* module is the responsible to perform the process of diagnosis, while the *Recommendation* module defines the process of recommendation of execution in order to achieve some desired goal. The *Artificial Intelligence Toolset* module, which has an API (Application Programming Interface) called BIGUS [Bigus, 2001], provides the following reasoning algorithms: forward chaining, backward chaining and fuzzy logic. These algorithms can be used by the diagnosis and recommendation modules, which are responsible for representing the processes of diagnosis and recommendation, respectively.

The last module is the Reputation, which provides reputations of other agents to the DRP-MAS and to the application that uses the framework. In our approach, the reputation model is based on the Fire model [Huynh et al., 2004] and on the Report reputation system [Guedes et al., 2006] [Silva et al., 2007]. Below, the data that can be provided by the *Requester agent* are explained, and on following the *Diagnosis*, *Recommendation* and *Reputation* modules are presented in details.

3 Data Set to Diagnoses and Recommendations

The domain-independent data that the *Requester agent* should send to the *Diagnostic agent* in order to help with the definition of diagnoses and recommendations are the following:

1. Resources and associated problems – As mentioned in [Horling et al., 2000], resources can be considered important data to support diagnoses. A possible reason for an execution failure can be an insufficient amount or the absence of a resource.
2. Quality of service – As defined by the TAEMS model [Lesser et al., 2007] [Wagner et al., 2003], a quantitative value can be used to define the quality of a task executed by an agent. Such value can indicate the success of the agent in performing the task. Therefore, this data can be useful in diagnosis and recommendations.
3. Goal – The execution of an agent's plan is always associated with a goal that the agent wants to achieve [Silva et al., 1999], being fundamental to both a diagnosis and the corresponding advice.
4. Plan executed – In order to understand the reason for the failure and to provide alternative execution strategies it is necessary to have the plan that was executed by the *Requester agent*.
5. Agents with whom the agent interacted – It may be the case that an agent fails to achieve a goal because of wrong or false information received from another agent. Thus, it is important to know which agents were involved during the execution of the *Requester agent* plan.
6. Services used – It may be the case that a failure is caused by the poor quality of a service provided by another agent. Therefore, it is important to know the services used in order to provide diagnoses and to recommend other agents to provide the same services. Each request performed by some agent can have important identifying data: (i) request identifier, (ii) date and hour of the request and (iii) grade of severity of the request.

7. Profile – Each agent has a profile, which can be used to represent some of its characteristics. The *Requester agent* profile can, for instance, stipulate the minimum reputation value that its partners must have. Thus, the *Recommendation agent* should consider such information while seeking partners.
8. Belief Base – The knowledge base used by the *Requester agent*, including the time of its last update, can be useful to perform diagnoses and to provide recommendations.

When applications are based on ubiquitous computing, the following information, already identified in DRP-MAS, can be provided by the *Requester agent*:

1. Device used – Because of the different characteristics of available devices, it is important to identify (i) the type of device used (ex: cell phone, etc), (ii) its model (ex: LG MG296 GSM, etc.) and (iii) the language in which the data must be provided by the agent (ex: English, Spanish, etc).
2. Connections – The characteristics of the connections, i.e., (i) its speed (ex: 56Kbps, etc), (ii) its technology (ex: wireless, etc.) and (iii) the IP address used, are important to construct a diagnosis and to provide recommendations.

3.1 Diagnosis Module

The diagnoses are provided by the *Diagnostic agent* available in the framework, and are based on data provided by the *Requester agent*. Although different data can be required in different application domains, it is possible to define a set of domain-independent data that can be reused by the different, so as mentioned in subsection 2.3.

Since many different types of diagnoses can be performed by using domain-independent and domain-dependent data, the DRP-MAS framework has hot spots (flexible points) to support definition of diagnosis strategies. Note that different strategies can also be used by the same *Requester* agent depending on the data available after execution of the plan.

To help with the implementation of domain-dependent strategies three different algorithms (backward chaining, forward chaining and fuzzy logic) are available in the Artificial Intelligence Toolset module defined in the framework, as mentioned in subsection 2.2. The framework offers special support to the use of the forward chaining algorithm. The service provided by the framework helps with the identification of more precise diagnoses even when the user has provided very little data. In order to employ such a service the user of the framework must supply a hierarchy rule base that will be used by the framework to discover related diagnoses.

To illustrate the use of this service, let's suppose that an agent wants to obtain the list of events (celebrations) that will happen at a month in a country. In other to receive the list, the agent requests the service provided by an agent called *promoter*. When the *promoter* does not meet events, it asks for recommendations. When the *Diagnostic agent* receives the request, it uses the forward chaining algorithm to infer the diagnosis from the data provided by the *promoter*. Let's suppose that the *promoter* has provided the quality of the service (that is equal to 0 since no event was met), the name of the country and its knowledge base (KB) with the events that it knows. After receiving this information the *Diagnostic agent* checks if the country provided has some event in the KB and uses its rule base, shown in Table 1, to infer the diagnosis. Suppose that any event was not found in the KB and the diagnosis inferred was Without_Event, i.e., there is not any event to the correspondent country.

If the *Promoter agent* had provided the time that the KB was updated by the last time, two other diagnoses could have been provided (Figure 3). Thus, DRP-MAS sent the Recommendation agent not only the real diagnosis but also the other two that could have been used if more information was provided.

Table 1. Rule Base of detection of events in countries

IF Quality_of_Service=0 **AND**
Base_Belief=FALSE **THEN**
DIAGNOSIS=WITHOUT_EVENT
IF Quality_of_Service=0 **AND**
Base_Belief =FALSE **AND**
Base_Belief_Updated=FALSE **THEN**
DIAGNOSIS=NECESSARY_UPDATE_KB
IF Quality_of_Service=0 **AND**
Base_Belief =FALSE **AND**
Base_Belief_Updated=TRUE **THEN**
DIAGNOSIS=NOT_POSSIBLE_TO_MEET_EVENT

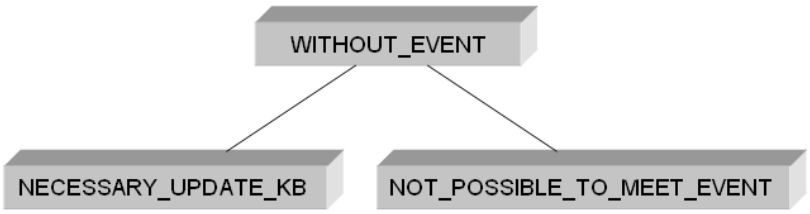

Figure 3. Relation between diagnoses defined on the rule base

3.2 Recommendation Module

The *Recommendation agent* includes a process for providing alternative ways to achieve a goal. This process is composed of three steps: (i) selecting plans, (ii) verifying whether the selected plans presuppose interactions with other agents and (iii) choosing appropriate agents to the necessary interactions in selected plans. On following each step is explained in details.

3.2.1 Selecting Plans

This step is the first to be executed when the *Recommendation agent* receives the diagnosis from the *Diagnostic agent* and needs to verify which plans can be used to achieve a specific goal. Based on the information provided by the *Requester agent*, the *Recommendation agent* analyzes the plans to be chosen. As different strategies of diagnosis can be developed, the DRP-MAS framework defines such a strategy as a hot-spot that can be instantiated based on the requirements of the application.

In order to be able to select plans based on the information provided by the *Requester agent*, each plan should be associated with a set of information that describes and classifies the plan. It is important to point out the services and resources used during the execution, the goal that the plan will try to achieve, related diagnoses, a value that specifies when the plan must be executed, and other domain-dependent information such as devices that the plan will use, types of associated connections, and a collection of possible problems that the plan can solve. Note that the information associated with each plan is strongly related with the information that the *Requester agent* can provide to the *Diagnostic agent*, as described in subsection 2.3.

DRP-MAS offers two simple services with the aim of helping with the definition of the recommendation strategy. One of the services is able to select plans that are related to given data and the other is able to select plans that are not related to such data. After selecting the recommended plans, the second step of the recommendation process should be executed. In the case where no plan could be recommended, a message is sent to the *Requester* and the process is aborted.

3.2.2 Verifying Selected Plans

After the selection of alternative plans, this step verifies whether the plans will require the use of services that will be provided by other agents. If no plan requires such services, a message is sent to the *Requester agent* with the recommendations that have been obtained. Otherwise, the *Recommendation agent* will need to discover the agents that are able to provide such services and to select the ones with the best reputation. To select these trustworthy agents the *Recommendation agent* provides to the *Reputation* agent (detailed in subsection 2.5) a set of information that is used to select agents from reputations.

3.2.3 Choosing Agents

This step is responsible for receiving the selected agents by the *Reputation agent* and on follow to perform a new filter of the correspondent agents in order to define which will be recommended. The strategy used to select the partners is the third important hot-spot of the framework. Since a different reputation model can be used and different profiles can be defined. The profile can define the minimum acceptable reputation for each different types of reputation. The framework offers a profile that allows specifying this information in relation the available reputation.

3.3 Reputation Module

The reputation module is the one responsible for selecting agents with good reputation from of data provided for some agent. In order to help the selection, the agent can receive the following data from another agent: (i) some profile, the services that the can-

didate agent must provide, and the agents used previously in some execution performed for some agent (*Requester agent*).

Although the framework offers a default implementation of the *Reputation agent* based on the reputations provided by the Report framework [Guedes et al., 2006] [Silva et al., 2007] and the Fire system [Huynh et al, 2004], any other reputation system can be used. The strategy used by the *Reputation agent* to select the reputations of the agents is a hot-spot of the framework. The default strategy provided by the framework uses three different types of reputation provided by the Fire model and one provided by the Report framework to evaluate the reputations of the agents.

The Report framework implements a centralized reputation mechanism that stores the reputations of the application agents. The agents in the system are able to provided information about the behavior of other agents and Report, as part of the Governance framework, is able to evaluate such behavior and store the associated reputations.

Based on the centralized reputation mechanism defined in Report, DRP-MAS defines a reputation base to store reputations provided by agents about the behavior of others. After interacting with their partners, the agents should evaluate their behavior and send such information to DRP-MAS. The main idea is to define a unique reputation (or global reputation) for each participant in an application, and to allow agents access to such global reputations.

The Fire model defines a decentralized reputation mechanism where the agents are able to evaluate the behavior of other agents and also to store their reputations. From the set of available trust and reputations types defined in Fire, DRP-MAS uses the interaction trust (resulting from past experiences of direct interactions), witness reputation (reports of witnesses about other agents' behavior) and certified reputation (references provided by other agents about an agent's behavior).

Fire was chosen because it defines certified reputations which is an extra category not provided by other reputation systems. Such reputations are fundamental when an agent wants to know the reputation of agents with which it has not interacted and when it does not know any other agent that has interacted with the desired agent.

By using the *Reputation agent*, the *Recommendation agent* is then able to request candidate agents that have of the four available reputation types defined as default in DRP-MAS. As stated before, other reputation systems and associated reputation types can be implemented.

4 Case Study: Ubiquitous Computing with MPS

The case study presented in this section is related to the domain of Mobile Process Service (MPS), a business process that brings together individuals solving a problem by sharing expertise while interacting through communication devices driven by agents. Since such agents are typically heterogeneous, potentially designed and developed by different developers and also distributed over the environment, it is reasonable to suggest that failures may occur. Therefore, this domain is appropriate to illustrate our diagnosis and recommendation framework. Our approach provides a diagnosis when an agent cannot complete its assigned task and alternative plans. The simple scenario used to illustrate the approach is the negotiation of the price of a music CD in a market place.

4.1 Scenario: Music Market Place

In this section, we explain the music market place scenario, which was implemented using the DRP-MAS framework. In this scenario a customer or *Buyer agent* that wishes to buy CDs contacts a *Seller agent* and provides the name of the music that must be on the CD, its category and the maximum price the *Buyer agent* or customer is willing to pay for that CD.

If the *Buyer agent* does not receive the desired CD, it requests recommendations from DRP-MAS by providing (i) the plan executed; (ii) the quality of such execution; (iii) the desired goal ("to buy a specific CD"); (iv) the identification of the *Seller agent*; (v) the agent profile that defines the amount of money to be spent buying CDs and the minimum acceptable reputation of each of its partners; (vi) the name of the music; (vii) the chosen CD category and (viii) the CDs provided by the seller, if any.

To perform a diagnosis, we used the forward chaining algorithm again. A rule base (see Table 2) was defined with three possible diagnoses: (i) seller does not know of any CD with the characteristics specified by the buyer, (ii) the CD provided is more expensive then the amount the buyer is willing to pay, and (iii) although the CD has the specified music it belongs to a different category.

The first diagnosis occurs when the buyer has not received any CD since there is no CD with the desired music (quality_execution = 0 indicating that no CD was provided). The second diagnosis occurs when the CD provided by the seller is more expensive then the buyer is willing to pay (quality of execution between zero and ten indicating that a CD was provided but it is not the one the buyer wants). The third diagnosis happens when the CD provided has the desired music but is in a different category. We can see that, in this scenario, the diagnoses are totally independent (Figure 4), unlike the example presented in subsection 2.4. On the other hand, in this scenario it is possible to have two diagnoses occur at the same time.

After receiving the diagnosis, the *Recommendation agent* updates the reputation of the seller according to the product it has sent to the buyer. If the seller has not provided any CD, its reputation decreases more than in the cases where a CD was provided.

Table 2. Rule base of the Music Marke Place scenario

IF Quality_Execution = 0 AND
THEN **Diagnosis = "Seller_Does_Not_Know_CD"**
IF Quality_Execution < 10 AND
Quality_Execution > 0 AND
Mash_profile = "false" AND
THEN **Diagnosis = "High_Price_CDs"**
IF Quality_Execution < 10 AND
Quality_Execution > 0 AND
THEN **Diagnosis = "Different_Category_of_Music"**

Figure 4. Diagnoses of the Music Market Place scenario

Since the three different diagnoses blame the seller for not providing the buyer with the desired CD, we can conclude that the plan the buyer has executed is not the problem. Therefore, the *Recommendation agent* recommends the same plan but suggests different sellers. The sellers are selected according to the buyer profile. In the case the buyer has not specified any maximum price in its profile, the *Recommendation agent* recommends expert agents, i.e., agents that will probably send more expensive CDs owing to their expertise on finding what their clients want. In the case where the buyer has specified a maximum price, the *Recommendation agent* recommends common sellers that have lower, but reasonable reputations.

5 Related Work

In [Li et al., 2004] a decentralized system is proposed to perform diagnosis and monitoring. Each component has a monitor (Monitoring Agent), which is responsible for collecting information about the component. Once it has obtained the information, it is provided to agents responsible for finding diagnoses while working together. When applied to open multi-agent systems this approach violates the agents' privacy. DRP-MAS, on the other hand, avoids the creation of monitors and lets the application agents themselves provide the information when failures occur.

The work [Horling et al., 2000] examines the use of domain-independent diagnoses in multi-agent systems. Their approach is based on the assumption that the correct behavior, or at least the expected one, should be previously described. They have defined a goal/task decomposition language called TAEMS that can be used to describe goals and sub-goals. This language provides an explicit representation for goals, and pathways to achieve the sub-goals; each branch of the goal tree terminates at an executable method. It is possible to represent explicitly the expected behavior of the method, its expected quality, cost and duration, and also the other methods with which it interacts. In DRP-MAS the methods are represented by plans that are used to attain goals. Each plan has a set of possible related information, such as, the type and amount of resources used, desired goal, and expected quality. Therefore, if a plan does not achieve the agent's goal, it is possible to determine when and where the failure happened. Our approach offers a larger set of information than the one proposed by [Horling et al., 2000], and also supports the choice of the agent's partners based on their reputations.

In [Roos et al., 2002] a set S used to diagnose the multi-agent system is defined as S = (C, M, Id, Sd, Ctx, Obs), where C is a set of components, M is a specification of a possible fault in each component, Id is a set of identifiers of points that connect components, Sd is the description of the system, Ctx is a specification of input values of the system that are determined outside the system by the environment, and Obs is a set of observed values of the system. DRP-MAS follows a similar idea by extracting the necessary information to perform diagnoses from the information provided by the user.

In [Pop et al., 2006] is presented the knowledge discovery from databases (KDD), which is a complex process composed of several phases: business understanding, data understanding, data preparation, modeling, evaluation and deployment. For each of the phases, there are many algorithms and methods available, the end-user having to select one of them. It is proposed a multi-agent system based intelligent recommendation system for selection of the most appropriate solving method for each phase. In order to provide the recommendations some current dataset is compared with other datasets of already existing scenarios in a knowledge base. In the DRP-MAS the recommendations also can be provided from comparison of data sets, however, more complex strategies can be defined.

6 Conclusions

In this paper we have illustrated the use of the multi-agent systems to create a hybrid diagnostic-recommendation system for agent execution applied to a mobile process service. This system performs different strategies of diagnostic, recommendation and reputation aiming to achieve the agent goals. The mobile process domain is used to illustrate our approach because it provides a representative set of scenarios.

Although the variety of benefits provided by the framework, two disadvantages are present in the approach: (i) can be difficult to define any diagnosis or recommendation even that the DRP-MAS framework offers a set of services, which can be used by the strategies outlined in the instances, and (ii) overload of memory can happen in the system due the amount of agents offered by framework (*Mediator*, *Recommendation*, *Diagnostic* and *Reputation agent*). This problem is mainly related to creation of an exclusive *Recommendation* and *Diagnostic agent* for each *Request agent*. However, this approach was proposed to avoid two situations: (i) agents to be allocated for a long time in list of wait until their requests to be treated and (ii) dependence on strategies between the processes of diagnosis and recommendation.

In future research intend to propose an approach that solves the disadvantages mentioned and that allows adaptation of agents when some diagnosis and recommendation is met. We also intend to determine the type of situations and problems that can occur when software agents are used in ubiquitous computing domains, in order to provide a better support.

References

BIGUS, J., BIGUS, J., Constructing Intelligent Agents Using Java, second edition, March, 2001.

BOELLA, G. AND TORRE, L., Regulative and Constitutive Norms in Normative Multi-Agent Systems. In Proceeding of 9th International Conference on the Principles of Knowledge Representation and Reasoning. California, 2004.

DURAN, F., SILVA, V., LUCENA, C.J.P., Using Testimonies to Enforce the Behavior of Agents, In Proceedings of the Coordination, Organizations, Institutions, and Norms in Agent Systems III., COIN 2007, Springer-Verlag, 218-231, 2007.

GUEDES, J., SILVA, V., LUCENA, C.J.P., A Reputation Model Based on testimonies, In Proceedings of the Agent Oriented Information Systems IV: Proc. of the 8th International Bi-Conference Workshop, AOIS 2006, Springer-Verlag, 37-52, 2006.

HORLING, B., LESSER, V., VINCENT, R., BAZZAN, A., XUAN, P., Diagnosis as an Integral Part of Multi-Agent Adaptability, In Proceedings of the DARPA Information Survivability Conference and Exposition, DISCEX'0, Volume 2, 211-219, 2000.

HUYNH, T. D., JENNINGS, N. AND SHADBOLT, N., *FIRE: an integrated trust and reputation model for open multi-agent systems*. In Proceedings of the 16th European Conference on Artificial Intelligence, Valencia, Spain, 2004.

HUYNH, T., JENNINGS, N. and SHADBOLT, N., 2004, Developing an integrated trust and reputation model for open multi-agent systems, In Proceedings of the 7th Workshop on Trust in Agent Societies, 2004.

JENNINGS, N., AND WOOLDRIDGE, M., Agent- Oriented Software Engineering; Proceedings of the 9th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, Multi-Agent System Engineering, (MAAMAW-99), Vol. 1647, Springer-Verlag: Heidelberg, German,. 1-7, 1999.

KOOGAN, A. AND HOUAISS, A., Encyclopedia and Dictionary. Delta Publisher Rio de Janeiro, 1995.

LESSER, V., HORLING, B., and et al. The TAEMS whitepaper / envolving specification. Available: http://dis.cs.umass.edu/research/taems/white/. Last access in November, 2007.

LI, T., PENG, Y., ZHAO, H., LI, K., Application of Multi-Agent in Control and Fault Diagnosis Systems. In Proceedings of the Third International Conference on Machine Learning and Cybernetics, Shanghai, 26-29, August 2004.

ROOS, N., TEIJE, A., BOS, A., CEES, W., An Analysis of Multi-Agent Diagnosis, In Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems, AAMAS'02, 986-987, 2002.

SILVA, V.; GARCIA, A.; BRANDAO, A.; CHAVEZ, C., LUCENA, C.J.P., ALENCAR, P., Taming Agents and Objects in Software Engineering, Software Engineering for Large-Scale Multi-Agent Systems, Springer-Verlag, 1999, 1-26, 1999.

SILVA, V., DURAN, F., GUEDES, J., LUCENA, C.J.P., Governing Multi-Agent Systems, In Journal of Brazilian Computer Society, special issue on Software Engineering for Multi-Agent Systems, n. 2 vol. 13, 19-34, 2007.

WAGNER, T., VALERIE, G., PHELPS, J., TAEMS Agents: Enabling Dynamic Distributed Supply Chain Management, The Electronic Commerce Research and Applications, Volume 2, Number 2, 114-132, 2003.

WOOLDRIDGE, M., AND CIANCARINI, P., Agent-Oriented Software Engineering: The State of the Art, in First Int. Workshop on Agent-Oriented Software Engineering, Vol. 1957, Springer-Verlag, Berlin, 1-28, 2000.

Pop, D., Negru, V., Sandru, C., Multi-Agent Architecture for Knowledge Discovery, *synasc*, Eighth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC'06), pp. 217-226, 2006.