DICOM Image Management Through Agents Based Systems

Dani Juliano Czelusniak[**], Erica Beatriz Fuscolim[**], Osiris Canciglieri Junior[*]

[*] Pontifical Catholic University of Paraná / Polytechnic School - Production and System Engineering Graduate Program (PUCPR/PPGEPS), Curitiba, Paraná, Brazil.

[**] Pontifical Catholic University of Parana, Polytechnic School, Production Engineering Undergraduate Program (PUCPR/EP), Curitiba, Paraná, Brazil.

dani.czelusniak@pucpr.br, ericafuscolim@hotmail.com, osiris.canciglieri@pucpr.br

Abstract— This paper presents a research that is developed inside "Products and Systems Design and Development" research group of Pontifical Catholic University of Parana (PUC-PR). In this research is studied, designed and developed a DICOM image loader with agents based system, that in a near future will be a important module of a dental prosthesis design decisions multi-agents software. In this paper, will be shown an overview of agent systems explaining its structure with a framework called JADE, used to provide agents software environment. The methods used to guide this research were, the bibliographical survey to list de concepts, and, states of art for the expert system project layout, develop and tests procedures. As a result, this paper shows that using agents' software techniques in expert systems development is a new way to manage data in a flexible way, offering forms to creating modular solutions that should have the capacity to deal with complex scenarios. It also allows, better software scalability and modularization for complex and specialized software solutions.

I. Introduction

Actually, is more evident in corporate environments, the application of software tools for automation and administrative control of processes in operational environment. The development and maintenance tools for software applications are in constant evolution to make the business processes development, more flexible and bug-free [10].

In this scenario, arises a new view for building applications, called agents software based systems [2]. This new architecture is gaining developers attention by the capacity to enable the applications development in a flexible form, because it permits that new software modules inserted in the agents environment can coexist with others, without necessity of modifying these agents "modules code", treated later as agent behaviors [3], [4].

Following, an introduction will be presented from the current business information technology context that aims for new challenges of software context with expert systems. Next, will be presented the JADE agents framework [1] and the advantages of its use before the conventional building software techniques.

This paper, therefore, presents how agent software behaviors need to be designed using agent-based systems techniques with JADE framework, to manage DICOM medical images portfolio. This acquaintanceship facilitates the constantly evolving of the developed system software [5]. To better understanding, this article presents this new software type, focusing on how agent software behaviors works, serving as a guide to develop agent's software to manage DICOM medical images portfolio, that is a module of a dental prosthesis design decisions multi-agentss system that is under development.

II. Agents Software Systems

Agents' software is a special kind of software that is work internally in autonomous ways, working well with the complexity of technological evolution. In high complex scenarios, is necessary use software, which has capacity of adaptation and provide mechanisms, which allows it, to take decisions in accordance with changes in their directives. This modality of information system commonly is known as "agents based systems" or "agents systems" [6], [7], including all kinds of agent, agents and multi-agents software. This way, agents systems are a software construction kind, which was born from Artificial Intelligence classes. These days, they are increasing distinguishability by the ability to permit applications development not only in modular, but modular and autonomous software modules.

Reviewing the literature about software development using artificial intelligence techniques to build agent based systems, seems that is the artificial intelligence is the study and creation of machines that displays human-like qualities, including ability to reasoning [11]. In this context, the agent software can maps the awareness in actions; their architecture varies by species and function of agent software, which can be a database, an intranet, or dedicated expert embedded software in some hardware. Its architecture is responsible for the agent's interaction. The relationship between software agents, intelligent software and architecture according with [7], shown in the Figure 1.

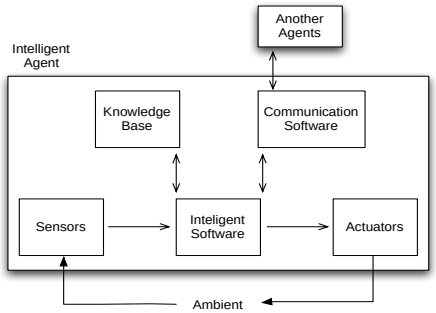

Figure 1 - Internal structure of agent software, adopted from [7]

Intelligent agents are software that have autonomous capacity to compare scenarios and execute actions, collect information, explore and learn in an environment according stimulus or detected perceptions, in the best way aiming execute it tasks faster and precisely mode as possible. Agent is anything that can be considered able to perceive their environment through sensors and act on this by actuators [7], for each possible perceptions, a rational agent must select an action that is expected to come to maximize their performance measure, given the evidence provided by a set of perceptions and by any internal knowledge of the agent.

JADE (Java Agent Development Environment) is one of the existing sets of software libraries, called by software developers as "framework", which is designed to develop applications with agents structures models [1]. Agents systems developed with this framework, will follow the standards defined by FIPA (Foundation for Intelligent Physical Agents). This is an IEEE (Institute of Electrical and Electronics Engineers) member organization, which arranges computational standards for agent, agents and multi-agents systems, guaranteeing interoperability between heterogeneous agents standards [6].

This framework is referenced in the software development tool that JADE is composed, by software elements that FIPA standards defined as necessary for their operation. This software elements set for agents software development, is called as agent platform by [1] and [2], with main reference architecture of FIPA [8] agent platform.

The conceptual JADE framework, to be presented by Figure 2, consists of agents and containers (that is a software environment where agents runs its lifecycle) the standard FIPA defines as necessary for its operation. All JADE Agent Platform has a special container, called the Main Container, which is the first container to be initialized when the application is loaded and. It contains two special agents and the communication service, which will provide functionality to the platform.

Conceptually, this framework was divided in some parts described below:

• AMS: System Management Agent, it oversees the access of other agents to the platform, and is responsible for control and authentication of agents' calls and the life cycle of services provided by the platform. Maintains in its structure a list with the identifiers of the agents along with their states.

• DF: Yellow Pages Service (or Directory Facilitator) is the service that provides the addressing mechanism of the agents, applying the metaphor of "yellow pages" of the agent platform.

• Message Transport Service (or ACC Agent Communication Channel), is the system component that controls the exchange of messages, managed the use of communication protocols.

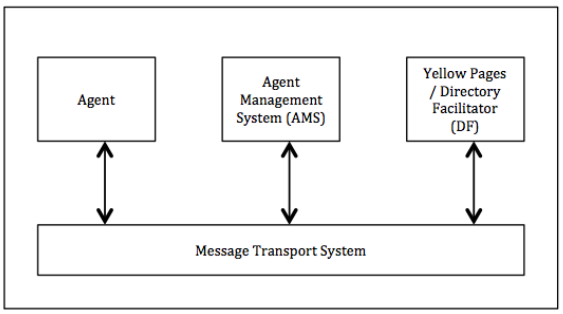

Figure 2 – Conceptual JADE Agent Platform, adopted from [1]

Following the specifications set by FIPA, the services of yellow pages and management agents communicate with the agents through language called FIPA-sl0. However, there may be used other communication language implementations from the FIPA standard.

Besides the basic elements needed for the operation of the JADE framework described above, below there is another figure that exemplify physical JADE container element. Each instance of the JADE environment, which is currently running, has at least one container, and agents residing therein. A set of assets containers is called Agent Platform, as shown in Figure 3. For example, in a computer network, instantiate a container computer and create mobile agents to flow through the network computers, seeking more interesting machine to perform certain processing.

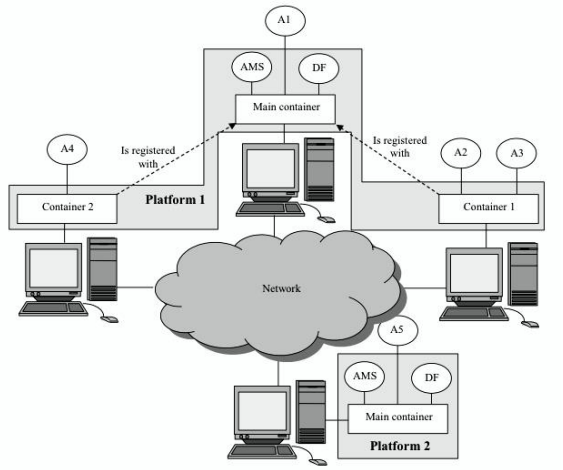

Figure 3 – Physical JADE Agent Platform, from [1]

Software tools with these characteristics have the possibility to make the analysis of large volumes of information, and can be executed and monitored, without requiring manuals tabbing and reading. For this, an agent software tool needs to be modular, to allow a shorter setting and adaptation to the desired environments, covering the concerns of professionals in short time periods; evolving to accompany the necessities, always guaranteeing best results.

According with [1] JADE provides a simplified interface for agents to access the AMS and DF services, simply by sending and receiving messages in standard-defined format. As these implementations are fully

standardized because they are very common, both classes can perform these tasks in a user interface, implemented through methods.

The Agent class in JADE, in the words of [1], is a *superclass*, which allows the software developer to instantiate its own agents. This implies that an agent inherits the characteristics, attributes and behaviors of its base class written in Java. In it, the actions are performed through behaviors, which are tasks that occur asynchronously and concurrently, booked by hidden internal mechanism to the software developer.

III. JADE Agent Software Structure

JADE Agents are instances of that perceive their environment through sensors and act when requested, being that its communication occurs through messaging [1], [6], [7], [10].

By codifying agents, it is necessary to inform the code of which class of JADE environment that the current agent is inheriting its basic structure [1]. In this article, the classes use the base class's Agent, by performance second policy inheritance the base class Agent, performing the second guidelines environmental policies, demonstrated below.

```
public class AgentSearch : Agent
```

In this context, seeking to understand the functioning of a system, its structure was divided into three distinct processes [1]:

Whenever a JADE agent is instantiated in a container, agent for acting Agent Controller automatically runs the *setup( )* method overload of the agent class, which is tasked to perform their initialization, as demonstrated by the following code fragment:

```
public override void setup()
```

During the execution of this method, operations that creating the agent description are performed so as to make your registration on Directory Facilitator. After these procedures, are initialized the behaviors of agents, through the add Behavior method, according in the following example, where it is added the behavior of receiving messages.

```
addBehaviour(new ReceiveMessageFromMControl(args));
```

After initiating the behaviors, they are executed and after finalization the method that performs the agent is executed. In the same form as the *setup ( )*, It is also necessary to put your overload into effect, and this method is called take down, as exemplified in the sequence:

```
public override void takeDown()
{
   FileAgent.Log("Agent" + getLocalName() + " has been
finished...");
   this.takeDown();
}
```

IV. JADE Agent Software Behaviors

The agents behaviors are actions that they have execute in face of a perceived stimulus in the environment [6], [7], as an incoming message, for example. Can be initialized from the *setup method ( )* as already discussed agent, but they can also be initialized at any moment from other behavior [1]

In its structure, basically reside two implementations for the behavior to perform actions. The first is by the method *action( )*, which owns the code that will be executed when the behavior is activated. The second implementation is responsible for finalizing the behavior and is implemented by the method *done( )*, It is named after the implementation of behavior, be it for having achieved an objective or by finalization of itself as defined implementation by type of behavior, described in the sequence.

"One Shoot" Behavior: The simplest type is implemented in a behavior. The method *action( )* the agent behavior is executed only once and at the end, the code is finalized directly. It, does not appear the method *done( )* as will be described in other behaviors that this always returns the logical value true by the end of its execution.

```
public class Name_Behavior : OneShotBehaviour
{
   public override void action()
   {
      // Do α
   }
}
```

"Ciclic" Behavior: This behavior never finalizes its execution, executing the action whenever it is activated. As opposed the previous type of behavior, this behavior always returns a logical value false.

```
public class Name_Behavior : Ciclic
{
   public override void action()
   {
      // Do β
   }
}
```

"Simple Behavior" Behavior: Is the behavior commonly used in systems development agents with the platform JADE. Have the possibility to perform different operations depending on your state, or the way it is activated; according demonstrates the code fragment below. It can be observed the overdid method *action( )* (1), and tests of conditions for execution of tasks in (2) e (3) and finalization of behavior in(4) with the execution of the method *done( )*.

```
public class ReceiveReplies : SimpleBehaviour
{
(1) public override void action(){
(2) if (conditionA == "value"){
       // Do γ
    }
    else{
(3)    if (conditionB == "value"){
            // Do δ
       }
    }
}
(4)  public override bool done(){
        return completed;
     }
}
```

V. JADE Agent Software Messaging

Messages in JADE follow the standards and rules set by FIPA, available at www.fipa.org. The message exchange mechanism is executed a synchronously between the agents. Each agent has a kind of "p.o. box" where the messages are received and every receiving message is notified to the agent, according to what presents the Figure 4 constructed from [1] that presents the form through which is effected the exchange of messages among the agents.

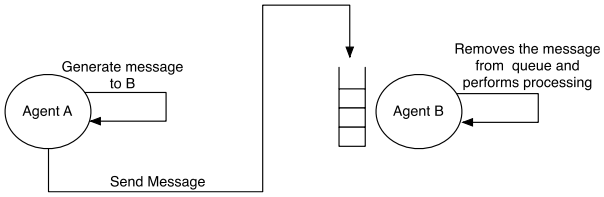

Figure 4 – Physical JADE Agent Platform, from [1]

The code fragment demonstrated below presents the form through which a message is mounted and sent. Initially, a message object is created (1) from the class, which has the structure that enables the handling message objects. Following are inserted in the message the receiver (2), the sender (3) and the content (4). After that, the message is sent (5) from the agent to your receiver.

```
(1)ACLMessage myMsg = new ACLMessage(ACLMessage.INFORM);
(2)myMsg.addReceiver(destination);
(3)myMsg.setSender(sender);
(4)myMsg.setContent(content);
(5)myAgent.send(myMessage);
```

For the agent to be able to put into effect the receiving of messages, it is also necessary to create a class-based Message Template method, as described in (1). So, is effected the receipt of the message in (2). In the case of system developed, one of the agents there which is the sender of the message to be able to realize the treatment of the data according can be seen in (3).

There is also the possibility to perform the processing of the personal data, according to the content of the message that was received by the agent, triggering another behavior. It is possible to be open the content of a message via the method *getContent( )*.

```
(1)MessageTemplate receivedMsg = MessageTemplate.MatchPerformative(ACLMessage.INFORM);
(2) ACLMessage myMessage = myAgent.receive(receivedMsg);
(3) if (aCLMessageReplay.getSender().getLocalName() == "sender"){
    // Performs the processing of the message
}
```

VI. Case Study: Design of an Agent Software With a Behavior to Load DICOM Images Data

Structuring an Expert Agent System Prototype to Support Dental Prosthesis Design Decisions. The prototype of an expert agent system to support dental prosthesis design decisions will consists of three autonomous agents, that will can exchange messages one with other and understand concepts in an ontology described in OWL format organized with Protege Ontology software. These agents will be denominated as Agent Master Control, Agent List File and Agent Search. This model of agents based software was worked firstly in [12] and updated in [13].

Specifically, the "Agent List File" will be responsible for read and load the files recorded in a directory. This files are generated by tomography and magnetic resonance equipment's, in the DICOM format, making possible the image processing according to the necessity, extracting characteristics or information that will be translated or shared with other systems. The DICOM standard, adopted from 1993, made the image processing treatment by algorithms since the information obtained directly from the hardware, to support health professionals [14], [15], [16].

These agents that will be developed in this first version of the expert system software are classified as "simple reactive agents", by the fact that the agents choose their actions on the basis of on his present understanding of the fact, derived from environmental stimulus, discarding operations coming from their historical frame of reference. There are plans to update the agent's behaviors of the next version of this expert system to a level of basis reasoning.

On agents' expert system loads, the main container that has agents is instantiated. Then, a control agent of proper JADE present in the framework so called Agent Controller. It instantiates all the agents in the container and activates them, by its *start( )* the container method. Once the method is instantiated and activated, the agent runs your own *setup( )* behavior and begin work.

Figure 5 illustrates the inner working of expert system agent software prototype. After the agents initiating, the "Agent Master Controller" request to the "Agent List File" to list the files that are in directory intended for DICOM files. This agent will returns to the "Agent Master Controller" a list with all files founded. Receiving this list, the "Agent Master Controller" opens each file, read it, extracts necessary data, performs its conversion to a format that will be used by the expert system and sends them, to the "Agent Analyzer". Receiving this information, the "Agent Analyzer", performs the information analyzing by requested criteria and returns it result to the "Agent Master Controller". After, the agents based expert system processing is completed.

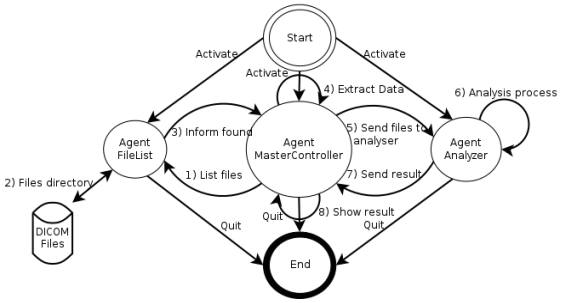

Figure 5 - Structural Diagram of Agent Expert System Prototype adopted from [12] and [13].

The agent that will be the sponsor by the analysis, the "Analyzer Agent", to perform it work will need data stored in ontology format. To improve this analysis behavior in agent software will be designed an ontology that provides patterns to the software agent analyzer behavior. With these patterns, the agent will have a prosthesis details collection, and will be able to "understand" different kinds of dental prosthesis parts needed for each situation.

VII. Conclusions and Results

This article has presented a research made to know the viability of develop an agents expert system with ontologies, to support DICOM image load and future analysis by agents based software. This research is a portion of a major Pontifical University of Parana, which is under design and development by Products and Systems Design and Development research group. This project that aims to make dental images treatment, to support design decisions in dental prosthesis processes. It shows that is incontestable the importance of information systems to support any kind of professionals during decision making processes. This need becomes more evident when have risk of errors that can seriously compromise human health or esthetics. Nevertheless, specifically in process of images portfolio management, like related DICOM format, where it becomes the need of rendering aspects that are not phrase. Will make necessary the interpretation of the graphical information with highly specialized skills of professionals.

About the agent software, its source code is presently under development. In first tests at the laboratory, shown that different agents can analyze the same DICOM images set at the same time, without an agent causes disturbs in the search and analysis process of another agent are running in an asynchronous way.

Under this scenario, aims that the agent systems can better assist in the correct classification of information precisely by the fact of effectively process large volumes of data by its characteristics of work in autonomous, modularized and associative way. Each agent of software is unique, but works by communicating (asynchronously, by predefined communication protocol) with the others, extracting information as request, a second agent, that passes this information to be treated by one third agent that can understand this kind of data, and so forth. This process might be concurrent and the professional with his experience skills will be able to choose the best result or even make a composition thereof.

About the JADE framework, is an alternative for developing agents based software, to meet FIPA specifications, guaranteeing interoperability with other systems which operating under this standardization, achieving a greater level of integration. In this sense, can be perceived that the ability of adaptation, reuse inherent in agents systems, in conjunction with the use of development and tools like JADE makes possible the expert software applications development which can be executed in most operating systems, allow for the development of software tools applied to the actual dental prosthesis professionals requirements. Is estimated that this new conceptions for expert software applications development, allied to the evolution of new technologies can build knowledge bases and practices, to feed processes of expert software developing with more efficient and flexible models.

References

[1] F. Bellifemine, G. Caire, T. Trucco, G. Rimassa. JADE Programmers Guide. Available at Jade Developers SVN Service at <https://avalon.cselt.it/svn/JADE/trunk> (2012).

[2] J. M. Corchado, R. Laza, L. Borrajo; J. C. Yañez, M. Valiño. Increasing the Autonomy of Deliberative Agents with a Case-Based Reasoning System. International Journal of Computational Intelligence and Applications. V 3, N 1 (2003).

[3] D. J. Czelusniak. Agents software architecture based for of information sources portfolio management on the web. Doctoral Thesis in Post Graduate Program in Production Engineering and Systems. Federal University of Santa Catarina. Florianópolis, Santa Catarina, Brazil (2013).

[4] D. J. Czelusniak. Agents system proposal to support skills management. Master Dissertation in Production Engineering. Federal Technology University of Paraná. Paraná, Brazil (2007).

[5] A. C. B. Garcia, J. S. Sichman. Agentes e Multiagentes, in: Sistemas Inteligentes: Fundamentos e aplicações. Organização Solange Oliveira Rezende. Barueri, SP. Editora Manole (2005).

[6] E. Rich, K. Knight. Artificial Intelligence. Ed. McGraw-Hill. International Edition. Singapure (1991).

[7] S. Russel, P. Norvig. Inteligência Artificial. 2ª. Edição. Editora Elsevier. Rio de Janeiro, RJ (2004).

[8] The Foundation for Intelligent Physical Agents. FIPA. Information at: <http://www.fipa.org/>

[9] M. Greaves. Semantic Web 2.0. IEEE. Intelligent Systems (2007).

[10] S. Liao. Technology management methodologies and applications: A literature review from 1995 to 2003. Elsevier. Technovation 25 (2005).

[11] A. C. B. Garcia, J. S. Sichman. Agentes e Multiagentes, in: Sistemas Inteligentes: Fundamentos e aplicações. Organização Solange Oliveira Rezende. Barueri, SP. Editora Manole (2005).

[12] D. J. Czelusniak. Agents' software architecture based for information sources portfolio management on the web. Doctoral Thesis in Post Graduate Program in Production Engineering and Systems. Federal University of Santa Catarina. Florianópolis, Santa Catarina, Brazil (2013).

[13] D. J. Czelusniak. Agents' system proposal to support skills management. Master Dissertation in Production Engineering. Federal Technology University of Paraná. Paraná, Brazil (2007).

[14] A. L. Szejka, M Rudek, O. Canciglieri Junior. A Reasoning System to Support the Dental Implant Planning Process. 19th ISPE International Conference on Concurrent Engineering. Trier, Germany (2012).

[15] D. Grauer, L.S.H. Cevidanes, W.R. Proffit. Working with DICOM craniofacial images. American journal of orthodontics and dentofacial orthopedics: official publication of the American Association of Orthodontists 136 (2009).

[16] R. N. J. Graham, R.W. Perriss, Scarsbrook AF DICOM demystified: A review of digital file formats and their use in radiological practice (2005).