6

Disruption Management in Airline Operations Control – An Intelligent Agent-Based Approach

António J.M. Castro[1] and Eugénio Oliveira[1]
[1]*LIACC-NIAD&R, FEUP, DEI, University of Porto*
Portugal

1. Introduction

Operations control is one of the most important areas for an airline company. Through operations control mechanisms an airline company monitors all the flights checking if they follow the schedule that was previously defined by other areas of the company. Unfortunately, some problems may arise during this stage (Clausen et al., 2005). Those problems can be related with crewmembers, aircrafts and passengers. The Airline Operations Control Centre (AOCC) includes teams of experts specialized in solving the above problems under the supervision of an operation control manager. Each team has a specific goal contributing to the common and general goal of having the airline operation running under as few problems as possible. The process of solving these kinds of problems is known as Disruption Management (Kohl et al., 2004) or Operations Recovery.

To select the best solution to a specific problem, it is necessary to include the actual costs in the decision process. One can separate the costs in two categories: Direct Operational Costs (easily quantifiable costs) and Quality Operational Costs (less easily quantifiable costs). Direct operational costs are, for example, crew related costs (salaries, lodgement, extra-crew travel, etc.) and aircraft/flights cost (fuel, approach and route taxes, handling services, line maintenance, etc.). The quality operational costs that AOCC is interested in calculating are, usually, related with passengers satisfaction. Specifically, we want to include in the decision process the estimated cost of delaying or cancelling a flight from the passenger point of view, that is, in terms of the importance that such a delay will have to the passenger.

In this chapter we present our intelligent agent-based approach to help the AOCC solving the disruption management problem. It is organized as follows: In Section 2 we present some related regarding operations recovery, a classification of current tools and systems in use in some airline companies and a brief summary of the current use of software agents' technology in other domains. Section 3 introduces the Airline Operations Control Centre (AOCC), including typical organizations and problems, the current disruption management (DM) process and a description of the main costs involved. Section 4 is the main section of this chapter and presents our agent-based approach to this problem. This section presents: (i) the reasons that made us adopt the software agents and multi-agent system (MAS) paradigm; (ii) the MAS architecture including the specific agents, roles and protocols as well as some relevant agent characteristics like autonomy and social-awareness; (iii) decision

mechanisms, including costs criteria and negotiation protocols and (iv) examples of the problem solving algorithms used. In Section 5 we present the experimental setup and, in Section 6, we evaluate our approach, presenting and discussing the results. Finally, in Section 7, we conclude and give some insights on the future work.

2. Related Work and Current Tools and Systems

The goal of this section is threefold. In Section 2.1 we present the related work regarding operations recovery. Research in this area has been made, mainly, through Operations Research (OR) techniques. Barnhart et al., (Barnhart et al., 2003) gives an overview of OR-based applications in the air transport industry. In Section 2.2 we describe and classify the current tools and systems in use at some worldwide airlines and in Section 2.3 we present some interesting examples of how agents are used in other applications domains and problems.

2.1 Related Work

We divided the bibliography we have analyzed in three main areas: aircraft recovery, crew recovery and integrated recovery. For a more detailed explanation of those papers as well as for older papers related with each of these subjects, please consult (Clausen et al., 2005).

Aircraft Recovery: In (Liu et al., 2008) the authors propose a "multi-objective genetic algorithm to generate an efficient time-effective multi-fleet aircraft routing algorithm" in response to disruption of flights. It uses a combination of a traditional genetic algorithm with a multi-objective optimization method, attempting to optimize objective functions involving flight connections, flight swaps, total flight delay time and ground turn-around times. According to the authors "(…) the proposed method has demonstrated the ability to solve the dynamic and complex problem of airline disruption management". As in other approaches, the authors do use the delay time in the objective functions but nothing is included regarding passengers' quality of services costs.

Mei Yang Ph.D. thesis (Yang, 2007) investigates the use of advanced tabu search methodologies to solve the aircraft-grounding problem and the reduced station capacity problem. The objective is to minimize the schedule recovery costs associated with flight schedule modifications and deviations from the original route. Mei introduces cancellation and delay costs in the objective function. For the delay costs, Mei uses a value of $20 if the delay is less than 15 minutes and $20 each minute if the delay is greater or equal to 15 minutes. For flight cancellations it uses a combination of lost revenue, loss of passenger goodwill and other negative effects, specific and predefined for each flight. The main difference regarding our approach is that we allow the definition of profiles for passengers of each flight (Mei and others, do not consider passengers' profiles). Each one with an associated cost formula, that reflects the delay costs from the passenger point of view.

In (Rosenberger et al., 2001) the authors formulate the problem as a Set Partitioning master problem and a route generating procedure. The goal is to minimize the cost of cancellation and retiming, and it is the responsibility of the controllers to define the parameters accordingly. It is included in the paper a testing process using SimAir (Rosenberger et al., 2002), simulating 500 days of operations for three fleets ranging in size from 32 to 96 aircraft servicing 139-407 flights. Although the authors do try to minimize flight delays, nothing is included regarding the importance of using quality costs.

Crew Recovery: In (Abdelgahny et al., 2004) the flight crew recovery problem for an airline with a hub-and-spoke network structure is addressed. The paper details and sub-divides the recovery problem into four categories: misplacement problems, rest problems, duty problems and unassigned problems. The proposed model is an assignment model with side constraints. Due to the stepwise approach, the proposed solution is sub-optimal. Results are presented for a situation involving a US airline taking into account 18 different problems. This work also omits the use of quality costs for deriving an appropriate solution.
Integrated Recovery: In (Bratu & Barnhart, 2006) the author presents two models that considers aircraft and crew recovery and through the objective function focuses on passenger recovery. They include delay costs that capture relevant hotel costs and ticket costs if passengers are recovered by other airlines. According to the authors, it is possible to include, although hard to calculate, estimations of delay costs to passengers and potential costs of loosing future ticket sales. To test those models an AOCC simulator was developed, simulating domestic operations of a major US airline. It involves 302 aircrafts divided into 4 fleets, 74 airports and 3 hubs. Furthermore, 83869 passengers on 9925 different passengers' itineraries per day are used. For all scenarios solutions are generated with reductions in passenger delays and disruptions. The difference comparing with our approach is that we propose a generic model to calculate the delay cost to passengers, based on their specific profile and opinion (obtained through frequent surveys).
In (Kohl et al., 2004) the author reports on the experiences obtained during the research and development of project DESCARTES (a large scale project supported by EU) on airline disruption management. The current (almost manual) mode of dealing with recovery is presented. They also present the results of the first prototype of a multiple resource decision support system. Passenger delay costs are calculated regarding the delay at the destination and not at departure (we include both in our proposal) and takes into consideration the commercial value of the passenger based on the booked fare class and frequent flyer information. The main difference regarding our proposal is that we use the opinion of the passengers when calculating the importance of the delay.
Lettovsky's Ph.D. thesis (Lettovsky, 1997) is the first presentation of a truly integrated approach in the literature, although only parts of it are implemented. The thesis presents a linear mixed-integer mathematical problem that maximizes total profit to the airline while capturing availability of the three most important resources: aircraft, crew and passengers. The formulation has three parts corresponding to each of the resources, that is, crew assignment, aircraft routing and passenger flow. In a decomposition scheme these are three parts of a master problem known as the Schedule Recovery Model. Although the author takes into consideration the passenger, it does so concerning finding the best solution for the disrupted passengers. The difference of our approach is that we use the opinion of the passengers regarding the delay (expressed through a mathematical formula) to reach the best possible solution concerning delaying the flight. We still do not approach (at least at present time) the, also important, issue of finding the best itinerary for disrupted passengers.

2.2 Current Tools and Systems

In previous work (Castro, 2008) we have classified the current tools (or systems that provide those tools) in use at AOCCs in one of these three categories:

1. Database Query Systems (DBQS)
2. Decision Support Systems (DSS)
3. Automatic or Semi-Automatic Systems (ASAS)

The DBQS – Database Query Systems (the most common situation at airlines) allows the AOCC human operators to perform queries on the existing databases to monitor the airline operation and to obtain other data essential for decision-making. For example, the aircraft and/or crew roster, aircraft maintenance schedule, passenger reservations, and so on. These systems are useful and relatively easy to implement and/or acquire but they have some important disadvantages, for example, to find the best solution and to take the best decision is completely dependent on the human operator. As we have explained in (Castro, 2008) there are two problems when airline companies use only this type of systems: (1) the solution quality is dependent on knowledge and experience of the human operator and, (2) due to the usual difficulty of the human being in leading with large volumes of data simultaneously, they do not use all the necessary information (variables) to take the best decision.

The DSS - Decision Support Systems, besides having the same characteristics of the DBQS, also include additional functionalities to support the human operators on the decision-making. For example, after a request made by a human operator, these systems are able to recommend the best solution to solve a problem related with a delayed aircraft. Some of them may just recommend a flight re-scheduling but others are able to justify the candidate solution as well as to present the solution cost. DSS systems eliminate some of the disadvantages of the DBQS systems. Namely, they are able to analyze large volumes of data and, because of that, propose solutions that take into consideration more information (variables). The decision-making still is on the human operator side but, now, he is able to take better decisions. Unfortunately, one of the big problems with airline companies is the absence and/or complexity of the computerized information system keeping all the operational information. These are of paramount importance for the success of the decision support tools. This problem, referred in (Kohl et al., 2004) as the Data Quality and System Accessibility Problem, gains more importance when we start to implement decision support tools and/or automatic or semi-automatic systems.

The goal of the third type of systems, ASAS – Automatic or Semi-Automatic Systems, is to automate as much as possible the AOCC, replacing the functional part by computerized programs. Specifically, these systems try to automate the repetitive tasks and also the tasks related with searching for the best solution (problem solving). In a totally automatic system, decision-making is also taken by the system. In a semi-automatic system, the final decision is taken by the human operator. In ASAS type of systems, the AOCC does not need as much human operators as in the previous ones, to operate correctly. Usually, roles or functions related with operation monitoring, searching for solutions related with aircraft, crew or passenger problems and re-allocation of resources, are performed by specialists agents (Castro & Oliveira, 2007) replacing the human specialists. The final decision regarding the application of the solution found by these systems on the environment (for example, making the necessary changes on the airline operational plan database) depends on the human supervisor. According to (Wooldridge, 2009) and (Castro, 2007) the agent and multi-agent systems paradigm is more appropriate to be used in this domain than any other paradigm.

2.3 Other Application Domains

To the best of our knowledge, we were the first to propose an organization of agents to represent all roles of an AOCC, including specialist agents that cooperate to achieve the common overall goal of solving the unexpected problems arising during airline operations (Castro, 2007), (Castro & Oliveira, 2007). However, agents and multi-agent systems have been applied both to other problems in air transportation domain and in other application domains. A brief and incomplete list of such applications follows. Tumer and Agogino developed a multi-agent algorithm for traffic flow management (Tumer & Agogino, 2007). Wolfe et al., use agents to compare routing selection strategies in collaborative traffic flow management (Wolfe et al., 2007). For ATC Tower operations, Jonker et al., have also proposed the use of multi-agent systems (Jonker et al., 2005). As a last example, a multi-agent system for the integrated dynamic scheduling of steel production has been proposed by Ouelhadj (Ouelhadj, 2003), (Cowling et al., 2003).

3. Airline Operations Control

In this section we introduce the airline operations control problem – AOCP (also known as airline disruption management problem). To contextualize, we start by briefly introducing the AOCP preceding problem known as the Airline Scheduling Problem (ASP). Then we explain what an airline operational control centre (AOCC) is and we present some typical AOCC organizations. The typical problems, the current disruption management process as well as the main costs involved are also introduced.

3.1 Airline Scheduling Problem

According to (Kohl et al., 2004) the scheduling process of an airline company is composed by the long and short-term phases presented in Figure 1. The scheduling process has three main dimensions or views: (1) passenger view; (2) aircraft view and (3) crew view. The first one represents the seats available to be sold to the airline customers. The other two views, represents resources that will be allocated.

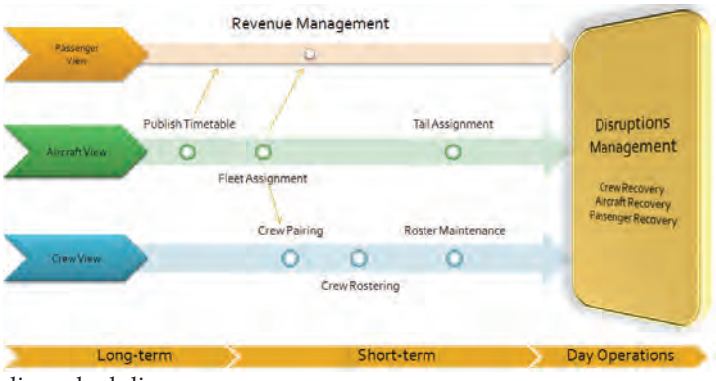

Fig. 1. The airline scheduling process

Everything starts with *publishing the flights timetable* for a specific period of time (usually six months). After publishing the timetable, the *revenue management* phase starts. Here the goal is to maximize the revenue obtained selling tickets. At the same time, the scheduling of the two most important resources starts: aircrafts and crew. Regarding the aircraft, the first step is the *fleet assignment*. Here, the goal is to assign the aircraft type or aircraft fleet that will perform the flights. It is an important step because the aircraft type/fleet will define the number of available seats in each flight. Near to the day of operations, the assignment of the specific aircraft to each flight is performed. This step is known as *tail assignment*. After the fleet assignment step, it is possible to start to schedule the crew. The first step is the *crew pairing*. The goal is to define the crew duty periods (pairings) that will be necessary to cover all the flights of the airline for a specific period of time (typical one month). Having the pairings, it is possible to start the *crew rostering* step that is, assign crewmembers to the pairings. The output of this step is an individual crew roster that is distributed or published in the crew web portal. Finally and until the day of operations, it is necessary to change/updated the crew roster (*roster maintenance*), to include any changes that might appear after publishing the roster. The airline scheduling problem (ASP) is composed of all the previous phases and steps and ends some hours or days (depends on the airline policy) before the day of operation. The global objective of the ASP is to maximize the airline operating profit. For more detailed information please consult (Grosche, 2009) specially Section 2.1 to Section 2.4.

3.2 AOCC Organization

The airline operations control problem (AOCP) starts where the airline scheduling problem stops. In Figure 1 the AOCP is represented by the disruption management square. If everything goes as planned the airline just needs to monitor the execution of the plan. Unfortunately, several unexpected events appear during this phase that can disrupt the plan. To monitor those events and solve the problems that arise from these disruptions and return to the previous plan as soon as possible, it is necessary to define and follow a disruption management process. Airline companies have an entity called Airline Operations Control Centre (AOCC) that is responsible for the disruption management process. There are three main AOCC organizations (Castro, 2008):

- *Decision Centre*: The aircraft controllers share the same physical space. The other roles or support functions (crew control, maintenance service, etc.) are in a different physical space. In this type of *Collective Organization* all roles need to cooperate to achieve the common goal.
- *Integrated Centre*: All roles share the same physical space and are hierarchically dependent of a supervisor. For small companies we have a *Simple Hierarchy Organization*. For bigger companies we have a *Multidimensional Hierarchy Organization*. Figure 2 shows an example of this kind of AOCC organization.
- *Hub Control Centre (HCC)*: Most of the roles are physically separated at the airports where the airline companies operate a hub. In this case, if the aircraft controller role stays physically outside the hub we have an organization called *Decision Centre with a hub*. If the both the aircraft controller and crew controller roles are physically outside the hub we have an organization called *Integrated Centre with a hub*. The main advantage of this kind of organization is to have the roles that are related

with airport operations (customer service, catering, cleaning, passengers transfer, etc.) physically closer to the operation.

The organization adopted depends on several factors like airline size, airline network type (for example, hub-and-spoke) and geographic distribution of the operation. In Figure 2 we present the organization of a typical *Integrated Operational Control Centre*. It is important to point out the role of the supervisor, a characteristic that makes this organization hierarchical and, also, the operation time-window that marks the responsibility boundaries of the AOCC. This operation time-window is different from airline to airline but, usually, ranges from 72 to 24 hours before to 12 to 24 hours after the day of operation.

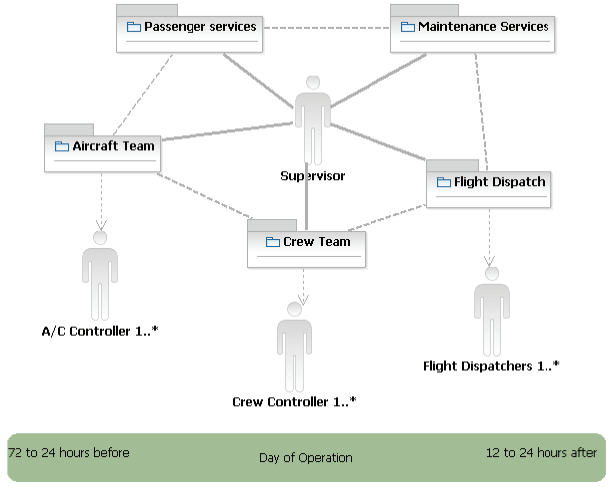

Fig. 2. Integrated airline operational control centre

The roles or support functions more common in an AOCC, according to (Kohl et al., 2004) and (Castro, 2008), are the following:

- *Flight Dispatch*: Prepares the flight plans and requests new flight slots to the Air Traffic Control (ATC) entities (FAA in North America and EUROCONTROL in Europe, for example).
- *Aircraft Control*: Manages the resource aircraft. It is the central coordination role in the operational control.
- *Crew Control*: Manages the resource crew. Monitors the crew check-in and check-out, updates and changes the crew roster according to the disruptions that might appear during the operation.
- *Maintenance Services*: Responsible for the unplanned maintenance services and for short-term maintenance scheduling. Changes on aircraft rotations may impact the short-term maintenance (maintenance cannot be done at all stations).
- *Passenger Services*: Decisions taken on the AOCC will have an impact on the passengers. The responsibility of this role is to consider and minimize the impact of the decisions on passengers. Typical this role is performed on the airports and for bigger companies is part of the HCC organization.

3.3 Typical Problems

In the previous section we presented typical AOCC organizations and the roles that exist on those organizations. Now, it is important to understand the typical problems that appear during the execution of the airline operation. From our observations in a real AOCC, and from (Kohl & Karisch, 2004), we found the typical problems presented in Figure 3. In this diagram we have also included the impact that each problem might have on flight arrival or departure delays as well as the relation that exist between them. The diagram also shows that the problems might propagate due to the relation between them, and generate new problems on different flights. This propagation characteristic makes the problem more difficult to be solved optimally in a real time and dynamic environment, like the one we have on the AOCC.

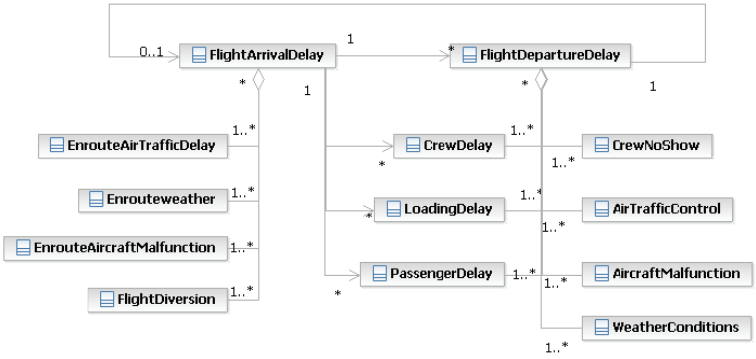

Fig. 3. Typical AOCC problems and relations

As we can see in Figure 3 there is an obvious relation between *Flight Arrival Delays* and *Flight Departure Delays*. Most of the flights are performed by aircrafts that are used in previous flights. If we have an arrival delay and the aircraft turn-around time at the airport is not enough, then, if the AOCC does not find an alternative solution, we will also have a departure delay. From the diagram we can also see that the main reasons for flight arrival delay (besides the delay on departure) are: *En-route air traffic*, *en-route weather*, *en-route aircraft malfunction* and *flight diversion*. In the previous cases and to minimize the arrival delay it is necessary a cooperation between the pilot, the AOCC and ATC. Regarding departure delays, the main reasons are: crew delays, cargo/baggage loading delays and passenger delays as a consequence of an arrival delay. Crewmembers that do not report for duty, air traffic control reasons, aircraft malfunctions and weather conditions (at departure or at arrival) are the other main reasons for departure delays.

3.4 Current Disruption Management Process

As we can see from the previous section, there are several problems that might cause flight delays. AOCCs have a process to monitor the events and solve the problems, so that flight delays are minimized with the minimum impact on passenger and, preferably, with the minimum operational cost. In Figure 4 we present the current disruption management process in use at most of the airlines. This process has five steps:

1. *Operation Monitoring*: In this step the flights are monitored to see if anything is not going according the plan. The same happens in relation with crewmembers, passenger check-in and boarding, cargo and baggage loading, etc.
2. *Take Action*: If an event happens, like for example, a crewmember is delayed or an aircraft malfunction, a quick assessment is performed to see if an action is required. If not, the monitoring continues. If an action is necessary than we have a problem that needs to be solved.
3. *Generate and Evaluate Solutions*: Having all the information regarding the problem the AOCC needs to find and evaluate the candidate solutions. Although there are several costs involved in this process, we found that the AOCC relies heavily on the experience of their controllers and in some rules-of-thumb (a kind of hidden knowledge) that exist on the AOCC.
4. *Take Decision*: Having the candidate solutions a decision needs to be taken.
5. *Apply Decision*: After the decision the final solution needs to be applied in the environment, that is, the operational plan needs to be updated accordingly.

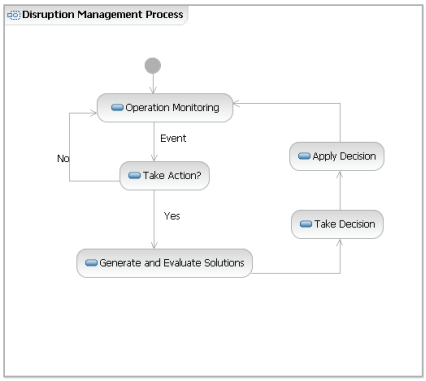

Fig. 4. AOCC disruption management process

In our opinion, this process can greatly benefit from an intelligent agent based approach to the problem, as we will explain in Section 4.

3.5 Main Costs Involved

In the step *Generate and Evaluate Solutions* of the disruption management process on the previous section, we should consider the main costs involved in generating and choosing from candidate solutions. According to our observations these are the main costs involved when generating and evaluating a solution for a specific disruption:

1. *Crew Costs*: the average or real salary costs of the crewmembers, additional work hours and *perdiem* days to be paid, hotel costs and extra-crew travel costs.
2. *Flight Costs*: airport costs (approach and taxing taxes, for example), service costs (cleaning services, handling services, line maintenance, etc.), and average maintenance costs for the type of aircraft, ATC en-route charges and fuel consumption.

3. *Passenger Costs*: passenger airport meals, passenger hotel costs and passenger compensations.

Finally, there is a less easily quantifiable cost that is also included: the cost of delaying or cancelling a flight from the passenger point of view. Most airlines use some kind of rule-of-thumb when they are evaluating the impact of the decisions on passengers. Others just assign a monetary cost to each minute of delay and evaluate the solutions taking into consideration this value. We propose a different way of calculating this cost component.

4. A MAS for Disruption Management in Airline Operations Control

In Section 3 we introduced the Airline Scheduling Problem and the Airline Operations Control Problem (or Disruption Management Problem). We have described the AOCC organization and roles as well as the typical problems that appear during the execution of the operational plan. The disruption management process used by airlines was presented as well as the main costs involved in generating and evaluating the solutions. In this section we present our intelligent agent based approach to solve the Disruption Management Problem in the airline domain. The MAS was developed using Java[1] and JADE (Bellifemine et al., 2004) as the development platform and as the run-time environment that provides the basic services for agents to execute.

4.1 Why an Agent and Multi-Agent System Paradigm?

Considering the agent and multi-agent system characteristics as specified in (Wooldridge, 2009) and (Elamy, 2005), the following ones make us adopt this paradigm to the Airline Operations Control Problem:

- *Autonomy*: MAS models problems in terms of autonomous interacting component-agents, which are a more natural way of representing task allocation, team planning, and user preferences, among others. In Figure 5 the *PaxManager*, *AircraftManager* and *CrewManager* agents (among others) are agents that can choose to respond or not to the requests according to their own objectives.
- *Agents are a Natural Metaphor*: The AOCC is naturally modelled as a society of agents cooperating with each other to solve such a complex problem.
- *Reactivity*: Agents are able to perceive and react to the changes in their environment. The *Monitor* agent in Figure 5 is an example of such an agent.
- *Distribution of resources*: With a MAS we can distribute the computational resources and capabilities across a network of interconnected agents avoiding problems associated with centralized systems. Airline companies of some dimension have different operational bases. We use a MAS for each operational base, taking advantage of this important characteristic. Due to the *social awareness* characteristics of some of our agents (for example, *Monitoring* agent in Figure 5) they are able to distribute their tasks among other agents with similar behaviour.
- *Modularity and Scalability*: A MAS is extensible, scalable, robust, maintainable, flexible and promotes reuse. These characteristics are very important in systems of

1 http://www.java.com

this dimension and complexity. Our MAS is able to scale in terms of supporting more operational bases as well as in supporting different algorithms to solve specific problems.

- *Concurrency/Parallelism*: Agents are capable of reasoning and performing tasks in parallel. This provides flexibility and speeds up computation. The *CrewSimAnneal*, *CrewCBR* and *CrewHillClimb* agents in Figure 5, are examples of concurrent agents. Additionally and according to (Stone & Veloso, 2000) "if control and responsibilities are sufficiently shared among agents, the system can tolerate failures by one or more agents". Our MAS can be totally or partially replicated in different computers. If one or more agents fail, the global objective is not affected.
- *Legacy Systems*: The AOCC needs information that exists in obsolete but functional systems. We can wrap the legacy components in an agent layer, enabling them to interact with other software components.

4.2 MAS Architecture

It is important to point out that we arrived to the architecture of our multi-agent system, after performing an analysis and design using an agent-oriented software methodology (Castro & Oliveira, 2008). The agent model and service model were the outputs of this process and the base for this architecture.

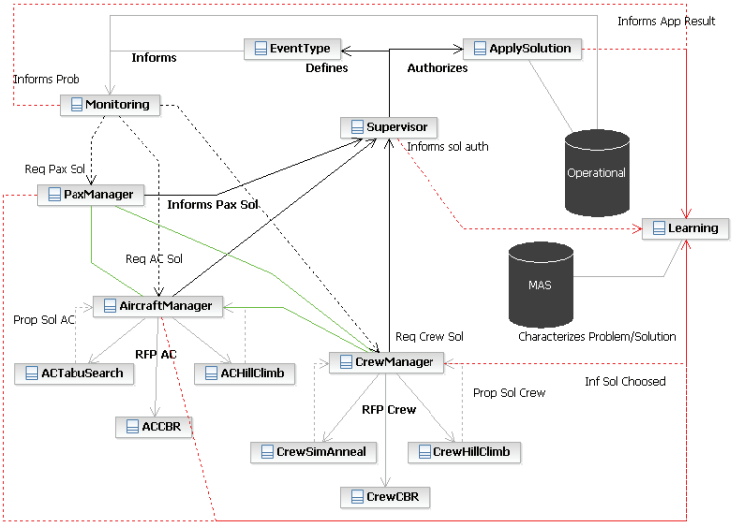

Fig. 5. MAS architecture

Figure 5 shows the architecture of our multi-agent system approach. The boxes represent agents and the narrow black dash lines represent requests/proposals made. The larger green lines represent the interaction between agents regarding negotiation and distributed problem-solving process. The narrow gray lines represent interaction within a hierarchy of agents and the normal black lines represent the interactions after a solution is found. It is important to clarify that Figure 5 represents only one instance of the MAS. We can replicate

almost all agents with the exception of the *Supervisor* agent because it is the one that interacts with the human supervisor (an application domain restriction). Each one of the agents *Monitoring*, *PaxManager*, *AircraftManager*, *CrewManager* and *Supervisor* has a specific role in the AOCC. The *Monitoring* agent monitors the operational plan looking for events that may represent any of the usual three problem dimensions, that is, aircraft, crew and/or passenger problems. In case there are other instances of this agent, they recognize and interact with each other, splitting the monitoring task. For example, if each instance corresponds to an operational base, each one will monitor the corresponding operation plan. This is one example of the social-awareness characteristic of our agents. The agent is autonomous in the sense that it will consider an event as a problem only if the event has certain characteristics.

The *PaxManager* agent has the responsibility to find solutions for passenger problems. The *AircraftManager* and *CrewManager* agents have the responsibility for finding solutions for aircraft and crew problems, respectively. These agents are autonomous in the sense that they can choose not to respond to the information received from the *Monitor* agent, i.e., if the problem is not related with their field of expertise or if they do not have local resources to solve that problem. These agents have similar social-awareness characteristics of the *Monitor* agent. Although not yet implemented, these agents may decide to participate with their expertise in the integrated and distributed problem solving approach of the system.

The *AircraftManager* and *CrewManager* agents manage a team of specialized agents (Castro & Oliveira, 2007). Each team should have several specialist agents, each one implementing a different problem solving algorithm, making them heterogeneous regarding this characteristic. The *ACTabuSearch* agent, *ACCBR* agent and *ACHillClimb* agent implements algorithms dedicated to solve aircraft problems and present the candidate solutions they find to the *AircraftManager* agent. The *CrewSimAnneal* agent, *CrewHillClimb* agent and *CrewCBR* agent implements algorithms dedicated to solve crew problems and present the candidate solutions to the *CrewManager*.

The agent *Supervisor* and agent *EventType* are the only ones that interact with a human user of the AOCC. The *Supervisor* agent presents the solutions to the human supervisor, ranked according to the criteria in use by the airline (more information on the next section), including details about the solution to help the human to decide. After getting approval from the human supervisor, the *Supervisor* agent requests *ApplySolution* agent to apply it on the environment.

All agents are able to act and observe the environment that is represented by the *Operational* and *MAS* database, in our diagram. The operational database includes information regarding the flight, aircraft and crew schedule as well as airport and company specific information. The other database is related with the learning characteristics of our system and is used, mainly, by the *Learning* agent. The learning characteristics of our system are not yet implemented. In Section 7, the interested reader can find more information about the way we expect to apply learning in our MAS. Finally, the protocols we use are the following FIPA compliant ones:

- *Fipa-Request*: This protocol allows one agent to request another to perform some action and the receiving agent to perform the action or reply, in some way, that it cannot perform it. Fipa-request is used in interactions between the *Monitor*, *PaxManager*, *AircraftManager* and *CrewManager* agents.

- *Fipa-Query*: This protocol allows one agent to request to perform some kind of action on another agent. It is used in the interactions that involve *PaxManager*, *AircraftManager*, *CrewManager* and *Supervisor* agent; *Supervisor*, *ApplySolution* and *EventType* agent and, finally, *EventType* and *Monitoring* agent.
- *Fipa-Contract.net*: "In the contract net protocol, one agent (the Initiator) takes the role of manager which wishes to have some task performed by one or more other agents (the Participants) and further wishes to optimize a function that characterizes the task"(Fipa, 2002). We use a simplified version of this protocol in the interactions that entail the *AircraftManager* and its specialized agents, i.e., *ACTabuSearch*, *ACCBR* and *ACHillClimb;* and *CrewManager* and its specialized agents, i.e., *CrewSimAnneal*, *CrewHillClimb* and *CrewCBR*. More information about how we use this protocol is presented in the next section.

4.3 Decision Mechanisms

Our system uses negotiation at two levels. The first level is the *Manager Agents* level, i.e., between *PaxManager*, *CrewManager* and *AircraftManager* agents. At this level the agents cooperate so that an integrated solution can be found. We define an integrated solution as one that considers the impact on the three dimensions of the problem, that is, aircraft, crew and passengers. As of the writing of this paper, we do not have this negotiation protocol completely implemented. Section 7 gives a glimpse of how we are implementing it. The second level is the *Specialist Agents or Team level*, i.e., between each manager agent and the specialist agents of the team. At this level we have used a simplified fipa-contract.net (Fipa, 2002) (Smith, 1980).

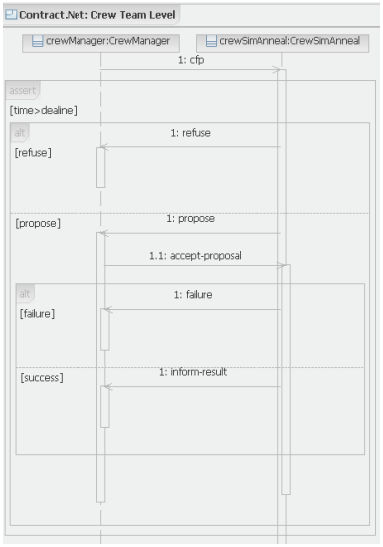

Fig. 6. Simplified contract net protocol

Figure 6 shows the simplified contract.net protocol applied to the *CrewManager* team (for simplicity only the interaction between *CrewManager* and one of the specialist agents is shown). After receiving a request from the *Monitoring* agent and case the *CrewManager* agent

decides to reply, a Call for Proposal (cfp) is issued to initiate the negotiation process. Table 1 shows an example of a message sent in this step.

```
(cfp
   :sender (agent-identifier :name CrewManager@masdima:1099/JADE
      :addresses (sequence http://masdima:7778/acc))
   :receiver (set (agent-identifier :name CrewSimAnneal@masdima:1099/JADE
      :addresses (sequence http://masdima:7778/acc)))
   :X-JADE-Encoding Base64
   :content "ABXN rO0AAAA eHB4h3CAAAAAsAAAAAeHB4"
   :language  fipa-sl
   :conversation-id  cfp_crew_solution)
```

Table 1. CFP message sent by CrewManager agent

Please note that the content of the FIPA-ACL message is a serialized Java object (see Table 2), that contains the event description, as well as the deadline for receiving an answer (propose or refuse) and the deadline for receiving the candidate solution (i.e., the *CrewSimAnneal* agent needs to send a candidate solution before a specific period of time).

```
import java.io.Serializable;
import java.util.ArrayList;
import java.util.GregorianCalendar;

public class CrewProblem implements Serializable {
   private ArrayList<Event> events;
   private int numSeconds;
   private int maxCost;
   private int numMinutesTimeWindow;
   private GregorianCalendar bidDeadline;
   private GregorianCalendar candSolutionDeadline;
   public ArrayList<Event> getEvents() {return events;}
   (…)
   public int getMaxCost() {return maxCost;}
   public void setMaxCost(int maxCost) {this.maxCost = maxCost;}
   public GregorianCalendar getBidDeadline() {return bidDeadline;}
   public void setBidDeadline(GregorianCalendar bidDeadline) {this.bidDeadline = bidDeadline;}
   public GregorianCalendar getCandSolutionDeadline() {return candSolutionDeadline;}
   public void setCandSolutionDeadline(GregorianCalendar candSolutionDeadline) {
               this.candSolutionDeadline = candSolutionDeadline;}
   public CrewProblem(ArrayList<Event> events, int numSeconds, int maxCost, int numMinutesTimeWindow,
         GregorianCalendar bidDeadline, GregorianCalendar candSolutionDeadline)
   {
      this.maxCost = maxCost;
      this.events = events;
      this.numSeconds = numSeconds;
      this.numMinutesTimeWindow = numMinutesTimeWindow;
      this.bidDeadline = bidDeadline;
      this.candSolutionDeadline = candSolutionDeadline;
   }
}
```

Table 2. Partial example of a Serialized Java object included in the CFP message

The *CrewSimAnneal* agent may choose to answer refuse or propose. In our approach the *CrewSimAnneal* propose performative only means that it will look for a candidate solution according to the conditions of the cfp. The *CrewManager* agent will automatically answer back with an accept-proposal. Here we simplified the contract.net protocol to speed-up the communication between our agents. In our case, the answer we get from specialist agents is a simple yes or no, because we want all available agents (i.e., that are not busy looking for candidate-solutions for other requests) to work in parallel to find candidate solutions.

Because of that we do not need to choose between all the answers received. If there is a problem during the execution of the task, the *CrewSimAnneal* agent issues a failure performative stating the reasons for the failure, in the serialized Java object included in the message content. If the agent is able to perform the task with success, it will issue an inform-result performative (Table 3) that includes the serialized object (Table 4) with the candidate solution.

```
(inform
  :sender (agent-identifier :name CrewSimAnneal @masdima:1099/JADE
    :addresses (sequence http://masdima:7778/acc))
  :receiver (set (agent-identifier :name CrewManager@masdima:1099/JADE
    :addresses (sequence http://masdima:7778/acc)))
  :X-JADE-Encoding Base64
  :content "eHB4h3CAAAAAsAAAAAeHABXNrO0AAAAB4"
  :language fipa-sl
  :conversation-id cfp_crew_solution)
```

Table 3. Example of a Failure and Inform message

```
import java.io.Serializable;
import java.util.ArrayList;

public class CrewSolution implements Serializable {
   private int cost;
   private int initialCost;
   private String description;
   private ArrayList<Flight> solution;

   public int getCost() {return cost;}
   public void setCost(int cost) {this.cost = cost;}
   public int getInitialCost() {return initialCost;}
   public void setInitialCost(int initialCost) {this.initialCost = initialCost;}
   public String getDescription() {return description;}
   public void setDescription(String description) {this.description = description;}
   public ArrayList<Flight> getSolution() {return solution;}
   public void setSolution(ArrayList<Flight> solution) {this.solution = solution;}
   public CrewSolution(ArrayList<Flight> solution, String description, int cost, int
initialCost)
   {
      setCost(cost);
      setInitialCost(initialCost);
      setDescription(description);
      setSolution(solution);
   }
}
```

Table 4. Serialized Java object included in the Inform Message

At the team level, the manager agent needs to select the best solution from the candidate solutions that were found by the specialist agents. As of the writing of this paper, we use the *Total Operational Cost* as the only criteria for the selection. Other criteria, like *AOCC Global Performance*, are being tested but we do not have any results at this moment.
The *Total Operational Cost (tc)* of a specific solution includes *Direct Operational Costs (dc)* and *Quality Operational Costs (qc)* and is given by Equation 1.

$$tc = dc + \beta qc \qquad \beta \in R, \beta \geq 0 \tag{1}$$

Coefficient β is used to define the weight of quality costs.

Direct Operational Costs (dc) of a specific solution are costs that are easily quantifiable and are related with the operation of the flights, namely, *Crew Costs (cc)*, *Flight Costs (fc)* and *Passenger Costs (pc)*. It is given by Equation 2.

$$dc = cc + fc + pc \tag{2}$$

The *Crew Cost (cc)* for a specific flight includes the salary costs of all crew members (*Salary*), additional work hours to be paid (*Hour*), additional perdiem days to be paid (*Perdiem*), hotel costs (*Hotel*) and extra-crew travel costs (*Dhc*). The *Crew Cost* for a specific solution is given by Equation 3.

$$cc = \sum_{i=1}^{|F|}\sum_{j=1}^{|C|}(Salary_{\{i,j\}} + Hour_{\{i,j\}} + Perdiem_{\{i,j\}} + Hotel_{\{i,j\}} + Dhc_{\{i,j\}}) \tag{3}$$

where
$i \in F; F = \{all\ flights\ in\ solution\}$
$j \in C; C = \{all\ crewmembers\ in\ flight\}$

The *Flight Cost (fc)* for a specific flight includes the airport costs (*Airp*), i.e., charges applied by the airport operator like approaching and taxing; service costs (*Service*), i.e., flight dispatch, line maintenance, cleaning services and other costs; average maintenance costs for the type of aircraft that performs the flight (*Maint*); ATC en-route charges (*Atc*); and fuel consumption (*Fuel*), i.e., fuel to go from the origin to the destination (trip fuel) plus any additional extra fuel required. The *Flight Cost* for a specific solution is given by Equation 4.

$$fc = \sum_{i=1}^{|F|}(Airp_i + Service_i + Maint_i + Atc_i + Fuel_i) \tag{4}$$

where
$i \in F; F = \{all\ flights\ in\ solution\}$

The *Passenger Cost (pc)* of the delayed passengers for a specific flight includes airport meals the airline has to support when a flight is delayed or cancelled (*Meals*), hotels costs (*PHotel*) and any compensation to the passengers according to regulations (*Comp*). The *Passenger Cost* of the delayed passengers for a specific solution is given by Equation 5.

$$pc = \sum_{i=1}^{|F|}\sum_{d=1}^{|D|}\left(Meals_{\{d,i\}} + PHotel_{\{d,i\}} + Comp_{\{d,i\}}\right) \tag{5}$$

where
$i \in F; F = \{all\ flights\ in\ solution\}$
$d \in D; D = \{all\ delayed\ passengers\ in\ flight\}$

Quality Operational Costs (qc) of a specific solution are costs that are not easily quantifiable and are related with passenger satisfaction. The quantification of this value is very important to increase the quality level of an airline company when facing a disruption. Equation 6 presents a generic expression that calculates this value according to (Castro & Oliveira, 2009).

$$qc = \alpha \sum_{i=1}^{|F|} \sum_{p=1}^{|PP|} \left(P_{\{p,i\}} * C_{\{p,i\}} \right) \tag{6}$$

where
$i \in F; F = \{all\ flights\ in\ solution\}$
$p \in PP; PP = \{flight\ passengers\ profiles\}$
$P = number\ of\ passengers\ of\ profile\ p$
$C = delay\ cost\ of\ each\ passenger\ on\ profile\ p$
$\alpha = coefficient\ to\ convert\ to\ monetary\ costs$

Now that we know the main costs involved, it is time to understand how each manager agent selects the best (or the best x candidate solutions). Once the participant agent has completed the task (for example, agent *CrewHillClimb* in Figure 5), it sends a completion message to the initiator (agent *CrewManager* in Figure 5) in the form of an *inform-result performative* (Table 3), with the details of the candidate solution (Table 4) including the *Total Operational Cost*. The manager agent sorts, in ascending order, all candidate solutions received by total operational cost. The top three solutions are selected (Castro & Oliveira, 2007).

4.4 Problem Solving Algorithms

As it is possible to see in Figure 5 (Section 4.2), the aircraft and crew dimension have, each one, a team of specialist agents. Each agent should implement a heterogeneous problem solving algorithm on the team they belong to. Preliminary results show that a single problem solving algorithm is not able to solve, dynamically and within the required time restriction, all types of problems that we have identified during our observations (see Section 3.3). Taking advantage of the modularity, scalability and distributed characteristics of the MAS paradigm, we are able to add as many specialist agents as required, so that all types of problems are covered. As we have seen in Section 4.2 and 4.3, the idea is to have all specialist agents of a team looking for solutions concurrently.
In this section we are going to show how we have implemented one of the specialist agents of the crew team, namely, *CrewHillClimb*. This agent implements a hill climb algorithm. For more details regarding how we have implemented this and other specialist agents, please read (Mota, 2007).

The hill climbing agent solves the problem iteratively by following the steps:

1. Obtains the flights that are in the time window of the problem. This time window starts at the flight date, and ends at a customizable period in the future. This will be the initial solution of the problem. The crew members' exchanges are made between flights that are inside the time window of the problem.
2. While some specific and customizable time has not yet passed, or a solution below a specific and customizable cost has not been found, repeats steps 3 and 4.
3. Generates the successor of the initial solution (the way a successor is generated is described below).
4. Evaluates the cost of the solution. If it is smaller than the cost of the current solution, accepts the generated solution as the new current solution. Otherwise, discards the generated solution. The way a solution is evaluated is described below.

5. Send the current solution to the *CrewManager* agent following the protocol as we have seen in Section 4.3.

The generation of a new solution is made by finding a successor that distances itself to the current solution by one unit, that is, the successor is obtained by one, and only one, of the following operations:

- Swap two crewmembers between flights that belong to the flights that are in the time window of the problem.
- Swap a crewmember of a flight that belongs to the flights that are in the time window of the problem with a crewmember that isn't on duty, but is on standby.

When choosing the first element to swap, there are two possibilities: (1) choose randomly or (2) choose an element that is delayed. The choice is made based on the probability of choosing an element that is late, which was given a value of 0.9, so that the algorithms can proceed faster to good solutions (exchanges are highly penalized, so choosing an element that is not late probably won't reduce the cost, as a possible saving by choosing a less costly element probably won't compensate the penalization associated with the exchange).
If the decision is to exchange an element that is delayed, the list of flights will be examined and the first delayed element is chosen. If the decision is to choose randomly, then a random flight is picked, and a crewmember or the aircraft is chosen, depending on the probability of choosing a crewmember, which was given a value of 0.85. When choosing the second element that is going to swap with the first, there are two possibilities: (1) swap between elements of flights or (2) swap between an element of a flight and an element that is not on duty. The choice is made based on the probability of choosing a swap between elements of flights, which was given a value of 0.5.
The evaluation of the solution is done by an objective function that measures the following types of costs:

- The crew cost according to Equation 3;
- The penalization for exchanging elements;
- The penalization for delayed elements. The cost associated with this aspect is the highest, because the goal is to have no delayed elements.

The *Hill Climbing Objective Function (hc)* is given by Equation 7.

$$hc = cc + excW * nExc + delayW * nDelay \tag{7}$$

where
cc = *crew cost according to equation 3*
$excW$ = *penalization for crew exchanges*
$nExc$ = *the number of crew exchanges*
$delayW$ = *penalization for delaying crewmembers*
$nDelay$ = *the number of delayed crewmembers*

Table 5 shows the implementation of the hill climbing algorithm in Java.

```
GregorianCalendar currentDate = new GregorianCalendar();
int secondsExecution = (int) ((currentDate.getTimeInMillis() - startDateResolution.getTimeInMillis()) / 1000);
while(!Shared.to(problem.getNumSeconds(), secondsExecution, problem.getMaxCost(), currentSolutionCost))
{
   // get successor
```

```
    successor = Shared.generateSuccessor(Shared.copyArrayList(currentSolution));
    // checks if successor has an inferior solution cost
    successorCost = Shared.calculateCost(successor, initialPlainSolution);
    System.out.println("Successor Cost: " + successorCost + "\n");
    if(sucessorCost < currentSolutionCost)
    {
       currentSolution = successor;
       currentSolutionCost = successorCost;
    }
    currentDate = new GregorianCalendar();
    secondsExecution = (int) ((currentDate.getTimeInMillis() - startDateResolution.getTimeInMillis()) / 1000);
 }
```

Table 5. Implementation of the Hill-Climbing algorithm in Java

5. Experimental Setup

To evaluate our approach we have setup a scenario that includes 3 operational bases (A, B and C). Each base includes their crewmembers each one with a specific roster. The data used corresponds to a real airline operation of June 2006 of base A. We have simulated a situation where 15 crewmembers, with different ranks, did not report for duty in base A. A description of the information collected for each event is presented in Table 6.

Attribute	Description
Event ID	A number that represents the ID of the event. For tracking purposes only
Duty Date Time	The start date and time of the duty in UTC for which the crew did not report.
Duty ID	A string that represents the ID of the duty for which the crew did not report.
Flt Dly	Flight delay in minutes
C Pax	Number of passengers in business class
Y Pax	Number of passengers in economy class
End Date Time	The end date and time of the duty in UTC for which the crew did not report.
Ready Date Time	The date and time at which the crew member is ready for another duty after this one.
Delay	The delay of the crewmember. We have considered 10 minutes in our scenario.
Credit Minutes	The minutes of this duty that will count for payroll.
Crew Group	The crew group (Technical = 1; Cabin = 2) that the crewmember belongs to.
Crew Rank	CPT = Captain; OPT = First Officer; CCB = Chief Purser; CAB = Purser.
Crew Number	The employee number.
Crew Name	The employee name.
Base ID	The base where the event happened. We considered all events in base A.
Open Positions	The number of missing crews for this duty and rank. We used a fixed number of 1.

Table 6. Description of the information collected for each event

The events did not happen at the same day and each one corresponds to a crewmember that did not report for a specific duty in a specific day. Table 7 shows the data for each of the events created. As you can see we have omitted the information regarding *Delay*, *Base ID* and *Open Positions* because we have used fixed values as indicated in Table 6. For example, the event 10 corresponds to the following situation: Crew Peter B, with number 32 and rank CPT (captain) belonging to the crew group 1 (technical crew), did not report for the duty with ID 1ZRH12X with briefing time (duty date time) at 15:25 in 15-06-2006. This flight did not delay on departure and has 5 passengers in business class and 115 in economy class. The event was created after a 10 minutes delay of the crewmember in reporting for duty and happened at base A. It is necessary to find another crewmember to be assigned to this duty. The duty ends at 09:30 on 17-06-2006 and the crewmember assigned to this duty will be ready for another one at 21:30 in 17-06-2006. The duty will contribute with 1318 minutes (21h58) for the payroll. The new crewmember must belong to the same rank and group.

After setting-up the scenario we found the solutions for each crew event using three methods.

	Duty DateTime	DutyID	Flt Dly	C Pax	Y Pax	End DateTime	Ready DateTime	Cred Min	Crew Grp	Rnk	Crw Nr	Crew Name
1	05-06 07:25	1ORY149S	0	7	123	05-06 13:35	06-06 01:35	370	2	CAB	80	John A
2	05-06 07:25	1ORY149S	10	11	114	05-06 13:35	06-06 01:35	370	2	CAB	45	Mary A
3	05-06 07:25	1ORY85P	0	10	112	05-06 13:35	06-06 01:35	370	1	CPT	35	Anthony
4	15-06 04:10	2LIS24X	30	0	90	16-06 16:15	17-06 04:15	1757	2	CAB	99	Paul M
5	15-06 04:10	3LIS25X	25	3	77	15-06 09:20	15-06 21:20	632	2	CAB	56	John B
6	15-06 12:50	2LHR63P	5	25	85	16-06 20:45	17-06 08:45	1549	1	CPT	57	Paul S
7	15-06 12:50	2LHR63P	0	20	95	16-06 20:45	17-06 08:45	1549	1	OPT	53	Mary S
8	15-06 14:15	1LHR31P	0	23	52	15-06 20:55	16-06 08:55	843	2	CCB	23	Sophie
9	15-06 15:25	2LHR19P	10	27	105	16-06 20:45	17-06 08:45	1341	2	CCB	34	Angel
10	15-06 15:25	1ZRH12X	0	5	115	17-06 09:30	17-06 21:30	1318	1	CPT	32	Peter B
11	25-06 05:20	1LIS16S	20	3	97	25-06 15:05	26-06 03:05	585	2	CAB	20	Paul G
12	25-06 05:20	1LIS16S	5	2	108	25-06 15:05	26-06 03:05	585	2	CAB	10	Alice
13	25-06 05:20	1LIS158T	0	4	92	25-06 15:05	26-06 03:05	585	2	CAB	15	Daniel
14	25-06 06:15	3LIS174S	0	1	129	27-06 16:15	28-06 04:15	1258	2	CAB	71	George
15	25-06 14:20	4LIS50A	0	2	83	28-06 19:40	29-06 07:40	219	1	OPT	65	Allan

Table 7. Events data used for testing

In the first method we used one of the best users from the AOCC, with current tools available, to find the solutions. The user uses software that shows the roster of each crewmember in a Gantt chart for a specific period. The user can scroll down the information, filter according to the crew rank and base, and sort the information by name, month duty, etc. Each user has a specific way of trying to find the solutions. However, we have observed that, in general, they follow these steps:

1. Open the roster for a one month period, starting two days before the current day. For example, let's suppose that the current day is 7th of June of 2006, they open the roster from the 5th of June until the 4th of July.
2. Filter the roster by crew rank and base, where the base is equal to the base where the crew event happened and crew rank is equal to the crewmember rank that did not report for duty.
3. Order the information by month duty, in an ascendant order and by seniority in a descendent order.
4. Visually, they scroll down the information until they found a crewmember with an open space for the period of time that corresponds to the duty to be assigned. This period of time takes into consideration the start and end time of the duty and also the time required for resting (ready date time).
5. If they do not found a crewmember in the base specified, they try to find it in another base, filtering the information accordingly.
6. They assign the duty to the crewmember with less *credit hours*.

The data collected using this method is presented in Table 8. We point out that the data in columns marked with an asterisk where calculated manually, according to the equations presented in chapter 4.3. The reason for this is that the information system that is available for the users does not include information related with any kind of costs.

	Duty ID	Base ID	Crew Grp	Rank	Hour Pay (*)	Perdiem Pay (*)	Quality Op. Cost	Op. Cost (*)
1	1ORY149S	A	2	CAB	0,00	72,00	0	72,00
2	1ORY149S	B	2	CAB	0,00	72,00	0	86,40

3	1ORY85P	A	1	CPT	942,90	106,00	0	1048,90
4	2LIS24X	A	2	CAB	939,00	144,00	0	1083,00
5	3LIS25X	B	2	CAB	0,00	72,00	0	86,40
6	2LHR63P	B	1	CPT	777,00	212,00	0	1186,80
7	2LHR63P	B	1	OPT	0,00	148,00	0	177,60
8	1LHR31P	A	2	CCB	687,65	72,00	0	759,65
9	2LHR19P	B	2	CCB	0,00	144,00	0	172,80
10	1ZRH12X	C	1	CPT	0,00	212,00	0	296,80
11	1LIS16S	A	2	CAB	0,00	72,00	0	72,00
12	1LIS16S	C	2	CAB	0,00	72,00	0	100,80
13	1LIS158T	B	2	CAB	0,00	72,00	0	86,40
14	3LIS174S	A	2	CAB	1051,60	216,00	0	1267,60
15	4LIS50A	A	1	OPT	246,40	296,00	0	542,40
	Totals				**4644,55**	**1982,00**	0	**7039,55**

Table 8. Data collected (partial) after using method 1 (human user)

In the second method we have used our approach as indicated in Section 4 but with β=0 in Equation 1 (*Total Operational Cost*), i.e., although we calculate the *Quality Operational Cost* as indicated in Equation 6 we did not considered this value in resolution as well as in the decision process. The data collected is presented in Table 9.

	Duty ID	Base ID	Crew Grp	Rank	Hour Pay	Perdiem Pay	Quality Op. Cost	Direct Op. Cost
1	1ORY149S	A	2	CAB	0,00	72,00	0	72,00
2	1ORY149S	B	2	CAB	0,00	72,00	501,31	86,40
3	1ORY85P	B	1	CPT	0,00	106,00	0	127,20
4	2LIS24X	C	2	CAB	563,40	62,00	1561,76	875,56
5	3LIS25X	B	2	CAB	0,00	72,00	1877,73	86,40
6	2LHR63P	C	1	CPT	0,00	212,00	658	296,80
7	2LHR63P	A	1	OPT	0,00	144,00	687,62	144,00
8	1LHR31P	B	2	CCB	229,17	72,00	0	361,40
9	2LHR19P	B	2	CCB	0,00	144,00	788,78	172,80
10	1ZRH12X	C	1	CPT	0,00	212,00	0	296,80
11	1LIS16S	A	2	CAB	0,00	72,00	961,95	72,00
12	1LIS16S	C	2	CAB	0,00	72,00	301,48	100,80
13	1LIS158T	B	2	CAB	0,00	72,00	0	86,40
14	3LIS174S	C	2	CAB	411,00	93,00	0	705,60
15	4LIS50A	B	1	OPT	0,00	296,00	449,84	355,20
	Totals				**1203,57**	**1773,00**	**7788,47**	**3839,36**

Table 9. Data collected (partial) after using method 2 (No Quality Costs)

In the third method we have used our approach as indicated in Section 4 but with β=1 in Equation 1, i.e., considering the *Quality Operational Cost* in the resolution as well as in the decision process. The *Quality Operational Cost* was calculated using two passenger profiles (business and economy classes) and with α=0,1. Equation 9 and Equation 10 are the formulas used to calculate the delay cost of each passenger in business and economy profile, respectively. For more information about how we reached these equations, please read (Castro & Oliveira, 2009).

$$C_{business} = 0.16 * x^2 + 1.38 * x \qquad x = \text{minutes of flight delay, } x >= 0 \tag{9}$$

$$C_{economy} = 1.20 * x \qquad x = \text{minutes of flight delay, } x >= 0 \tag{10}$$

The data collected is presented in Table 10.

	Duty ID	Base ID	Crew Grp	Rank	Hour Pay	Perdiem Pay	Quality Op. Cost	Direct Op. Cost
1	1ORY149S	A	2	CAB	0,00	72,00	0	72,00
2	1ORY149S	B	2	CAB	0,00	72,00	501,31	86,40
3	1ORY85P	B	1	CPT	0,00	106,00	0	127,20
4	2LIS24X	C	2	CAB	503,50	144,00	1060,92	906,50
5	3LIS25X	C	2	CAB	0,00	72,00	1420,78	100,80
6	2LHR63P	B	1	CPT	102,90	212,00	272,10	377,88
7	2LHR63P	B	1	OPT	37,22	144,00	0	217,46
8	1LHR31P	B	2	CCB	229,17	72,00	0	361,40
9	2LHR19P	B	2	CCB	0,00	144,00	788,78	172,80
10	1ZRH12X	C	1	CPT	0,00	212,00	0	296,80
11	1LIS16S	A	2	CAB	0,00	80,00	593,30	80,00
12	1LIS16S	C	2	CAB	0,00	80,00	144,34	112,00
13	1LIS158T	B	2	CAB	0,00	72,00	0	86,40
14	3LIS174S	C	2	CAB	411,00	93,00	0	705,60
15	4LIS50A	A	1	OPT	138,83	288,00	0	426,83
	Totals				**1422,62**	**1863,00**	**4781,53**	**4130,07**

Table 10. Data collected (partial) after using method 3 (Quality Costs)

6. Results and Discussion

Table 11 shows a comparison of the results obtained through the above methods. We point out that in method 1 (*human*) we did not calculate the quality costs, and in method 2 (*agent-no-quality*) we did calculate the quality costs but they were not used to find the best solution, although we use that value for comparison purposes. From the results obtained we can see that on average, method 3 (*agent-quality*) produced solutions that decreased flight delays in aprox. 36%. *Agent-quality* is, on average 3% slower than *agent-no-quality* in finding a solution and produces solutions that represent a decrease of 23% on the total operational costs, when compared with *agent-no-quality*.

From the results (Table 11) we can see that our approach obtains valid solutions faster and with fewer direct operational costs when compared with the current method used in a real airline company (*human*). *Agent-no-quality* represents a decrease of aprox. 45,5% and *agent-quality* a decrease of aprox. 41%. *Agent-quality* has a higher direct operational cost than *agent-no-quality* because it uses the quality operational cost in the decision process. If we read this number without any other consideration, we have to say that the goal of having less direct operational costs was not achieved. An 8% increased on direct operational costs can represent a lot of money. However, we should read this number together with the flight delay figure. As we can see, although *agent-quality* has increased the direct operational costs (when compared with *agent-no-quality*) in 8% it was able to choose solutions that decrease, in average, 36% of the flight delays. This means that, when there are multiple solutions to the same problem, *agent-quality* is able to choose the one with less operational cost, less quality costs (hence, better passenger satisfaction) and, because of the relation between quality costs and flight delays, the solution that produces shorter flight delays.

	Human (M1)		Agent-no-Quality (M2)		Agent-Quality (M3)	
	Total	%	**Total**	%	**Total**	%
Base of the solution:						
- From the crew event base (A)	7	47%	3	20%	3	20%
- From base B	6	40%	7	47%	7	47%

- From base C	2	13%	5	33%	5	33%
Time to Find Solution (avr sec)	**101**	**100,00%**	**25**	**24,75%**	**26**	**25,74%**
Flight Delays (avr min):			**11**	**100,00%**	**7**	**63,64%**
- Base A (avr)			14	40%	7	30%
- Base B (avr)			9	26%	4	17%
- Base C (avr)			12	34%	12	52%
Total Direct Operational Costs:	**7039,60**	**100,00%**	**3839,36**	**54,54%**	**4130,07**	**58,67%**
Total by Base:						
- Base A	4845,55	92,42%	288,00	11,23%	578,83	14,02%
- Base B	1796,40	34,26%	1275,80	49,77%	1429,54	34,61%
- Base C	397,60	7,58%	2275,56	88,77%	2121,70	51,37%
Total Quality Operational Cost:			**7788,47**	**100%**	**4781,53**	**61,39%**
Total by Base:						
- Base A			1649,57	21,18%	593,30	12,41%
- Base B			3617,66	46,45%	1562,19	32,67%
- Base C			2521,24	32,37%	2626,04	54,92%
Total Operational Costs:			**11628,01**	**165%**	**8911,60**	**126,6%**
Total by Base:						
- Base A			1937,57	16,66%	1172,13	13,15%
- Base B			4088,42	35,16%	2991,73	33,57%
- Base C			4796,80	41,25%	4747,74	53,28%

Table 11. Summary of the results obtained by each method

From this conclusion, one can argue that if we just include the direct operational costs and the expected flight delay, minimizing both values, the same results could be achieved having all passengers happy. In general, this assumption might be true. However, when we have to choose between two solutions with the same direct operational cost and delay time, which one should we choose? In our opinion, the answer depends on the profile of the passengers of each flight and on the importance they give to the delays (quality operational cost), and not only in minimizing the flight delays and direct operational cost. *Agent-quality* takes into consideration this important information when making decisions. This is the reason why we think that one of the main contributions of our work is the generic approach to quantify the passenger satisfaction regarding delaying a flight, from the passenger point of view. It is fair to say that we cannot conclude that our MAS will always have this behaviour. For that we need to evaluate a higher number of scenarios, at different times of the year (we might have seasonal behaviours) and, then, find an average value.

Additionally, we found that the cooperation between different operational bases has increased with our approach, because we evaluate all the solutions found (including the ones from different operational bases where the event happened) and we select the one with less cost. In *human*, they choose the first one they find with less credit hours, usually from the same base where the event was triggered. This cooperation is also possible to be inferred from the costs by base. In Table 11 is possible to see that the direct operational costs of base C using *human* represents only 7,58% of the costs of all bases, whilst in *agent-no-quality* and *agent-quality* it represents 88,77% and 51,73%, respectively. The same is possible to be inferred from the other bases (although with different figures). This means that our MAS uses more resources from other bases than the base where the problem happened (base A).

7. Conclusion and Future Works

In this chapter we have introduced the Airline Operations Control Problem as well as the Airline Operations Control Centre (AOCC), including typical organizations and problems, the current disruption management (DM) process and a description of the main costs involved. We described our agent-based approach to this problem, including the reasons that make us adopt an agent and multi-agent system (MAS) paradigm; the MAS architecture with agents, roles and protocols as well as some agent characteristics like autonomy and social-awareness; the decision mechanisms, including the costs criteria and negotiation protocols used and examples of the problem solving algorithms. Using data from a real airline company, we tested our approach and discussed the results obtained by three different methods. We have shown that our approach is able to select solutions that contribute to a better passenger satisfaction and that produce shorter flight delays when compared with methods that only minimize direct operational costs.

We are working on several improvements. Some of them are already implemented. However, we did not perform, yet, enough tests to have meaningful results. These are our goals:

- Improve autonomy and learning characteristics of the *Monitor* agent, so that he is able to consider new events (or change existing ones) according to the experience he gets from monitoring the operation, without relying exclusively on the definition of events created by the human operator.
- Working on a protocol at the *Manager Agent* team level that allows a better coordination and improves the distributed problem solving characteristics of our approach. For example, including in each team, knowledge provided by other teams to improve the objective function of each specialist agent, with parameters of the other dimensions (aircraft, crew and passenger).
- Solving problems learning by example, applying Case-Based Reasoning (CBR).
- Increase robustness of future schedules by applying the knowledge gathered from learning by example.
- Study the behaviour and compare the results, of several problem solving algorithms, including the ones that implement heuristics to specific problems. The idea is to classify the algorithms according to their success rate in solving specific types of problems in this domain.

8. Acknowledgements

The first author is supported by FCT (Fundação para a Ciência e Tecnologia) under research grant SFRH/BD/44109/2008. The authors are grateful to TAP Portugal for allowing the use of real data from the airline company.

9. References

Abdelgahny, A., Ekollu, G., Narisimhan, R. & Abdelgahny, K. 2004. A Proactive Crew Recovery Decision Support Tool for Commercial Airlines during Irregular Operations, *Annals of Operations Research*, Vol.127, pp. 309-331.

Barnhart, C., Belobaba, P. & Odoni, A. (2003). Applications of Operations Research in the Air Transport Industry, *Transportation Science*, Vol. 37, pp. 368-391.

Bellifemine, F., Caire, G., Trucco, T. & Rimassa, G. (2004). JADE Programmer's Guide, *JADE 3.3,* TILab S.p.A.

Bratu, S. & Barnhart, C. (2006). Flight Operations Recovery: New Approaches Considering Passenger Recovery, *Journal of Scheduling*, Vol. 9, No. 3, pp.279-298.

Castro, A.J.M. (2007). Designing a Multi-Agent System for Monitoring and Operations Recovery for an Airline Operations Control Centre, *MsC Thesis*, pp.1-133, Porto, Portugal: University of Porto, Faculty of Engineering.

Castro, A.J.M. & Oliveira, E. (2007). Using Specialized Agents in a Distributed MAS to Solve Airline Operations Problems: a Case Study, *Proceedings of IAT 2007 (Intelligent Agent Technology Conference)*, pp. 473-476, Silicon Valley, California, USA 2-5 November 2007, IEEE Computer Society, ISBN: 0-7695-3027-3.

Castro, A. (2008). Centros de Controlo Operacional: Organização e Ferramentas, *Monograph for Post-graduation in Air Transport Operations*, ISEC - Instituto Superior de Educação e Ciências, Outubro 2008 (In Portuguese).

Castro, A. & Oliveira, E. (2008). The rationale behind the development of an airline operations control centre using Gaia-based methodology, *International Journal of Agent-Oriented Software Engineering*, Vol. 2, No. 3, pp. 350–377.

Castro, A.J.M & Oliveira, E. (2009). Using Quality Costs in a Multi-Agent System for Airline Operations Control, *11th International Conference on Enterprise Information Systems*, Artificial Intelligence and Decision Support Systems, pp. 19-24, Milan, Italy, May 6-10, 2009. ISBN: 978-989-8111-85-2

Clausen, J., Larsen, A. & Larsen, J. (2005). Disruption Management in the Airline Industry – Concepts, Models and Methods, *Technical Report, 2005-01*, Informatics and Mathematical Modeling, Technical University of Denmark, DTU.

Cowling, P.I., Ouelhadj, D., & Petrovic, S. (2003). A Multi-Agent Architecture for Dynamic Scheduling of Steel Hot Rolling, *Journal of Intelligent Manufacturing, Special Issue on Agent-Based Manufacturing Process Planning and Scheduling*, Vol. 15, No. 5, pp. 457-470.

Elamy, A. (2005). Perspectives in Agents-Based Technology, *AgentLink News 18*, August 2005, ISSN: 1465-3842.

Fipa. (2002). *FIPA Contract Net Interaction Protocol Specification*, http://www.fipa.org/specs/fipa00029/SC00029H.html, Accessed in 2009/06/21.

Grosche, T. (2009). Computational Intelligence in Integrated Airline scheduling, *Springer-Verlag Berlin Heidelberg*, 978-3-540-89886-3, Germany.

Jonker, G., Meyer, J.-J., & Dignum, F. (2005). Towards a Market Mechanism for Airport Traffic Control, *12th Portuguese Conference on Artificial Intelligence (EPIA 2005)*, Covilha, Portugal.

Kohl, N. & Karisch, S. (2004). Airline crew rostering. problem types, modeling, and optimization, *Annals of Operations Research*, Vol. 127, pp. 223-257.

Kohl, N., Larsen, A., Larsen, J., Ross, A. & Tiourline, S. (2004). Airline Disruption Management – Perspectives, Experiences and Outlook, *Technical Report, CRTR-0407*, Carmen Research.

Lettovsky, L. (1997). Airline Operations Recovery: An Optimization Approach, *Ph.D. dissertation*, Georgia Institute of Technology, Atlanta, USA.

Liu, T., Jeng, C. & Chang, Y. (2008). Disruption Management of an Inequality-Based Multi-Fleet Airline Schedule by a Multi-Objective Genetic Algorithm, *Transportation Planning and Technology*, Vol. 31, No. 6, pp. 613-639.

Mota, A. (2007). Multi-Agent System for an Airline Operations Control Centre, *MsC Thesis, Porto*, Portugal: University of Porto, Faculty of Engineering.

Ouelhadj, D. (2003). A Multi-Agent System for the Integrated Dynamic Scheduling of Steel Production, *Ph.D. dissertation*, The University of Nottingham, School of Computer Science and Information Technology, England, August 2003.

Rosenberger, J., Johnson, E. & Nemhauser, G. (2001). Rerouting aircraft for airline recovery, *Technical Report, TLI-LEC 01-04*, Georgia Institute of Technology.

Rosenberger, J., Schaefer, A., Goldsmans, D., Johnson, E., Kleywegt, A. & Nemhauser, G. (2002). A Stochastic Model of Airline Operations, *Transportation Science*, Vol. 36, No. 4, pp. 357-377.

Smith, R. (1980). The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver, *IEEE Transactions on Computers*, Vol. C, No. 29.

Stone, P. & Veloso, M. (2000). Multi-Agent Systems: a Survey from a Machine Learning Perspective, *Autonomous Robotics*, Vol. 8. No. 3.

Tumer, K., & Agogino, A. (2007). Distributed Agent-Based Air Traffic Flow Management, *6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007)*, Honolulu, Hawaii.

Wolfe, S. R., Jarvis, P. A., Enomoto, F. Y., & Sierhuis, M. (2007). Comparing Route Selection Strategies in Collaborative Traffic Flow Management., *Proceedings of IAT 2007 (Intelligent Agent Technology Conference)*, Silicon Valley, California, USA 2-5 November 2007, IEEE Computer Society, ISBN: 0-7695-3027-3.

Wooldridge, M., (2009). When is an Agent-Based Solution Appropriate? *An Introduction to Multiagent Systems*, West Sussex, England: John Wiley & Sons, Ltd., 2nd Edition (15 May 2009), pp. 183, ISBN: 978-0470519462.

Yang, M. (2007). Using Advanced Tabu Search Techniques to Solve Airline Disruption Management Problems, *Ph.D. dissertation*, The University of Texas at Austin, December 2007, USA.