# Multi-Agent Plan Diagnosis and Negotiated Repair*

# (Demo Paper)

Huib Aldewereld
Depart. of Computer Sciences
Universiteit Maastricht
Maastricht, The Netherlands
h.aldewereld@cs.unimaas.nl

Pieter Buzing
Faculty of EEMCS
Delft University of Technology
Delft, The Netherlands
p.c.buzing@tudelft.nl

Geert Jonker
Inst. of Computing Sciences
Universiteit Utrecht
Utrecht, The Netherlands
geertj@cs.uu.nl

## ABSTRACT

In the complex, dynamic domain of Air Traffic Control (ATC) many unexpected events can happen during the execution of a plan. Sometimes these disruptions make the plan infeasible and require a change of the original plan. Unexpected events may disrupt the execution of a plan leading to conflicts concerning the use of shared resources. By monitoring the possibly disrupted execution of a plan, air traffic controllers identify and repair conflicts before they occur, making the plan 'healthy' again. Model-based diagnosis helps to identify the causes of observed disruptions in the execution of a plan. This information enables the creation of better plan repairs. These repairs should *efficient*, but moreover they should be *fair*, i.e., one airline should not be the victim of conflicts caused by another. Due to the complexity of planning tasks, it is beneficial to provide a distributed solution such that the workload is spread instead of centralised. Moreover, since the choice between various possible solutions to a conflict in the plan execution directly influence different parties (with diverting interests), the decision about which solution to choose should not be made by a single (central) decision maker, but agreed upon by the different parties involved. The Multi-Agent Diagnosis and negotiated repair (MAD) demonstrator combines our previous research done on model-based diagnosis, planning and scheduling techniques, and methods for multi-agent negotiation to solve this problem in a distributed manner. The resulting tool is a system to support the control and adaptation of distributed plan execution in the domain of ATC.

## Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent Systems*

## Keywords

Plan Repair, Conflict Diagnosis, Negotiation, Multi-Agents

*Supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Ministry of Economic Affairs (the Netherlands). Project DIT5780: Distributed Model Based Diagnosis and Repair.

**Cite as:** Multi-Agent Plan Diagnosis and Negotiated Repair (Demo Paper), Huib Aldewereld, Pieter Buzing, Geert Jonker, et. al., *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16.,2008,Estoril,Portugal,pp.1659-1660.


## 1. INTRODUCTION

Plan development and plan execution in complex, dynamic environments are difficult tasks, which explains the tendency to use intelligent heuristics to support these tasks. Currently, the (initial) plan development in fields such as Air Traffic Control (ATC) is largely supported by software using such heuristics, but for plan execution, however, such support is not widely available, even though the execution of plans in these environments requires continuous control and adaptation. During the execution of plans, errors may occur that influence the outcome of the plan. As a result the observed execution of a plan may differ from the expected execution. Plan diagnosis, cf. [6, 3], aims at identifying the cause of these differences. Information gained by this may then be used to try to find adaptations to the plan (*repairs*) [2, 1]. The demonstrator presented in this paper intends to provide the missing support to plan execution, implementing previous research done on adaptation and control of plan execution by using model-based diagnosis techniques integrated in multi-agent environments [2, 4, 3, 1, 5, 6]. The use of distributed systems is a strong trend in ATC research these days. In order to reduce the workload on air traffic controllers, the planning problems need to be solved locally instead of centrally and by the parties involved instead of by single decision makers. Therefore the use of multi-agent systems is an obvious solution.

Although plan execution problems exist in many different domains, the example domain of the Multi-Agent Diagnosis and negotiated repair demonstrator tool (henceforth referred to as the MAD demonstrator) is the field of *tactical airport planning*. This phase of planning is concerned with the sequencing of arriving and departing aircraft and their scheduling on the gates. In this domain the environment has an important influence on the intended execution of a plan. Unforeseen changes in the state of the environment, such as snow or heavy headwinds, may influence the execution of the plan. Due to the frequent occurrence of such *disruptions* to the plan execution, the domain contains some interesting problems, making the need for repairs particularly high. Moreover, due to multiple parties involved in the plan execution, each with their own interests (e.g., competing airlines), the domain is inherently distributed. Changes to the plan cannot be made without some form of consent from the different parties involved (e.g., resource managers and airlines). This means that some form of negotiations are necessary to select the best possible repair.

Most of the time the airlines can help each other. However, since we assume that agents (representing the different

airlines) are selfish (due to their competitive nature), some sort of mechanism is required to guarantee that providing help will be paid back. In essence, a means needs to be introduced to stimulate co-operation among the agents; to this end, the MAD demonstrator makes use of a specialised monetary system to facilitate reciprocal co-operation. In short this means that when, for instance, KLM flight K1232 is delayed and therefore not able to finish its procedures at the gate in time, the next scheduled flight (in this case, L241 from Lufthansa) could agree to wait with going on-gate instead of forcing K1232 to leave. In return, KLM pays the cost of this repair to Lufthansa. A crucial aspect of the plan repair domain is that resources may not be economically exploited. E.g., an airline may not obtain a slot only to sell it against a high price. For this reason, standard currency can not be used. We developed a novel currency mechanism in which the value of an individual credit depends on the reputation of the agents that have used it. This currency mechanism guarantees equitable reciprocal co-operation under relatively weak assumptions [4, 5].

## 2. THE MAD DEMONSTRATOR

The main component of the demonstrator is the plan representation (see Figure 1 for a representation of the MAD demonstrator and the execution flow). The plan consists of distributed reservations to the airports resources, which maintain the details of those reservations. The different resources are modelled as different agents of the system. The reservations represent a sort of 'contract' of an airline with a resource for the handling of one of the flights of that particular airline (each flight has three distinct reservations; one for landing, one for de-boarding/boarding, and one for take-off). At the program start, the number of resources and airline agents can be specified, after which a random problem instantiation with the specified constraints is created.

After initialisation, the MAD demonstrator works in different stages as shown in Figure 1. The main loop of the program consists of the generation of a random disruption to simulate unforeseen occurrences that take place during the execution of the plan. The disruptions are handled and the plan is updated, which is then checked for consistency. If a conflict is found (e.g., the reservations of two different flights on the same gate overlap), the conflict is handed to the repair and diagnosis engines, which provide the input for the negotiation between the agents.

The repair engine determines the best possible repairs (more than one repair candidate may be generated for a given conflict) based on the current state of the plan. Candidates are generated by moving and swapping reservations on different gates/runways.

The diagnosis engine examines the conflict to determine the cause of the disruption (useful for predicting other similar events, e.g., all flights from the same direction are affected similarly by weather changes during flight) and the agent responsible for the conflict (the responsible agent is penalised for the conflict by being forced to pay for the costs of the necessary repairs).

The negotiation part of the program contains the co-ordination between all parties involved in the selection of the repair candidate that is most desired. Upon initialisation (i.e., input is received from the negotiation and diagnosis engines), the negotiations between the agents are initiated and, based on their preferences, the agent responsible for the conflict, and fairness, the repair solution that suits the agents best is chosen. Note that the fairness should only be measured over a period of time, after multiple repairs. An airline delaying one of its flights to help solve a conflict now could be helped by another airline at a later time. Fair repair is achieved when for every participant the amount of given help is equal to the amount of received help. The chosen candidate is used to repair the plan and the main cycle can start over.

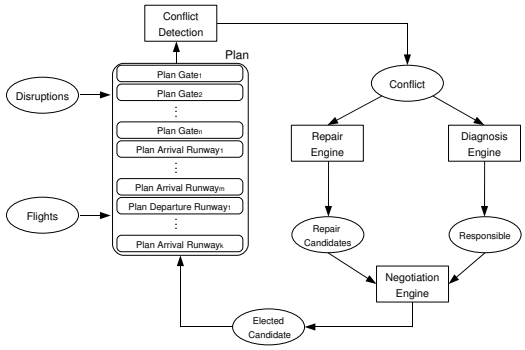


**Figure 1: Demonstrator program structure.**

## 3. ADDITIONAL AUTHORS

Additional authors: Femke de Jonge (Universiteit Maastricht, e-mail: f.dejonge@unimaas.nl), Frank Dignum (Universiteit Utrecht, email: `dignum@cs.uu.nl`), John-Jules Ch. Meyer (Universiteit Utrecht, email: `jj@cs.uu.nl`), Nico Roos (Universiteit Maastricht, email: `roos@unimaas.nl`), and Cees Witteveen (Delft University of Technology, email: `c.witteveen@tudelft.nl`).

## 4. REFERENCES

[1] P. Buzing, A. ter Mors, J. Valk, and C. Witteveen. Coordinating self-interested planning agents. *Journal of Autonomous Agents and Multi-Agent Systems*, 12(2):199–218, 2006.

[2] P. C. Buzing and C. Witteveen. Temporal plans and resource management. In A. Tuson, editor, *Proc. of the 24th Annual Workshop of the UK Planning and Scheduling Special Interest Group (PlanSIG 2005)*, pages 115–124, 2005.

[3] F. de Jonge, N. Roos, and H. Aldewereld. Using DES for temporal diagnosis of multi-agent plan execution. In *Multiagent Systems Technologies. 5th German Conference, MATES 2007*, LNAI 4687, 2007.

[4] G. Jonker, H. Hesselink, J.-J. Ch.. Meyer, and F. Dignum. Preventing selfish behaviour in distributed tactical airport planning. In *Proc. of the 7th USA/Europe R&D Seminar on Air Traffic Management (ATM'07)*, 2007.

[5] G. Jonker, J.-J. Ch.. Meyer, and F. Dignum. Achieving cooperation among selfish agents in air traffic management domain using signed money. In *Proc. of the 6th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2007)*, 2007.

[6] N. Roos and C. Witteveen. Models and methods for plan diagnosis. *Journal of Autonomous Agents and Multi-Agent Systems (JAAMAS)*, to appear, 2007.