

AIR TRAFFIC SERVICE MANAGEMENT BASED ON COLLABORATIVE AGENTS AND PASSENGER DELAY CRITERIA

Fábio Silva Carvalho*
Leonardo Mayeda
Ítalo Romani de Oliveira
Paulo Sérgio Cugnasca
Liria Matsumoto Sato

University of Sao Paulo
Polytechnic School – Computer and Digital Systems Engineering Department

Abstract

The management of Air Traffic Service (ATS), done for independent operators of diverse air companies, uses several computational systems to allow aircraft identification and flight monitoring. These systems provide levels of efficiency and satisfaction considered reliable. However, not much has been done in order to improve the ATS systems when it comes to the specific needs of each passenger, for example, to prevent losses of connection of flights generated for delays, or even to compensate delays. The CDM (Collaborative Decision Making) concept, which allows the air company to participate in traffic management, provides a better quality of service to meet the necessities of the passengers. This work describes a use case of the CDM concept, presenting diverse elements of an efficient computational infrastructure for this application.

Keywords: *Collaborative Decision Making, Multi-Agent Systems, Distributed Database Systems, Parallel File System*

* Corresponding author:
University of Sao Paulo
Polytechnic School – Computer and Digital Systems Engineering Department
Avenida Professor Luciano Gualberto, travessa 3, n°. 158
Prédio da Engenharia Elétrica - Sao Paulo, SP, Brazil
email: fabio.s.carvalho@poli.usp.br
Tel: (+ 55 11) 3091-5617

1. INTRODUCTION

The main goal of this article is to discuss, using a short case study, passenger connection losses and flight delay minimization. This work is being developed in the GIGA Project context, in which the use of high speed interconnection technology is an important feature. Giga Project consists of the development of optical network technologies, applications and telecommunication services associated with IP technology and broadband. The article aims to show a case study which represents the aeronautics system behavior, verifying the functionality of a Collaborative Decision Making (CDM) system prototype on the air traffic system management in a gigabit network platform.

Through a simulation based on this case study, it will be possible to verify the developed prototype functionality, besides analyzing the information sharing among plenty of aeronautics systems components, such as aircrafts and airports, in operation optimizations.

The decision making rules oriented to the passenger, the essential principle of the case study, are described in the second section. The usage of these rules becomes easier with the support of an air traffic management decision making architecture proposed in (Cugnasca *et*

al., 2005), used in the system prototype development.

This architecture has a parallel file system - Network Parallel File System (NPFS) as an essential element described in the third section, and a parallel and distributed database management system - SisBDPar - described in the fourth section.

Besides this architecture, an environment that allows the integration between distributed database systems and multi-agent systems is also used. This system, called Distributed data Agent Service Environment (DASE), is described in the fifth section.

The sixth section describes the study case proposed, its constraints, and its abstraction, besides the solution proposed through an initial design of the simulation that will be implemented in the future.

2. COLLABORATIVE DECISION MAKING RULE

The main characteristic emphasized in this section, and explored by the case study described in the sixth section, is the connection loss and the passenger flight delay minimizing; thus, only the necessary information to represent this matter is considered during the collaborative decision making rules description.

A flight represents airplane traffic between

two different airports. A flight without stops is always related to exactly two different airports, but a flight with stops is related to at least three different airports. An airport can be related to any quantity of flights.

In order to reach the desired location in a satisfactory way, a passenger can fly only once, or take several flights. This second case represents a connection in accordance with the following rules:

- The flights are subsequent. This means at no moment their foreseen length intervals are overlapped.
- A flight destination airport is the same as the next flight origin one.
- The foreseen interval between a flight and the next one must always respect a minimum time interval, otherwise it won't be possible for the passenger to get out of a flight and get in the next one, becoming extremely vulnerable to delays, what fatally will cause the connection loss.

2.1. Relevant flight steps to the case study

In a simple way, every flight can be described through the following operations:

1. The airplane take-off in the origin airport.
2. The airplane traffic between the origin airport and the destination airport vicinities.
3. The landing request in the destination

airport made by the airplane and allowed by the destination airport.

4. The airplane landing in the destination airport.
5. The airplane parking in the destination airport, in the proper location to the disembark.

It is important to emphasize that for a passenger who has a connection, there is a sixth operation which considers the full time between his first flight disembark to his next departure. This interval represents his break or any other activity he may want to have during it. The flight steps described in this subsection are shown in the figure 1.

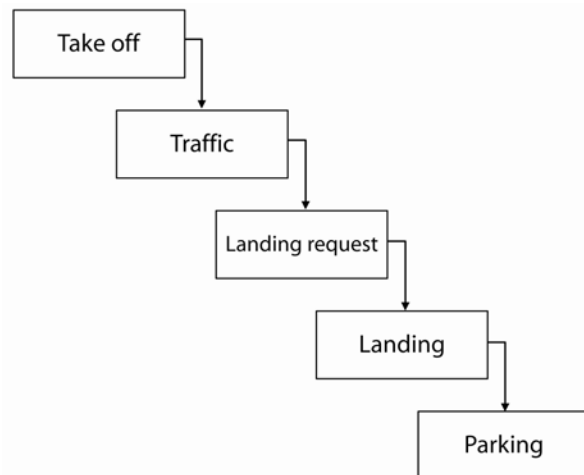


Figure 1: Flight steps

2.2. Events that causes delays during flights

An airplane can find an adverse and unexpected situation when leaving the origin airport and getting authorization to land in the destination airport. Besides it, the foreseen

interval during the traffic between the origin airport and the destination airport is not also precise.

Therefore, nothing can be said about the exact time an airplane leaves the origin airport, the time spent during the air traffic, and the time needed to get landing authorization in the destination airport.

The previously mentioned events that generate delays during flights and the operations that cause them are the following:

1. Take-off: depends on a lot of events that occur randomly.
2. Air space: climate conditions and traffic jam affect it.
3. Landing request: excess of traffic can generate delay in obtaining landing authorization.
4. Disembark place allocation: in the case of excess number of airplane in the airport, the disembark and embark operations can experience delays, even if the airplane has already been on the ground.

The three following sections describe briefly the three systems that, together, compose the architecture which supports the CDM prototype, mentioned in the introduction. This prototype will be used in the development of a case study simulation. The case study is described in the sixth section.

3. NPFS

The NPFS, *Network Parallel File System* (Guardia, 1999), is a parallel file system which provides high performance input/output parallelization. NPFS was designed to be integrated to distributed applications that process large amounts of data. This system is appropriate to distributed memory architecture, composed of a set of nodes interconnected by high speed interconnection network.

Each node has its local disk and memory. The nodes are used to process execution and to data storage. There are three kinds of processes: master, server and client, which interact among themselves following the client-server paradigm.

The master process has only one instance and its location is determined during the system activation. This process is the responsible for the management of centralized control operations, such as the parallel file open and close operations.

The server process is hosted in each node, and its function is to provide the writing and reading of parallel file distributed segments. It is responsible for activate the application processes that will be executed in the host. The figure 2 illustrates the NPFS architecture.

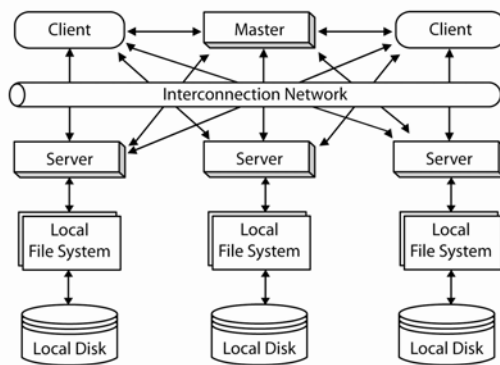


Figure 2: NPFS architecture

The client process is responsible for access the parallel files executing calls to NPFS primitives. Thus, the reading requests are sent sequentially to the server processes involved. The server processes execute the reading in a parallel way, and, afterwards, the read data are sent sequentially from servers to the clients.

In the data writing operation, the data to be written are sent sequentially from client processes to proper server processes. Each server sends a writing confirmation message to the client when it completes the writing in the local disk.

The data distribution in NPFS is done through the stripping technique, and the decision of data location inside parallel file segments is done without the storage table support, simplifying the query processing. Thus, the parallel file size is not limited by the mapping table size.

Therefore, the server processes and the master process do not need to store information about

the data location in the parallel file segments, and their activities are only reading or writing bytes sequence in parallel file segments positions.

4. SISBDPAR

The SisBDPar (Lubacheski, 2005) is a parallel and distributed database management system designed and implemented over NPFS, which distribute the query and sub-query processing, increasing the database performance.

The queries can be split and distributed among several CPUs. In consequence, this database system offers considerable scalability because the storage capacity increases according to the quantity of storage resources.

The SisBDPar architecture is based on client-server paradigm. The distribution of responsibilities between client and server is defined in a strong coupled way. The client has access to the database through an application programming interface (API).

The client can perform queries using API functions and a SQL-like language. During the query processing, performed in the client, the information stored in the catalog is verified, so is possible to decide which local servers will be used. The local servers are considered data servers, and are responsible in handle the parallel query processing.

Clients can access the stored information from

any node, and the local data servers are started in each node that contains a NPFS server. The figure 3 shows the SisBDPar architecture. In this figure “DB” means database, “M” means memory and “P” represents the CPU.

The queries are analyzed in the API/client before to be sent to the local data servers. In API/client and local data servers there are components responsible for the communication among themselves.

The relational operators, gotten from the queries, are sent to local database servers to be processed and to generate a provisory result. The interaction among API/clients and local data servers during the processing of a query occurs in the following way:

1. The query requested is analyzed and verified by API/client, which determines if should use local data servers, or NPFS servers or both. Besides it, a provisory result set is created to store the query result.
2. The local data servers process the relational operator, and store the result in the provisory result.
3. The API/client gets the result and forward to the client through an internal structure, allowing it to show or process the data.

In the case of NPFS server use, the API/client makes reading or writing request of byte sequences, and it is also responsible for whole query coordination and processing.

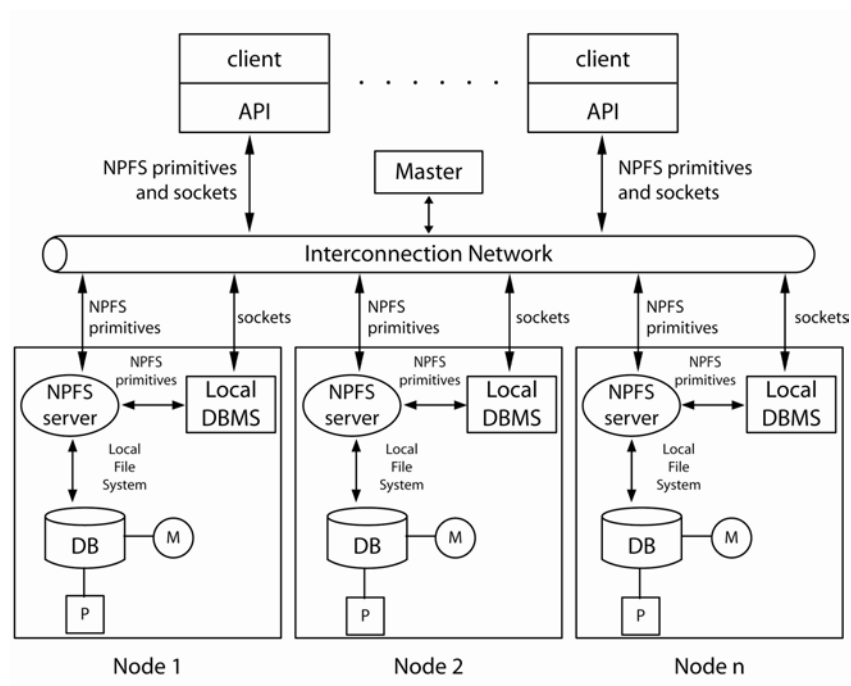


Figure 3: SisBDPar architecture

5. DASE

The Distributed data Agent Service Environment DASE, (Carvalho e Sato, 2006) is a system developed using Java programming language that offers a set of computational services integrated to agents of a multi-agent system platform (Sycara, 1998). Agents provide transparent, well defined and efficient distributed data access. Besides this, DASE complies with any standard used by the agent platform.

The two main objectives are:

1. Offer to agents of a multi-agent system access to distributed data transparently.
2. Offer additional resources related to data access, such as data consistency guarantee through distributed concurrency control.

The JADE (Bellifemine *et al.*, 2001) is an agent execution platform chosen by DASE project. The JADE is fully implemented using Java programming language, and is compatible with FIPA2000 specifications, Foundation for Intelligent Physical Agents (Aparicio *et al.*, 1999).

It has implementations of an Agent Management System (AMS) and a Directory Facilitator (DF), which represent respectively the white pages and yellow pages services, in comply with the FIPA standard. Besides to be an execution environment, it has also controlling, testing and monitoring agent

tools.

The following items represent the DASE main features:

1. The services are offered through agents in comply with FIPA standard.
2. Any complexity related to the data access is abstracted, guaranteeing the transparency during data access.
3. The physical data locality is transparent to the client agents even if the data are distributed.
4. Access to the data from client agents in a standardize way, independently of the resource manager type or version.
5. Compliance with a plenty of different resource managers, since they have a Java Database Connectivity (JDBC) driver (Ellis *et al.*, 2001).
6. Simultaneous support to different database systems.
7. Access to the data from client agents always through SQL statements or any SQL-like language adopted by the resource manager.
8. Support to distributed concurrency control.
9. Support to load balancing.

The DASE interacts with three different systems: the client agents, which are the agents using the DASE services to access distributed data, the JADE agents and the

distributed database system. The client agents use services provided by DASE and JADE agents. The DASE agents also use services offered by JADE agents, besides use database system services. The figure 4 shows the interaction among these systems.

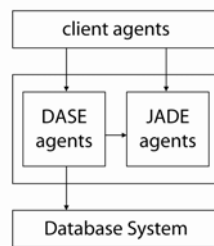


Figure 4: Systems interaction

The JADE services used by client agents and DASE agents are essentially life cycle control, identification, service publication and discovering, and agent communication infrastructure. The services offered by DASE agents are discovered by client agents through the JADE DF publication. The SisBDPar, which acts over the NPFS, represents the database system. Thus the distributed data are offered to the agents.

6. CASE STUDY PROPOSED

The case study aims at providing the basis of a simulation that justifies the architecture adoption. This architecture, described in the three previous sections, supports the CDM system, responsible for supporting the decision making in critic tasks related to the context described in the second section.

This case study consists in a simplification of the Brazilian air space control system, “*Sistema de Controle do Espaço Aéreo Brasileiro*” (SISCEAB), considering the Brazilian civil aviation operations. The simulation will approach synthetically the verification of airplane traffic behaviors among sets of airports, emphasizing the decision process optimization during the airplane landing request authorization in order to minimize the connection loss and passengers flight delays.

Some constraints are used in order to reduce and simplify the context reached by this case study, improving the perception of the behavior and analyzed consequences. Besides it, several constraints are defined, simplifying the real cases, for example, the disregard of flights with stops and the usage of a same airplane in more than one flight. Therefore, all affirmations in this study consider this context.

A flight represents airplane traffic between two airports, and an airplane is exclusive of a one flight, what means that an airplane is related one flight only. A flight never will have stops, thus it always is related to exactly two different airports, while an airport can be related to any quantity of flights.

An airport supports, only one airplane landing in a moment, in other words, it doesn't occur

simultaneous landings. In the case of take-off, they are independent among themselves and don't compete for authorization. Hence, this situation represents an ideal airport for the simulation in which there is only one private landing runway, and infinite private take-off runways.

A passenger always has one or two flights. In the case of two flights, necessarily represents a connection which is related to a passenger independently of other passengers, and obeys the following rules:

- The two flights are subsequent. It means that in any moment their foreseen length intervals are upon themselves.
- A flight destination airport is equal to the next flight origin airport.
- The foreseen interval between a flight and the next one is at least forty five minutes.

Subsequent flights related to the same passenger who is not in accordance with this rule are not accepted.

The interval of forty five minutes between two flights of a connection is justified by the addition of the last column maximum value from the last two lines in table 1, showed in the next subsection. Besides this, it is considered that a flight ends always after its airplane landing.

6.1. Events that make delays during flights

The same five operations which describe a flight, presented in the second section, are considered in this case study, even to the additional operation, described in section two, in case of passengers who have connections.

Nevertheless, for a viable simulation application, the events set that causes delays to these operations is reduced. For each operation is assigned its duration interval, which can be constant, variable respecting a limit, or undetermined. These numbers were estimated by authors but, considering the method used here, they are just parameters and can be easily modified after a more precise analysis. Such information is showed in table 1.

Table 1: Flight operations characteristics considered for the case study

Operation	Events that make delays	Length
Take-off	Several events which occur randomly.	5
Air space	Climatic conditions and traffic congestions.	-
Landing request	Traffic excess can generate delay to obtaining the landing authorization.	-
Landing	None.	12
Disembarkation place allocation	In the case of airplane excess in the airport, the disembarkation and the embarkation can have delay even if the airplane has already been on the ground.	5-15
Passenger waiting for a connection embarkation	Queues and other randomly unforeseen reasons.	15-30

The length column in table 1 has numbers, number intervals, or just a hyphen. A number in this column indicates a constant length necessary for the conclusion of the operation. In this case there is no kind of delay.

The number of intervals represents the minimum and maximum values to the conclusion of the operation. In this case the length of the delays varies according to the nature of the delay; however it is always inside the interval. The length of the operations represented by a hyphen is also variable, according to the delays. Nevertheless, in this case there is no boundary, as maximum or minimum.

Delays in take-off change the foreseen flight departure time. However, it is not harmful to the landing authorization process, described in the next section, because this process simply will consider the new value assigned.

Thus, the take-off operation length is ranked as 5 in table 1, even with the possibility of delays, represented by the second column from take-off row in the same table. Besides this, as said before, there are no delays for take-off.

6.2. The landing authorization process

Considering that more than one flight can have the same airport as destination, more than one flight can occur at the same time,

several events may cause delays during the flights (see table 1), and just one landing can occur in each airport in a moment, then arising the necessity of each airport to be able to control efficiently the landing requests and landing authorizations.

Besides the natural necessity of landing authorizations, described in the former paragraph, there is another detail that turns this decision so important: this process is crucial to define the quantity of connection loss and passenger flight delays. It occurs because the best criterion to landing authorization will be the one that considers the impact on future flights, in other words, the criterion that minimizes the connection loss and delays in future flights.

It is important to point out that every other flight conditions, as well as the involved airplanes, are considered equivalent and irrelevant during the landing authorization process, turning irrelevant to this consideration factors like fuel level of involved airplanes.

In order to facilitate the comprehension of this case study and the solutions proposed in this subsection and in the next one, it is presented a summary of the most important adopted constraints:

1. Every airport has only one exclusive landing runway. This runway does not

- allow simultaneous landings.
 - 2. There is no delay during take off.
 - 3. A landing takes twelve minutes and is started after the landing authorization emission.
 - 4. It is considered that a flight does not wait to take-off for passengers with connection loss.
- LC_{\max} maximum waiting time for a passenger with a connection.
 - LC_{\min} minimum waiting time for a passenger with a connection.
 - $T_{p,j}$ foreseen moment for airplane j to finish the landing.
 - T_d^i foreseen moment for i passenger to be available for the connection.
 - T_s^i subsequent flight exit moment in that i passenger will do the connection.
 - A^i available time to i passenger connection after the airplane landing.
 - P_c^i time in i passenger connection occurs.

When an airplane requests landing authorization, information about the flight is uploaded. This information will allow the decision process about the priority this airplane landing must receive, in case of landing queuing. Information on the flight passenger list is used to infer about the connections or the amount of passenger delayed.

6.2.1. Airplane landing priority calculation

Always that there is an excess of airplanes intending to landing, the airplane landing authorization uses a priority process to sort out the flights. The priority process is defined exclusively considering the connection loss and the minimization of delays. To quantify the delays of passenger connections, the following variables are defined:

- PV_{\max} flight disembark place allocation maximum value.
- PV_{\min} flight disembark place allocation minimum value.

The PV_{\max} , PV_{\min} , LC_{\max} e LC_{\min} values can be obtained from table 1. The $T_{p,j}$ value is always updated in accordance to the distance between the airplane and the landing runway, and its speed. The T_s^i value corresponds to the passenger connection next flight foreseen exit time. The T_d^i , A^i e P_c^i variables are calculated in the following way:

$$T_d^i = T_{p,j} + PV_{\max} + LC_{\max} = T_{p,j} + 15 + 30 = T_{p,j} + 45 \quad (1)$$

$$A^i = T_s^i - T_d^i \quad (2)$$

$$P_c^i = (PV_{\min} - PV_{\max}) + (LC_{\min} - LC_{\max}) = -25. \quad (3)$$

There are three options for a passenger with a connection, in accordance to A^i value, as is showed in table 2.

Table 2: Possibilities for a passenger with a connection

$A^i > 0$	The passenger is not late.
$0 > A^i \geq P_c^i$	The passenger is late, but maybe has still conditions to do the connection.
$A^i < P_c^i$	The passenger has not condition to do the connection.

According to Table 2, the lower the A value of a passenger with connection, the sooner or later to this situation he will be.

Each airplane landing priority is defined using the following criteria and in this sequence:

1. Flights with at least one late passenger with connection.
2. Flights with at least one passenger that can do his connection normally.
3. Flights with delay according to their flight plan.

In case of none of the above mentioned situations for landing requests, a higher priority will be assigned to the first airplane request for landing authorization.

In order to evaluate the three referred criteria, passengers without conditions to do the connections, in accordance with table 2, are considered passengers without connection.

To the first criterion, for each flight j is assigned a delay coefficient A_j , calculated through the addition of the A^i value of each late passenger with connection, in accordance with formula 4, being that N_j is the number of passengers of flight j . Thus, the airplane with the smaller delay coefficient value will have

priority to landing.

$$A_j = \sum_{i=1}^{N_j} A^i \quad (4)$$

In the second criterion, flights will be prioritized considering the number of passenger of each flight with the shortest available time to do his connections, in other words, the shortest A^i value.

Passengers that have a long connection interval, in other words, whose A^i value is longer than a hundred and twenty minutes, are considered passengers without connection.

In the case that the third criterion is considered, a flight delay according its flight plan will have the highest priority. An air traffic management data modeling proposal can be found in (Junior *et al.*, 2005).

7. CONCLUSIONS

In this article a case study related to the Brazilian air space control system, “*Sistema de Controle do Espaço Aéreo Brasileiro*” (SISCEAB), considering the Brazilian civil aviation operations, was presented. The case study emphasized the decision making process optimization during the airplanes landing authorization, in order to minimize the connection loss and passenger flight delays.

The case study proposed allows to verify the functionality of a collaborative decision making (CDM) system prototype, in the air

traffic management subject, whose three essential elements were briefly described. The case study application allows a simulation and its design and results will be presented in future works. This simulation will allow increasing the perception of the civil aviation landing request process, and the consequences of that process, such as the influence in future flights.

ACKNOWLEDGMENTS

This article authors thanks RNP – *Rede Nacional de Pesquisa*, supported by FINEP, for the GIGA Project participation - “*Projeto RNP/GIGA Prototipagem de um sistema de apoio à decisão compartilhada para uma infra-estrutura aeronáutica sobre bases de dados paralelas e distribuídas*”. Project 2468.

REFERENCES

Aparicio, M., L. Chiariglione, *et al.* FIPA - Intelligent agents from theory to practice. Geneva: Telecom 99 1999.
 Bellifemine, F., *et al.* JADE - A FIPA2000 Compliant Agent Development Environment. AGENTS’01, May 28-June 1, p.216-217. 2001.
 Carvalho, F. S. e L. M. Sato. DASE – Distributed data Agent Service Environment: Escola Politécnica da Universidade de São

Paulo 2006.
 Ellis, J., *et al.* JDBC™ 3.0 Specification. October. 2001.
 Guardia, H. C. Considerações Sobre as Estratégias de um Sistema de Arquivos Paralelos Integrado ao Processamento Distribuído. Escola Politécnica, Universidade de São Paulo, São Paulo, 1999.
 Junior, J. R. D. A., *et al.* Modelagem de Dados para Gerenciamento de Tráfego Aéreo. São José dos Campos, SP: IV SITRAER Simpósio de Transporte Aéreo 2005.
 Lubacheski, F. A. G. Uma Infra-Estrutura para Banco de Dados Baseada em Arquivos Paralelos e Distribuídos. Escola Politécnica, Universidade de São Paulo, São Paulo, 2005.
 Cugnasca, P. S., *et al.* Uma Arquitetura Lógica para Auxílio À Tomada de Decisão no Gerenciamento de Tráfego Aéreo. São José dos Campos, SP: IV SITRAER Simpósio de Transporte Aéreo 2005.
 Sycara, K. P. Multiagent Systems. 1998.