3ª ed

MIM

Optimization of an Agent Based Clinical Data Retrieval System

José Hilário Patriarca de Almeida

MESTRADO EM
INFORMÁTICA MÉDICA
2º CICLO DE ESTUDOS

OUT|2012

3ª ed

MIM

Optimization of an Agent Based Clinical Data Retrieval System

José Hilário Patriarca de Almeida

MESTRADO EM
INFORMÁTICA MÉDICA
2º CICLO DE ESTUDOS

ORIENTADORES:
Ricardo João Cruz Correia - FMUP
Pedro Manuel Vieira Marques - FMUP

OUT|2012

Contents

Acronyms

AID Agent Identifier

ACC Agent Communication Channel

ACL Agent Communication Language

AP Agent Platform

API Application Programming Interface

BDI Belief-Desire-Intention

CIDES Department of Department of Health Information and Decision Sciences

CINTESIS Center for Research in Health Technologies and Information Systems

DAI Distributed Artificial Intelligence

DB Database

DF Directory Facilitator

DIS Departmental Information Systems

FIPA Foundation for Intelligent Physical Agents

FMUP Faculdade de Medicina da Universidade do Porto

JADE Java Agent Development Framework

JAF Java Agent Framework

MAID Multi-Agent System for Integration of Data

MAS Multi-Agent System

MASS Multi-Agent System Simulator

PDF Portable Document Format

VEPR Virtual Electronic Patient Record

List of Tables

List of Figures

Acknowledgments

To my supervisors Prof. Ricardo Cruz Correia and Msc. Pedro Manuel Vieira Marques, who's dedications, motivation and patience helped me overcome the obstacles found.

To all my colleagues at CIDES and CINTESIS for their availability and help.

To my family and friends for all their support.

A special thanks to Ana, Bruno and Shiva for the constant inspiration. "You are never alone or helpless. The force that guides the stars guides you too".

The work presented in this MSc thesis has been funded by FEDER funds (Programa Operacional Factores de Competitividade COMPETE) and by National funds (FCT Fundacao para a Ciencia e a Tecnologia) through project SAHIB - Enhancing multi-institutional health data availability through multi-agent systems [PTDC/EIA-EIA/105352/2008].

Personal Motivation

The work on this thesis is the development of a personal interest in the way healthcare information is created, stored and then used. In everyday use we see multiple information systems storing patient data and the integration of all this data is a current and interesting area of work.

Through my work within CIDES and CINTESIS namely on the optimization of an agent based system for the retrieval of clinical data (MAID), in the development of HL7 based integration solutions, in the organization of workshops related to HL7 and openEHR and with my current work within the scope of the SAHIB project with the aim of enhancing multi-institutional health data availability through multi-agent systems, and future work on a project involving the creation of openEHR based healthcare information system I aim to solve problems related to interoperability and integration of health data so that healthcare professionals may have access to relevant patient data during the delivery of care.

Abstract

A Virtual Electronic Patient Record (VEPR) has been deployed at Hospital de S. João and is running since 2005. One of its modules is a Multi-Agent system for the Integration of Data (MAID) that provides automatic report retrieval from departmental systems.

The current version of MAID uses a static interval for the agent's report references retrieval actions. As each department differs in report production rate throughout the day there was a need to optimize and adapt the report reference retrieval actions to each departmental system profile.

Given the sensible nature of the context this optimization could not be done within the deployed system. Hence a simulation environment was developed that enables the testing and comparison of different optimization options using properly anonymised real past report data as reference.

The main contributions of this work were the development of the agent based simulation environment and the development of the adaptive Scheduler agent that allow the agent's actions for the retrieval of report reference to adapt themselves to the departmental report production patterns. A web interface for MAID agent's configuration was also developed.

A pilot of the simulation environment was deployed with the Scheduler Agent and the necessary modifications made to MAID, enabling the execution of simulations and comparison of simulation data between the currently deployed system and possible optimization paths.

The use of a simulation environment is a viable path for the development and testing of optimization algorithms as it allows the comparison of variables related to an optimization process. Using past data as a way to optimize a retrieval process is also a viable path as it allows the adaptation of the agent's actions to the departmental profiles. The development and deployment of a Scheduler Agent enables the optimization of the report retrieval behaviors.

Resumo

Em 2005 um Registo Clínico Virtual foi instalado no Hospital de S. João. Um dos seus módulos é um sistema multi-agente que executa a recolha automática de relatórios de diversas proveniências departamentais.

A versão atual do

The current version of MAID uses a static interval for the agent's report references retrieval actions. As each department differs in report production rate throughout the day there was a need to optimize and adapt the report reference retrieval actions to each departmental profile.

Given the sensible nature of the context this optimization could not be done within the deployed system. Hence a simulation environment was developed that enables the testing and comparison of different optimization options using real past report data as reference.

The main contributions of this work were the development of the agent based simulation environment and the development of the adaptive Scheduler agent that allow agent's actions to adapt themselves to the departmental report production patterns. A web interface for MAID agent's configuration was also developed.

The use of a simulation environment is a viable path for the development and testing of optimization algorithms as it allows the comparison of variables related to an optimization process. Using past data as a way to optimize a retrieval process is also a viable path as it allows the adaptation of the agent's actions to the departmental profiles. The development and deployment of a Scheduler Agent enables the optimization of the report retrieval behaviors.

Outcomes

Main Outcomes

- J. Patriarca-Almeida, P. Vieira-Marques, R. Cruz-Correia. Simulation Environment for the Optimization of the Data Retrieval Capabilities of an Agent Based System in a Healthcare Setting. Presented at the 5th International Conference on Industrial Applications of Holonic and Multi-Agent Systems - HoloMAS 2011, August 29 - 31, 2011, Toulouse, France. Published in Lecture Notes in Computer Science, 2011, Volume 6867/2011, 124-132
- J. Patriarca-Almeida, P. Vieira-Marques, R. Cruz-Correia. Optimization of an agent based clinical data retrieval system. Presented at the 7th Iberian Conference on Information Systems and Technologies - CISTI 2012, June 20-23, 2012, Madrid, Spain. Published in Information Systems and Technologies, Proceedings of the 7th Iberian Conference on Information Systems and Technologies, 2012, Volume 1, Tome 2, 890-895

Other Outcomes

- Development of a failsafe report retrieval software for MAID;
- Organization and lecturing on workshops related to interoperability, HL7 and openEHR.

Outline

This thesis is organized into five chapters as described in the following points:

- The first chapter is the introduction and includes a background section that is further divided into two subsections. The first subsection includes an overview of agents and multi-agent systems (MAS), of the Java Agent Development Framework (JADE) and of simulation in the MAS domain. The last subsection includes a description of the Virtual Electronic Patient Record (VEPR) including its MAS, the Multi-Agent System for Integration of Data (MAID) platform. The chapter concludes with two other sections that contain the definition of the research questions and the objectives of this thesis;

- The second chapter details the development of the a simulation environment created for the monitoring of MAID execution and measurement of several variables related the optimization work. It includes a description of the architecture of the simulator and a detailed description of its components. It also includes a description of the deployment process of a simulation;

- The third chapter details the development of the adaptive scheduling algorithm. It details the changes made to MAID to accommodate the optimized scheduling process. It also includes a description of the development of the Scheduler Agent, including its behaviors for schedule generation and messaging;

- The forth chapter details the discussion of the simulation results, focusing on resource occupation and on the efficiency of the optimization;

- In the fifth chapter we can find the main findings of this thesis as well as a discussion on its limitations, of future work and the closing arguments.

Introduction

Chapter 1

Introduction

As the work on this thesis involved the optimization of an agent-based system it begins with a background that includes the description of what agents and multi-agent systems are. It follows with a description of the framework used to develop MAID, the JADE MAS development framework, as well as the standards at its origin, the Foundation for Intelligent Physical Agents (FIPA) specifications. This is followed by a detailed description of the VERP and of one of its main modules - MAID. Concluding this chapter we have the stating of the research questions and of the objectives.

1.1 Background

1.1.1 Agents and Multi Agent Systems

Overview

Agent theory and the development of Multi-Agent Systems has as its foundation the study of distributed artificial intelligence (DAI) in the 1970's. DAI encompasses many fields as varied as computer science, philosophy, sociology or economics.

As a broad subject it is difficult to precisely define this field but the following definition by Gerhard Weiss [1] is a good starting point: " DAI is the study, construction, and application of multiagent systems, that is, systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks". In the following sections the concepts of intelligent agents, multi-agent systems, ther

Intelligent Agents

The concept of agent evolved as there was a necessity for many computer applications to adapt themselves to their working environment in order to satisfy their design goals. There was a need for applications that were still functional while operating in unpredictable environments where

adaptation to failure was a required feature.

From this was born the concept of an agent as "a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objective"[4].

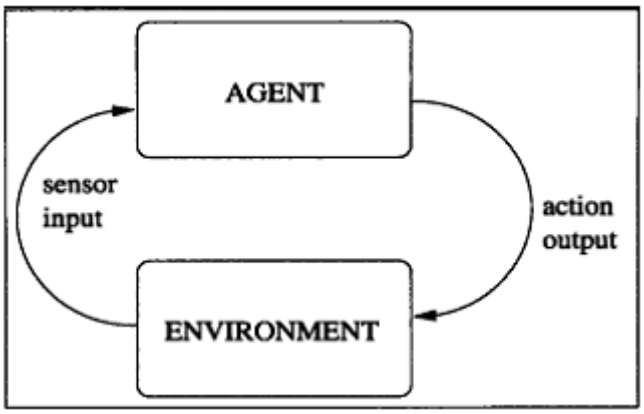


Figure 1.1: Agent interactions with its environment.[1]

Two main items arise from the definition:

First, as seen figure 1.1, agents have a degree of interaction with their environment in a way that, although not having complete control of it, they can influence it. Due to interactions of the agents with their environment and the fact that they can only partial control it, we can say that usually the environment can be thought as non-deterministic[1].

Second, autonomous action means that agents are able to operate without direct intervention of humans or other computer systems in that they have control over their own internal state and over their behaviors[4]. They are are however somewhat limited by a set of actions they can execute and that they can use to possibly influence their environment. The way an agent acts - what action it chooses to execute - depends on its decision making system that takes into account a series of pre-conditions that must be fulfilled and environmental properties that can affect it, so that a specific action is chosen and executed.

Evolving the concept of agent a bit further we can define a few properties other than its autonomy. Taking into account its interactions with the environment, we can see that an agent will need a certain degree of flexibility so it can handle the non-deterministic aspects of what surrounds it. The concept of "intelligent agent" can then be defined as an agent with flexible autonomous action[1]. Wooldridge and Jennings[4] defined a few properties that an intelligent agent must possess in order to be defined as such. They are the following:

- social ability: intelligent agents must be capable of interacting with other agents (and possibly humans) using an agent communication language;
- reactivity: intelligent agents perceive their environment and respond in a timely fashion to changes that occur in it;

- pro-activeness: intelligent agents do not simply act in response to their environment, they are able to exhibit goal-directed behavior by taking the initiative.

For an intelligent agent having a social ability means that in its environment it may encounter other and interact with other agents with the same or different goals. For an agent to achieve its goal it may have to cooperate or negotiate with others.

The reactivity of an intelligent agent appears to be in direct opposition to its pro-activeness but, as with humans, both are an integral part of its function. The reactive function of an agent is directly related to its ability to monitor environmental changes and adapt its behaviors/actions to these changes if necessary and even change its goals based on these changes. In an environment where uncertainty is present, for example, where multiple agents interact with the environment, reactivity is required, as an agent must not blindly execute its actions.

To say that an agent is pro-active is to say that it has goal-oriented behaviors that allow it to fulfill the function that it was designed for. Internally its pro-activeness is represented as a plan or set of plans with defined pre-conditions that are executed to achieve a state defined as a set of post-conditions: its goal.

The development of an intelligent agent involves balancing each of these features. Taken individually they are easy to develop but the design of an agent that takes into account all these facets is obviously a more complex task. As is said "The whole is greater than the sum of its parts".

There are several architectures that try to define how each property is to be implemented.These will be further developed in the following sections pertaining to agent architectures, multi-agent systems and agent communication and interaction.

Agent Architectures

In order to put agent theory into practice as to be able to build agent-based computer systems a series of architectures were envisioned using the concepts of agent theory. One of the most cited definitions of "agent architecture" is given by Maes[5]. He defines an "agent architecture" as "a particular methodology for building such an agent." He further specifies that an architecture can be thought as a collection of modules to be used in construction of agent based systems and that "The total set of modules and their interactions has to provide an answer to the question of how the sensor data and the current internal state of the agent determine the actions (effector outputs) and future internal state of the agent. An architecture encompasses techniques and algorithms that support this methodology."

Agent architectures approach the problem of building agent-based systems in a multitude of ways. Wooldridge and Jennings[4] grouped various architectural approaches.

The first are deliberative architectures. In these the agents have a explicitly represented

symbolic model of the world and their decision process uses pattern matching and symbolic manipulation and is made using logical reasoning. Using these architectures implies having to accurately represent, in a symbolic way, a complex and rich environment, with its entities and processes. It also implies developing agents that have reasoning systems that are able to use the information coming from a complex environment such as real world and produce the desired results from that information.

Another approach are the reactive architectures developed in response to the perceived complexity associated with a deliberative approach that would not allow, for example, to build agents that operate in time constrained environments. In contrast to the deliberative architectures, they don't include a complex symbolic world representation and reasoning systems. The foundation of reactive architectures comes from the work of Rodney Brooks [6][7] that states that agent's behaviors can be generated without symbolic representations, and that intelligent behavior is the result of the agent's interaction with it's environment. In reactive architectures the agent's decision process can be implemented through a series of actions, each corresponding to a task accomplishing behavior. These behaviors take environmental input and are triggered by an event defined by the status of the environmental input. The triggering of events can be simultaneous so a behavioral hierarchy is implemented to handle this - a subsumption hierarchy[6].

Another approach are layered architectures in which the decision process is realized through multiple subsystems, involving reactive and proactive behaviors and resulting in a decomposition of functionality but also increased complexity of the agent systems due to the interactions between the layers [1]

Another approach are Belief-Desire-Intention (BDI) architectures in which the decision process depends on the manipulation of data representing the beliefs, desires, and intentions of the agent. Agent systems based on this type of architecture are developed around the concept that the decision process of an agent revolves around the determination of its goals and how to achieve them, forming a reasoning process for the agents[1]. These agents have: a set of goals is then mapped as the agents intentions, or what it wants to achieve; a set of beliefs that represents the information about it's current environment; a set of desires that represent the options that can be chosen by the agent, that are generated from its current intentions and its current beliefs. Agent actions revolve around a decision process that culminates in the selection of an action to be executed based on its current intentions.

Multi-Agent Systems

Due to the nature of agents and the problems they aim to solve it is common for agents to be organized in Multi-Agent Systems (MAS). In these systems there is a need to develop an infrastructure for agent operation and interaction that take into account individual and group goals and actions. These systems are centered around communication and interaction protocols

that allow the exchange of messages between autonomous agents that results in conversations between agents, enabling them to advance their own goals and/or the goals of a group of agents. These systems also supply directory services that allow agents to find others to communicate with as well as means for the coordination (cooperation or competition) of agents[1]

Foundation for Intelligent Physical Agents Specifications

Due to the need for interoperability between agent systems there was a need to standardize the communication protocols between agents as well as the services and ontologies made available by different platforms. To this end the Foundation for Intelligent Physical Agents (FIPA) created a set of specifications [8] for the development of MAS platforms. They define the infrastructure and language required for open interoperation of agent platforms as this is one of the goals of FIPA.

In the [2] it defines a set of agent management services for the Agent Platform (AP) that may themselves be agents and include, as seen in figure 1.2, the following:

- Directory Facilitator (DF) - provides yellow pages services. An agent registers its services with the DF agent so that other agents may find them. An agent can query the DF for services offered by other agents;
- Agent Management System (AMS) - acts in a supervisory role of the AP and the life cycle of its agents. It maintains a directory of Agent Identifiers (AIDs) for agents registered with the AP. It also offers a white pages service enabling agents to find each other and initiate a communication process.
- Message Transport Service (MTS) - the default communication method between agents on different APs

FIPA also defines an Agent Communication Language (ACL) that is used within an AP for the exchange of messages between agents. It has very well defined semantics and its messages are encoded in a textual form enabling communication between agents running on different platforms [9]. A set of interaction protocols[8] are also defined in the specifications. These protocols (p.e. Request Interaction Protocol, Query Interaction Protocol, Propose Interaction Protocol) define a series of communication patterns between agents standardizing a series of common interaction patterns.

JADE Framework

In the development of MAID an agent development framework was used: the Java Agent Development Framework (JADE). This framework provides middleware functionalities that simplify the development of applications that use an agent based paradigm [10] [4].

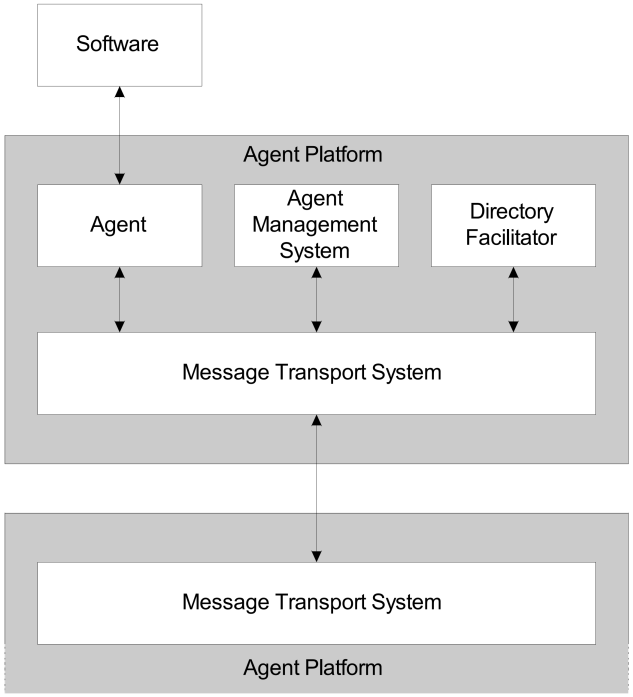


Figure 1.2: Foundation for Intelligent Physical Agents - Agent Management Reference Model [2]

The platform was created in 1998 as a means of validating the FIPA specifications but later evolved into a full middleware platform for the the development of agent applications in compliance with the FIPA specifications, with emphasis on the simplicity and usability of the provided API [10][11]. JADE implements specifications such as the Agent Management Specification including the AMS through an AMS agent, the DF through a DF agent and the MTS through an Agent Communication Channel (ACC).

JADE also provides additional features not included in the FIPA specifications including a distributed, fault tolerant, container architecture, persistent message delivery, security mechanisms, agent mobility support, web-service interaction and graphical interfaces for the AP[10].

Each agent within the a JADE based system is an autonomous entity, having its own thread of execution and using it to control its own life cycle and decide autonomously when to perform which actions[10]. Upon its creation it resides in a agent container that provides all the services described above. It also has an associated set of behaviors that represent tasks that the agent can carry out as seen in figure 1.2.

In JADE these behaviors are developed through an extension of a "Behaviour Class"[11]. This class provides a skeleton common to every task to be performed. It exposes three complementary methods: the *action* method that allows to represent the task to be accomplished by the specific behavior, the *done* is used by the agent scheduler and must return true if the behavior has terminated its execution or false when it has not terminated and the *action* method is to be executed again and the *reset* method is used to restart the behavior.

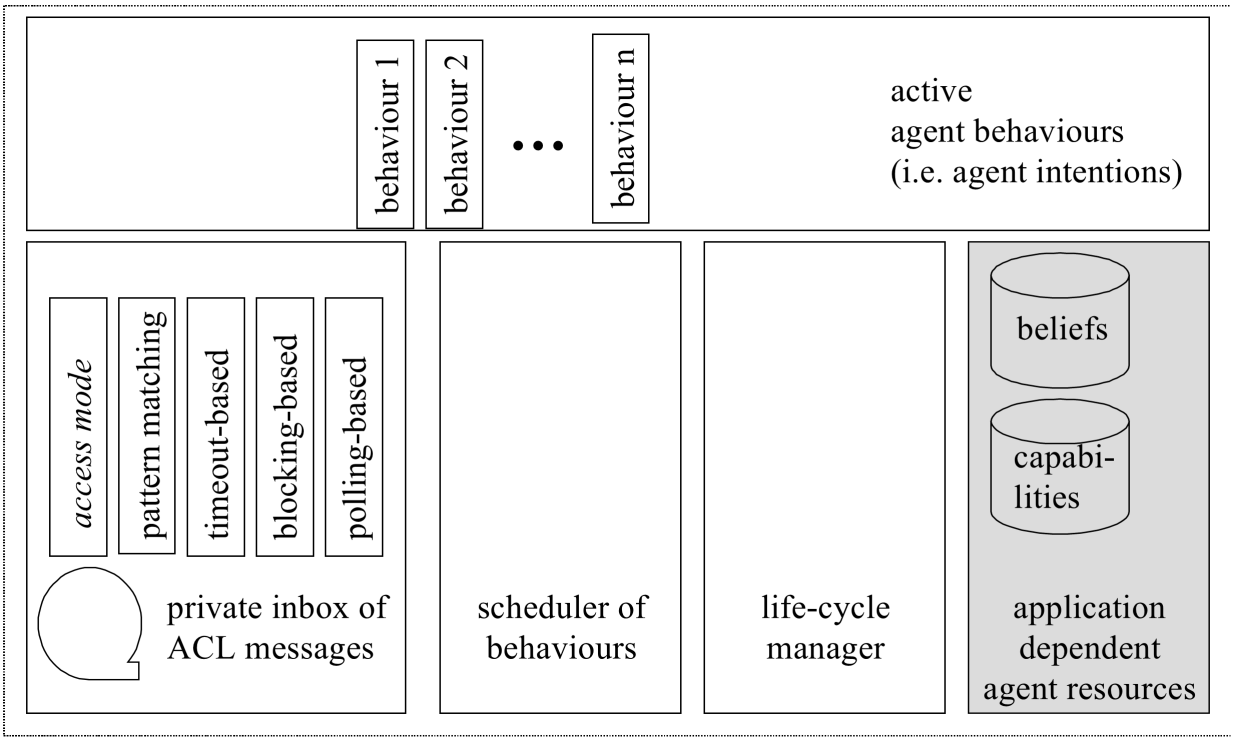


Figure 1.3: Java Agent Development Framework agent architecture [2]

Simulation Environments for Multi-Agent Systems

As the MAS field began to grow there was a necessity to develop tools for measuring MAS system performance. Two common research tools for the study of MAS are benchmarks which are standardized tasks and testbeds which are environments in which agent systems can be studied[12]. These tools enable the comparison of different systems, highlighting differences in performance between systems.

With these tools the researcher introduces variation within the benchmark or testbed and measures the results and impact that the introduced changes have on system performance. There may be concerns about the viability of these approaches as they may only be viable for a subset of agent systems where the complexity of the system is low. Also to note is the degree of complexity of the real world deployment and how it relates to the complexity of the simulation environment and the benchmarks executed.

Two approaches may be used in order to take advantage of the performed benchmarks. One is to implement simulations that can be systematized but also allow for the results of the experiment to be applied to the real world systems. Other is to conduct tests on the real systems and observe their behavior.[12]. On this basis one can find frameworks that are developed to simulate agent systems while also enabling the evaluation of such systems.

One of these is Java Agent Framework (JAF)[13] that was designed to "allow an agent's behavioral logic to perform without the knowledge that it was operating under simulated conditions"[13].

It uses a component-based design to attain the separation of agent-dependent behavior logic from the underlying support code and implements a series of features that a simulation environment should have such as allowing for real-world complexity, a well defined model of time and allow deterministic experimentation of multiple scenarios. Within this work was developed a Multi-Agent System Simulator (MASS)[**?**] that was developed aiming to solve two conflicting objectives: the "ability to accurately measure the influence of different multi-agent coordination strategies in an unpredictable environment" and "to realistically model adaptive behavior in multi-agent systems within a static environment"[**?**].

In Closing

An agent is a computer system, capable of autonomous action in some environment. It can sense its environment and execute a set of actions in order to meet its design objectives and in the process may modify its environment.

To aid the development of agent based systems and their interoperability FIPA defined a series of specifications that were implemented by various frameworks/platforms, one of which being JADE. JADE enables the development of agent based systems providing a FIPA compliant framework that eases the development of MAS.

The use of simulation environments within the MAS field allows for comparison of different systems and more specifically, in the case of the work of this thesis, the comparison of changes within the same system.

1.1.2 Multi-Agent System for Integration of Data

With the necessity for integration of data in a healthcare context a project for integration of clinical record data was started at Hospital de S. Joao(HSJ). This was needed as many of the departmental information systems (DIS) were not designed to be inter-operable. These DIS are heterogeneous regarding the communication protocols, database systems and file formats. As there were benefits in the integration of reports from multiple departments in a single visualization interface a Virtual Electronic Patient Record (VEPR) was developed and installed in 2005 at HSJ[3] aiming to retrieve and integrate patient data within the hospital. A general architechture of the VERP can be seen in figure 1.4.

One of the main functionalities of the VERP is document retrieval. This task is preformed by a Multi-agent system named Multi-Agent system for Integration of Data (MAID)[14]. A general architechture of MAID can be seen in figure 1.5). MAID was developed using the Java Agent Development Framework (JADE)[11], described in the previous section, that complies with the FIPA specifications [8].

A Multi-agent system approach was chosen due to the agent's flexible communication capabilities and autonomous behavior capacities [3] and because they are known for addressing issues

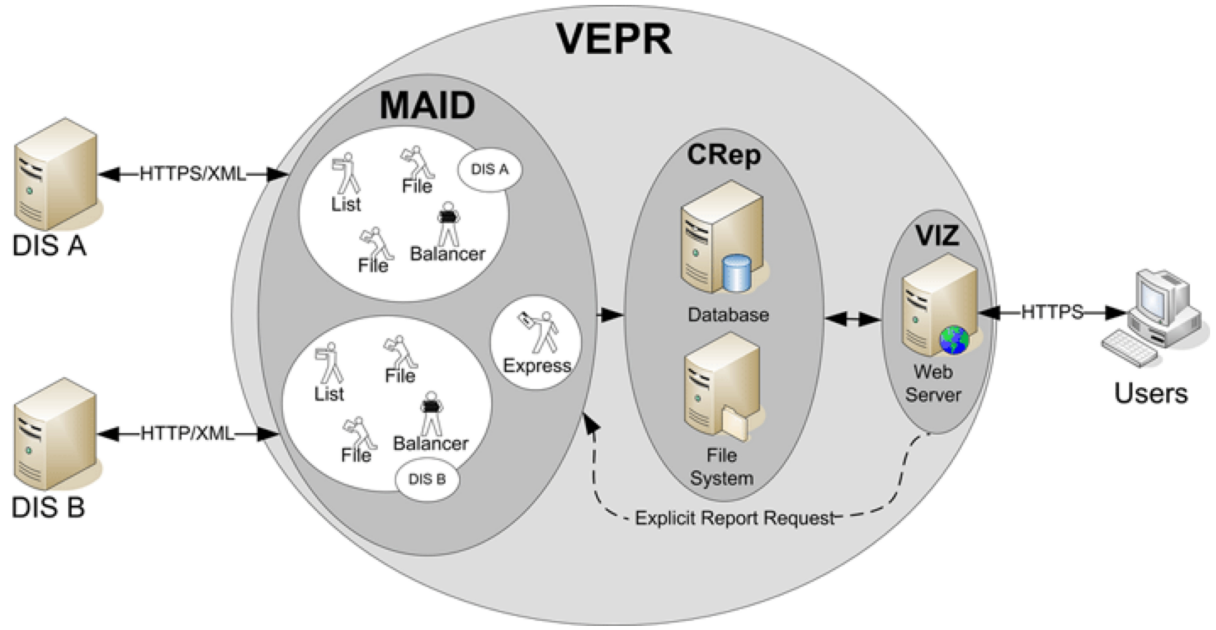


Figure 1.4: General architecture of the Virtual Electronic Patient Record (VEPR) system including the Multi-Agent system for Integration of Data (MAID), the Visualization (VIZ) module and the Central Repository (CRep)[3].

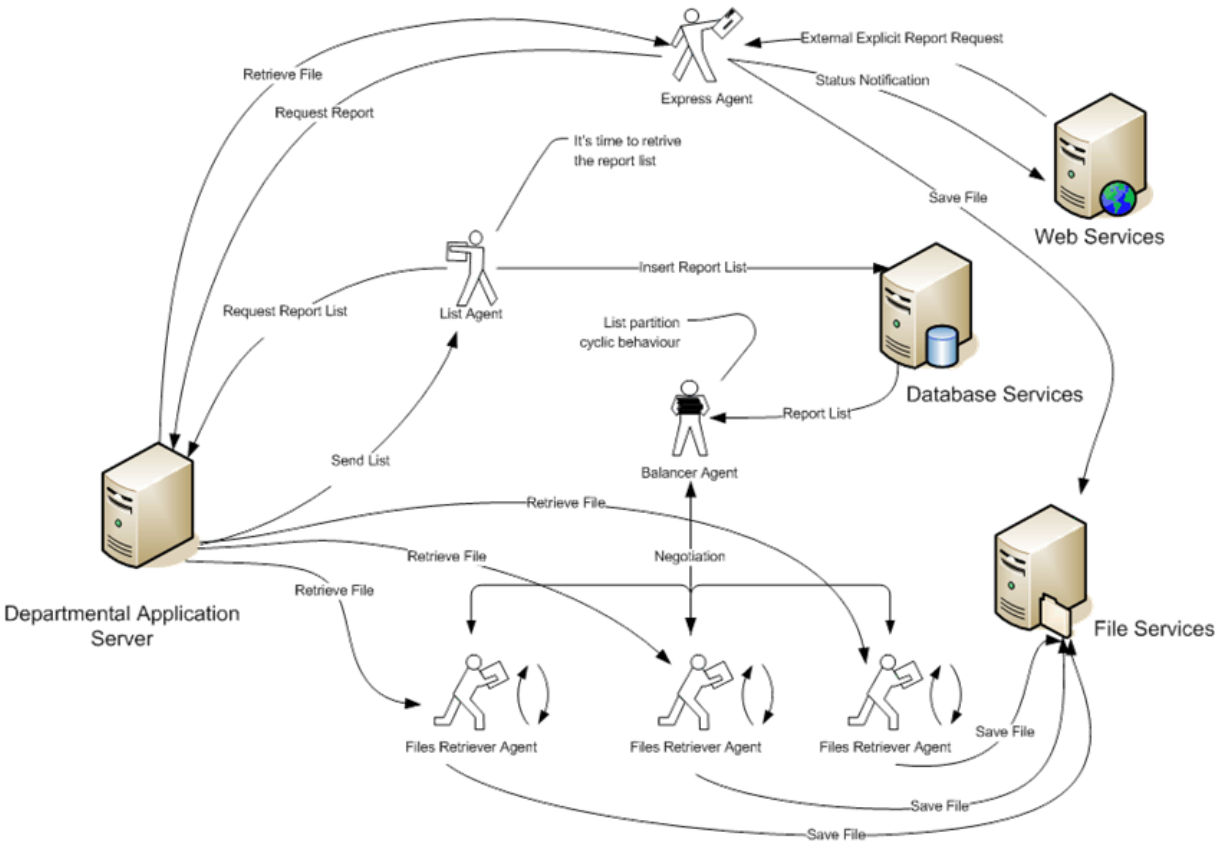


Figure 1.5: General architecture of the Multi-Agent system for Integration of Data (MAID)

related to distributed information sources [15, 16] proving their use in a healthcare setting [17].

MAID contains agents for system setup (Control agent), retrieval of report references (List agents) and retrieval of the report files (Balancer and File agents), each departmental system having a set of List, Balancer and File agents associated with it (see Fig. **??**). The Control agent is responsible for loading the other agents configurations and for their creation.

The MAID agents are as follows:

- Control Agent - Has a system setup behavior that manages the startup of the agent platform;
- List Agents - Each department has a corresponding List agent. These agents have a report list retrieval behavior that communicates with the departmental system and retrieves a list of report references of the reports produced during a specific requested timeframe,
- Balancer Agents - Each department has a corresponding Balancer agent. These agents have a balancer behavior that distributes the report references retrieved by the departmental List agent to a series of File agents. They are also responsible for the creation of the File agents corresponding to each department,
- File Agents - Each department has a set of file agents. The balancer agent corresponding to each department creates these agents. These agents have a report retrieval behavior that, upon distribution of the report references by the balancer agent, communicates with the departmental system for the retrieval of the actual report file.
- Express agent - This agent listens for report file requests that are triggered by the report visualization module when a report listing is available but the report file has nor been retrieved. It has an express retrieval Behavior that requests the report file from the departmental system.It does not take part in the cyclic nature of the report retrieval process. Its purpose is to respond promptly to report requests that only have their references in the system.

As seen if figure 1.6 At system setup the Control agent loads the configurations, contained in properties files, for each of the departmental systems and creates the respective List and Balancer departmental agents based on the their corresponding template. Each Balancer agent is then responsible for the creation of the corresponding File agents for each department.

After MAID startup the List agent of each department will, with a List Retrieval Behavior, autonomously and periodically request a list of reports produced during a given interval. This interval is defined by the last report reference request and a variable that defines the periodical report retrieval, defined statically in the properties file for each department.

MAID's environment can be thought of as mostly accessible in a way that MAID is aware of the state of the external departmental systems status. The environment can also be thought as

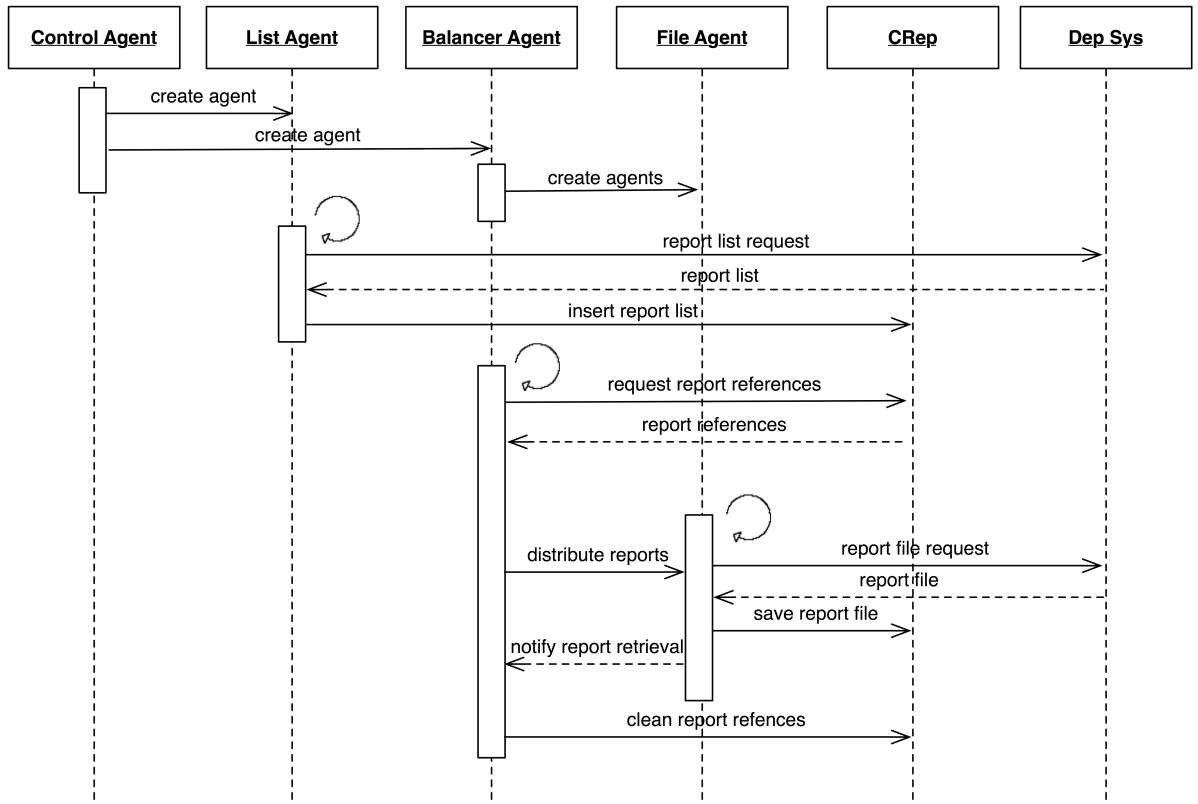


Figure 1.6: Sequence diagram of the Multi-Agent system for Integration of Data (MAID)

dynamic and non-deterministic due to the nature of the departmental systems as they can for example, not be available for report retrieval due several circumstances, thus having an impact in the agent's retrieval capabilities.

This system has been working for years in a very sensitive environment in a healthcare setting. Due to this fact, it is not advisable to test optimization solutions within the production environment.

To test different optimization options a simulation environment was developed that will enable the study of the influence that different variables, for example, the number of reports produced per hour, have in the retrieval of report references, resulting in the development of a scheduling algorithm that will enable the agents to adapt themselves to the report production patterns reducing the strain that the repeated requests may cause to the external systems.

1.2 Research Questions

There are varying patterns in the production of reports throughout the day and on the weekends as seen in figure 1.7. The represented patterns correspond to two departments, and it can clearly be seen the differences within a department and between departments. These are two of the production patterns found, as they vary for each department, accompanying departmental report production. As there is an increased focus on the importance of information quality,

including availability and timeliness as well as time savings, in the overall system quality [18] there was a need to optimize the current system. One of the areas of possible optimization was the report retrieval process by adapting it to each of the departmental production flows. The motivation is then to optimize the report list retrieval behavior.

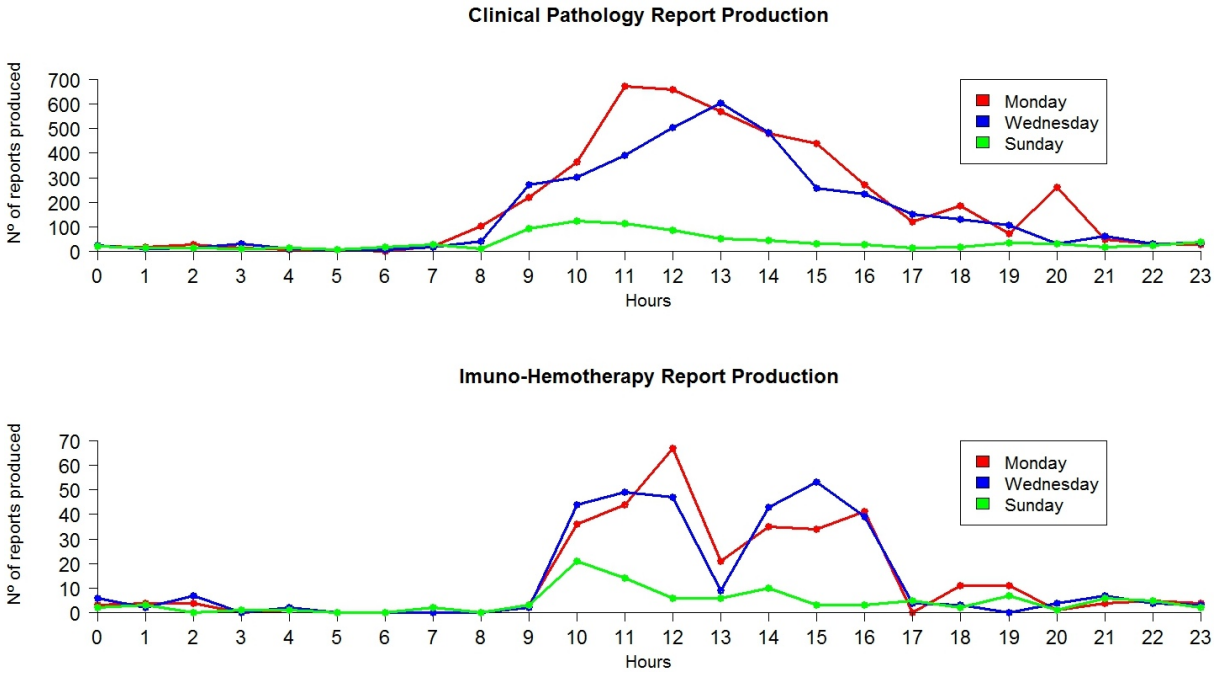


Figure 1.7: Report production of two different hospital departments on the 41st week of 2008 for Monday, Wednesday and Sunday. Above the Clinical Pathology department, bellow the Imuno-hemotherapy department.

To that end a few questions had to be answered:

- How to better adapt the report list retrieval process to the departmental systems profiles;
- How to alter MAID agents behavior so that they take into account an optimized report list retrieval schedule;
- How to test, monitor and evaluate different optimization paths while maintaining a simulation scenario similar to real world use.

1.3 Objectives

The main objective of this work is the optimization of the MAID report list retrieval process.

To this end three sub-objectives have to be attained:

- The development of a simulation environment that will allow the comparison of different simulation executions.

- The development of a Scheduling Agent that will generate hourly schedules for each service based on past report production patterns and weights attributed to a series of relevant variables.
- The development of agent behaviors that allow the communication of the generated schedules to the List Agents altering the report list retrieval behavior from static to dynamic, adapting it to the report production patterns of their corresponding department.

Agent Environment Simulator

Chapter 2

Agent Environment Simulator

In this chapter will be described the simulation environment that allows us to compare different optimization routes.

As we intend to replicate as much as possible the real world scenario, we include in the simulation environment two key elements: the departmental systems and the agent system so that it can be possible to accurately reproduce the departmental systems production patterns and the report retrieval process.

The aim is then to develop a simulation environment that will allow the comparison of different simulation executions as it is not possible to test different optimization solutions with the currently deployed system.

2.1 Simulator Architecture

For the accurate simulation of the agents environment there was a need to replicate the database and filesystem structure of CRep. This implied using a database with the same structure as the one used by MAID in the production environment as well as the same filesystem structure used to store the report files.

A migration of the database procedures used in MAID was performed as well as changes to the agents configuration files regarding the database connections, filesystem paths and departmental system's URI addresses (corresponding to the previously created departmental scripts) so that the agents can properly communicate with the simulated departmental systems.

There was also the configuration of the database connections to a simulated external database system called SONHO that is used to validate patient's demographic information in the reports. The filesystem structure of CRep (see figure 1.4) was also duplicated so that MAID's File agents can properly store the report files. The organization of the CRep filesystem is based on the patient identification number, each patient having a directory in the filesystem[3].

The following subsections will detail the components of the simulation environment that can

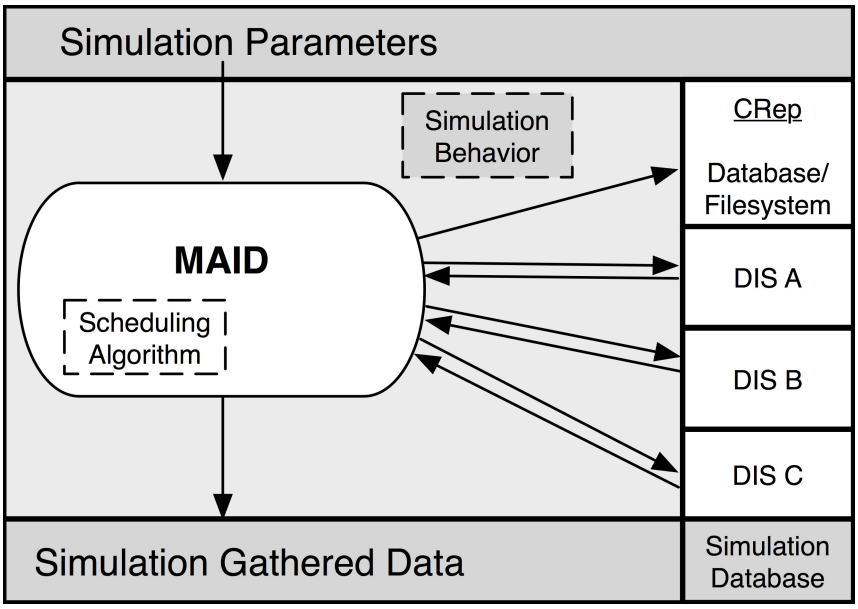


Figure 2.1: Simulation environment showing the simulation parameters, the simulation behavior, the MAID agent system, the Central Repository (CRep), the departmental systems (DIS) and the simulation data.

be seen in figure 2.1.

2.1.1 Departmental Systems Simulation

As it is not possible to use the real-time data from the different hospital departments there was a need to simulate their report production patterns. This was achieved through the creation of a database (HSJ_SIM) that will have the report data from the production database (HSJ).

Using past report production data is essential to understand the production patterns and will enable a similar experience to the actual retrieval process. The HSJ_SIM database has a subset of the real data, properly anonymized, that will enable us to use daily data for up to two years of report production, from 2007 to 2009.

There was also a need to create a series of scripts, one for each department, that will return the list of reports produced for a given time period. These will be used to simulate the returned XML report references (see Fig. 2.2) that an agent receives upon each report list request (see Fig. **??**).

In these scripts a database query to HSJ_SIM is executed for the corresponding department, using the interval supplied by the List agent's request, and a XML file will be constructed from the query results. The exchanged XML files have a standard format used throughout all the departments for the request and retrieval of report references (see figure 2.2).

Due to the clinical information contained in the actual reports it is not possible to collect them for the actual simulation. The report file reference contained in the "document" element (see Fig. 2.2) is a generic html or pdf document with content similar to the actual reports.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE MessageHSJXXI SYSTEM "mensagemHSJXXI_4.dtd">
<MessageHSJXXI id="1300202511654" emission-date="15/03/2011 14:03:24" dtd-version="4">
        <response emitter="3" request="1300202511654">
                <reportList initial-date="15/10/2008 10:03:00" final-date="15/10/2008 10:03:25">
                        <report id="5866909" type="Analise de Patologia Clinica" emission-date="15/10/2008 10:03:22">
                                <patient birth-date="05/06/1964" sex="F">
                                        <name>Nome Generico do Doente 530035</name>
                                        <identification code="530035" dictionary="SONHO.NUM_SEQUENCIAL"/>
                                </patient>
                                <documentList>
                                        <document type="HTML">http://newdevel.med.up.pt/~agent/sampledoc.html</document>
                                </documentList>
                        </report>
                        <report id="5866905" type="Analise de Patologia Clinica" emission-date="15/10/2008 10:03:01">
                                <patient birth-date="18/01/1955" sex="F">
                                        <name>Nome Generico do Doente 630009</name>
                                        <identification code="630009" dictionary="SONHO.NUM_SEQUENCIAL"/>
                                </patient>
                                <documentList>
                                        <document type="HTML">http://newdevel.med.up.pt/~agent/sampledoc.html</document>
                                </documentList>
                        </report>
                        <report id="5866906" type="Analise de Patologia Clinica" emission-date="15/10/2008 10:03:06">
                                <patient birth-date="07/06/1934" sex="F">
                                        <name>Nome Generico do Doente 1492311</name>
                                        <identification code="1492311" dictionary="SONHO.NUM_SEQUENCIAL"/>
                                </patient>
                                <documentList>
                                        <document type="HTML">http://newdevel.med.up.pt/~agent/sampledoc.html</document>
                                </documentList>
                        </report>
                        <report id="5866907" type="Analise de Patologia Clinica" emission-date="15/10/2008 10:03:16">
                                <patient birth-date="11/07/1981" sex="F">
                                        <name>Nome Generico do Doente 247119</name>
                                        <identification code="247119" dictionary="SONHO.NUM_SEQUENCIAL"/>
                                </patient>
                                <documentList>
                                        <document type="HTML">http://newdevel.med.up.pt/~agent/sampledoc.html</document>
                                </documentList>
                        </report>
                </reportList>
        </response>
</MessageHSJXXI>
```

Figure 2.2: XML retrieved from the Clinical Pathology departmental system, generated from the departmental simulation script.

2.1.2 MAID Modifications for Simulation

There was a necessity to make some changes to MAID itself, in order to adapt it for a simulation environment. These changes were made in order to be able to complete a simulation execution, gathering all of its outcomes at the end.

Changes were made to the Control agent's behaviors. A simulation behavior was added to the Control agent for monitoring of resource occupation of the MAID platform using the SIGAR API[19]. The monitoring occurs every minute. The retrieved data is used in the development of the optimization of MAID as the retrieved values will aid in the comparison of different optimization paths. This behavior also enables him to receive messages from other agents. These indicate that their simulation run is complete in order to have the Control agent gather all the simulation data and shutdown MAID. Methods were developed that will gather all the simulation data and record it a simulation database.

As the deployed system uses the current system time as a basis for the report reference retrieval, receiving report lists from the time it last asked for those lists to the current time, using the fixed period, changes were made to the List agent's list retrieval behavior so that it could retrieve past report references. In the simulation environment the behavior will start the retrieval on the date set in the simulation parameters and then use as time increments a

previously set static period (similar to the deployed system) or one configurable by the the optimizations generated by the scheduling algorithm.

2.1.3 Web Interface for Simulation Configuration

To facilitate the configuration of the simulation, a web-based configuration tool was developed that allows the editing of MAID's configuration files, the simulation parameters and the weights of the impact variables of the scheduling function.

The forms of the configuration tool are generated dynamically from the properties/configuration files. In figure 2.3 to Fig. 2.7 we can see the different configuration pages.

In "MAID Configuration" we can alter the service list configurations and the Control, List, Balancer and File agent's configurations.

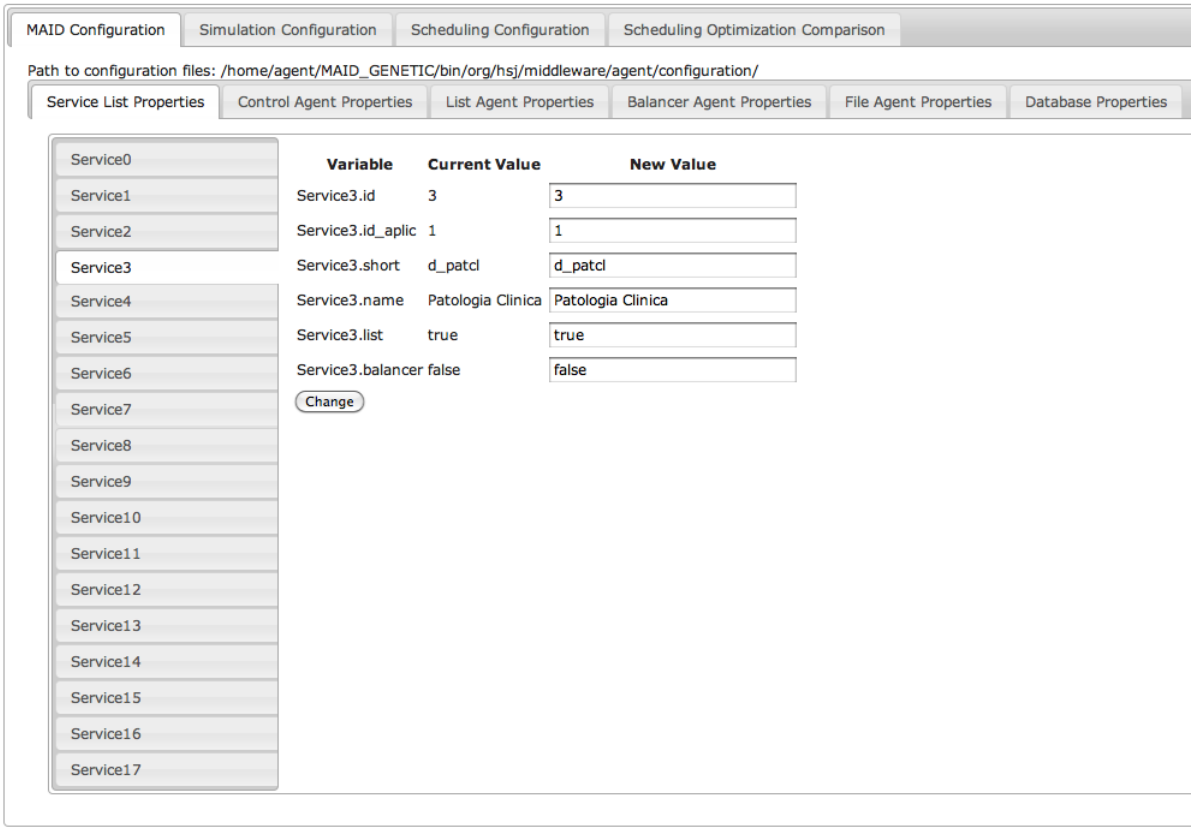


Figure 2.3: Configuration of MAID's active service list

In the service list configuration page (see figure 2.3) we can manage the identification of each department/service as well as the state of their respective List and Balancer agents. If set to false these variables will indicate to MAID that the list retrieval and report file distribution and retrieval will not be active.

In the Control Agent's configuration page (see figure 2.4) we can manage the Control Agent configuration properties.

In the List Agent's configuration page (see figure 2.5) we can manage the configuration of

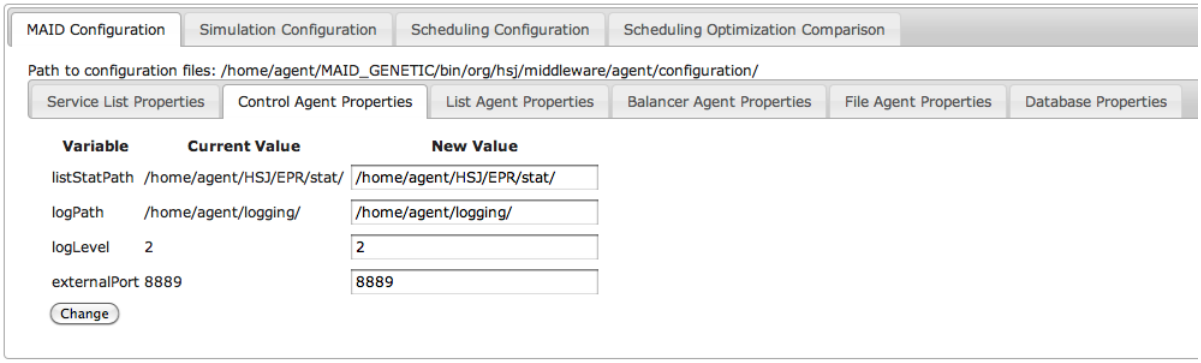


Figure 2.4: Configuration of MAID's Control Agent

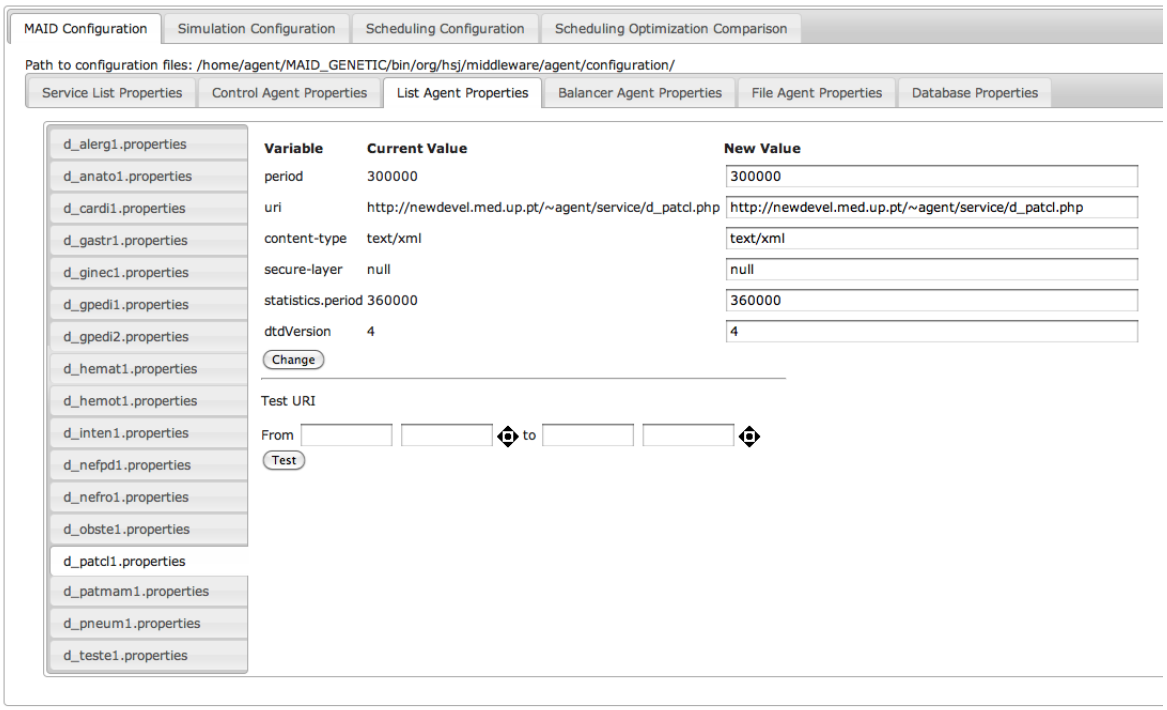


Figure 2.5: Configuration of MAID's List Agents

the List Agent's properties of each department. We can also test the list retrieval process for the department, obtaining the generated report list XML.

In the "Simulation Configuration" tab (see figure 2.6) we can configure the simulation environment defining the period of simulation, the databased used in the simulation, and the location of the simulation output files. It also enables the execution of simulations without resorting to the command line that is usually used to deploy MAID.

In the "Scheduling Configuration" tab (see figure 2.7) we can configure the scheduling parameters for each departmental system. These will be used in the schedule generation by the Scheduler agent in the schedule generation behavior.

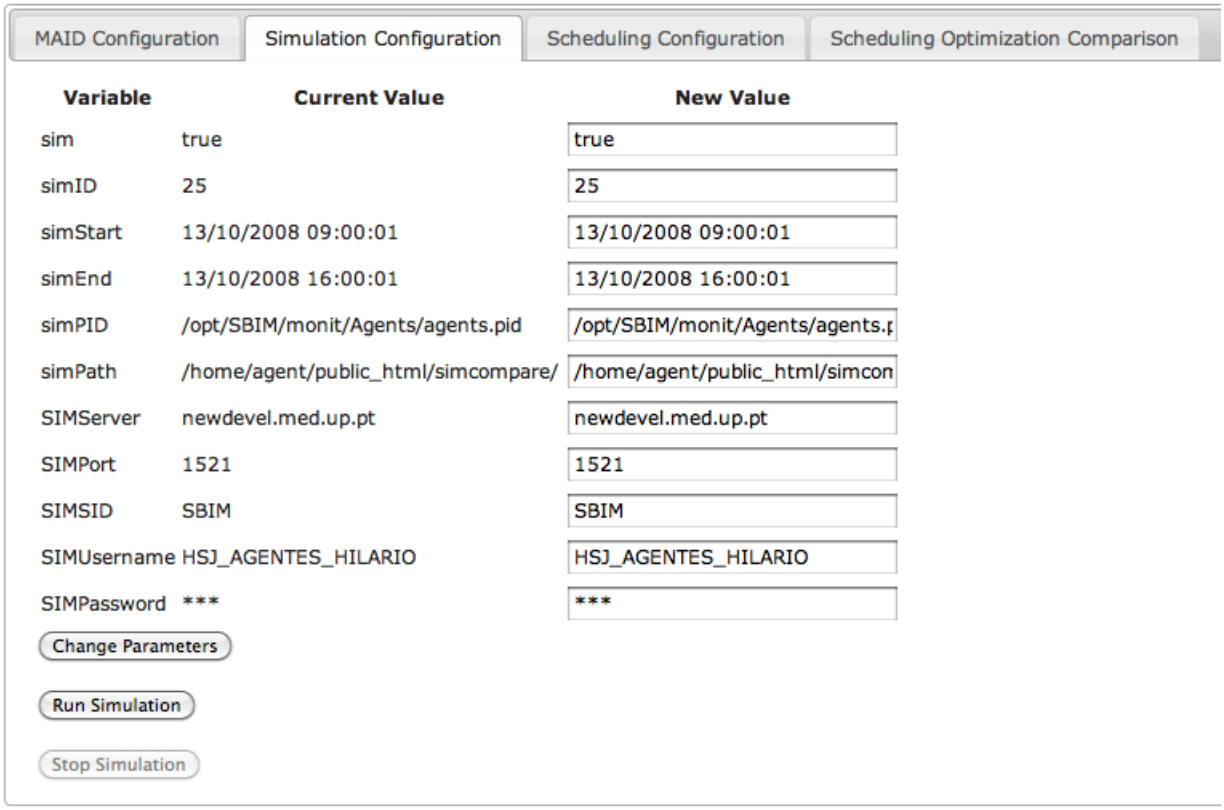


Figure 2.6: Configuration of the simulation environment

2.2 Simulation Execution

The execution of a simulation is divided into three different steps (see Fig. 2.1). These include the configuration of the simulation, the actual execution of the simulation and, finally, the gathering and pre-processing of simulation data. These will be detailed in the following subsections.

2.2.1 Simulation Configuration

The simulation configuration is made by the adjustment of a simulation properties file used by the simulation behavior. In the configuration step of the simulation a series of parameters are defined including the data retrieval time frame that the simulation will run, directly related to the time frame of report production of the simulated departmental systems. Also defined is the database configuration for the storage of the simulation results.

It is also possible to parametrize the scheduling algorithm. We can configure the weight that several variables may have in the reference retrieval process. These variables are to be used in the scheduling algorithm calculations and are related to the current time frame within the simulation and are gathered for each of the departments. The selected variables include report production count, report retrieval count, median time difference between production and retrieval and median time difference between sequential report production.

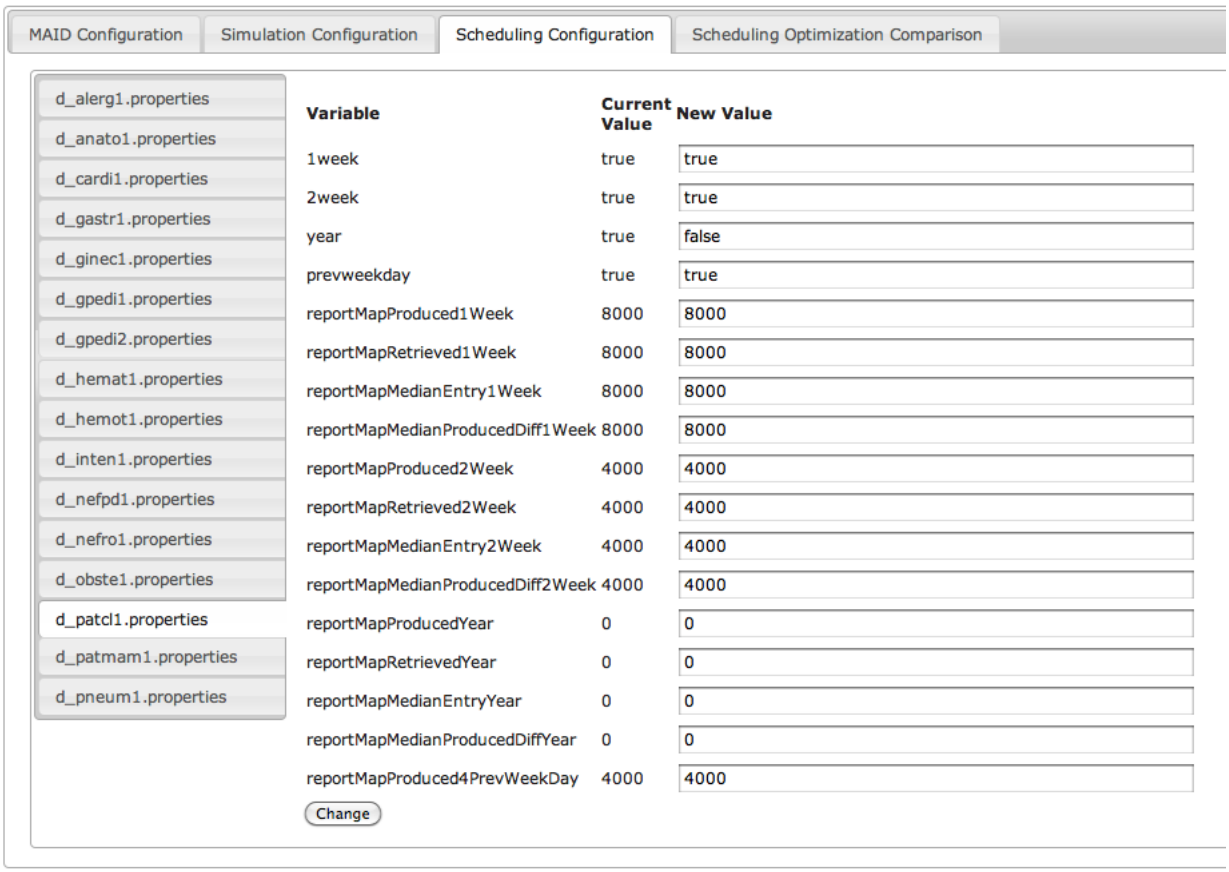


Figure 2.7: Configuration of the weights of the impact variables

The variables are gathered per hour and retrieved for the previous week (see figure **??**), two weeks prior, the corresponding week in the previous year as well as the four previous corresponding weekdays for the current simulated date.

2.2.2 Simulation Run

During the execution of the simulation MAID behavior is similar to the deployed version but includes some changes (see Fig. 2.1).

One was the execution of a simulation behavior by the Control agent as a part of the simulation environment. In it the Control agent awaits the communication of the end of the other agents simulation. When the execution of their respective cycles ends, and the time frame of the simulation is past, each agent informs the Control agent that it has terminated its simulation execution.

Other changes in MAID include the use of the scheduling algorithm as some of the data gathered from the simulation is related to the execution of the scheduling calculations and use of these by the List agents.

The variables gathered during the execution of the scheduling process will be the main terms of comparison between the different simulation executions and the baseline currently deployed

system.

2.2.3 Simulation Results Gathering

When all the agents complete their tasks the Control agent executes the simulation termination methods, gathering all the relevant simulation data.

For each departmental system, during the simulation run, data for each variable will be retrieved by the scheduling process to a corresponding map and used as input of a scheduling algorithm to create a corresponding scheduling table.

The gathered simulation data includes the retrieved variables and the scheduling algorithm output. All of these are stored, when the simulation ends, in the simulation database tables.

2.3 Discussion

The simulation environment enables the comparison of different simulation executions by generating a series of tables and graphics based on the simulation data retrieved from different simulation executions.

It is possible, for example, to compare various simulation executions regarding a specific variable using the gathered simulation data. In this case we can visually compare the median difference between production and retrieval date for two different simulation executions having also recorded the changes made to the scheduling algorithm between the two executions.

While the simulation environment is essential in the optimization process of MAID some limitations are to be taken into account, such as the impact that the introduction of simulation messages has in the overall simulation environment. This could be mitigated by monitoring the exchanged messages using tools provided by the JADE platform for monitoring agent message exchanges, so that these can be quantified and taken into account when using the simulation data.

The simulation environment will enable us to find the most representative variables that influence the report reference retrieval and attribute a weight to each. Various iterations of simulation execution and result analysis will lead to the optimization of the scheduling algorithm to enable MAID List agents to adapt themselves to their respective departmental services.

Adaptative Scheduling of Agent Actions

Chapter 3

Adaptative Scheduling of Agent Actions

3.1 Scheduler Agent

The Scheduler Agent is the agent responsible for the generation and delivery of the retrieval periods of the List agents.

During setup the agent registers itself with the Directory Facilitator (DF) service of the MAID platform as per the agent lifecycle define by FIPA. Also during setup this agent loads a configuration file. In this file is defined if its schedule generation and schedule messaging behaviors are active and what are their respective execution cycle timers. These timers define the periods of execution of the corresponding behaviors.

This agent has a method for determining which agents are fit to receive the schedule. This method receives a service descriptor, in this specific case a "List Retriever" descriptor. It then searches the DF service for any corresponding agents.

It also has method for sending a schedule value to a List agent. In this method the Scheduler agent creates a FIPA compliant message with a List agent as a receiver and with the schedule value as its content. It sets the message type as being a schedule message and sends it to the corresponding agent.

The main task of the Scheduler Agent is to generate weekly map, for each hour of each day of the week, of list request frequency/schedule values for the List agent's retrieval behavior. For this it generates a series of maps that then uses to calculate the schedule values to be used by the List agents. This task will be better detailed in the following sections.

3.2 Schedule Generation Behavior

The Scheduler Agent uses this behavior to generate the schedules. In this behavior the agent will generate maps with a series of variables that will be used to calculate the schedules.

It iterates through a list of active services / List agents. For each active service it retrieves a

series of maps. These maps will have, for each weekday of a given timeframe and each hour of the day a calculated or retrieved value. These are the following:

- Map of report production: contains the report production count of a given hour. This is obtained by querying the VERP database for a count of the reports that were produced between a specific period, using the emission date of the reports as a parameter;
- Map of report retrieval: contains the report retrieval count of a given hour. This is obtained by querying the VERP database for a count of the reports that were retrieved between a specific period, using the retrieval date of the reports as a parameter;
- Map of median time to retrieve: contains the median time in seconds between production and retrieval of reports during a given hour. This is obtained by querying the VERP database for reports that were produced between a specific period, using the emission date of the reports as a parameter and calculating the difference between their production and retrieval date. It then is calculated the median value of these values for each hour;
- Map of median time difference between consecutive produced reports: contains the median time difference between consecutively produced reports. This is obtained by querying the VERP database for reports produced between a given timeframe and then calculating the difference in seconds between the production times of consecutive reports. It is then calculated the median of these values for each hour.

These maps are generated in relation to the current time for the previous week, two weeks prior and the corresponding week of the previous year as these contain relevant patterns associated with current report production.

The agent then generates a schedule. It iterates through a list of active services and for each reads a schedule configuration file containing the weight each variable will have in the calculations.

Changes to the weights contained in the file will alter the results of the scheduling algorithm. It then, for each weekday and each hour, retrieves the corresponding values from each of the maps.

It combines each retrieved variable value with its corresponding impact value. It then combines these values to produce the schedule value. It takes into account that this value must not fall out of a fixed upper and lower bound. Depending on the departmental characteristics, it must not be too small so as to not overload the external departmental systems with requests or too large that the reports do not arrive in time for the process of care delivery.

It then saves the schedules to the database of the VERP repository (CREP) as well as the reference values used to calculate the scheduling tables. The saved schedule includes the identification of the corresponding department (Service ID and Service Application), the weekday, the

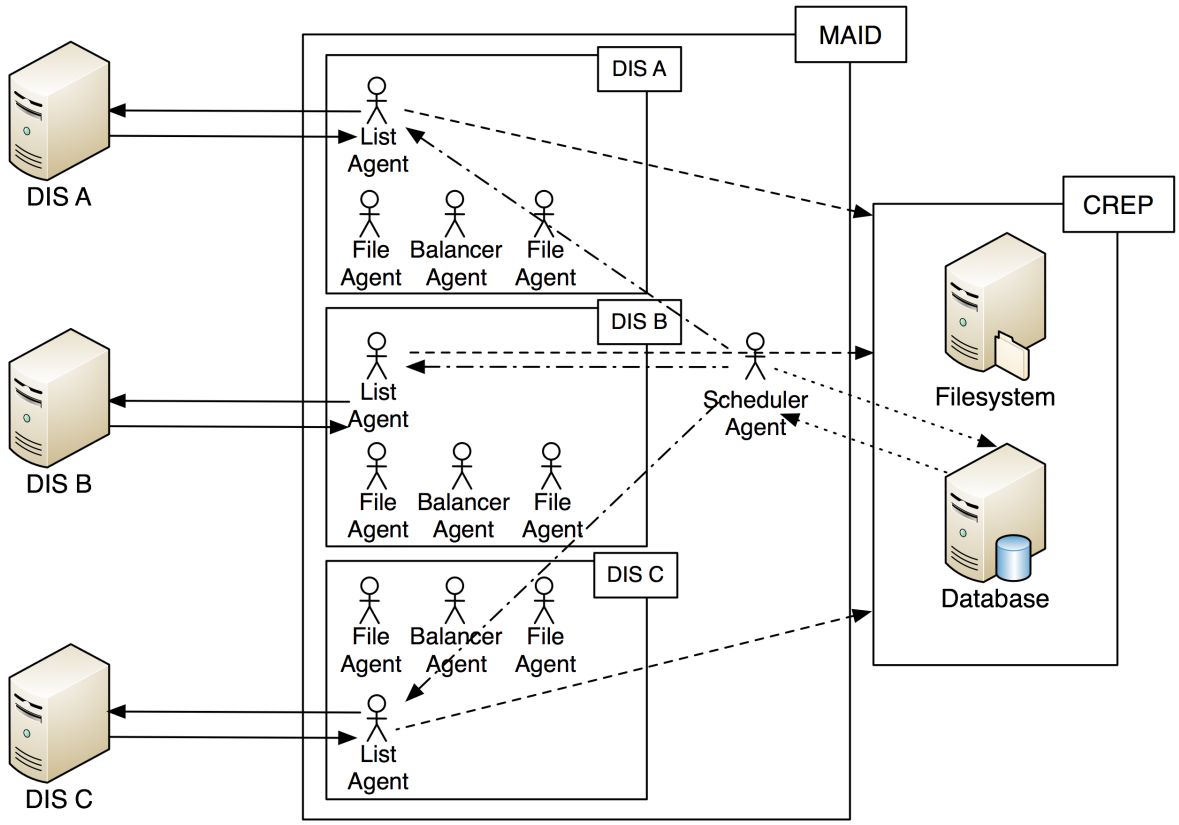


Figure 3.1: Schedule delivery process of the Scheduler Agent

hour and the generated schedule value. It also optionally creates a csv file of the schedule tables for monitoring purposes.

3.3 Schedule Messaging Behavior

The Scheduler Agent uses this behavior to periodically send the schedule values to the List agents. In this behavior the agent obtains a list of agents that are available to receive the schedule. It does this by querying the DF for "List Retriever" agents. It then prepares a scheduling message to send to the corresponding agents.

To do so it obtains the Service ID and Service Application of a List agent and obtains the current weekday and time from the system clock. It then queries the CREP database with these parameters for the schedule value.

Having obtained this value the agent constructs a FIPA compliant message that includes a sender identification (its own Agent Identifier - AID), a Receiver identification (the receiving agent's AID), a message type identification (a "schedule" message) and as content the schedule value. The behavior then send the message (see figure 3.1) and blocks itself until it is time to send the next batch of messages to the List agents.

3.4 Schedule Reciever Behavior

The List agent has a Schedule Receiver Behavior that is used to listen for messages coming from the Scheduler Agent (see figure 3.1). It is prepared to only receive scheduling messages.

Upon receiving a scheduling message it sets its own retrieval period to the value contained in the message contents.

3.5 Control, List and Balancer Agent Changes and List Retrieval Behavior Changes

There was a need to change the List agent and its List Retrieval Behavior to accommodate for the scheduling optimization. To this end a Schedule Receiver Behavior was added to the List's agent behaviors. The List Retrieval Behavior now uses as its retrieval period the one set by the Schedule Receiver Behavior.

3.6 Discussion

As seen in figure 3.2 the optimization process is centered around the Scheduler Agent. With its behaviors for generation and delivery of schedules to the List Agent it enables an adaptive report list retrieval process, adjusted to the report production patterns of each department.

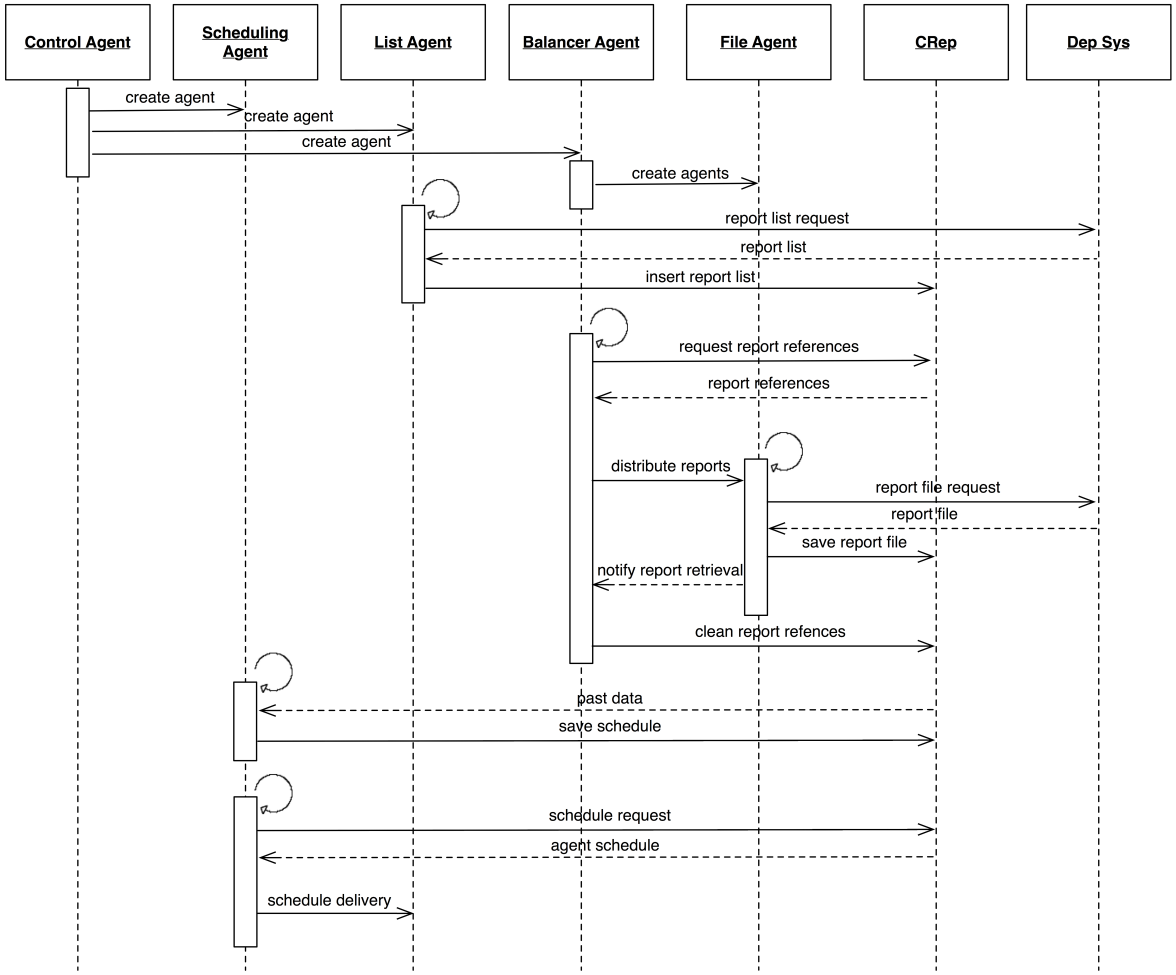
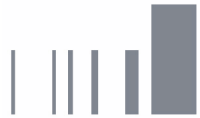


Figure 3.2: Sequence diagram for the optimized version of the MAID platform



Results

Chapter 4

Results

4.1 Results Overview

Having developed the simulation environment and the Scheduler Agent and its behaviors it is possible to execute simulations that will enable the evaluation of resource occupation and of optimization efficiency, testing the optimal weights for the impact variables that lead to different optimization paths, and also to monitor the impact that these changes have in system resource occupation.

4.2 Resource Occupation

In order to monitor the impact that the changes to the retrieval process have on system resources a system monitoring method was developed and included in the simulation behavior, using the SIGAR API[19], to monitor a series of variables.

These include CPU percentage usage and memory usage of the MAID process as well as the network interface data transmission and reception values. The values of these monitoring variables are retrieved continuously and registered for future analysis. These values are also used to compare different optimization paths.

4.3 Optimization Efficiency

To be able to compare the efficiency of different optimization paths in relation to the original report retrieval process as well as between them, a series of variables are retrieved and/or calculated for comparison.

These include the number of report list requests, the number of retrieved lists without reports (empty report lists), the mean and standard deviation of the time difference between report production and list retrieval and the time difference between report list request and report list response.

Table 4.1: Example of comparison between the non-optimized version of MAID and an optimization path in the retrieval of report lists from the Clinical Pathology Department for the period between 09:00 And 16:00 of 2008/10/13

Recorded Variables	Simulation Values		Difference between non-optimized and optimized (%)
	Non-Optimized	*Possible Optimization Path*	
Resources			
Mean CPU % Usage (Std. dev.)	0.0051 (0.0116)	0.0053 (0.0131)	+0.0002 (3.9%)
Mean Memory Usage in Megabytes(Std. dev.)	35.75 (0.94)	35.78 (0.97)	+0.03 (0.1%)
Network Interface Total Megabytes Received	208.68	206.80	-1.88 (-0.9%)
Network Interface Total Megabytes Sent	193.62	193.05	-0.57 (-0.3%)
Efficiency Variables			
Number of report list requests	85	79	-6 (-7.1%)
Number of empty report lists	3	3	0 (0%)
Mean time difference between report production and list retrieval in minutes (Std. dev.)	2.45 (1.44)	2.56 (1.51)	+0.11 (4.5%)
Mean time difference between list request and list retrieval in seconds (Std. dev.)	3.39 (0.03)	3.42 (0.04)	+0.03 (0.9%)

4.4 Discussion

In Table 4.1 we can see the results of a simulation for the optimization of the Clinical Pathology Departmental system report retrieval process. The variables were gathered for the period between 09:00 and 16:00 of 2008/10/13. In this table are demonstrated the differences between the non-optimized and the optimized report list retrieval process.

Concerning resource occupation and regarding mean CPU usage there is an increase from 0.0051 to 0.0053, giving a 3.9% increase in CPU usage in the optimized version. Regarding mean memory usage there is an increase from 35.75 MBs to 35.78 MBs, giving a 0.1% increase in memory usage.

In regards to the number of MBs received by the network interface, there is a decrease from 208.68 to 206.80, giving a 0.9% decrease in Megabytes received. In regards to the number of MBs sent by the network interface, there is a decrease from 193.62 to 193.05, giving a 0.5% decrease in Megabytes sent.

Concerning the efficiency variables and regarding the number of report list requests there is a decrease from 85 to 79, giving a 7.1% decrease in the number of report list requests.

Regarding the number of empty report lists the number, 3, is equal in both the non-optimized and the optimized process.

Regarding the mean time difference between report production and list retrieval there is an increase from 2.45 to 2.56 minutes, giving a 4.5% increase in the mean time difference between report production and list retrieval.

Regarding the mean time difference between list request and list retrieval there is an increase

from 3.39 to 3.42 seconds, giving a 0.9% increase in the mean time difference between list request and list retrieval.

Conclusion

Chapter 5

Conclusion

5.1 Summary

This chapter will analyse the results of the simulation execution presented in the previous chapter. It will also describe the limitations of the work of this thesis and discuss future work.

5.2 Main Findings

Analyzing the obtained simulation results we can compare the non-optimized version with the optimized version of the MAID report list retrieval process both in resource usage and efficiency of the report list retrieval process.

The goal of the chosen optimization path was to reduce the number of list requests to the external system as to not overburden it. We can see that with this optimization path this number has gone from 85 to 79, down by a significant 7.1%.

Concerning resource usage we can see an increase CPU and memory usage of 3.9 and 0.1% respectively, but the increase in these values, due to the total CPU and memory available, is negligible. The network interface received and sent values have decreased by 0.9 and 0.3% respectively. This may be a consequence of the decrease in report list requests and is also an intended effect as it reduces network strain.

Regarding the efficiency variables there is a significant increase from 2.45 to 2.56 minutes (4.5%) in the mean time between report production and retrieval of the report list but this is expected, as the number of requests for the same time period has diminished, the time between them naturally increases. There is also a slight increase (0.9%) in the time between report list request and retrieval. This may be due to, although there are fewer requests, the number of produced reports is the same, so the requested lists contain more reports in each list, increasing their size marginally and also increasing the time to obtain them.

The Scheduling agent will allow an optimized report list retrieval process that is adjusted

to the report production patterns of each department. The process of determining the best optimization path for each department will have to take into account the balancing between not putting too much stress on the departmental systems, minimizing the number of empty report lists and retrieving the reports in a timely fashion.

In this case we took into account the impact that more frequent requests made by MAID can have in the departmental systems. Boundaries must be set according to each departmental system as to not overload them, as each request can have a significant impact on these systems.

5.3 Limitations

Limitations of this work include the data sources used in the simulations. The anonymised dataset used as source for the report lists is somewhat limited, has a few gaps in the data and is somewhat dated (includes data from 2007 to 2009). There may have been some adjustments to the report production patterns since then.

Another limitation of the work is the number of simulation executions that were possible to analyze. This is due to the manual process of generating the statistics based on the simulation data.

5.4 Future work

Future work includes the testing of various optimization paths for each of the departmental systems, changing the impact variables weights and minimum and maximum retrieval periods to better suit each department until the best path of optimization for each is determined. It will be also important to measure the impact that current scheduling will have in future system execution.

The simulation environment may also be improved, automating the generation of simulation results, eliminating the need for the manual processing of the results.

The optimization of the Balancer Agent will also have to be taken into account. Some work has already been made on this, within the scope of the OPTIM project, enabling the agents to send the report lists to a tool that classifies them by their relevance and enables the balancer to distribute to the file agents the more relevant ones for retrieval of the report files.

5.5 In Closing

The research questions to be answered were the following:

- How to better adapt the report list retrieval process to the departmental systems profiles;

This was done using past report production and retrieval data for the the schedule generation.

- How to alter MAID agents behavior so that they take into account an optimized report list retrieval schedule;

This was done by introducing a Scheduler Agent that generates a retrieval schedule for each service and delivers it to the corresponding List Agent.

- How to test, monitor and evaluate different optimization paths while maintaining a simulation scenario similar to real world use.

The monitoring of system performance is done continuously by the Control Agent. At the end of a simulation execution several variables are also retrieved. All of these allow for the comparison of different simulation executions. The simulation uses data from an anonymised dataset of report data from HSJ and mimics the behavior of the deployed agent system.

References

Bibliography

[1] G. Weiss, Multiagent Systems - A Modern Approach to Distributed Modern Approach to Artificial Intelligence. MIT Press, 1999.

[2] FIPA, "Fipa agent management specification," 06 2012.

[3] R. Cruz-Correia, P. Vieira-Marques, P. Costa, A. Ferreira, E. Oliveira-Palhares, F. Araujo, and A. Costa-Pereira, "Integration of hospital data using agent technologies - a case study," Ai Commun, vol. 18, pp. 191–200, 2005.

[4] M. Wooldridge and N. Jennings, "Intelligent agents: Theory and practice," Knowledge Engineering Review, vol. 10(2), pp. 115–152, 1995. http://www.csc.liv.ac.uk/ mjw/pubs/ker95/ker95-html.html.

[5] P. Maes, "The agent network architecture (ana)," SIGART Bulletin, vol. 2, no. 4, pp. 115–120, 1991.

[6] R. A. Brooks, "Intelligence without representation," Artif. Intell., vol. 47, no. 1-3, pp. 139–159, 1991.

[7] R. A. Brooks, "Intelligence without reason," in IJCAI, pp. 569–595, 1991.

[8] FIPA, "Standard fipa specifications," 06 2012.

[9] FIPA, "Fipa communicative act library specification," 06 2012.

[10] F. Bellifemine, G. Caire, and D. Greenwood, Developing Multi-Agent Systems with JADE. John Wiley & Sons Ltd, 2007.

[11] F. Bellifemine, A. Poggi, and G. Rimassa, "Developing multi-agent systems with jade," Lecture Notes in Artificial Intelligence, vol. Intelligent Agents VII - Agent Theories Architectures and Languages, pp. 89–103, 2001.

[12] S. Hanks, M. E. Pollack, and P. R. Cohen, "Benchmarks, test beds, controlled experimentation, and the design of agent architectures," AI Magazine, vol. 14, no. 4, pp. 17–42, 1993.

[13] R. Vincent, B. Horling, and V. R. Lesser, "An agent infrastructure to build and evaluate multi-agent systems: The java agent framework and multi-abent system simulator," in Revised Papers from the International Workshop on Infrastructure for Multi-Agent Systems: Infrastructure for Agents, Multi-Agent Systems, and Scalable Multi-Agent Systems, (London, UK, UK), pp. 102–127, Springer-Verlag, 2001.

[14] P. Vieira-Marques, R. Cruz-Correia, P. Costa, E.Palhares, A. Ferreira, and A. Costa-Pereira, "Maid - multi agent for the integration of data," Proceedings of the 1st Iberian Conference on Information Systems and Technologies, vol. 1, pp. 603–614, 2006.

[15] J. Ambite and C. Knoblock, "Agents for information gathering," Ieee Expert, vol. 12, pp. 2–4, 1997.

[16] C. Hayes, "Agents in a nutshell - a very brief introduction," IEEE Transactions on Knowledge and Data Engineering, vol. 11, pp. 127–132, 1999.

[17] D. Isern, D. Sanchez, and A. Moreno, "Agents applied in health care: A review," Int J Med Inform, vol. 79, pp. 145–166, 2010.

[18] M. V. D. Meijden, H. Tange, J. Troost, and A. Hasman, "Determinants of success of inpatient clinical information systems: A literature review," J Am Med Inform Assoc, vol. 10, pp. 235–243, 2003.

[19] SIGAR, "Sigar - system information gatherer and reporter," 06 2012.

Appendix

Chapter 6

Appendix

6.1 Simulation Environment for the Optimization of the Data Retrieval Capabilities of an Agent Based System in a Healthcare Setting

6.2 Optimization of an Agent Based Clinical Data Retrieval System

U.PORTO
FMUP FACULDADE DE MEDICINA
UNIVERSIDADE DO PORTO

U.PORTO
FC FACULDADE DE CIÊNCIAS
UNIVERSIDADE DO PORTO