Multi-Agent Technology for Air Traffic Control and Incident Management in Airport Airspace

Vladimir Gorodetsky
SPIIRAS
39, 14-th Liniya, 199178
St. Petersburg, Russia
7-812-3233570l
gor@mail.iias.spb.su

Oleg Karsaev
SPIIRAS
39, 14-th Liniya, 199178
St. Petersburg, Russia
7-812-3233570l
ok@mail.iias.spb.su

Vladimir Samoylov
SPIIRAS
39, 14-th Liniya, 199178
St. Petersburg, Russia
7-812-3233570l
samovl@iias.spb.su

Victor Skormin
CAIT
Binghamton University
Binghamton, NY 13902, USA
1-607-777-4471
vskormin@binghamton.edu

ABSTRACT

The paper presents a model, multi-agent architecture, implementation approach and software prototype of a multi-agent system for autonomous air traffic control within airport airspace capable of automatic detection of potential violations of safety policies by individual aircraft and consequent incident management. It features a model facilitating practical implementation of the concepts of openness and agent-based autonomy of air traffic control, social rules, distributed safety policy for conflict resolution, as well as predictive analysis and P2P interaction–based coordination of aircrafts' motion. The main results are validated by simulation.

Categories and Subject Descriptors

H.4.2 [**Decision Support**]: Distributed autonomous control system – *distributed multi-agent system, autonomous real-time control, peer-to-peer behavior coordination.*

General Terms

Algorithms, Management, Design, Experimentation, Security.

Keywords

Multi-agent system, autonomous air traffic control, P2P agent interaction, safety policy, incident detection and deconfliction.

1. INTRODUCTION

Due to ever increasing intensity of air traffic and increasingly rigid safety requirements, development of novel principles of Air Traffic Control (ATC) currently became a well recognized problem. Indeed, Air Traffic Control Operators (ATCO) are currently overloaded with their responsibilities and perform at the limit of their capacity. That is why the expected increase of air traffic intensity will inevitably exceed the capacity of existing ATC systems. An additional factor making the control problem highly critical is the increased frequency of abnormal situations, such as aircraft hijacking. In such situations, due to their highly dynamic and unpredictable nature, ATCO may completely fail to monitor and control the situation.

It is well recognized that satisfactory resolution of the described situation hinges upon providing individual aircraft as much control autonomy as possible and delegating them end-to-end routing and collision avoidance from the very take-off and to landing. Consequently, the free flight concept [1] for aircraft routing during cruising was formulated in the professional community ([8], [9]). This concept implies that every aircraft is provided some routing flexibility and the collision avoidance task is delegated to the autonomous pilot-assisting software based on distributed safety policy. Unfortunately, little attention is paid to the development of new principles of ATC within the airport airspace (AAS), where air traffic density is much higher while control processes are highly dynamic and the physical space is very limited.

Recent achievements in Multi-Agent System (MAS) thory provide a convenient framework for modeling and a technology for software implementation of autonomous ATC system within AAS. Indeed, agent-based modeling of collective behavior of distributed autonomous entities constrained by social rules and supported by distributed policy for conflict resolution, is the focus of many recent MAS research [6]. It provides adequate framework for autonomous ATC systems in question. Additionally, recent results in Peer-to-Peer (P2P) agent systems, in particular, development of reference model of P2P agent platform [7] and its subsequent software implementation ([2], [4]) provide unique architecture and technology for development of *open* systems with highly transient population of autonomous entities of MAS. It is important to note that the last property is intrinsic for ATC tasks.

The paper presents a conceptual model, multi-agent architecture, specification technology, and software prototype implementing ATC system within AAS. Together, these technologies implement the principles of openness and autonomy based on social rules, distributed safety policy for conflict resolution, predictive analysis and P2P interaction–based coordination of aircraft motions. Section 2 outlines basic domain knowledge and separation standards intended to assure the safety of aircraft motion. Section 3 describes typical behavior patterns of "normal" and hijacked aircraft and offers an organization concept of an ATC focused on agent-based autonomous path planning and P2P conflict resolution strategy. Section 4 outlines the developed distributed conflict resolution policy. Section 5 describes the developed architecture of a multi-agent ATC. Section 6 illustrates graphical user interface of the developed software prototype implementing basic ideas of the paper. Section 7 provides the conclusion describing the paper contribution and future work.

2. AIR TRAFFIC CONTROL DOMAIN KNOWLEDGE

2.1. Airport Airspace Topology

The high level notion of the *airspace topology* is intended to specify admissible trajectories of aircrafts sharing the airport airspace. It is worth noting that airspace topology does not address real-time air traffic configuration that concerns positions, speeds and courses of the set of aircrafts operating within AAS. Fig. 1 and 2 exemplify airspace topology (in horizontal and vertical projections respectively) in the New York City area uniting three airports, JFK, LaGuardia and Republic.

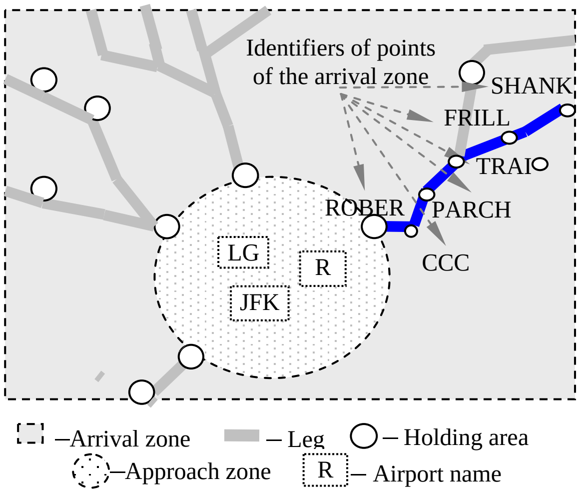

Figure 1. Airspace topology within New York City area (Horizontal projection), and arrival and approach

Airport airspace encompasses two zones: (i) arrival zone and (ii) approach zone. Arrival zone comprises *Arrival schemes*. E.g., Fig. 1, shows nine arrival schemes. Every arrival scheme begins with the entry point and is specified as a sequence of legs [3] ending with the *holding area*.

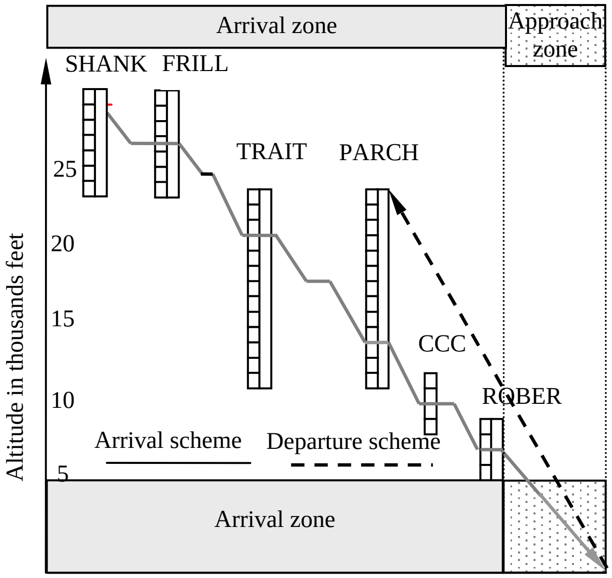

Figure 2. Airspace topology within New York City area (Vertical projection) and arrival and approach zones)

Approach zone comprises *approach schemes*. These schemes are not depicted in Fig. 1 due to too small scale of the figure. Each approach scheme begins at the approach zone entry point, consists of sequence of legs and ends at an airport runway.

Movement schemes within each approach zone can be classified in two categories (with some vagueness), (i) *standard* approach schemes and (ii) *missed* approach schemes, where the latter occurs in exceptional situations (technical problems, hijacking, etc). As a rule, a missed approach results in the necessity to use a holding area. *Transition schemes* bind the destination points of the arrival schemes and entry points of approach ones. As a rule, each arrival scheme is bound with several approach schemes. Transition schemes are used for binding different arrival schemes.

Fig. 2 depicts movement schemes (arrival and departure) projected onto vertical plane. In the left part of the figure, along the vertical axis, the echelon scale (from 0 till 30,000 feet with quantization step of 1000 feet) is depicted. The vertical projection of landing path through the arrival and approach schemes passing through SHANK, FRILL, etc. points is given by solid line.

The specification of the airport airspace topology also determines admissible echelons, i.e. admissible altitude ranges for passing through exit points of the legs. For example, while passing through the SHANK point, aircraft are required to use the echelons in between 24, 000 – 30,000 feet. Some legs may be bound with holding areas. E.g., all legs of an arrival scheme shown in Fig. 2, excluding the CCC leg are bound with the holding zone.

Airport airspace topology specification also contains departure schemes. They begin at a runway and end at exit points of airport airspace. Since climbing rate of an aircraft typically exceeds its descending rate, the exit points are located (in horizontal plane), between outer boundaries of the approach and arrival zones.

2.2. Separation standards

Separation standards defined for various air traffic–related situations constitute the basis for air traffic safety. They must be observed at any time by all pairs of aircraft that autonomously follow the distributed rule-based safety policy (see subsection 4.3) thus assuring conflict-free air traffic. Let us outline the separation standards for pair-wise motion of aircrafts for various situation cases.

a. Horizontal movement of aircraft occupying different echelons. An attribute determining minimal admissible *vertical distance* between pair of aircrafts if they are flying strictly horizontally is further denoted by the symbol D_A (Fig. 3).

b. "Following" motion of aircrafts within the same echelon of altitude. The attributes determining separation standards for this

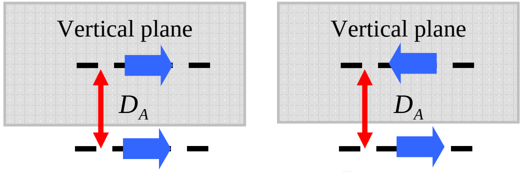

Figure. 3. Distance D_A

case are D_B –minimal longitudinal distance measured along the axis line of the legs and D_C –minimal distance between trajectories of aircrafts measured in directions orthogonal to the longitudinal axes of aircrafts (Fig. 4).

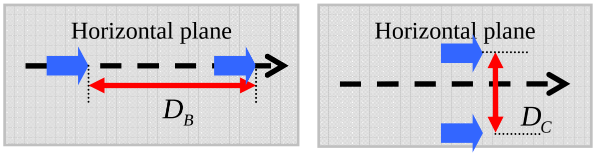

Figure 4. Distances D_B and D_C

c. Transversal motions of aircraft occupying the same altitude echelon It is said that the aircrafts are moving along the cross-cut trajectories if the angle value between the trajectories in horizontal plane is more than of 70 and less than of 110 degree (Fig. 5). The attribute determining the separation distance between such aircrafts is denoted as D_D. It represents the distance from an aircraft to the trajectory crossing point when one of the aircrafts has reached the crossing point.

d. Head motion of aircraft one of which is changing the altitude echelon. It is said that aircrafts have *head motion* if one of them is moving horizontally while the other one is climbing or descending with a vertical speed V_A if the angle between the course of horizontally flying aircrafts and projection of the course of the other aircraft onto horizontal plane is more than of 110 degrees. The distance D_E corresponds to horizontal distance between aircrafts when one of them has reached the trajectory crossing point. Two cases are to be distinguishes here: (1) the aircraft that earlier reached the crossing point is the one changing the echelon; (2) the aircraft that earlier reached the crossing point is the one flying horizontally. The difference between these cases is that D_E in the first case has to be greater than in the second case. Denote corresponding values of D_E as D_{E1}, and D_{E2} respectively (see Fig. 6 and 7 respectively). It is important to note that admissible values of D_{E1} and D_{E2} depend on the vertical speed V_A of the aircraft changing the echelon.

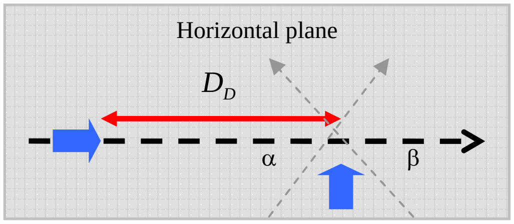

Figure. 5. Distance D_D

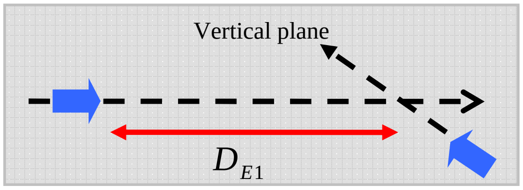

Figure 6. Distance D_{E1}

Generally, admissible values of distances D_A, D_B, D_C, D_D, D_{E1} and D_{E2} depend on different air traffic–related situation attributes. The following admissible values of these distances have been assumed:

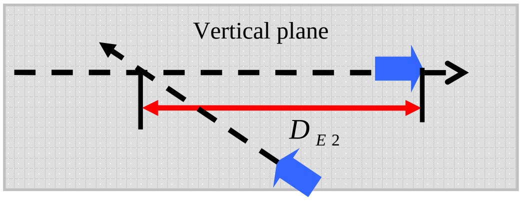

Figure 7. Distance D_{E2}

- D_A = 0.3 km;
- D_B = 10 km in the arrival zone and 5 km in approach one;
- D_C = 10 km in the arrival zone and 5 km in approach one;
- D_D = 20 km in arrival zone and 10 km in approach one;
- D_{E1} = 30 km if V_A <10 m/sec and 60 km otherwise;
- D_{E2} =15 km if V_A < 10 m/sec and 30 km otherwise.

The same attributes are used to represent separation standards between normal aircraft and the abnormal one (hijacked, technically-challenged, etc.). Moreover, the same policy providing safety of normal aircrafts in the presence of a hijacked one is used.

3. TYPICAL BEHAVIOR PATTERNS OF NORMAL AND HIJACKED AIRCRAFTS

Existing model of an aircraft movement intended for landing or take-off comprises the typical behavior patterns and negotiation procedures with corresponding ATCO as it is described below.

a. Landing: Entry into airport airspace

As the aircraft is approaching the arrival zone, its pilot informs the ATC operator of the corresponding sector of the arrival zone about the intended altitude and entry point of arrival. Depending on the situation, the pilot does or does not receive the approval of his intention and the assigned arrival movement scheme.

b. Landing: Behavior patterns within arrival zone

Within arrival zone, aircraft is moving along the axes of legs constituting the assigned arrival scheme. During the movement, the aircraft is passing through the arrival zone points, exit points of the previous legs and entry points of the subsequent ones.

Every arrival scheme point is assigned the admissible altitude echelons and therefore, while *passing through a scheme point*, the aircraft is permitted to pass through this point using one of the echelons assigned by the arrival zone ATCO. At some of these points, the holding areas exist. While approaching such a point, the aircraft either receives permission to enter the subsequent leg, or a request to enter the corresponding holding area where aircraft has to wait for ATCO permission to continue movement along the next leg of the assigned scheme.

While moving inside legs, the aircraft holds assigned altitude echelons and changes them during descending according to designation made by ATCO.

When one *aircraft has to pass another one* (e.g. due to the difference in admissible speeds for aircrafts of different classes) both have to deviate from the leg axis at predefined distances to the different sides. When passing is completed both aircrafts have to return to the leg axis and continue the movement. An important requirement is that both aircrafts have to return to the leg axis prior the current leg exit point. For this behavior pattern the aircrafts are permitted to simultaneously perform the passing and echelon change evolutions.

When an aircraft is *moving inside a holding area* it spends some time performing several circles within the holding zone depending on the situation. Within the zone, the aircraft stays at a single altitude echelon, but it is also may descend to a lower one.

Vectoring is an important behavior pattern that violates the leg boundaries. When vectoring is completed, the aircraft has to enter a leg of the same or other arrival scheme. Every vectoring requires building a new trajectory. Typical vectoring caused by weather conditions, technical problems, etc., implies turning at 30 degrees from the leg axis in horizontal plane, flying 20 km, and returning to the former course using the same or other echelon.

c. Landing: Movement inside approach zone
Entry into approach zone requires permission of responsible ATCO. While having no permission, the aircraft has to wait inside a holding area of the arrival zone. *Movement inside the approach zone* is carried out according to the designated approach scheme.

If, due to a reason, an aircraft that entered an approach zone *cannot perform landing*, it continues movement using a scheme of *missed approach* linked to its approach scheme while returning to one of the landing trajectories. In any case, to entry a new (or next) landing trajectory, the aircraft needs permission of ATCO of the respective approach zone. Otherwise, it has to wait within a holding zone specifically designated for missed approach case.

d. Take-off
Prior to take-off the aircraft pilot is assigned a movement scheme, informs ATCO about expected take-off time and *waits for the permission* for take-off. Depending on the current air traffic situation, the permission may be received with some delay. *Inside the approach zone* the aircraft uses the predefined departure scheme. *While moving inside arrival zone*, the taking-off aircraft uses the predefined departure scheme ending at the selected exit point of departure from the airport airspace.

e. Behavior patterns of hijacked aircraft
An important difference between the motion patterns of normal and hijacked aircraft is that the latter may ignore commands of ATCO and violate the rules of air traffic within AAS by not using predefined legs, waiting zones, entry and exit points, violating the predefined echelon altitudes, etc. In the paper, a limited set of typical behavior patterns are simulated. They include (a) motion of hijacked aircraft within the arrival zone, and (b) patterns using "broken line" trajectory. Nevertheless, even such geometrically simple patterns significantly complicate the air traffic control task.

4. ATC ORGANIZATIONAL PRINCIPLES

4.1. Existing Organizational Principle of ATC

Organizational principles of ATC determine how air traffic control functions are divided between ATCO and a pilot operating within AAS. Thus, two main roles of the ATC domain organization, "*pilot*" and "*air traffic operator*"[1], are defined. According to the currently existing ATC organizational principles, the main control operations performed by ATCO are:

a. Commands to aircraft approaching to the airport airspace:

A. Permission to entry into the AAS.

b. Commands to an aircraft operating within arrival zone:

B. Permission to transit into next leg.
C. Directives to transit into lower altitude echelon.
D. Coordinating evolutions of aircrafts in the passing situations.
E. Permission to entry the approach zone for the subsequent landing.
F. Changing the aircraft speed.
G. Performing vectoring.

c. Commands to an aircraft operating within approach zone:

H. Permission to a taking-off aircraft to take-off.

Unfortunately, such ATCO–centered organization of ATC is too inflexible and unable to support a significant increase of air traffic intensity and safety.

4.2. Advanced ATC Organizational Principle

The proposed ATC organization is focused on achieving openness and autonomy of ATC system supported by distributed safety policy for conflict resolution and P2P interaction–based autonomous coordination of aircrafts' movement. Like the existing ATC organization system, the main participants of the proposed one are *Aircraft pilots* responsible for autonomous solution of the tasks A, B, C, D, F and G within the approach zone and *Air traffic operator* whose responsibilities address control functions regarding tasks E in the arrival zone, and the tasks H within approach zone. Two important issues constitute the basis of control functions of the aircraft pilot role: 1) organization of information exchange and 2) safety policy determining the aircraft's autonomous behavior. Let us consider these issues.

Autonomous behavior of an aircraft in constrained environment, i.e. the airport airspace, assumes that each aircraft has to possess the information on current positions, courses and anticipated movement plans of other aircrafts operating within AAS, at least those that potentially may violate the separation standards. In the proposed ATC organization, this information is gathered by *Aircraft pilot* on the P2P basis. The list of potential peers may include only the aircraft that follow the same or overlapping arrival schemes, and this fact can be used for a significant decrease of information exchange. The latter is achieved via decomposition of the aircraft of the arrival zone in *independent groups.* The formation of groups and, hence, the *decomposition* may be achieved *on the sector basis.* Every sector is composed of the sequence of legs between two consequent entry points of the holding zones and the name of sector's exit point is assigned as

---

[1] This notice is important since "*role–based*" Gaia methodology [13] for multi-agent ATC system design is below used.

identifier of the sector. Thus, every sector is composed of sequence(s) of legs belonging to one or several arrival schemes that end in particular entry point of holding zone and start in an exit point of the previous holding zone. Therefore, the total count of the *sectors* is equal to the total count of holding areas. Note that the whole approach zone is considered as a sector.

The aircrafts operating within the same sector constitute a *group*(*Sector*). Since arrival scheme may include several sectors, each aircraft can belong to several groups depending on its behavior strategy and current air traffic situation. That is why aircraft groups may be overlapping. Each group *group*(*Sector*) is assigned the name of the corresponding sector. To further decrease the computational complexity and communication overhead, it is assumed below that, every aircraft, at any time instant *t*, takes into account potential conflicts within two groups, namely the sector of its current location id_1 and the next one it plans to transit to, id_2,. Thus, in the developed approach, to compute its own conflict–free behavior in the arrival zone, the aircraft relies upon information exchange with aircrafts of no more than two groups determined on P2P interaction basis.

Table 1. Information to exchange among aircrafts of a group

Aircraft's related data	
Aircraft	*<Aircraft's identifier>*
Class	*<Aircraft's class>*
Current sector	*<id of sector in which aircraft is currently located>* ,
Next sector	*<id of sector into which the aircraft has to overcome next>*
Update time	<*time of information update*>
Movement related data	
On Altitude	<*Current altitude echelon*>
To Altitude	<*Next selected altitude echelon*>
In holding area	<Holding area usage>
Information about transition into the next sector	
Transition point	<*Name of entry point*>
Transition time	<*Next sector transition time*>
Transition status	<*Intention /Decision*>
Approach	<Flight Scheme within the next zone> (For the aircraft of the approach zone)
Schedule delay	
S-Delay	<*Accumulated delay*>
F-Delay	<*Total accumulated delay of the flight*>

According to the proposed ATC organizational structure, within the arrival zone, the aircrafts have to *autonomously* solve the tasks A, B, C, D, F, G using P2P communication both for group discovery and conflict resolution if any. The information circulating among aircrafts of the same group is given in Tab. 1.

Let us note that every aircraft has to possess the information about all aircraft in the group(s) to which it belongs at the current and next movement step. Note that the above data are updated and sent to group peers when aircraft is making a decision from the set {A, B, C, D, F, G}. After receiving the updated information, the aircraft software has to assess the impact on safety of its own planned movement. If a conflict occurs, to avoid it, the aircraft starts P2P negotiation implementing the safety policy (see below).

4.3. Outline of Safety Policy and Deconfliction Algorithm

Safety policy is a set of rules determining priorities of the aircraft of the same group to be addressed by the current movement plan. It is implemented as a distributed deconfliction algorithm. To reduce the complexity of the deconfliction task, the algorithm is performed in two steps. At the first step, every aircraft computes its own pair-wise priorities regarding to all peers and at the next step the aircraft of the highest priority is automatically "granted" permission to use the sector's "resources" of the arrival zone (legs, holding zones) according to its current plan. Then these steps are iteratively repeated by the rest of the group aircrafts while taking into account the resources already reserved by aircrafts of the same group that have higher priority. Note that described safety policy concerns only landing aircrafts. Presently, the safety policy for taking-off aircrafts is still being developed.

The general approach to priority assignment to normal aircraft is as follows. First, relative orders for any pair of the airspace sectors are introduced. They are determined as "geometrical" precedence of the sectors of airport airspace starting from entry points and ending at runways. According to a general rule, an aircraft that entered through a particular entry point of its trajectory proceeds along a uniquely predefined sequence of the sectors of the arrival zone. Thus, it is said that sector X_i immediately precedes the sector X_j ($X_i < X_j$) if the former is the next sector in the aircraft landing scheme. This relation determines the order in which sectors and, therefore, groups are deconflicted. At the next step, the aircrafts of the same group are prioritized according to the set of safety policy rules given below.

Rule 1

If $Y_1 \in group(X_1)$ and $Y_2 \in group(X_2)$ and $X_1 < X_2$ then aircraft Y_1 is of higher priority than aircraft Y_2

Rule 2

If sector X_i is a sector of the arrival zone and both aircrafts, Y_1 and Y_2, belong to the sectors that immediately precede the sector X_1 then their priorities are determined either by *Rule 3* if no hijacked aircraft exists in AAS or *by Rule 4* otherwise.

Rule 3

Let two aircrafts, Y_1 and Y_2, have the exit times from the current sector $t_{Sector\ Exit}(Y_1, t_c)$ and $t_{Sector\ Exit}(Y_2, t_c)$ respectively scheduled. If $t_{SectorExit}(Y_1) < t_{Sector\ Exit}(Y_2)$ then priority of the aircraft Y_1 is higher than Y_2 one.

Let us note that, at a time instant, the aircrafts may belong to different sectors but plan to use the same sector as next one. In this case the *rule 3* is also applicable.

If two aircrafts occupy different sectors but potentially may conflict with hijacked aircraft then special functions $Conf(t, SectorX_1)$ and $Conf(t, SectorX_2)$ representing "the degree of conflict" are introduced for those sectors. Values of these functions are used as the arguments of the next rule,

Rule 4

If normal aircraft $Y_1 \in SectorX_1$ and $Y_2 \in SectorX_2$, and $Conf(t, SectorX_1) > Conf(t, SectorX_2)$ then priority of the aircraft Y_1 is higher than priority of the aircraft Y_2.

It is worth to note that in practice more rules and more aircraft attributes have to be used in the ordering process, e.g.,

- Class of aircraft (it determines the range of aircraft's speed depending on the flight altitude and therefore influences of aircrafts' preferences);
- Current echelon occupied by aircraft;
- Current deviation of the aircraft attributes from the scheduled ones;
- Fuel status, etc.

The mandatory requirement to any set of rules is that they have to provide conflict–free movement of the aircrafts. The selected rule sets can differ in resulting efficacy of ATC according to a criterion of multiple ones. A natural criteria, for instance, of air traffic control is maximal averaged capacity in terms of the total count of landing and departing aircrafts subject of safety requirements. But this task is out of the paper scope so far.

5. ARCHITECTURE OF MAS ATC: FORMAL SPECIFICATION

For design of Multi-agent ATC and Airspace Deconfliction System (ATC–AD MAS), extended Gaia methodology [10], was used. According to it, MAS architecture is specified in terms of a diagram representing its *meta–model* (Fig. 8) and "*liveness expressions*" specifying architecture in more detail. The meta–model specifies MAS architecture in terms of the roles to be assigned to agent classes, active software entities and the interaction of these components. Liveness expressions specify the role scenarios in different use cases. For the system in question, each role is mapped onto particular agent class. Therefore, the terms "*role*" and "*agent class*", in this application, may be denoted by the same identifier, however, the term "agent class" is below mainly used. In the developed system, three agent classes and one active software entity are introduced as described below:

- *Pilot assistant agent class* (*PA-agent class*); each agent of this agent class assists to the pilot of particular aircraft in autonomous ATC and in deconfliction situations;
- *Air traffic control operator agent class* (*ATCO-agent class*); each agent of this class assists the ATCO in decision making within the approach zone (sector);
- *Hijacked aircraft agent class* (*HA-agent class*); each agent of this class simulates and monitors of hijacked aircraft behavior.

Simulation server plays here the role of an active program entity. It is intended for simulation and visualization of real time situation in the airport airspace. It initiates real time events reflecting the results of operation of entities involved in air traffic and air traffic control. Simulation server also provides the interface to user; in particular, it supports the following functions:

- Visualization of the current air traffic situation within the airport airspace;
- Generation of the hijacked aircraft trajectory;
- Visualization of conflicts occurring between pairs of normal aircrafts and between normal aircrafts and hijacked ones.

According to Gaia methodology used in this development, formal specifications of agent classes (roles) are done in terms of liveness expressions. They specify the basic scenarios of agent classes' behavior in various tasks (use cases). In particular, specification of *PA-agent class* consists of 14 liveness expressions (*Initialization*, *Simulation cycle*, *Aircraft grouping*, *Arrival timetable monitoring*, etc.) presented in Fig. 8. *ATCO agent class* model consists of two liveness expressions, *Query* and *Permission.* Agent class simulating movement of the hijacked aircraft includes also two *liveness expressions* that are *Initialization* and *Trajectory forecasting*.

Specification of target system architecture in terms of liveness expressions is developed in detail but omitted in the paper due to lack of the space. These descriptions are done in context of the *Use cases* in which these liveness expressions are involved. In the developed model, seven such use cases (tasks of the target system), *U1–U7* (see Fig.8), are identified:

- (*U1*) Initialization of *PA-agent* class instances and initialization of the agent instance simulating the movement of the hijacked aircraft;
- (*U2*) Execution of simulation cycle;
- (*U3*) Grouping of aircrafts (instances of *PA-agent* class) that is used to reduce the overall information exchange traffic and to achieve the reduction of computational complexity of the airspace deconfliction algorithm;
- (*U4*) Autonomous planning of its own movement within the arrival zone by *PA-agent* class instance representing individual aircraft;
- (*U5*) Re–planning of own movement within the arrival zone by *PA-agent* class instances in order to avoid conflicts with normal and hijacked aircrafts;
- (*U6*) Normal aircraft's take-off control;
- (*U7*) Control of the arriving aircrafts during their movement within the approach zone (from the time when aircraft requests permission to entry the approach zone until the landing).

While performing these tasks, corresponding agents implement behavior specified in terms of liveness expressions.

Two types of liveness expression *Initiation* are used:

- Agent initiates running of a liveness expression in response to input messages, e.g. (Fig. 8), *PA- agent* class instance initiates running of the liveness expression "*Take-off*" when receives the message from *ATCO–agent* with "*Take-off permission*".
- Agent itself initiates a liveness expression as a result of its proactive emergent behavior, i.e. as a result of occurrence of some event(s) within the environment. An example is the initiation of *PA-agent* class instance liveness expression "*Grouping*" after transition of the normal aircraft from a sector to another one.

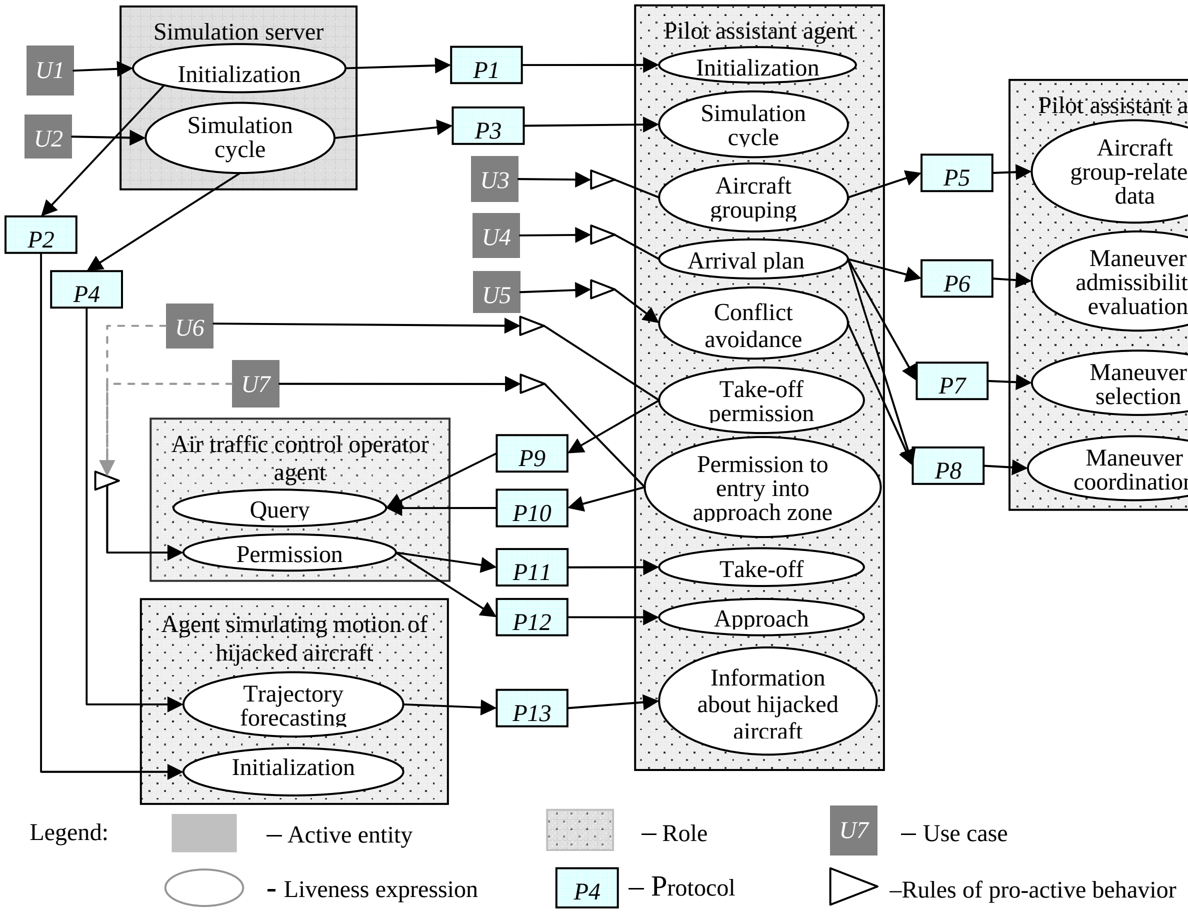

Figure 8. Meta-model of Multi-agent Airspace Deconfliction System

6. GRAPHICAL USER INTERFACE

Main window (Fig. 10) is used for visualization of the followings:

- Airport airspace topology (horizontal projection);
- Current positions of the aircrafts at the simulation time instant, and
- Detected conflicts.

If, at the current time instant, a conflict between a pair of the aircrafts is detected then this conflict is depicted by red line connecting the conflicting aircrafts when it exists. Interface also depicts some "statistics" of the detected conflicts. For this purpose, the sequence of the executed simulation cycles is depicted in the lower part of the window; the cycles exhibiting conflict(s) are depicted in red color whereas conflict–free cycles are depicted in green color.

The program component supporting graphical user interface operation checks separation standards while doing this independently of the ATC MAS and depicts the results; this functionality may also be used for agent behavior validation.

Graphical Interface Options assume image scaling, optional filtering of data visualized on image, and altitude-based filtering of data represented in horizontal projection mode.

Simulation control includes such functional capabilities as scaling of the simulation speed; simulation process interruption in the case if a conflict happens, simulation mode control (continuous or cycle-based simulation; detection of the time instant when hijacked aircraft appears. Additionally, *Simulation mode control* assumes the selection of a movement scheme and visualization of aircrafts' movements in vertical plane and manual input of hijacked aircraft movement data.

An example of visualization of an arrival scheme-related situation in vertical plane is given in Fig. 10. In this figure, the arrival scheme corresponding to sequential passing through points

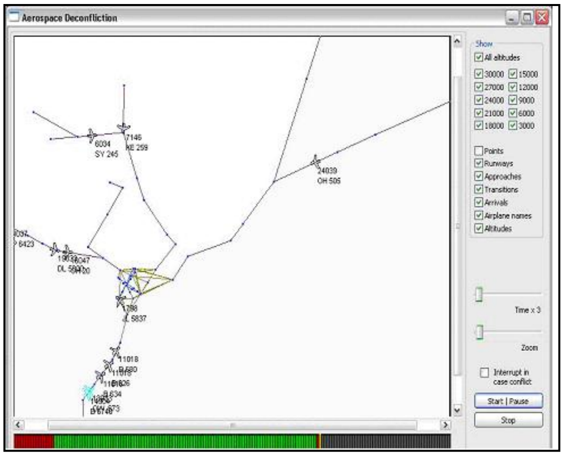

Figure 9. Main window

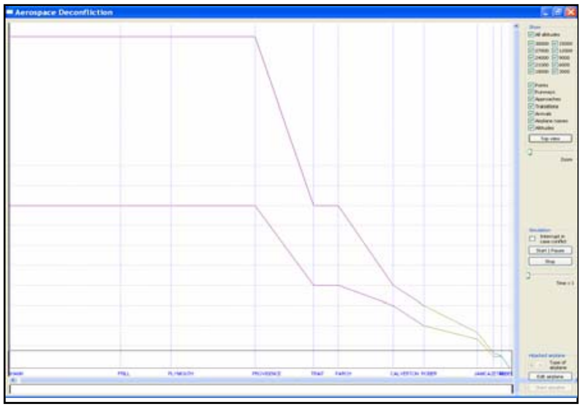

Figure 10. Visualization of movement scheme – related situation in vertical plane

HANK, FRILL, PLYMOUTH, PROVIDENCE, TRAIT, PARCH, CALVERTON, ROBER (see Fig. 9) of the JFK (New York City) airport is depicted. In this picture, the trajectories of the aircrafts situated in the "proximity" of the legs (at distances less than 5 km) constituting this scheme are depicted.

On the background of this window, horizontal lines depict the echelons ("every third" one is depicted in order to make the picture readable). In particular, 10 echelons presenting altitudes of 3000 feet, 6000 feet etc. till the altitude 30000 feet)are depicted in Fig. 10. The lines represent altitude boundaries of admissible echelons for corresponding legs assumed by the JFK AAS topology. Since flexible selection of the echelons is considered as a basic strategy of ATC, this graphical interface may be also utilized for graphical validation of the agents' behavior and the overall deconfliction algorithm.

Trajectory of the hijacked aircraft movement is specified prior to the simulation process. This specification is done in two steps. In the first step, its trajectory is specified in horizontal projection. This is done via selection of a sequence of the points of the trajectory. During the second step, the trajectory is specified in vertical projection. For this purpose special interface (not shown in the paper) is used. It defines the altitudes of the points selected at the first step of the procedure. The time instant corresponding to the appearance of the hijacked aircraft is defined manually during the simulation procedure. The hijacked aircraft is selected from the database presenting the aircraft speed as the function of the altitude and aircraft class.

Specification of particular air traffic situation is based on the use of real life timetable of arrival and departure. This is done using an editor of graphical user interface. In the developed version, a timetable of the JFK airport of New York City was utilized.

8. CONCLUSION

The paper offers a model, multi-agent architecture, formal specification methodology, and a software prototype implementing ATC system. The main paper contribution is that it suggests a feasible realization of ATC system featuring such important properties as openness and autonomy based on social rules, distributed safety policy for conflict resolution (collision avoidance), as well as on predictive analysis and P2P interaction–based autonomous coordination of aircrafts' motions. Design of the MAS in question is accomplished using Multi-Agent System Development Kit (MASDK 4.0), the recent version of the software tool that is being developed by the authors since 2000. This software tool implements extended version of Gaia methodology [10] in purely graphical style [5].

The further efforts are need to bring the model of air traffic control to the reality. Many aspects should be taken into consideration. The most important of them is the necessity to enrich the set of the aircrafts' behavior patterns while including vectoring, taking-off patterns, "missing approach" behavior, pattern intended to change the target airport, influence of weather conditions. It is also necessary to enrich the behavior patterns of hijacked aircrafts.

The set of rules determining behavior of aircrafts according to distributed security policy has also to be enriched and investigated from two view points, computational efficiency and quality of the air traffic control, e.g. from capacity view point. They should constitute the topics of further research. Nevertheless, the paper ideas will basically be preserved.

ACKNOWLEDGMENTS

This research was conducted under a contract from the Information Institute of the AFRL at Rome NY, USA. The authors are grateful to Mr. John Graniero of the AFRL for his support of this effort. The authors are also grateful to V.V. Kupin (St. Petersburg University of Civilian Aviation) for his thoughtful consultations on the existing air traffic control and challenges.

REFERENCES

[1] AEEC: Engineering Standards for Avionics and Cabin Systems, http://www.arinc.com/aeec
[2] http://space.iias.spb.su/ap/, 2007
[3] Draft 2 of Supplement 19 to ARINC Specification 424: Navigation System Data Base. Aeronautical Radio, INC, http://www.arinc.com/aeec
[4] Gorodetsky, V., Karsaev, O., Samoylov, V. ,Serebryakov, S. 2007. P2P Agent Platform: Implementation and Testing. The AAMAS Sixth International Workshop on Agents and P2p Computing (AP2PC 2007), Honolulu, 2007, pp. 21-32.
[5] Gorodetsky, V., Karsaev, O., Samoylov, V., Konushy, V., Mankov, E., Malyshev. 2005. A. Multi Agent System Development Kit. In R.Unland, M.Klusch, M.Calisti (Eds.) Software Agent-Based Applications, Platforms and Development Kits. Whitestein Pub.
[6] Kephart J. 2007. Multiagent Systems for Autonomic Computing. Invited talk at AAMAS'2007.
[7] FIPA P2P NA WG6: Functional Architecture Specification Draft 0.12. http://www.fipa.org/subgroups/P2PNA-WG-docs/P2PNA-Spec-Draft0.12.doc
[8] Tumer, K. and Agogino, A. 2007. Distributed Agent-Based Air Traffic Flow Management. Sixth Intl. Joint Conf. on Autonomous Agents and Multiagent Systems, 330-337.
[9] Tomlin, C., Pappas, G., and Sastry, S. 1997. Conflict resolution for air traffic management: A case study in multi-agent systems". IEEE Transactions on Automatic Control.
[10] Wooldridge, M., Jennings, N., and Kinny, D. 2000. The Gaia Methodology for Agent-Oriented Analysis and Design. International Journal of Autonomous Agents and Multi Agent Systems, 3(3), 285-312.