



RISE OF FLIGHT

The First Great Air War

“RISE OF FLIGHT” MISSION EDITOR USER MANUAL

ROF Editor user manual v1.0

TABLE OF CONTENTS

1. Introduction.....	5
2. Description of Editor’s Logic.....	6
<u>2.1. MCU.....</u>	<u>6</u>
2.1.1. MCU – Mission Control Unit.....	6
2.1.2. Object (Entity).....	6
2.1.3. Command.....	6
2.1.4. Trigger.....	6
2.1.5. Translator.....	6
<u>2.2. Links.....</u>	<u>6</u>
2.2.1. Object Link.....	6
2.2.2. Target Link.....	7
<u>2.3. Messages and their types.....</u>	<u>7</u>
3. Basic Rules of Mission Building.....	8
4. Interface and Editor Controls.....	9
<u>4.1. Instrument Panel of the Editor.....</u>	<u>9</u>
<u>4.2 Main Menu of the Editor.....</u>	<u>12</u>
4.2.1. “File” Menu.....	12
4.2.2. “View” Menu.....	13
4.2.3. “Search and Select” Menu.....	14
4.2.4. “Find missions entities” Menu.....	14
4.2.5. “Find location entities” Menu	16
4.2.6. “Draw” Menu.....	17
4.2.7. “Tools” Menu.....	18
4.2.8. “Location Database” Menu.....	20
<u>4.3. Editor Cameras and Navigation.....</u>	<u>20</u>
4.3.1. 3D Mode.....	20
4.3.2. 2D Mode.....	21

4.3.3. Cameras.....	21
4.4. Dropdown Menu on Right Mouse Click.....	22
4.5. Bottom Service Menu of the Editor.....	24
5. Mission.....	25
5.1. Basic Mission Properties and Settings.....	25
5.2. Mission Files, Mission Localization Files, and their Format.....	27
5.2.1. *.Mission.....	27
5.2.2. *.msnbin.....	29
5.2.3. [mission_name].eng.....	29
5.2.4. [mission_name].list.....	29
5.2.5. Working with Mission Localization Files.....	29
5.2.6. Localization ID.....	30
6. Working with Objects.....	31
6.1. Objects Selection.....	31
6.2. Moving and Positioning of Objects.....	32
6.3. Copy, Paste, Delete.....	33
6.4. Grouping of Objects.....	33
6.5. Uniting Moving Objects into Wing or Group.....	35
6.6. Saving to File, Importing and Inserting of Pre-saved Groups.....	35
6.6.1 Saving to File.....	35
6.6.2 Importing Objects into a Mission.....	36
7. Editor Objects.....	37
7.1. Objects Library Menu of the Editor.....	37
7. 2. Basic Properties for Editor's Objects.....	38
7.2.1 Planes/Vehicles/Artillery/Balloons.....	38
7.2.2. Trains.....	39
7.2.3. Firing Point.....	40
7.2.4. Buildings/Blocks/Airfields/Battlefield/Bridges.....	41
7.2.5. Helpers.....	43

7.2.6. Locations.....	46
<u>7.3. Advanced Properties for Editor's Objects.....</u>	<u>54</u>
7.3.1. Planes/Vehicles/Artillery/Aerostats.....	54
<u>7.4. Mission Command Units (MCUs) of the Editor.....</u>	<u>58</u>
7.4.1. Basic Properties for MCUs.....	58
<u>7.5. Advanced Properties for MCUs.....</u>	<u>59</u>
<u>7.6. MCU Command.....</u>	<u>59</u>
7.6.1. Command: Attack	59
7.6.2. Command: Attack Area	62
7.6.3. Command: Behavior.....	65
7.6.4. Command: Cover.....	68
7.6.5. Command: Damage.....	71
7.6.6. Command: Effect.....	73
7.6.7. Command: Force Complete.....	75
7.6.8. Command: Formation.....	77
7.6.9. Command: Land	80
7.6.10. Command: Move.....	82
7.6.11. Command: Play Animation.....	82
7.6.12. Command: Play Sound	84
7.6.13. Command: Take off.....	86
<u>7.7. MCU Translator.....</u>	<u>88</u>
7.7.1. Translator: Animation Operator.....	88
7.7.2. Translator: Camera Operator.....	91
7.7.3. Translator: Icon	95
7.7.4. Translator: Media	102
7.7.5. Translator: Mission Objective.....	106
7.7.6. Translator: Mission begin.....	108
7.7.7. Translator: Mission End.....	110
7.7.8. Translator: Subtitle.....	111

<u>7.8. MCU Trigger</u>	115
7.8.1. Trigger: Activate.....	115
7.8.2. Trigger: Camera point.....	117
7.8.3. Trigger: Check zone.....	119
7.8.4. Trigger: Counter.....	124
7.8.5. Trigger: Deactivate.....	126
7.8.6. Trigger: Proximity.....	128
7.8.7. Trigger: Spawner.....	132
7.8.8. Trigger: Timer.....	134
7.8.9. Random Timer Scheme.....	135
7.8.10. Trigger: Waypoint.....	137
8. Building Cooperative Mission	141
9. Frequently Used Mission Blocks, Examples and Building Tips	142
<u>9.1. Taking Off, Flying through Waypoints, Landing</u>	142
<u>9.2. Formation and Changing Formation in Flight</u>	143
<u>9.3. Attack Command En-route</u>	144
<u>9.4. Airstart</u>	144
<u>9.5. Vehicles Column on a Road</u>	145
<u>9.6. Train Movement</u>	146
<u>9.7. Building Animated Scene as Exemplified by Helper (Object in the Air)</u>	147
9.7.1. Animated Object is Seen Instantly After Mission Start.....	147
9.7.2. Animated Object is Seen Not Instantly After Mission Start	149
<u>9.8. Building Animated Scene as Exemplified by the Scene from Training Campaign</u>	149
<u>9.9. Screen Picture during Mission Start</u>	151
<u>9.10. Basic Mission – All Towns, Airfields and Bridges are on the Map</u>	152
<u>9.11. Files used in Media and Animation Operator</u>	152
9.11.1. Media Files in Game Compilation.....	152
9.11.2. *.chr Files for Animation Operator.....	152
10. General Information	154

1. Introduction.

The Rise of Flight Mission Editor contains the following functions:

- Universal trigger system of command execution and redistribution;
- Instrument to activate and deactivate mission objects by certain events, which greatly increases performance in missions containing large numbers of objects;
- Backwards compatibility of single and multiplayer missions (missions work by the same logic and are created with the use of universal instruments within the Editor);
- Ability to set priorities for command execution by AI, which in turn, determines non-linear mission progress. This feature allows AI to observe its present situation and react accordingly, which in turn creates the illusion of more natural (human-like) AI behavior, especially in multiplayer missions. For example, if there is no threat from the side of the player or other mission entities, the AI will continue executing his mission. But if a threat appears, AI actions can either be strictly determined by mission design or the AI can be left to act on its own, and after the threat is gone the AI will proceed with his mission.
- Visualization of gameplay with interface helpers (objects icons, waypoint icons, etc.). There are a variety of icon types, and the ability to preview mission briefings and debriefings. One can also add descriptive text as pop-up tips on the mission map;
- Built-in system of cameras, which allows creating scripted clips within the game engine. Basically, cameras are part of a mission and can act as part of the game process without interrupting it;
- Option to insert text, graphics and video into a mission as part of the game process. All of these options allow gameplay simplification, as well as create a specific atmosphere and custom plot line in a mission;
- Provides diverse means for editing missions such as:
 - Ability to edit object properties in groups such as moving, rotating, and copying (single objects, as well as the whole missions);
 - Ability create custom groups of unit formations and reuse them as templates (for example: a standard group of planes or cars, standard types of buildings);
 - Ability to import missions – this provides the functionality to combine the efforts of several designers, creating one complete mission. It also accelerates building complex missions (one makes a clip describing mission plot, the second is placing mission environment, the third creates the functional part of the mission and so on).

All the abovementioned properties of the Editor allow you to implement the widest range of artistic ideas in terms of mission and campaign design, as well as gameplay.

2. Description of Editor's logic.

To create dynamically changing missions, Full Mission Editor (FME) uses scripted language based on the interactions of objects in the simulator's environment, with certain logical elements (MCU) in the mission script – commands, triggers and translators.

2.1. MCU

2.1.1. MCU – Mission Control Unit – is a script that assigns various actions/properties to mission objects and the mission itself, depending on the type of MCU.

2.1.2. Object (Entity) – any game entity that can interact with the player – includes planes, buildings, and vehicles. Objects can be assigned with a model of behavior or a sequence of actions during a mission. For objects to interact in this way, or become editable, they must be brought to life with the help of the Create Linked Objects button, which can be found in the Basic Properties menu of the object (see section 7.2).

To understand how the script engine works, it's important to know that an object only becomes an object within a mission that has been loaded. In the Editor, we basically put in place a translator through which the script later assigns behavior models to the objects in the game engine.

2.1.3. Command – is an MCU that assigns specific a behavior model to an object (commands like “engage nearby enemies”) or a sequence of actions (commands like “go to waypoint”). In other words, the command “tells” an object what it has to do and in which order. In the MCU list, commands have a «Command:» prefix.

2.1.4. Trigger – is an MCU that activates a command or translator if certain conditions are met. Triggers can be either dependent on the object (for example: activate a command/event when object enters specific area) or independent (trigger activates command/event upon expiration of specific time passed after previous command/event). In the MCU list, commands have «Trigger:» prefix.

2.1.5. Translator – is the link between a script and an object. The translator is a script that is bound by the means of the Editor to an object and influences this object in a mission by the given rules.

2.2. Links

To write mission scripts correctly, one needs to understand linking logics of logical items with objects and between each other. There are 2 types of links – object link and target link.

2.2.1. Object Link (hotkey Shift+O) Object link (further OL – is marked by a line with green arrow) – is established between command, trigger, translator and object (from MCU to object i.e. you need to select the current MCU=> Shift+O=>object) that it influences. Object link (through a translator link to an object) determines which object has to execute the current MCU.

For example, OL from “go to waypoint” command to “airplane” object defines that in the mission, this plane has to proceed to the specified point. In case the link is established from a trigger, it can also mean the object is participating in conditions that turn this trigger on.

2.2.2. Target Link (hotkey Shift+T) Target link (further TL - is marked by a line with red arrow) is established between MCUs, determining the sequence of their execution.

Therefore, you have to select the MCU that the TL comes from and specify the MCU that will be next in the logical chain. Simply speaking, drag one MCU onto another MCU.

It's important to remember that for correct execution sequence of a script, the link has to be set from the previous MCU to the next one, not the opposite way. There can be numerous MCUs linked to one single MCU in different combinations.

Note: not all MCUs can provide TLs, see section 7.5 for detailed description of MCU properties.

2.3. Messages and their types.

Message – (is marked by a line with blue arrow) – is yet another type of link that sets sequence of MCU execution.

But unlike a TL the cause of executing this link is not the execution of “parent” MCU, but the change of certain condition of the object (through object’s translator – its essence) as a result of receiving a certain command or change in object’s properties (for example - its death).

In other words through a message an object tells the script “I’ve done that”. Via interface a message is established in a following way – select an object (of course with entity), right click on its icon, in context menu click Select Object Menu, then select the needed message (messages have “Message” postfix).

There are several types of Messages:

- Add On Land Message – message stating the fact that command to land was executed;
- Add On Take off Message – message stating the fact that command to take-off was executed;
- Add On Move Message – message stating the fact of movement;
- Add On Killed Message – message stating the fact of “death” of an object;
- Add On Damage Message – message about the actually received damage;
- Add On Spawned Message – message stating the fact of “birth” of an object;
- Add On Target Attacked Message – message stating the fact that command to attack the target was executed;
- Add On Area Attacked Message – message stating the fact that command to attack the area was executed;

When being set Message uses double link – source and target. In most cases it’s enough to use target (to skip source – press Esc while in setting Message mode – the cursor will change it’s type, then specify target). But sometimes you’ll need to use source also – in case you need to use a message by the fact of execution of a certain command.

For example if there are two commands for take-off for one plane you can specify as source – the first take-off command and target – the first waypoint of plane’s flight, therefore when the plane will have to make the second take-off you’ll have to specify as source already another command. In this example the object will give a message after executing a certain sequence of actions. You can manage other executed commands also in such a way.

3. Basic rules of mission building.

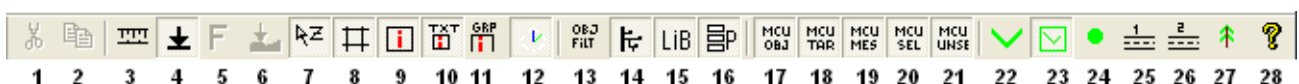
When building a mission it’s important to remember the following basic rules:

- you need at least one Mission Begin translator, but their number is unlimited. For example you can import stand-alone functional blocks of mission with the block itself having its own Mission Begin;

- each command has to be linked to an object that executes this command, i.e. the command has to “know” who it belongs to;
- command or trigger are executed (turn on) only in case they are started by TL or Message from another object;
- you should avoid giving commands to single object simultaneously and determine a sequence of commands using Timer triggers (1-2 seconds);
- deactivated object doesn’t receive commands, keeping those received before deactivation;
- when activating an object there needs to be a delay before giving him new commands (1-2 seconds) because activation process takes more then one step of modeling time;
- when using MCU Spawner remember that you can’t spawn a player (player controlled plane), you can’t spawn groups;
- don’t place more then 20 objects of the same type (everything that is in the lists: Bridges, Buildings, Airfields, Battlefield of the Library window) into one “screen frame” (approximately 5 km radius);
- it’s important to consider the priority of the command given to the object;
- don’t place more then 35 objects of different type into one “screen frame”;
- AI will attack only an object that belongs to opposite (enemy) coalition, it’s important to remember about it when building missions;
- it’s important to adhere the limitations for MCU of the Editor, which are listed in X section of this manual;

4. Interface and Editor Controls.

4.1. Instrument panel of the Editor.



- 1) – «cut» object – delete object from the map and place it into clipboard.
- 2) – copy object – place a copy of an object into clipboard.
- 3) – Measure Mode – ruler that allows measuring distance between points on the map, with possibility to set any broken line by points, with dynamic distance measuring and displaying the distance between points, where:

D – last section length,

L – total length of measured distance,

Hdg – heading (direction) of measuring.

Hotkey to use the ruler – Shift+M

To remove the last section and further all the previous sections – consecutive Backspace key. Exit from this mode – single Esc key.

4) – Toggle Camera – switches camera view in the Editor from isometric 3D to orthogonal 2D and vice versa. Hotkey for switching from 2D to 3D – F9.

5) – Focus on Selection – focusing camera on the selected object. Object is placed in the center of the monitor.

6) – Set on Ground – sets an object (or a selected group of objects) on the ground.

7) – Zoom to Cursor – if this key is on then camera zooms to the point where cursor is, if it's off – then to the center of the screen.

8) – Draw Grid – turns map grid on/off.

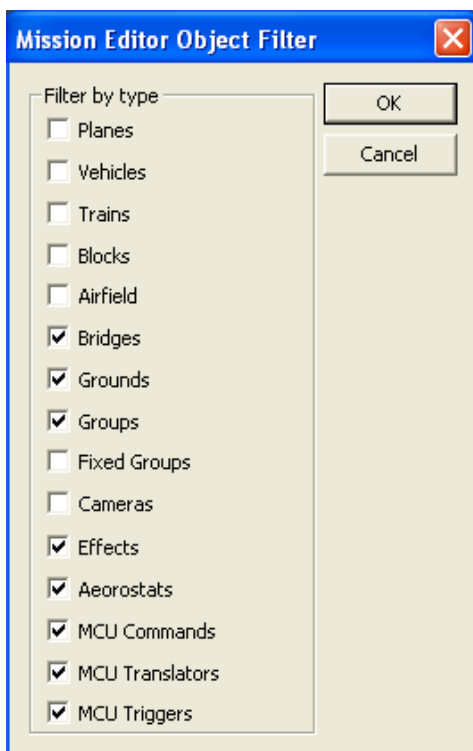
9) – Draw Icons – turns icons of the objects placed onto the map and Editor's MCUs on/off.

10) – Draw Icon Text – turns text with names and IDs of the objects placed on the map on/off.

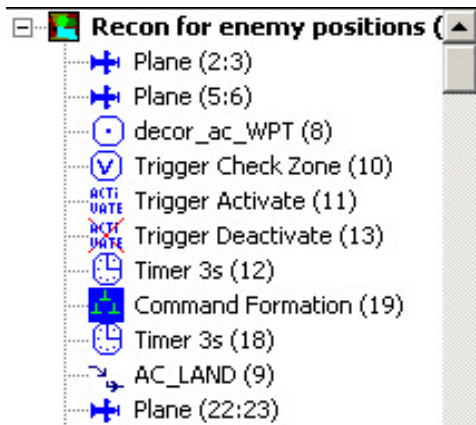
11) - Draw Icons in Group – turns icons of objects in groups on/off (off by default).

12) – turns compass on/off, where blue line – direction to north, moreover in 3D mode direction is also shown by digits – where the camera is pointed (heading angle), green line – direction of wind at surface level.

13) – Object Filter – opens Mission Editor Object Filter window that allows disabling icons of selected types of objects and MCUs on the map. This filter also hides icons of objects so that they don't disturb while editing if they are not needed at the moment (basically – object can't be selected). For example you can hide icons of bridges and blocks (or groups of ground buildings) since in the process of mission building they will occupy Editor's working space and simply disturb:

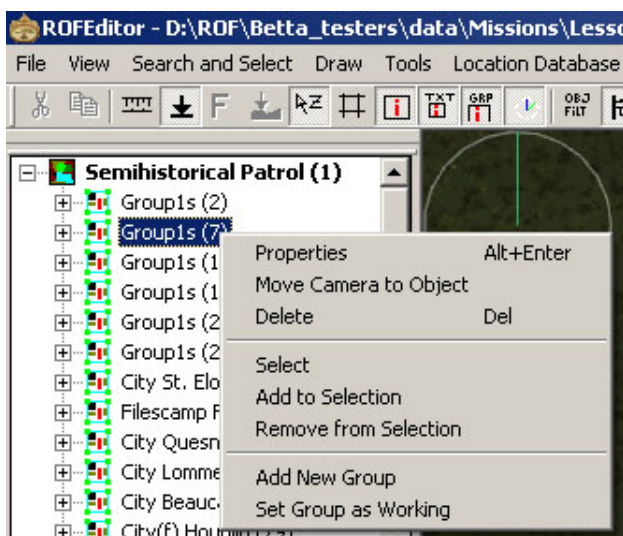


14) – Mission Tree – turns Mission Tree window on/off, which represents the list of all objects and MCUs installed on the map.



First number in brackets is object ID, the second is entity ID of an object. If it's MCU then it's MCU ID.

Also, besides displaying a list of objects this window allows quick transfer to the desired object by double-clicking on it in the list. If you select an object in this menu and right click it then a pop-up window will appear:



Properties – properties of the selected object (MCU);

Move Camera to Object – move camera specifically to the selected object (MCU);

Delete – deleting selected object (MCU);

Select – selecting selected object (MCU);

Add to Selection – adding this object (MCU) to the list of already selected objects (MCU);

Remove from Selection – removing this object (MCU) from the list of previously selected objects (MCU);

Add New Group – not available;

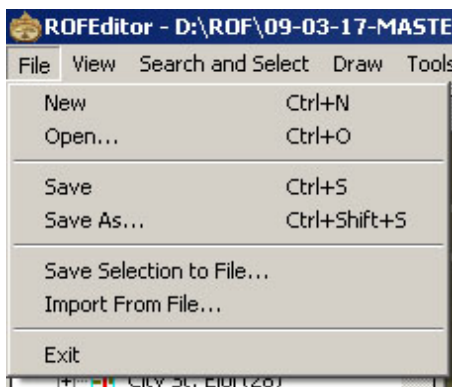
Set Group as Working – possibility to edit a group of objects (see Section 6 “Working with objects”).

15) – Objects Library – open/close Objects Library window.

- 16) – Property Dialog – open/close Basic Properties window.
- 17) – MCU OBJ – turns displaying of object links on/off.
- 18) – MCU TAR – turns displaying of target links on/off.
- 19) – MCU MES – turns displaying of “message” type links on/off.
- 20) – MCU SEL – turns displaying of links of the selected entities on/off.
- 21) – MCU UNSE – turns displaying of links of unselected entities on/off.
- 22) – basic autocheck of mission graph.
- 23) – turns autocheck at each mission save/load on/off.
- 24) – Unit Tests
- 25) – enables marking (road lighting) with red mask.
- 26) – not available.
- 27) – not available.
- 28) – Info about the Editor.

4.2 Main menu of the Editor

4.2.1. File menu:



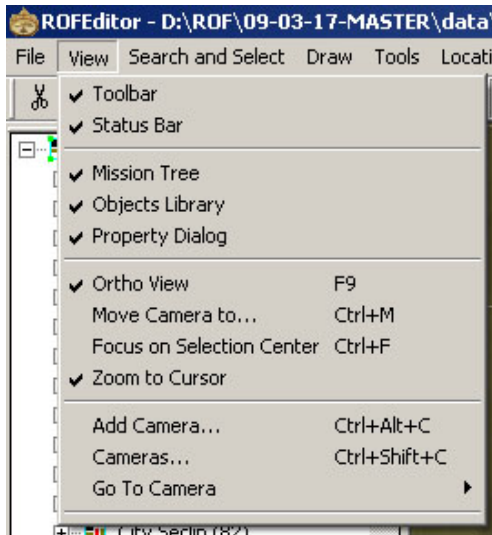
- New – create new mission file;
- Open... – open mission file;
- Save – save mission file;
- Save As... – save mission file under new name...;
- Save Selection to File... – save selected objects (entities) to a separate file for future pasting or importing into a mission.

Here objects are saved into a group with *.Group extension. For more on working with groups see Section 6.

By default *.Group files are saved to \Data\Missions directory;

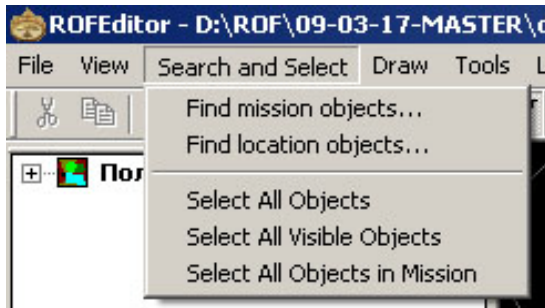
- Import From File... – import *.Group files into a mission;
- Exit – exit Mission Editor.

4.2.2. View menu:



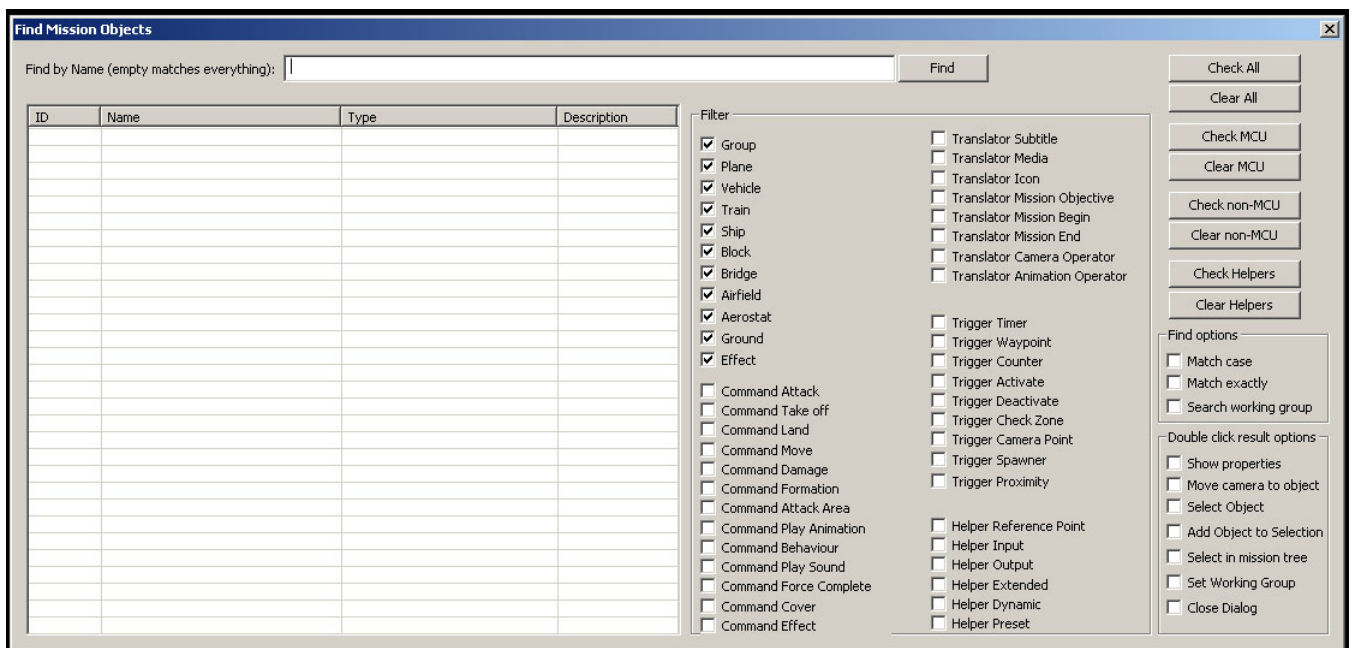
- Toolbar – show instrument panel of the Editor;
- Status Bar – show service information panel in Editor’s working space (see Section 4.5);
- Mission Tree – show mission “tree” window;
- Objects Library – show list of objects of the Editor (objects, commands, translators etc.);
- Property Dialog – show mission and objects properties window;
- Ortho View – 2D orthotropic view mode;
- Move Camera to... – move camera to the desired coordinates;
- Focus on Selection Center – centering of view (focusing) on the selected object or MCU;
- Zoom to Cursor – if this option is on then camera zooms to the point where the cursor is, if it’s off then to the center of the screen;
- Add Camera... – add a camera;
- Cameras... – edit camera properties in the list of pre-set cameras;
- Go to Camera – select and go to the camera that was set by Add Camera function (see Section 4.3).

4.2.3. Search and Select menu:



- Find missions objects... – search and select objects in mission;
- Find location objects... – search and select locations;
- Select All Objects – select all entities of the mission, all object of the game environment and also select all objects in workgroup;
- Select All Visible Objects – select all visible objects in the mission displayed by Mission Editor Object Filter (see above), and also select all objects in workgroup;
- Select All Objects in Mission – select all objects in mission.

4.2.4. Find Mission Objects menu:



- Find by Name (empty matches everything) field – search by entity's or object's name;
- Find button – find entity or object by the given name;

- ID, Name, Type, Description window - window displays found entities or objects, ID in the mission, name, type and description;

Filter field – adjust search and select by object type filters.

Check All button – select all checkboxes of the filter;

Clear All button – clear all checkboxes of the filter;

Check MCU button – select all checkboxes of MCU filter (commands, translators, triggers);

Clear MCU button – clear all checkboxes of MCU filter;

Check non-MCU button – select all checkboxes of objects filter;

Clear non-MCU button – clear all checkboxes of objects filter;

Check Helpers button – select all checkboxes of “helpers” filter;

Clear Helpers button – clear all checkboxes of “helpers” filter.

Find options field– search filter settings:

- Match case checkbox – case sensitive search;

- Match exactly checkbox – search by exact name match;

- Search working group checkbox – search only within workgroup;

Double click result options field – set actions that will happen after you double-click the found object in the list of found objects;

- Show properties checkbox – show properties window when you double-click the found object;

- Move camera to object checkbox – move camera to the found object when you double-click it;

- Select Object checkbox – select the found object when you double-click it;

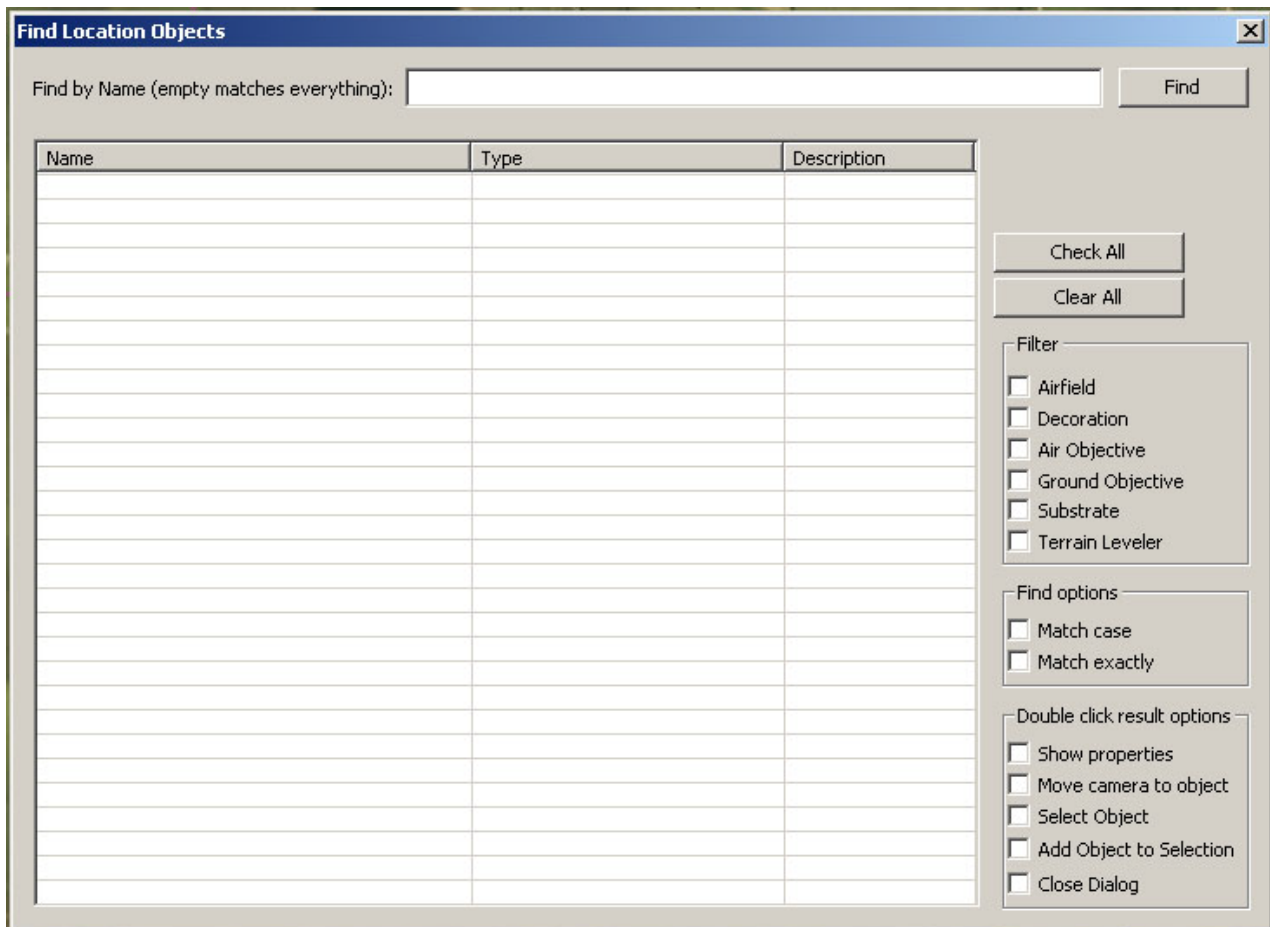
- Add Object to Selection checkbox – add the object to the list of selected objects when you double-click it;

- Select in mission tree checkbox – select the found object in mission “tree” window;

- Set Working Group checkbox – set editing mode for the found object when you double-click it (if it’s in a group);

- Close Dialog checkbox – close Find Mission Object window when you double-click the name of the found object (in the list).

4.2.5. Find location objects... menu



This mode allows finding and selecting a location on the map that will appear on the map after the location database is loaded - Default.ldf, Substrate.ldf and TerrainLeveler.ldf. Before starting a search you need to load these files to the editor.

Find by Name (empty matches everything) field – enter the full (partial) name of the object.

Find button – find location by the given name.

Name, Type, Description window – found locations window displaying its name, type and description correspondingly.

Check All button – select all checkboxes of the filter;

Clear All button – clear all checkboxes of the filter;

Filter field – locations search filter of the Editor.

Airfield checkbox – search airfield locations:

Decoration checkbox – search decoration locations of the game world;

Air Objective checkbox – search Air Objective type locations;

Ground Objective checkbox – search Ground Objective type locations;

Substrate checkbox – search Substrate type locations;

Terrain Leveler checkbox – search terrain leveler areas (locations);

Find options field – additional filter for searching and selecting locations in the Editor.

Match case checkbox – case sensitive search;

Match exactly checkbox – search by exact name match;

Double click result options field – set actions that will happen after you double-click in the ID, Name, Type, Description window.

Show properties checkbox – show Properties window for the found location;

Move camera to object checkbox – move camera to the found location;

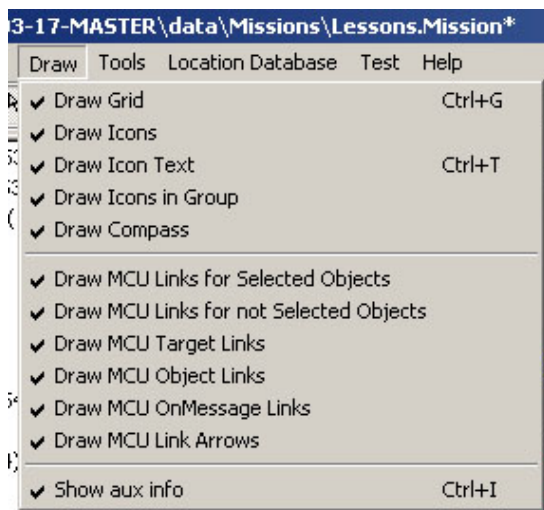
Select Object checkbox – select the found location;

Add Object to Selection checkbox – add location to the list of selected;

Close Dialog checkbox – close Find Location Object... window.

This search mode directly intersects with such Editor's objects as Helpers and Locations, to learn more see Sections 7.2.5 and 7.2.6.

4.2.6. Draw menu – menu specifying how to draw different objects and MCU in the Editor's working space.



Draw Grid – draw coordinate grid in 2D mode;

Draw Icons – draw object and MCU icons;

Draw Icon Text – show icon names for objects and MCUs;

Draw Icons in Group – draw group icons;

Draw Compass – draw game world compass;

Draw MCU Links for Selected Objects – draw links for the selected objects;

Draw MCU Links for not Selected Objects – draw links for unselected objects;

Draw MCU Target Links – draw Target links;

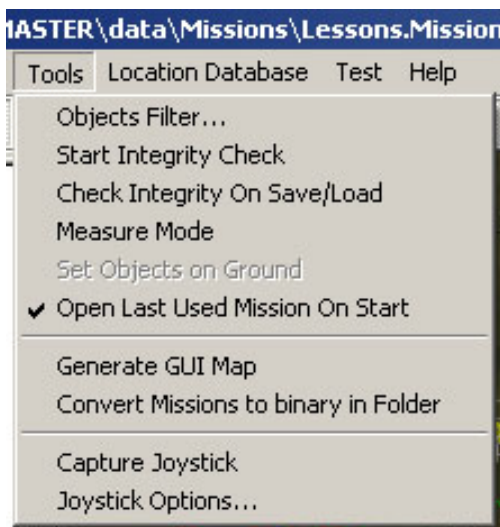
Draw MCU Object Links – draw Object links;

Draw MCU OnMessage Links – draw Message links;

Draw MCU Link Arrows – draw arrows of logical direction of links;

Show aux info – show information about framerate.

4.2.7. Tools menu – menu of Editor's tools.



Object Filter – open Mission Editor Object Filter window, that allows disabling drawing of selected object types on the map (see above).

Start Integrity Check – starts a program that checks assigned links between entities in the mission and displays results in the Editor's main menu;

Check Integrity On Save/Load – starts a program that checks assigned links between entities in the mission and displays results in the Editor's main menu at each mission save or load;

Measure Mode – ruler allows measuring distance between points on the map. You can specify any broken line by points. Distance is measured dynamically and displayed for each section. To remove the last section and further all the previous ones press Backspace consequently.

Exit this mode – single Esc key;

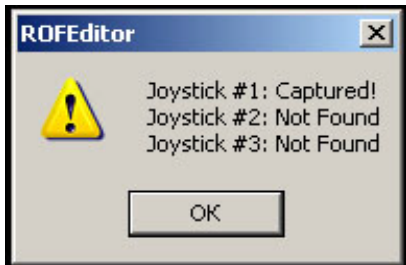
Open Last Used Mission On Start – at Editor's start up last open mission will be loaded;

Generate GUI Map – for internal, developers use;

Convert Missions to binary in Folder – convert mission files into binary format – with serious changes in Editor’s version binary format of missions is also often changed (see 5.2.2). This option allows automatic re-saving of old missions in the new format.

To do that choose folder with missions you’d like to re-save and the Editor will automatically re-save all the missions that are in this folder and its subfolder;

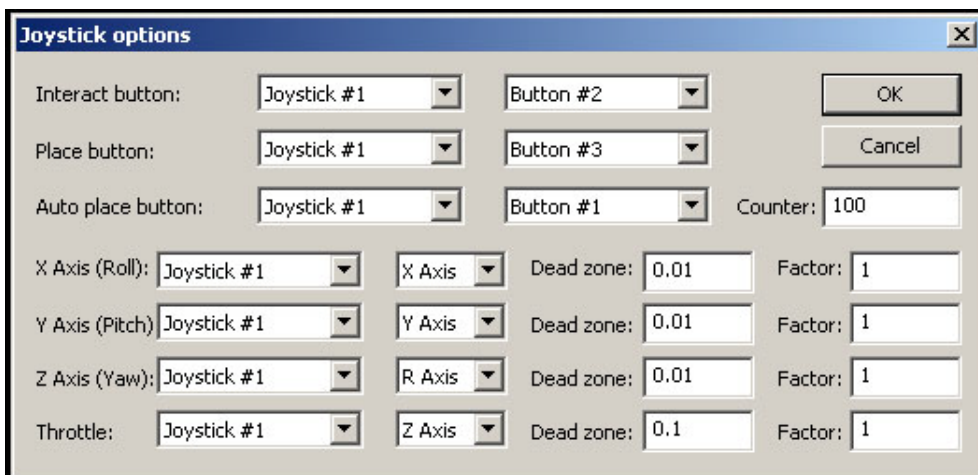
Capture Joystick – control camera movement in the Editor with joystick.



In this mode joystick is “intercepted” by Mission Editor; two modes – simple flight in game world and Camera Operator translator flight. In the second case if you set to the map a translator: Camera Operator – Rightclick – Selected Object Menu – Look from this Object, you’ll be able to fly over desired terrain in 3D mode, at this the Editor allows placing all intermediate camera waypoints (Trigger Came Point) in automatic or manual modes.

Also you can use this mode to place Waypoints for planes and vehicles. To do that you have to select an object, which you will be placing points for (or the last waypoint of this object).

Joystick Options – joystick control options menu for flying in simple and Camera Operator modes.



Interaction button – set joystick button to start moving;

Place button – button for manual placement of intermediate camera flight waypoints;

Auto place button - button for automatic placement of intermediate camera flight waypoints, automatically places waypoints during “flight”;

Counter – time in milliseconds between intermediate waypoints (only for Auto place button mode);

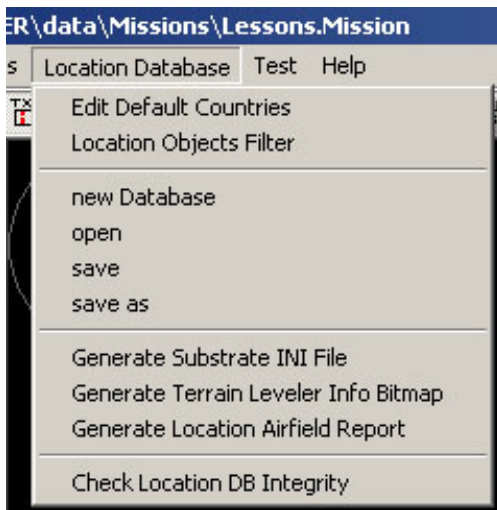
Factor – joystick sensitivity settings;

X, Y, Z Axis – joystick axes settings;

Throttle – speed of flight;

Dead zone – «dead» zones in joystick controls.

4.2.8. Location Database menu – menu for working with locations of mission generator.



Edit Default Countries – edit members of coalitions Opposite and Order (see 5.1.);

Locations Object Filter – locations filter;

New Database – create new database;

Open - open;

Save as – save as;

Generate Substrate INI file – generate “substrate” textures file;

Generate Terrain Leveler Info Bitmap – generate leveler areas location file for the tool leveling terrain under the objects (in .bmp format);

Generate Location Airfield Report – creates a report file (data\LocationDatabase\ Plane-Squad-Date_report.csv) by the information from airfield database (dates of squadrons stationed at the airfields and dates of their transfer);

Check Location DB Integrity – check and display of errors in locations database, the file is saved to data\LocationDatabase\IntegrityCheck.txt.

4.3. Editor’s camera and navigation

4.3.1. 3D mode:

- hotkey for switching between 2D and 3D modes – F9 or button in the instrument panel (Section 4.1).
- in isometric mode – left mouse click + mouse movement adjust camera position;

- right mouse click + mouse movement allow moving camera in horizontal space, is backed up by W, S, A, D – forward, back, left, right; SHIFT accelerates movement;
- mouse wheel moves camera by vertical axes. Camera heading can be determined by compass (circle in top left corner of the screen, green line shows direction to north, blue – wind direction).

4.3.2. 2D mode:

In 2D mode – similarly to isometric mode, with the exception of the ability to change camera orientation. Zooming is performed with mouse wheel.

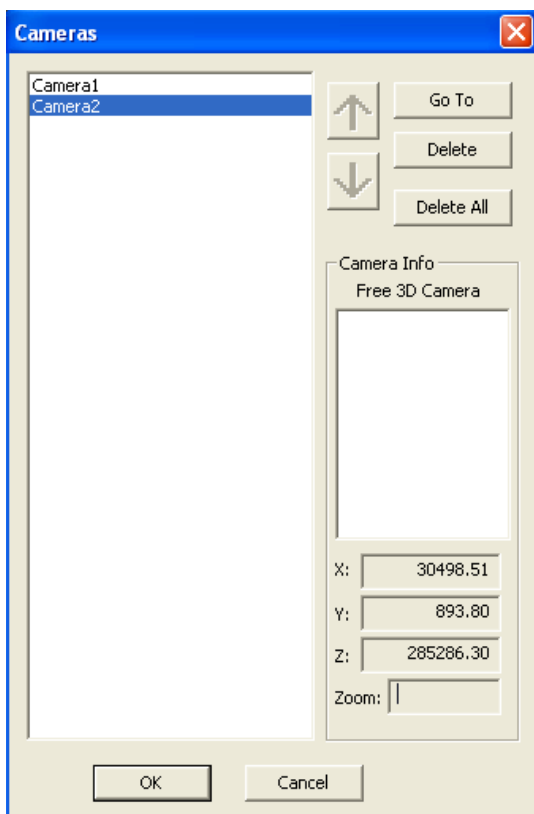
To jump between map areas quickly you can use pre-set cameras. To do this left mouse-click the desired map area and select Add Camera... (or Ctrl+Alt+C) in drop menu, type camera name in the appeared window and click OK. To jump to the desired camera select Go to Camera in the drop menu after you right click and then in the appeared dropdown menu select the desired camera.

Important: (!) During switching camera will work in the same mode it was set – i.e. if the camera was set in isometric mode it will switch the Editor to isometric mode each time will jump to it. The same is true for flat mode also (!).

4.3.3. Cameras

The list of pre-set cameras can be viewed and edited in Cameras window (Ctrl+Shift+c or right-click =>Cameras).

After you add a camera to the list of cameras of the Editor you can use Ctrl+1, Ctrl+2...0 hotkeys to jump to camera position. Order number (camera position in sequence called by hotkey) can be edited by moving cameras in the list with Up and Down arrows in Cameras menu.



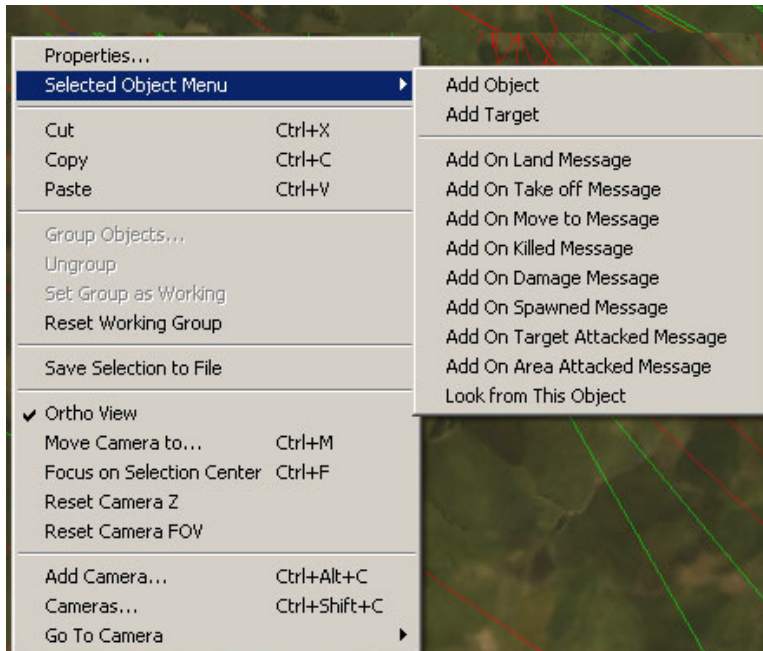
In this window:

Go To button allows jumping to camera selected in the list field in the left part of screen.

Delete and Delete All buttons delete selected camera and all cameras correspondingly.

Camera Info group displays camera type (Free 3D for cameras set in isometric mode, Ortho for cameras set in flat mode), camera notes (can be added/edited manually) and camera coordinates (X:, Y:, Z; fields). Zoom: field shows camera zoom to surface scale (only for Ortho cameras).

4.4. Dropdown menu by right mouse click.



If you right-click anywhere on the map dropdown menu will appear, where:

Properties... - opens window with list of properties to edit (see above);

Selected Object Menu – menu of actions over the selected object:

Add Object – set object link;

Add Target – set target link;

Add On Land Message – set message about the fact of landing type link;

Add On Take off Message - set message about the fact of takeoff type link;

Add On Move Message - set message about the fact of movement type link;

Add On Killed Message - set message about the fact of “death” type link;

Add On Damage Message - set message about actual damage type link;

Add On Spawned Message - set message about the fact of “spawn” type link;

Add On Target Attacked Message - set message about the fact of attacking target type link;

Add On Area Attacked Message - set message about the fact of attacking area type link;

Look from This Object – switch to 3D mode and view from the selected object.

Cut – cut.

Copy – copy.

Paste – paste.

Group Objects... – group selected objects in one group.

Ungroup – ungroup selected entities.

Set Group as Working – ability to edit group.

Reset Working Group – exit edit group mode.

Save Selection to File – save selected entities to a separate file.

Ortho View – flat mode.

Move Camera to... – move camera to the specified coordinates.



Where X and Z are map coordinates where camera will go to after you press OK.

Focus on Selection Center – focus on selected object.

Reset Camera Z – reset camera horizontal tilt into normal position.

Reset Camera FOV – reset camera viewing angle to default.

Add Camera – add camera.

Cameras... – edit camera properties in the list of pre-set cameras.

Go to Camera – select and go to camera set by Add Camera function.

4.5. Bottom service window of the Editor.

Mouse(m):	1855.35	80815.08	Grid(m):	5000	Camera (X Z Zoom):	47273.45	86757.64	2.758	
	1	2		3		4	5	6	7

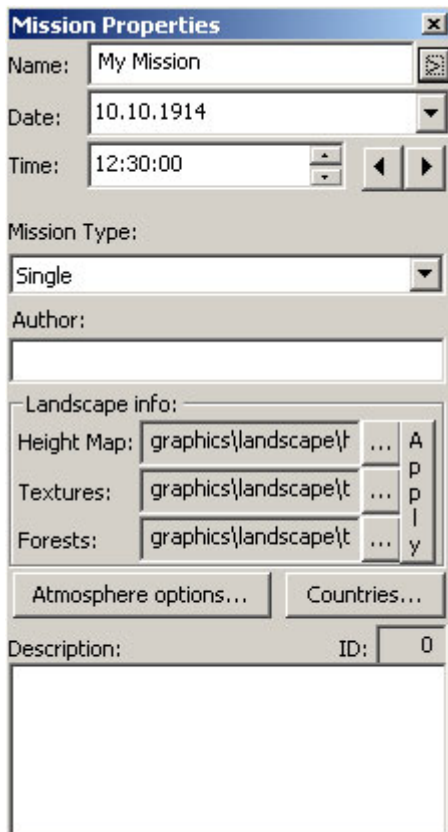
It is located in the lower part of screen of the Editor:

- 1) – mouse cursor coordinates by X axis;
- 2) – mouse cursor coordinates by Z axis;
- 3) – Grid(m) – size of grid cell at current zoom aspect;
- 4) - Camera (X Z Zoom): - coordinates and position of the camera from the first person:
 - 5 – camera coordinates by X;

- 6 – camera coordinates by Z;
- 7 – camera coordinates by Y (height), in 2D mode this window shows camera zoom aspect.

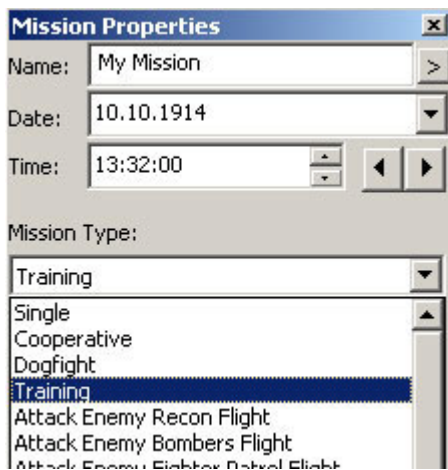
5. Mission.

5.1. Basic mission properties and settings.



When creating a mission you can see mission settings window in the lower left corner of the Editor:

- Name – mission name. This name will be displayed in mission list in game interface. If the name is blank then mission will be displayed by mission file name;
- Date – set date in the mission. You can set date in mission window or through the calendar, which will appear if you click date properties (to the right in date window);
- Time – set mission start time;
- Mission Type – set mission type;



When creating missions use only first four mission types (Single, Cooperative, Dogfight and Training), other mission types in the list are used by mission generator.

When creating a mission you should set mission type properly because if you for example set Cooperative type to a mission and place it into the folder for Single type missions (data\missions) then this mission won't be visible in the mission list in game interface.

- Author – field for specifying mission author;
- Landscape Info – menu for adding a map of another type;

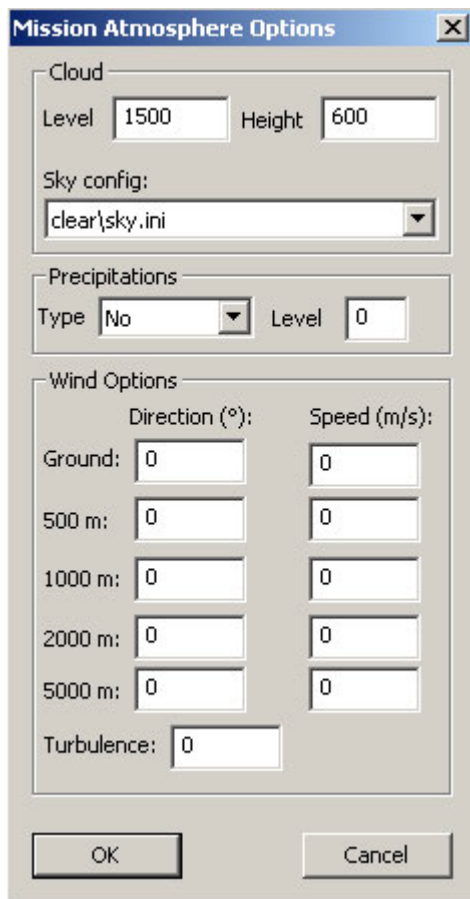
When adding another type map all files describing the new map must be placed into data\graphics\[terrain_name] folder. In our case the map is inside Landscape folder and is described by three files:

Height Map – (height.ini) file describes surface properties of a map, roads, water surfaces, forests and also a set of textures covering map surface.

Textures – (textures.ini) file describes a set of textures for components of a map, such as forests, water, surface masks, textures of town substrates etc.

Forests – (trees\summer.wds) file sets tree types depending on the season, in this case these are summer trees.

Atmosphere Options – weather settings in a mission;



The image shows a Windows-style dialog box titled "Mission Atmosphere Options". It contains three main sections: "Cloud", "Precipitations", and "Wind Options".

- Cloud section:**
 - Level: 1500
 - Height: 600
 - Sky config: A dropdown menu showing "clear\sky.ini".
- Precipitations section:**
 - Type: A dropdown menu showing "No".
 - Level: 0
- Wind Options section:**
 - Direction (°): A column of five input fields, all containing "0".
 - Speed (m/s): A column of five input fields, all containing "0".
 - Labels for the rows: Ground, 500 m, 1000 m, 2000 m, 5000 m.
 - Turbulence: An input field containing "0".

At the bottom of the dialog are "OK" and "Cancel" buttons.

Cloud – section with clouds and weather type settings:

- level – height of the lower edge of clouds above the ground;
- Height – thickness of cloud layer;

Sky config – choose cloud type, clear – cloudless, average – rear clouds, heavy – cloudy, precipitation – overcast, thunderstorm – overcast and fog.

Precipitations – downfall and its intensity settings section:

- Type – choose downfall type, rain/snow;
- Level – choose downfall intensity, from 0 to 10.

Wind Options – section with wind settings for the mission:

- Direction – direction of wind (where it blows to);
- Speed (m/s) – wind speed;
- Ground/500m/1000m/2000m/5000m – wind layers by altitudes, in columns to the right are wind direction and speed in layers correspondingly;
- Turbulence – turbulence force.

Countries button – coalition settings in the mission:



Using [>>>] and [<<<] buttons you can add the abovementioned countries into Order and Opposite coalitions.

Restore Default Countries – will reset coalition settings to default.

Description window – in this window you can type mission description, which will be displayed in game interface during mission selection in missions list and also as a text in mission description during mission loading.

5.2. Mission files, mission localization files, their format.

During mission building the Editor creates several files.

5.2.1. *.Mission – mission file in text format. Actually this is mission source; therefore always try to save it if you are going to use the mission in future.

Contents of text format of mission on the example of empty mission (*.Mission mission file can be edited in any text editor):

```
# Mission File Version = 1.0;
```

Options

```
{
    LCName = 0; // mission name localization line index
    LCDesc = 1; // mission description localization line index
    LCAuthor = 2; // mission author localization line index
    PlayerConfig = ""; //script of the plane available for the player in the mission
    Time = 13:32:0; //mission start time
    Date = 10.10.1914; //date in the mission
    HMap = "graphics\landscape\height.ini"; //file describing mission map;
    Textures = "graphics\landscape\textures.ini"; // file describing mission map;
```

```

Forests = "graphics\landscape\trees\summer.wds"; // file describing mission map;

MissionType = 3; // mission type, in this case 3=Training (0=Single, 1=Cooperative, 2=Dogfight)

CloudLevel = 1500; //altitude of clouds

CloudHeight = 600; //clouds thickness

PrecLevel = 0; //downfall intensity

PrecType = 0; //downfall type

CloudConfig = "clear\sky.ini"; //weather type

Turbulence = 0; //turbulence force

WindLayers //wind settings by layers (altitude : direction : speed)
{
    0 :    0 :    0;

    500 :    0 :    0;

    1000 :    0 :    0;

    2000 :    0 :    0;

    5000 :    0 :    0;
}

OppositeCountries //indexes of countries belonging to Opposite coalition
{
    501; //Germany

    502; //Austria-Hungary
}

OrderCountries // indexes of countries belonging to Order coalition
{
    101; //France

    102; //England

    103; //USA

    104; //Italy

    105; //Russia
}

```

}

end of file

5.2.2. *.msnbin – mission file in binary format. Functioning of a file of such type depends on mission format (meaning – if an updated version of the project is out – this file would be unworkable). If the Editor gives an error message when reading mission then, in case you have mission source left, delete *.msnbin file and resave the mission. The Editor will automatically create a new version of binary file.

To add to this, binary mission is loading considerably faster than the text one.

Mission will load as in case of both types of files present (at this *.msnbin will have a reading priority over *.Mission), so that if either this or that file type is present separately.

5.2.3. [mission_name].eng (or *.rus, *.ger, *.fra, *.spa) – localization files of the mission that store all the text data that are used in the mission. Therefore if you have created a mission in Russian language and want, for example, to create an English localization version then simply make a copy of this file and change its extension from *.rus to *.eng.

5.2.4. [mission_name].list – file necessary for cooperative mission to work correctly (see Section 8).

5.2.5. Working with Mission Localization Files.

When working with localization files pay extra attention!

Work with a mission (edit or re-save in the Editor) in that language (settings.exe – language), which it was created in. Otherwise if you'll open and re-save the mission in an Editor that is switched to another language there can be problems with localization lines IDs.

Localization file can be opened in simple Notepad, as it's in text format.

Localization files format has to be UCS-2 Little Endian.

5.2.6. Localization ID (index).

During mission creation all texts of the mission are saved in localization files, where IDs are assigned to blocks of text (in fact, inside the file it looks like a number with a colon – line index).

Therefore if you create several types of localization languages of a mission watch for IDs of all localization files to match and be absolutely equal to the ID of the original localization file. Also when re-saving a mission – check the ID, as they may change when adding new objects (though they shouldn't; but it's better to control it all the way to avoid problems).

Example of ([mission_name].rus) localization file:

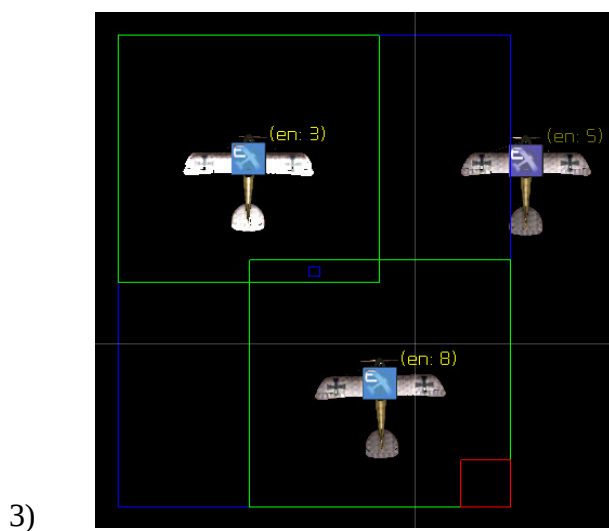
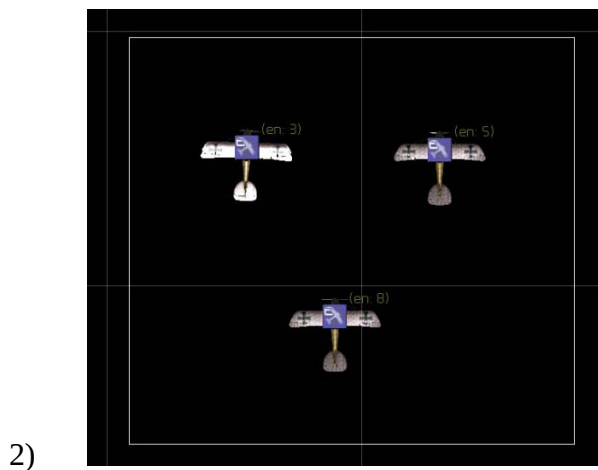
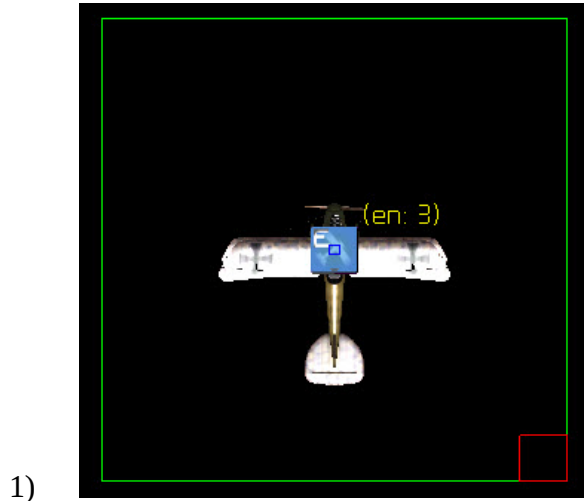
0:Attention! Incoming enemy fighters! Intercept them!
1:Excellent job! Enemy is destroyed!
2:Enemy fighters
3:
4:Airstart
5:Get ready for action – follow the waypoints!
6:
7:Primary objective failed

8: Our recon planes have to obtain very important information about enemy forces. Don't let enemy fighter near them!

6. Working with objects.

6.1. Selecting objects.

Select an object with left click, at this you 'll see a green frame around the object (1).



You can select several objects at once with a frame (2) or you can add objects to selecting frame by holding Ctrl key, at this each object will have its own green frame and the whole group of selected objects will be marked by a blue frame, which will have a small blue square marking the center of the group (3).

6.2. Moving and positioning of objects.

Moving and positioning of objects can be made as in the Basic Properties window (see Section 7.2), so that using mouse and keyboard.

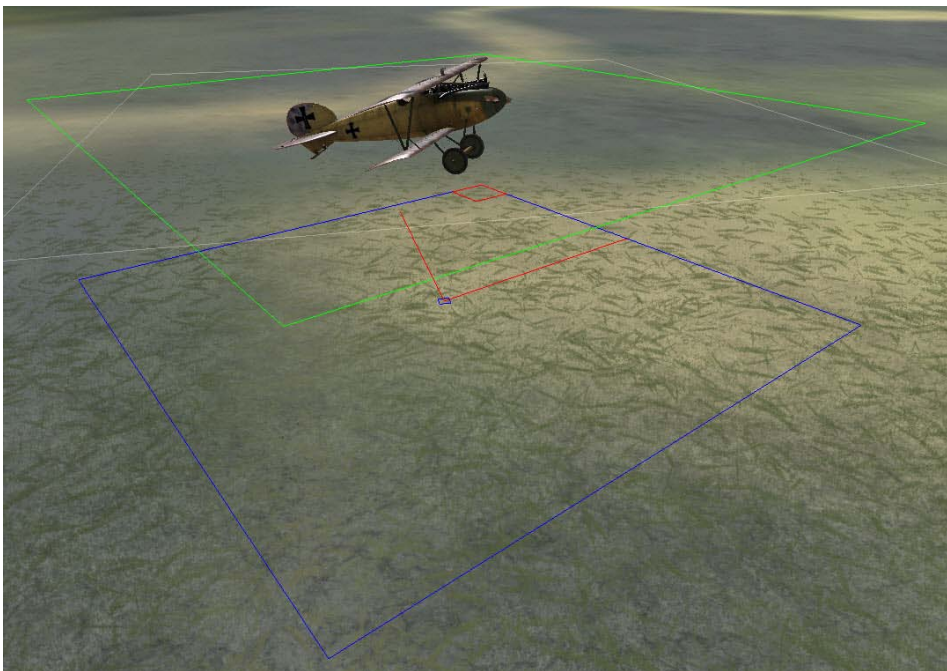
2D mode.

To move an object you need to select it and holding left mouse button move it to the desired place. To turn an object around its axis – pull the red square in the corner of the frame around the selected object (during this mouse cursor will turn into R), at this two red lines from the center will visually show turning angle and the gray frame – former position of the object.

To move an object using a keyboard, use its “arrow” keys. Select an object. Holding Shift key will accelerate movement. Holding Ctrl key will slow down movement. «<» and «>» keys turn the object around its axis.

PgUp and PgDown keys adjust the altitude (YPos).

3D mode.



When selecting an object in 3D mode the green frame shows the actual position of the object and the blue frame shows projection to the surface.

To move an object you have to select it and holding Shift+left-mouse-button move it to the desired place.

To turn an object around its axis select it and holding Ctrl+left-mouse-button move the mouse (cursor will change into R).

In 3D mode, just like in 2D mode, you can use hotkeys to move objects. These are the “arrows” for moving objects by X and Z, PgUp and PgDown keys for moving along Y axis, “<” and “>” keys for turning the object. To slow down hold Ctrl, to accelerate hold Shift.

6.3. Copy, Paste, Delete.

Any selected object (or a group of any selected objects) can be copied into clipboard and pasted anywhere on the map.

Hotkeys: Ctrl+C – copy, Ctrl+V – paste. You can also perform these actions through the dropdown menu by left mouse click. In this case the objects will be placed in a spot where mouse cursor is.

Also you can repeatedly paste the same object onto a map, if you hold Ctrl when selecting the object from objects library and placing it on the map.

6.4. Grouping of objects.

Grouping of objects helps creating some sort of blocks inside a mission from a large number of objects to work with them more comfortably.

You can group objects in 2D and 3D modes.

To group objects on a map select these objects, then left mouse click => Group Objects. After this all objects are gathered under Group icon and later you can move them by moving this icon. So this basically binds objects together. It's very convenient, for example, when creating a big city from a large number of small blocks or when creating some closed functional mission block.





You can “discharge” mission space by turning off blocks through objects filter (Section 4.1). At this the group will be displayed by Group icon.

This way you can also edit already created groups.

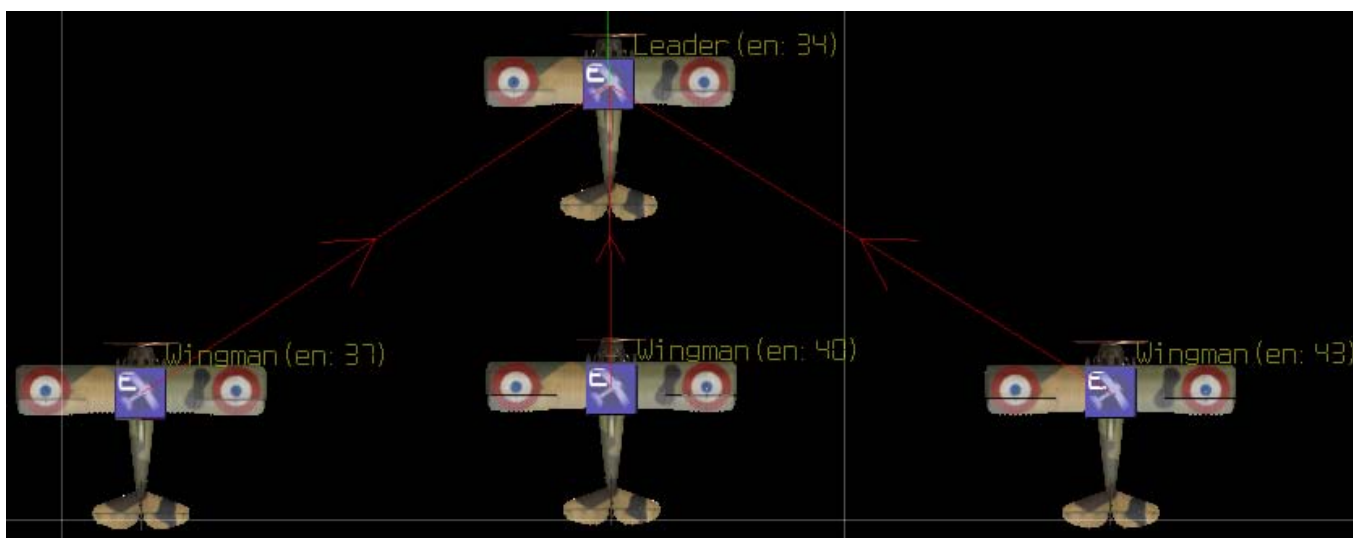
To do this select a group, right-mouse-click => Set Group as Working (if you are going to add/move blocks inside a group – don’t forget to turn on the corresponding filter in objects filter, otherwise you won’t be able to move objects).

After you click Set Group as Working, the group will be marked by yellow frame. In this mode you can move, delete and add objects to the group. For this simply place an object into the Editor’s working field and it will automatically be added to the working group.

Mandatory!: - don’t forget to close group editing mode (since this mode suggests working with this group only at this moment), to do that: right-mouse-click=>Reset Working Group.

6.5. Uniting moving objects into Wing or Group.

In this case group means that several objects (in the given example – planes) are subordinate to one leader and execute all commands that the leader gets.



To unite any moving objects into a wing or a group you need to “attach” all subordinate objects of the group to the group leader with target link.

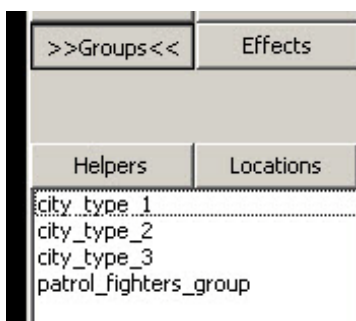
The leader will transmit all commands he receives from a mission to his subordinates. Therefore further mission-group interaction goes through the group leader and all direct commands are given to the group leader (there are some notes that are going to be mentioned further).

In the same manner all the other objects, such as cars, tanks and artillery, with the exception of trains and balloons, are united (see Section 7.2.1).

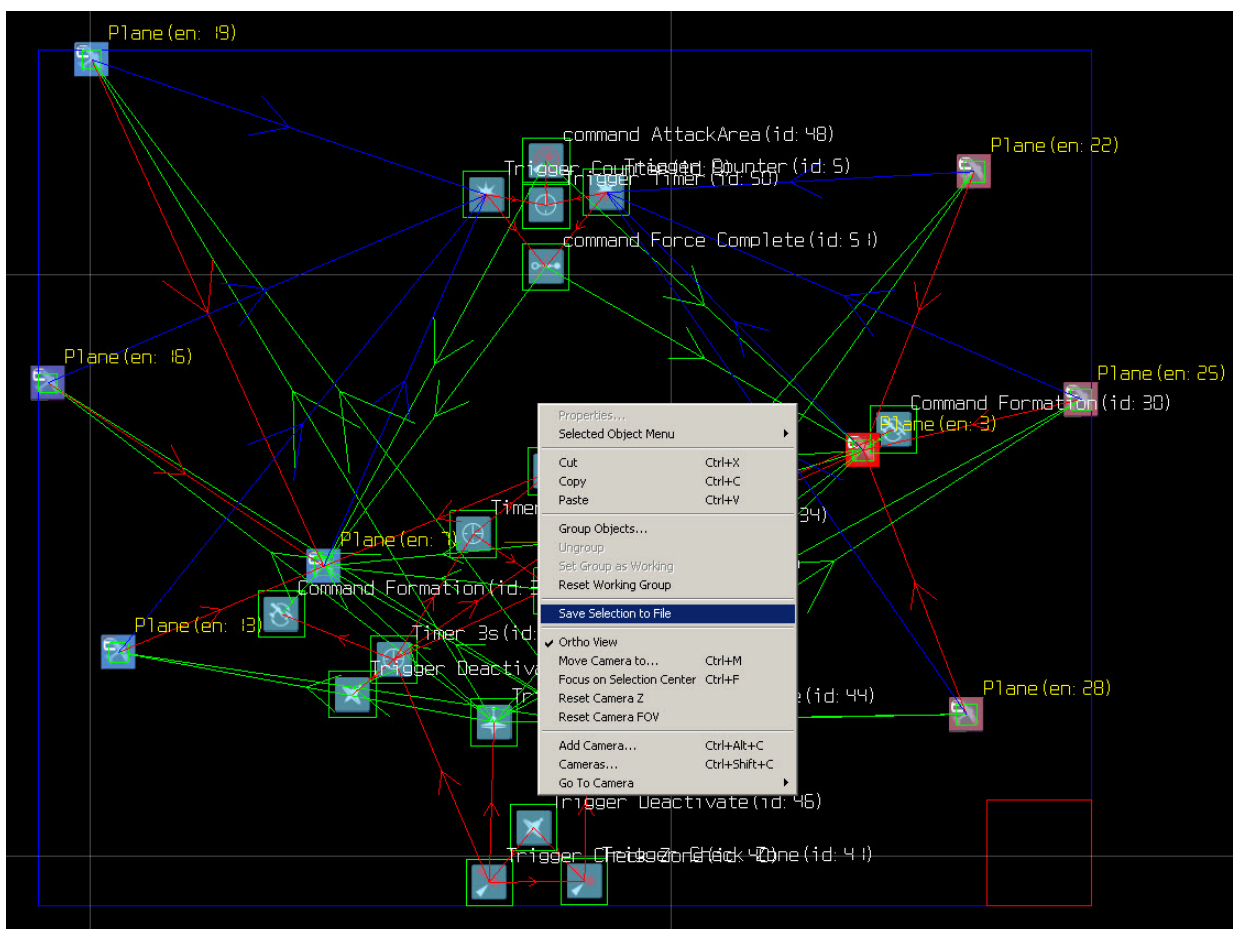
There is a possibility to adjust wingmen positions in the group (see Section 7.2.1) and to give an order to change formation in a wing or a group (see Section 7.7.8).

6.6. Saving to file, importing and inserting of pre-saved groups.

6.6.1 Saving to file



You can save to file any parts of a mission. The saved files are stored and saved by default to data/missions directory and can be seen in Groups section of the list in the right part of the Editor's screen (Objects Library).



For saving objects to a file: right-mouse-click => Save Selection to File

With this you can insert saved groups from Groups section into a mission as any ordinary objects (functional parts of mission, templates, standard groups etc.) and also import them.

6.6.2 Importing objects to mission.

You can also import objects saved to *.Group file into any other mission.

In this case, when importing (Import from File...Section 4.2.1.) such block into the next mission all objects and entities are imported exactly into that coordinates and places where they have been saved to file.

It's very convenient to use in case you want to put a template of one mission over the other mission. This way you can do parallel work over one mission or import mission templates made beforehand.

7. Editor's objects.

Each MCU or object placed on a map in the Editor has a number of editable properties.

The Editor allows group editing of a set of basic properties (further Basic Properties) of identical objects, in other words you can select with a frame or with CTRL several similar objects and set the same properties to them all.

Basic Properties window automatically opens when selecting an object or MCU and is by default in bottom left corner of the Editor.

7.1. Objects Library menu of the Editor

Planes	Buildings
Vehicles	Blocks
Trains	Airfields
Ships	Battlefield
Artillery	Firing-point
Aerostats	>>Bridges<<
MCUs	Grounds
Groups	Effects
Helpers	Locations

Planes: objects - planes.

Vehicles: Editor's objects – ground objects, cars and tanks.

Trains: objects – trains and carriages.

Ships: objects – ships.

Artillery: objects – field and howitzer artillery, AAA, machineguns and anti-aircraft machineguns.

Aerostats: objects – recon balloons.

MCUs: mission objects– Mission Command Units, triggers, commands, translators.

Groups: objects – files saved to *.Group in data/missions folder (see Section 6.6.).

Buildings: objects – single standing ground buildings.

Blocks: objects – buildings united in blocks, basic blocks of cities, villages, churches.

Airfields: objects – airfields or blocks of airfields.

Battlefield: objects – components of a battlefield, barbed wire, firing points, tents.

Firing point: objects – firing points, permanent fire position, earth-and-timber-emplacements.

Bridges: objects – bridges, car (HW) and railway (RW) bridges of various length and type.

Grounds: - objects from the Grounds menu

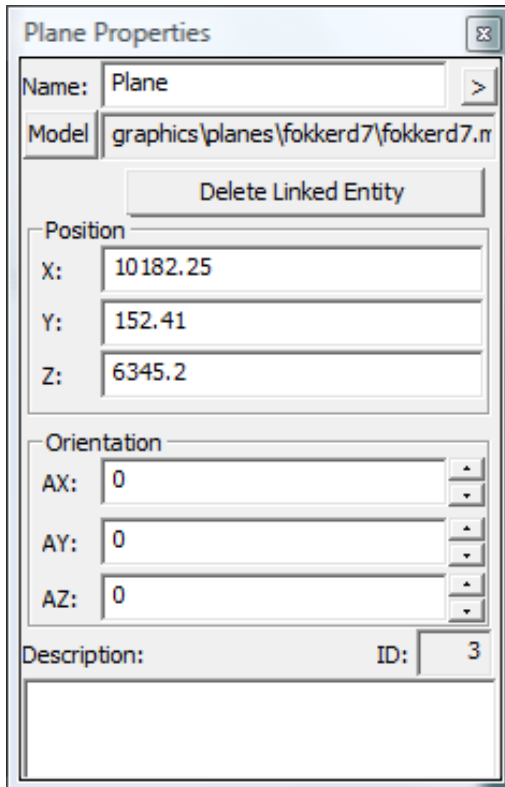
Effects: objects – effects emitters in a mission: sounds, smoke, explosion, etc. (see Section 7.6.6.).

Helpers: objects intended for interface linking with mission generator (see Section 7.2.5.).

Locations: objects intended for placing of location types into the database that is used by mission generator (see 7.2.6.).

7. 2. Basic Properties for Editor's objects.

7.2.1 Planes/Vehicles/Artillery/Aerostats



«>» button – opens Additional Properties window, we'll describe it below.

Name – objects name. In this field you can specify airplane name. Also in case of multiplayer mission – it's the name of airplane selection slots.

Model – this field shows the directory where the model that belongs to the object is. Model button allows assigning another model (including internal script) with saving of all properties of the object. Moreover, when selecting objects of the same type, for example when selecting all Plane type or Vehicle type objects, you can edit the whole group or change type of planes in the group from one to another.

Create Linked Entity/Delete Linked Entity – creates/deletes objects entity in the game world.

Limitation: An object without entity can't interact with game world, can't be linked with MCU or other objects and won't even visually exist in a mission. Exceptions from this are buildings – they will be displayed in game and objects will be able to interact with them even if they have no entity. However in this case buildings are not able to change their state.

Therefore this button has to be pressed right after you place an object to a map if this object (plane, tank etc.) participates in the mission, executes commands or gives messages in the mission. Deleting game entity doesn't have any obvious sense and can be used only perhaps to break all links of an object.

Fields Position X:, Y:, Z: - show coordinates of current object position in game world, where Y – altitude above sea level.

Fields Orientation AX:, AY:, AZ: - show object orientation in game world. The values are changing clockwise from 0 to 360 degrees. X axis corresponds to tilt axis (object turn around its longitudinal axis),

You can add (<<<) or remove (>>>) carriages from formation. The list of attached carriages is displayed in the left window (Carriage). With Up and Down buttons you can change the sequence of carriages in train formation.

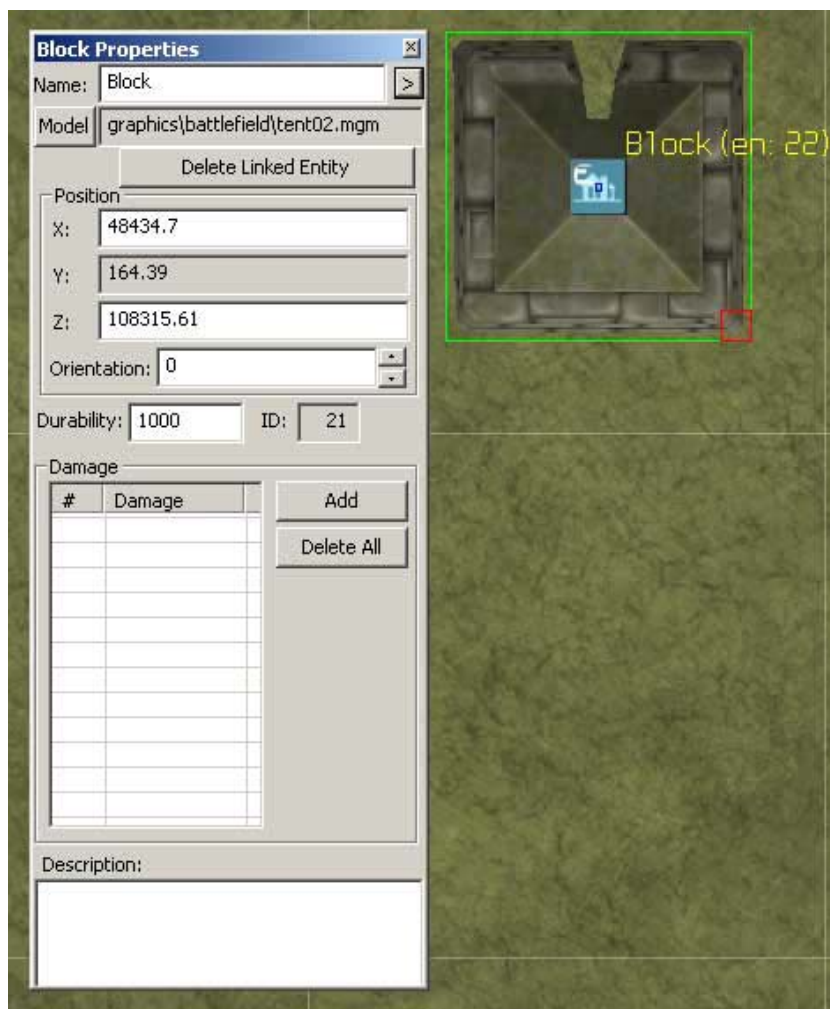
7.2.3. Firing Point

When Entity is being created it's in fact an independent ground object with all the properties of ground vehicles, only not being able to move.

7.2.4. Buildings/Blocks/Airfields/Battlefield/Bridges.

Ground objects-buildings (Buildings) have the same main properties as any other objects but with one feature we're going to describe below.

All objects of this type will be visible in a mission without created entities, but without it they will be undestroyable. If these objects have Entity, then in Durability field you can set “resistance” to enemy fire (basically its life points).

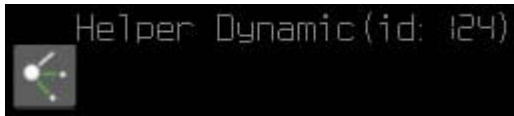


and in Damage field using Add buttons you'll be able to set damage level to an object, where:

7.2.5. Helpers

Helpers are meant for interface linking of the Editor and mission generator. You can read more detailed description of practical usage of helpers in “ROF Template Editor Users Manual”

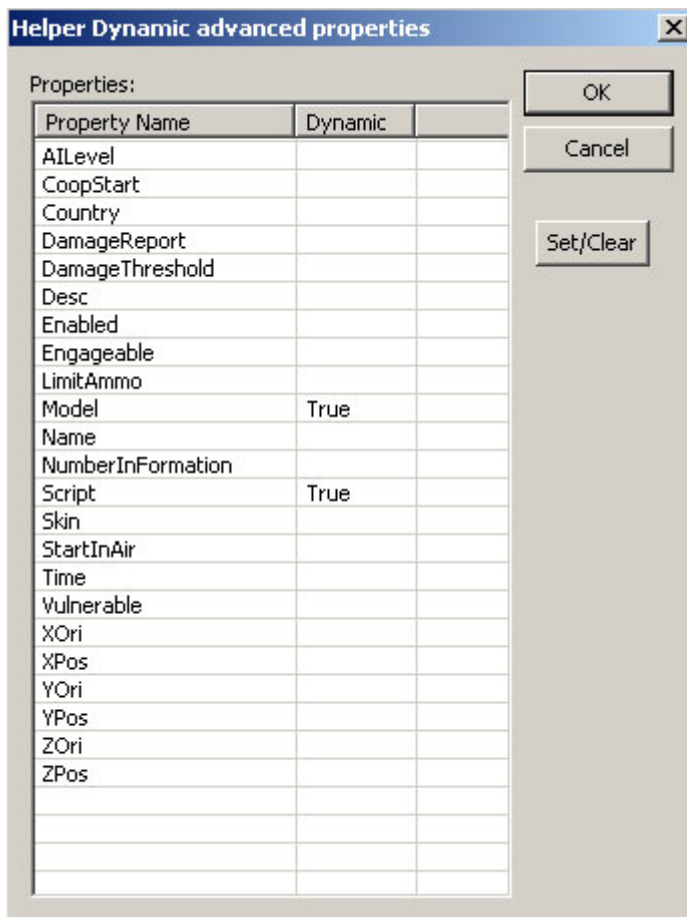
Helper: Dynamic



Sets dynamic properties of an object and is used during mission generation.

It is linked to an object or a group of similar objects (TL with each one) by Target Links.

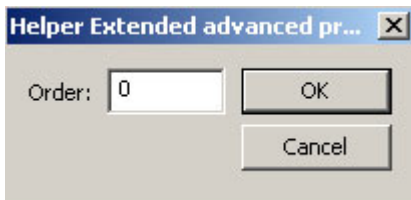
In Advanced Properties menu you can see the list of properties of the generated object (“Property name”) and “Dynamic” section, where you set those properties that are used during generation. True = use and is displayed in Template Editor GUI.



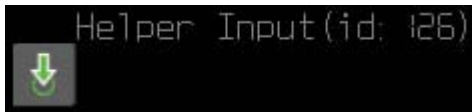
Helper: Extended



This one is usually used to determine/set the number of objects in a group. Sets IDs of objects for a preset, which will be added/removed from a preset during mission generation.



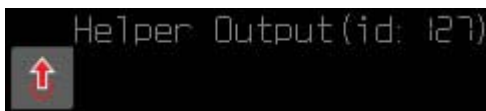
Helper: Input



This helper shows all links incoming to a preset.

It is used for interface connection between Mission Editor and Template Editor.

Helper: Output



This helper shows all links coming out of a preset.

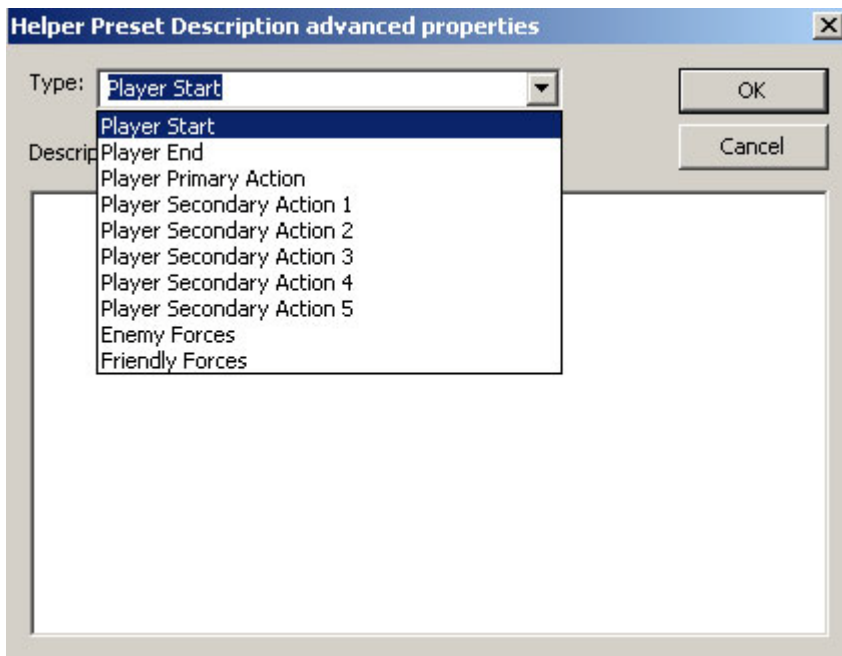
It is used for interface connection between Mission Editor and Template Editor.

Helper: Preset Description

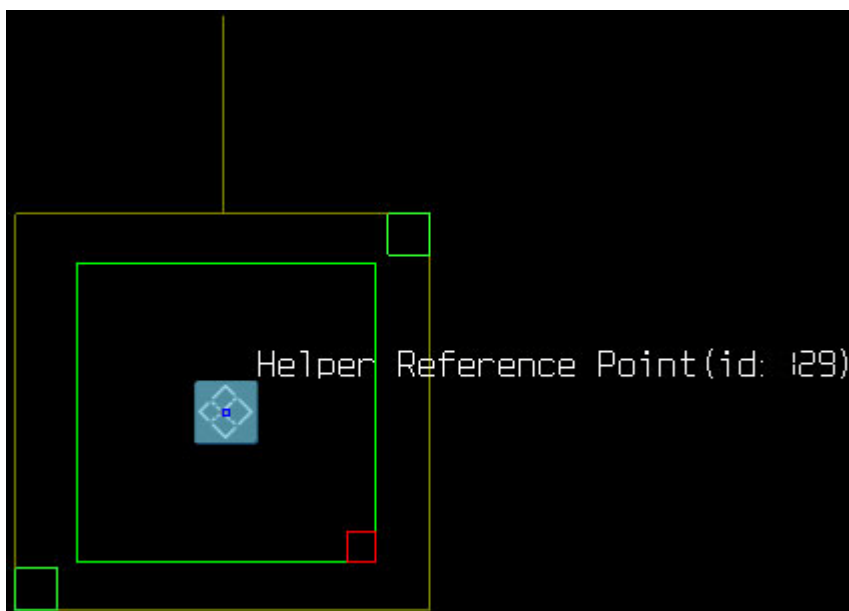


This helper sets text description of a preset for Mission Generator (component of Mission Briefing).

In Advanced Properties you set preset type -“Type” and enter the text describing the preset in “Description Text” field.



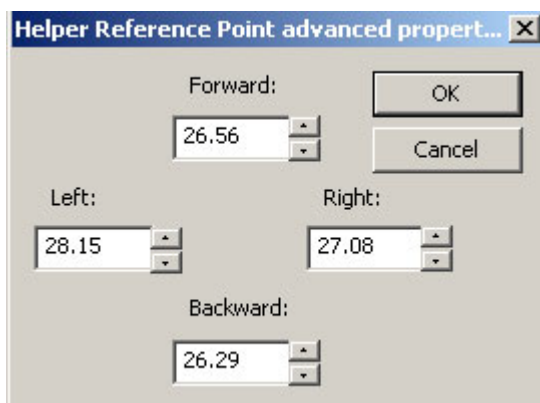
Helper: Reference Point



This helper marks center, orientation and size of a preset and is used during reading the rules of placing a preset to a location.

By holding left mouse button and pulling the green markers of external frame you can edit the size of a Reference Point. By clicking on an icon and red marker of internal frame you can move and rotate a Reference Point.

Sizes can also be set with “Advanced Properties” dialog.



Helper Reference Point advanced property...

Forward: 26.56

Left: 28.15

Right: 27.08

Backward: 26.29

OK Cancel

7.2.6. Locations

Locations are meant for creating a locations database, which is used by mission generator.

All locations have a basic set of properties: “Basic Properties”.



Air Objective

Name: >

Position and Orientation

X:

Z:

AY:

Length: Width:

Order Countries: Edit Countries

Opposite Countries:

Description:

Name – location name.

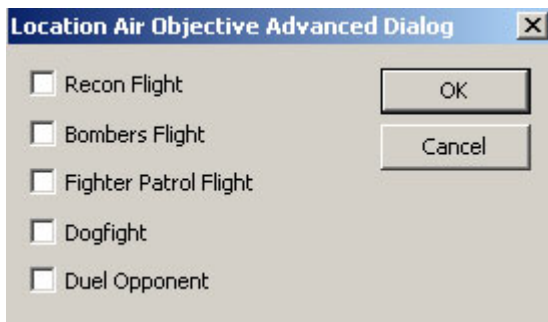
Fields for specifying location position (X;Y) and orientation (AY).

Length and Width – length and width of a location.

In “Edit Countries” menu of a location you set associated countries that are used during mission generation. At this location icon will change its color depending on the specified coalitions (Opposite/Order) and will remain grey if non-standard coalition settings are applied.



44



Recon Flight – enemy recon planes type

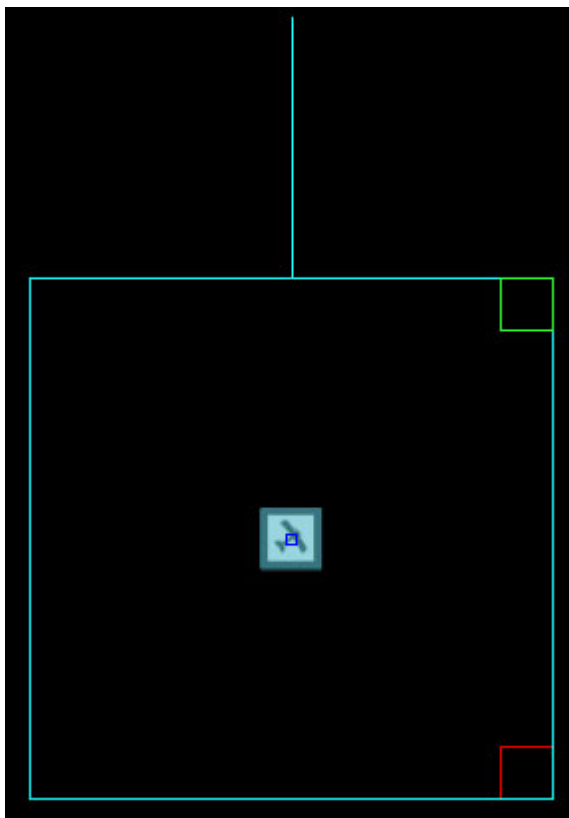
Bombers Flight – enemy bombers type

Fighter Patrol Flight – enemy fighters type

Dogfight – dogfight type

Duel Opponent – duel type

Airfield



It is used during airfield-squadrons DB creation.

In Advanced Properties menu you set airfield settings, with this a DB is created that is used during mission generation.

Add Associated Squadron

Country Name	Squadron Name	Plane Types Number	Squad Start	Squad End	
France	Escadrille 167	1	22.08.1918	11.11.1918	
France	Escadrille 154	1	01.03.1918	11.11.1918	
France	Escadrille 152	1	01.07.1918	11.11.1918	
France	Escadrille 124	2	01.12.1917	11.11.1918	
France	Escadrille 103	1	01.04.1917	11.11.1918	
France	Escadrille 102	1	01.03.1918	11.11.1918	
France	Escadrille 99	1	01.01.1917	11.11.1918	
France	Escadrille 94	2	01.03.1918	11.11.1918	
France	Escadrille 88	1	01.03.1918	11.11.1918	
France	Escadrille 87	2	01.05.1918	11.11.1918	
France	Escadrille 84	1	01.06.1917	11.11.1918	
France	Escadrille 81	1	01.12.1917	11.11.1918	
France	Escadrille 77	2	01.02.1918	11.11.1918	
France	Escadrille 73	1	01.11.1917	11.11.1918	
France	Escadrille 68	2	01.03.1918	11.11.1918	
France	Escadrille 67	2	01.04.1918	11.11.1918	
France	Escadrille 65	1	01.02.1918	11.11.1918	
France	Escadrille 48	1	01.02.1917	11.11.1918	
France	Escadrille 38	1	01.10.1917	11.11.1918	
France	Escadrille 37	1	01.02.1918	11.11.1918	
France	Escadrille 31	1	01.09.1917	11.11.1918	
France	Escadrille 26	1	01.03.1918	11.11.1918	
France	Escadrille 23	2	01.08.1918	11.11.1918	
France	Escadrille 15	1	01.06.1917	11.11.1918	
France	Escadrille 12	1	01.02.1918	11.11.1918	
France	Escadrille 03	1	01.08.1917	11.11.1918	
Germany	Jasta 99	1	01.01.1917	11.11.1918	
Germany	Jasta 80	3	15.02.1918	01.07.1918	
Germany	Jasta 65	3	23.01.1918	01.09.1918	
Germany	Jasta 64	3	01.04.1918	01.08.1918	
Germany	Jasta 61	3	30.07.1918	11.11.1918	
Germany	Jasta 35	5	07.01.1917	01.01.1918	
Germany	Jasta 34	5	20.02.1917	01.12.1917	

Buttons: Add/Set, Cancel

Squadron Planes Details

Coming Date: 01.06.1916

Leaving Date: 01.06.1916

With this a list of available squadrons is displayed with number of plane types at squadron service and with Coming and Leaving dates of a squadron from the battlefield.

Therefore Coming Date field specifies arrival date, Leaving Date – date of departure.

In Squadron Planes Details you can see squadron planes list by time periods.

Squadron Plane Info

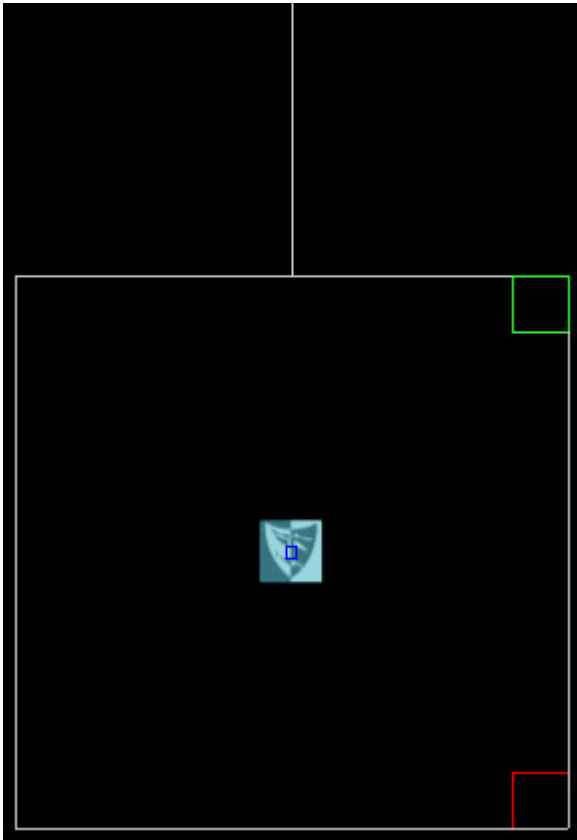
Squadron Name: Escadrille 152

Squadron Country: France

Buttons: OK, Cancel

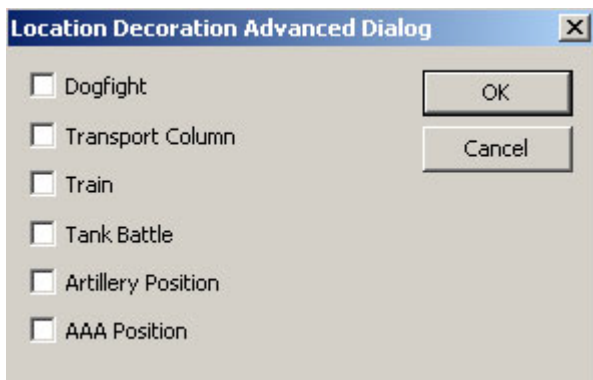
Plane Name	Start Date	End Date
SPAD XIII	01.07.1918	11.11.1918

Decoration



It is used when setting corresponding types of presets to a mission, in this case - “Decoration”.

In “Advanced Properties” you can set “Decoration” type of a location.



Dogfight – dogfight type

Transport Column – transport column type

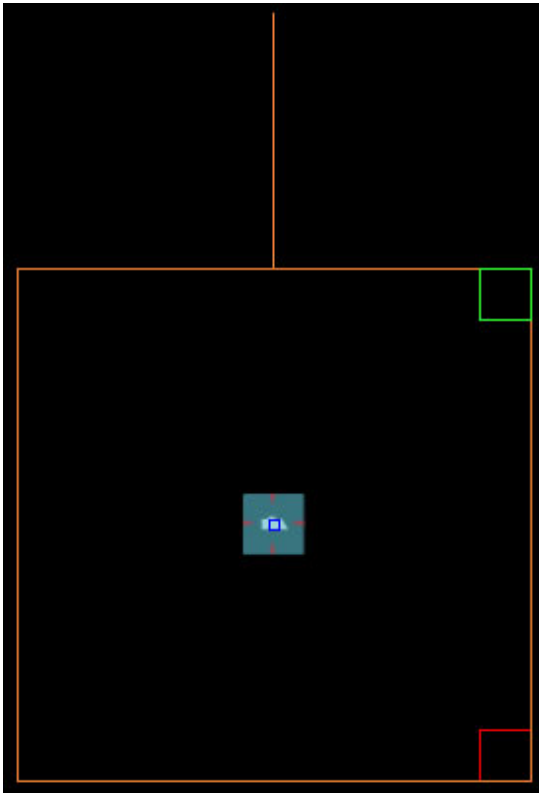
Train – train type

Tank Battle – tank battle type

Artillery Position – artillery position type

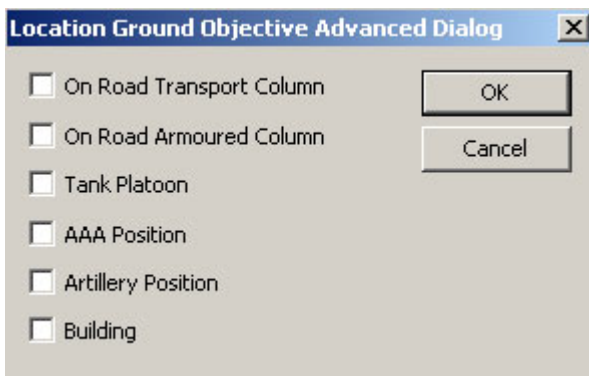
AAA Position – anti-aircraft artillery position

GroundObjective



It is used when setting corresponding types of presets to a mission, in this case - “Ground Objective”.

In “Advanced Properties” you can set “Ground Objective” type of location



On Road Transport Column – transport column on a road type

On Road Armored Column – armored column on a road

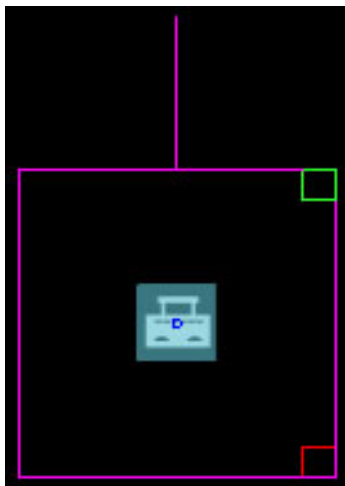
Tank Platoon – tank platoon type

AAA Position – anti-aircraft artillery type

Artillery Position – artillery position type

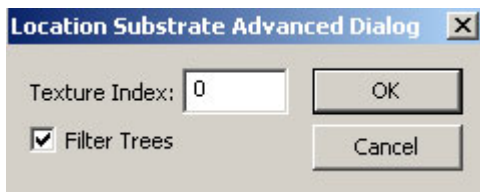
Building – building type

Substrate



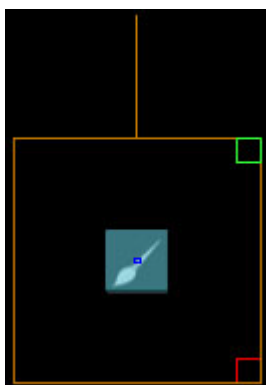
This location is for generating substrate-textures. During generation by “Generate Substrate INI file” command (see 4.2.8) layers.ini file is generated, which substrate-texture coordinates on a map are saved to. Substrate textures themselves are placed in their places (become visible in the Editor).

In Advanced Properties



Texture index is specified that will be used for the substrate. “Filter Trees” checkbox cuts/doesn’t cut trees during *.ini file generation.

TerrainLeveler



Is used to generate a map of heights. It shows coordinates for generation of leveling area.

After setting all locations you need to check the DB for errors with “Check Location DB Integrity” command (see 4.2.8.). Log file of checking is saved to IntegrityCheck.txt (data\LocationDatabase\).

7.3. Advanced Properties of Editor’s objects.

7.3.1. Planes/Vehicles/Artillery/Aerostats

Basic Properties window → left-click on “>” button, which is to the right of Name field.

In the appeared (Plane) Advanced window there are settings of the selected object, where:

Plane Advanced (1 object(s))

Linked TR ID: 11 ☒ Enabled

AI: Normal

Country: France

in Formation: 0

☐ Cooperative Start ☐ Start in the Air

☒ Vulnerable ☒ Engageable ☒ Limit Ammo

☒ Damage: (Threshold)

Damage report: 50

☐ AI Return to Base Decision

Script: LuaScripts\WorldObjects\spad13.txt

Skin: <default>

Payload: 0 empty

Route Time (minutes): 60

On Messages Table:

Message Type	Command ID	Target

Target ID: -1

Name:

Description:

Linked TR ID field – ID Linked Entity is automatically assigned by the Editor when Entity is being created for the given object;

Enabled field – if it's checked the object will appear immediately during mission loading, if not – the object can appear later in mission activated by Trigger:Activate (see 7.8.1.), when it's going to be needed according to the scenario.

AI field – AI level of an object (Low, Normal, High Ace, Player)

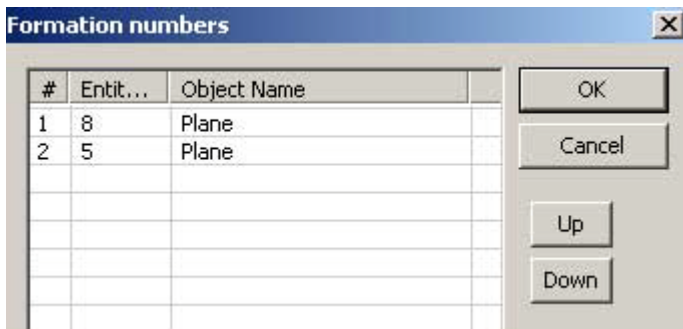
If you want to player controlled airplane in a mission you need to specify Player in AI field.

There is a possibility to change AI settings dynamically (except for Player setting) using Behavior command (see 7.6.3.).

Country field – select a country the object belongs to.

in Formation field – object's place in formation (group).

Change Formation button – changing formation (place inside a group) of wingmen in a group, works only for group leader. Select the line that specifies order number in formation of a wingman and click Up and Down to move it in the list.



Cooperative Start field – check this field if the mission is multiplayer. This allows making plane available in the lobby of a multiplayer game (list of planes available for selection) (see Section 8).

Start in the Air field – airstart of an object (only for planes). Don't forget to set plane altitude in Basic Properties (Y) so that the plane starts in the air at the desired altitude.

Vulnerable field – whether object is vulnerable to damage or not. There is a possibility to change this option dynamically by Behavior command (see 7.6.3.).

Engageable field – whether an object will be attacked by AI or not. It means that if an object has this option disabled it will not be attacked by other AI (planes, AAA etc.). This option can be dynamically changed by Behavior command (see 7.6.3.).

Limit Ammo field – unlimited ammo or not. This option can be dynamically changed by Behavior command (see 7.6.3.).

Damage field – Threshold or Step. Threshold – reaction on remaining life points (set in Damage report window) of an object in percents from initial condition. Step – step by step reaction (cyclic) on life removal in percents (set in Damage report window).

Damage report field – percents for Damage Report (Threshold/Step) conditions.

Damage report field is used to specify conditions when Damage Message is activated. If for example Damage report (Threshold) is 25 – the object in mission will send a message if his life points will decrease or fall below the threshold of 25% from initial condition. If Damage report (Step) is set to 25 – the object will send messages into a mission each time he receives a portion of damage equal to 25% of initial life points. Initial condition means 100% of life points.

AI Return to Base Decision field – if you select this option then AI will return to the nearest friendly airfield in automatic mode if in the course of a mission it receives critical damage or is out of ammo.

Script field – object script file directory.

Skin field – object paint scheme selection.

Payload field – plane armament configuration (ammo, bombs).

Route Time (minutes) field – estimated combat range (time) for AI (not used yet).

On Messages Table field – menu of Messages that are assigned to an object.

Add by Dialog button (in case of group leader) or Set by dialog (for wingman) – allow linking of wingmen to wing leaders or through selection in the dialog window. In other words a group can be formed by the dialog window.

Name – not used yet.

Description – not used yet.

Target ID field – in case of wingman – wing leader (his ID) for this wingman. Wing leaders always have this field set to -1.

An example in text format of a player controlled plane with ground spawning in a mission (Plane + MCU_TR_Entity):

Plane //object type - plane

```
{
  Name = "Plane"; //plane name will be shown in his icon in multiplayer if the plane will be AI controlled
  Index = 34; //plane ID
  LinkTrId = 35; //Entity ID
  XPos = 2170.13; //X coordinates of a plane
  YPos = 116.53; //Y coordinates of a plane, in this case this parameter will equal ground level that plane
  is being set to. If it's a car or an artillery position this value will always equal ground level that this
  object is set to;
  ZPos = 4263.06; //Z coordinates of a plane
  XOri = 0; //plane position (bank angle) by X axis,
  YOri = 0; //plane position (heading angle) by Y axis
  ZOri = 12.8; //plane position (pitching angle) by Z axis
  Script = "LuaScripts\WorldObjects\se5a.txt"; //plane script path
  Model = "graphics\planes\se5a\se5a.mgm"; //plane 3D path
  Country = 101; //country index assigned to the plane
  Desc = ""; //plane description
  Skin = ""; //plane skin path
  AILevel = 0; //plane AI level, in this case 0=Player (1=Low, 2=Normal, 3=High, 4=Ace)
  CoopStart = 0; //availability of the plane for coop mission, in this case the plane (plane slot to be
  selected by player) will be not available for a player
  NumberInFormation = 0; //position in formation
  Vulnerable = 1; //vulnerability (1=yes, 0=no)
  Engageable = 1; //will it be engaged by other AI (if = 0, then other AI won't attack this plane)
  LimitAmmo = 1; //limited ammo (yes=1, no=0)
  StartInAir = 0; //airstart yes/no; if you set =1 then you must specify Ypos for the plane, otherwise it will
  appear on the ground
  Time = 60; //flight time limit (for AI)
```

```

DamageReport = 50; //damage limit (level) to be used by On Damage Message (if enabled)
DamageThreshold = 1; //damage report; in this case from full value (100%), if =0 then the report will be
sent gradually (step)
}

```

MCU_TR_Entity //plane Entity, if you don't create it (create linked entity) then the plane won't appear in a mission

```

{
  Index = 35; //entity ID
  Name = "Plane entity"; //entity name (actually not used in a mission)
  Desc = ""; //entity description
  Targets = []; //an object that the Entity points at by Target Link
  Objects = []; //an object that the Entity points at by Object Link
  XPos = 2170.13; //matches object position that it's assigned to, in this case the plane
  YPos = 116.73; // matches object position that it's assigned to, in this case the plane
  ZPos = 4263.06; // matches object position that it's assigned to, in this case the plane
  XOri = 0; //not used
  YOri = 0; // not used
  ZOri = 0; // not used
  Enabled = 1; //initial condition of the object (plane), if =1 then it will appear immediately after mission
start, if =0 then it will have to be activated
  MisObjID = 34; //ID of the object that Entity belongs to
  OnMessages //list of messages that the plane will execute
  {
    OnMsg – one of the assigned to the plane messages
    {
      Type = []; - type of the assigned message
      CmdId = []; - ID of the command (source) after execution of which the next action (target) follows (if
exists)
      TarId = []; - ID of the command (target) that the plane will execute
    }
  }
}

```

7.4. Mission Command Units (MCUs) of the Editor.

7.4.1. Basic Properties of MCUs.

MCU Basic Properties

Name: Trigger Waypoint >

Position

X: 11419.86

Y: 252.41

Z: 13645.73

Orientation

AX: 0

AY: 0

AZ: 0

Targets: Find target

id	Name	Description
13	Trigger Waypoint	

Objects: Find object

id	Name	Description

Description: ID: 12

In Basic Properties of an MCU, compared to Objects, there are no Model and Create Linked Entity/Delete Linked Entity elements, but Targets and Objects lists are added that display name lists and IDs of objects that MCU have TL or OL with.

Find target and Find object buttons allow specifying an MCU for TL and Object for OL respectively. In fact these buttons have the same functions as SHIFT+T and SHIFT+O hotkeys correspondingly.

Double clicking a field in TL and OL lists will allow adding links by directly specifying MCU ID. Also if you double-click a link in the lists will let you edit them.

For some types of MCU their position is very important (CheckZone, AttackArea). This and other MCU properties will be explained in the next chapter of the Manual.

7.5. Advanced Properties of MCUs.

Let's look closely what each command does and what settings they have.

At first, let's look at the command priority settings - Priority.

Priority (execution priority) Low/Medium/High – specifies the priority of command execution by AI object. The higher the priority the more concentrated AI will be on executing this command up to ignoring the surrounding situation.

It's important to remember that for a command to be executed (accepted for execution by an object) it needs to be activated, i.e. “pulled” by TL or Message.

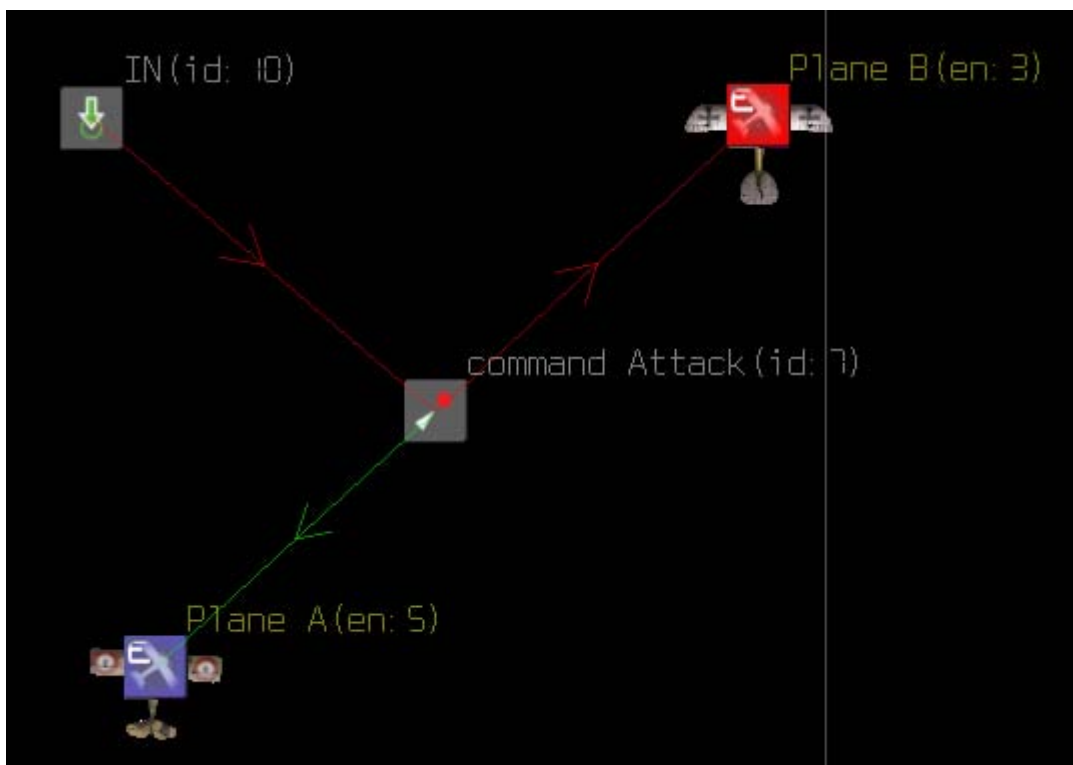
The moment of MCU activation has to be defined by the logic that an author puts into a mission and by mission plot.

7.6. MCU Command

7.6.1. Command:Attack



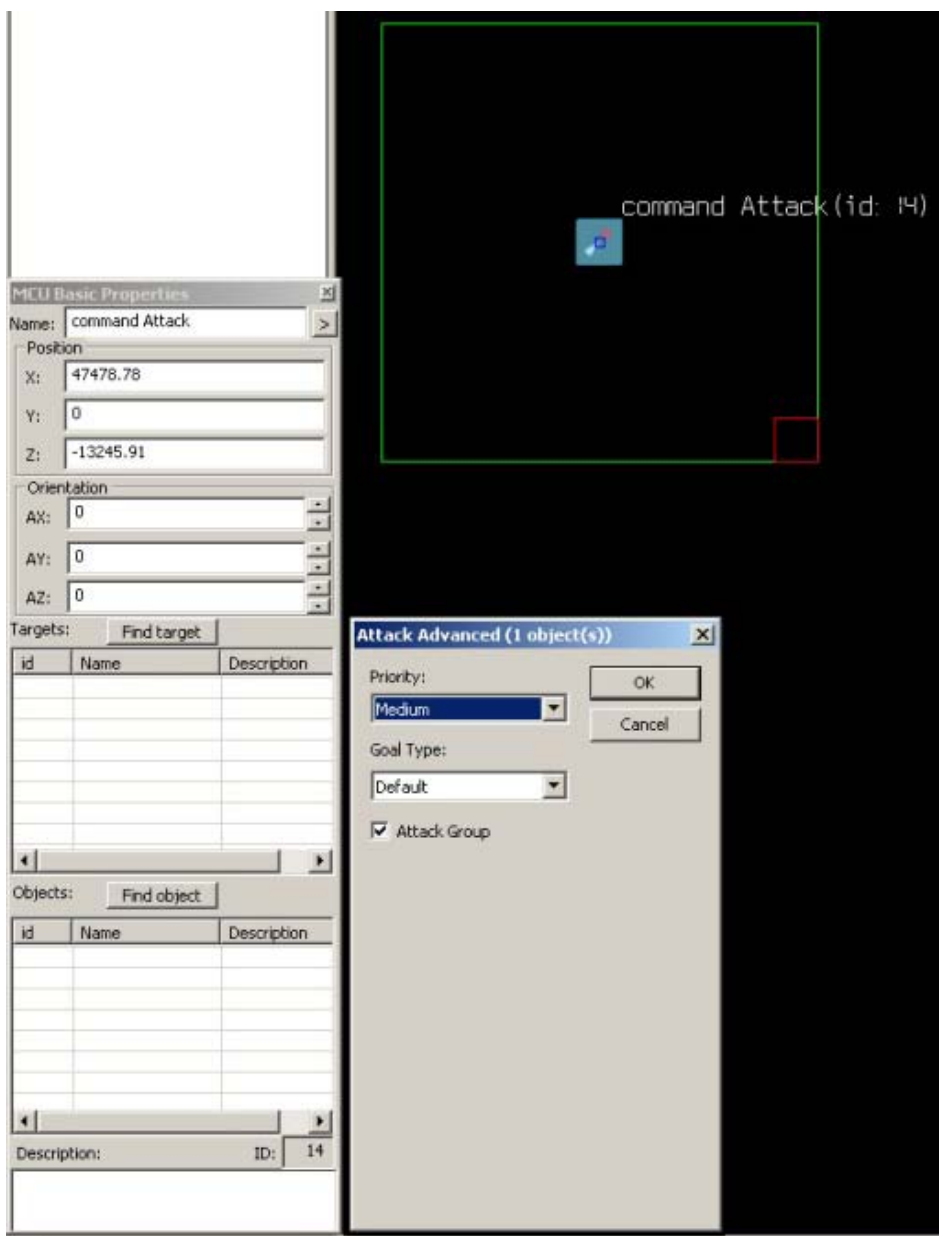
This command tells who (OL) has to attack who (TL). The feature of this command is that TL from it is set not to a command or a trigger, but to the attacked object. The object that receives this command will be attacking target-object until its destruction.



In the example on the picture – command for one plane to attack another plane, where:

- Plane A – attacking plane, Attack command refers to this plane by OL;
- Plane B – attacked plane, Attack command refers to this plane by TL;
- IN – any Message coming out of a mission that activates Attack command.

Double-clicking command's icon will open Attack Advanced Properties menu.



Advanced Properties of a command:

Priority – priority of command execution:

Goal Type: Default, Primary, Secondary. Displays team icon assigned to some object during game. This is made for convenience of visual determination of target object and doesn't influence mission results in any way.

Default – icon looks like



Primary – icon looks like



Secondary – icon looks like



Attack Group checkbox – when checked the attacker will attack the whole group that Target Linked object belongs to, and not only the object that TL points directly to.

Command priorities depending on AI level:

Considering different level of AI, which receives this command the attack will be executed only if there is certain distance between target and attacker (command execution range). With Priority set to Medium it will be:

- AI Low – 4 km
- AI Normal – 5 km
- AI High – 6 km
- AI Ace – 7 km

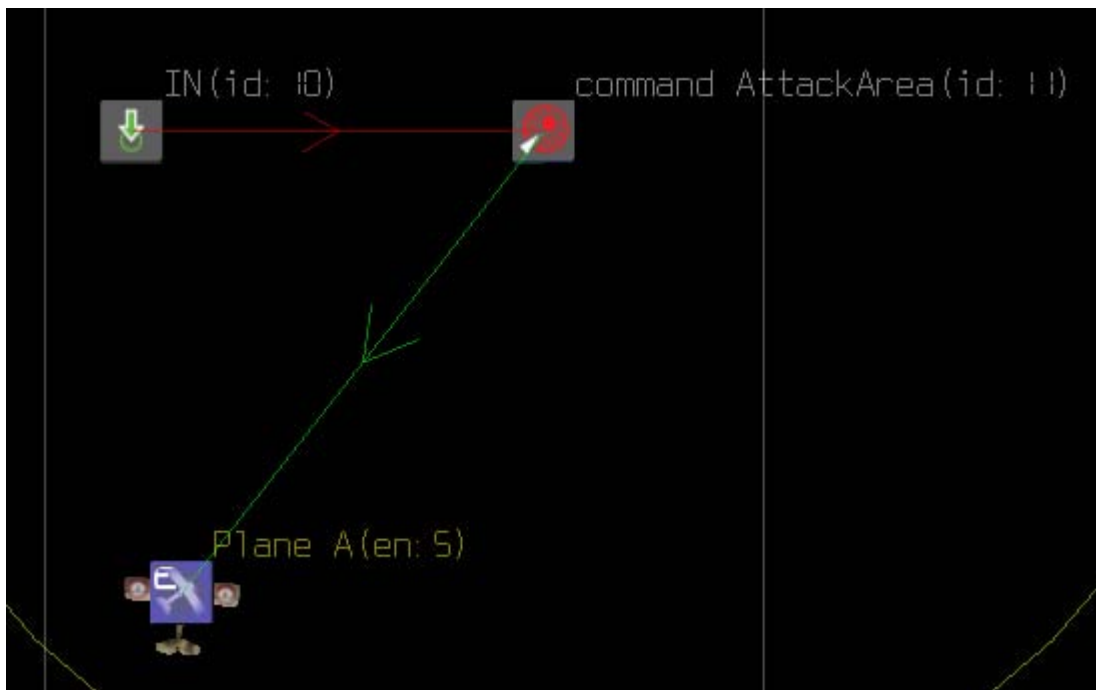
Text format of the command:

```
MCU_CMD_AttackTarget //MCU type
{
  Index = 5; //MCU ID
  Name = "command Attack"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2283.32; // X axis coordinates
  YPos = 119.646; // Y axis coordinates
  ZPos = 4280.93; // Z axis coordinates
  XOri = 0;
  YOri = 0;
  ZOri = 0;
  AttackGroup = 1; //attack group (if TL is pointed at the group leader)
  GoalType = 0; //goal type, affects how icon is displayed in GUI
  Priority = 1; //priority settings for command execution
}
```

7.6.2. Command:AttackArea



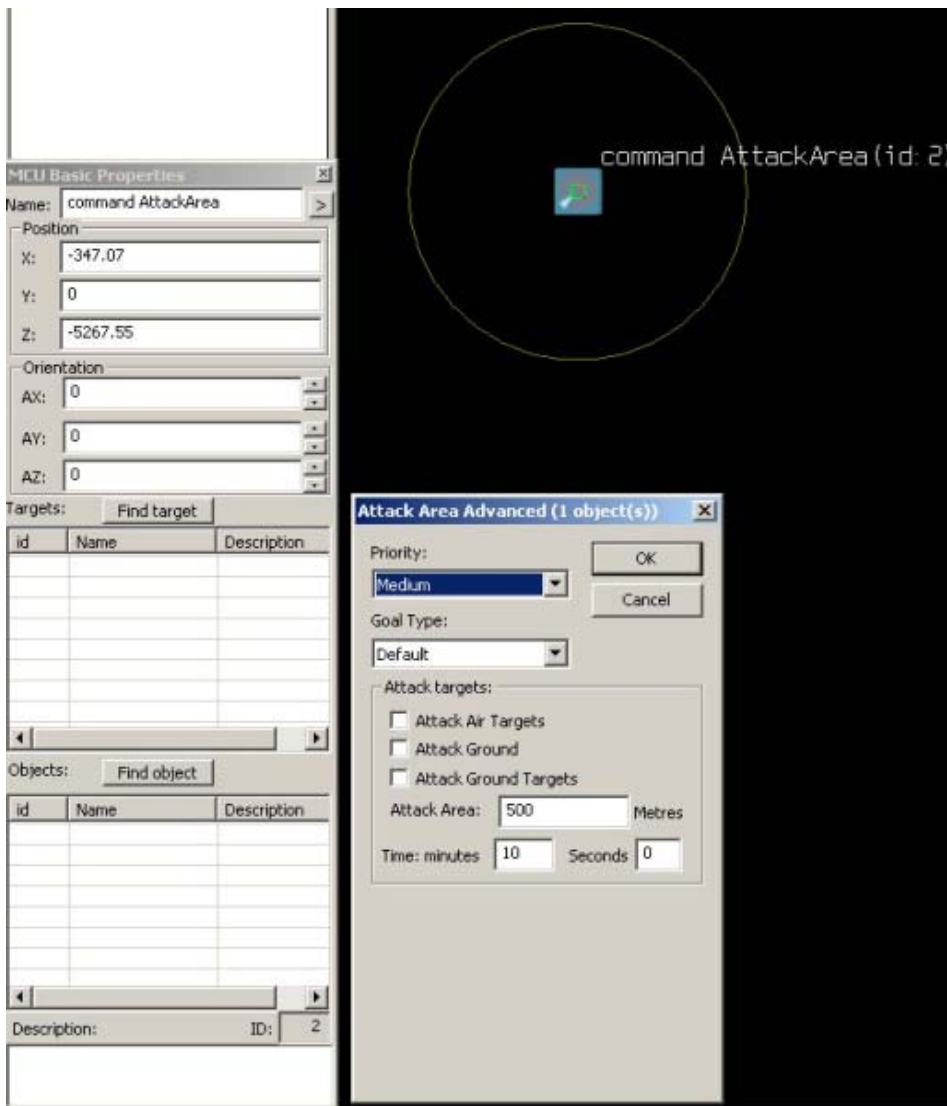
This command shows an area to an object where it has to attack enemy objects.



In the example on the picture – Attack Area command, where:

- Plane A – attacking plane, AttackArea command points at this plane by OL;
- IN – any Message coming out of a mission that activates AttackArea command.

Double-clicking command's icon will open Attack Area Advanced Properties.



Advanced Properties of the command:

Priority – priority of command execution:

Goal Type: Default, Primary, Secondary. Displays team icon assigned to some object during game. This is made for convenience of visual determination of target object and doesn't influence mission results in any way.

Default – icon looks like



Primary – icon looks like



Secondary – icon looks like



Attack Air Targets, Attack Ground and Attack Ground Targets checkboxes – when these checkboxes are selected the object will attack air targets, ground area and ground targets respectively.

You should consider that Attack Ground option will have different effect depending on the type of armament of the object:

- machineguns and AAA will attack occasional dots within a sphere with radius set by Attack Area setting and with center being in the point where command marker is placed. This issue should be considered to avoid situations when objects attack points under the ground;
- field cannons and howitzers will attack an occasional point on the surface in an area with radius being specified by Attack Area setting and with center in a point of orthogonal projection of command marker to the surface;
- planes will drop all their bomb load to a point of orthogonal projection of command marker to the surface. I.e. if the goal is to drop bombs from horizontal flight at an altitude then you should set this command at the altitude of drop with center being over the target.

Attack Area field – sets radius of attacked area in meters.

Time: minutes and Seconds fields – set time of life of a command in minutes and seconds respectively.

Text format of the command:

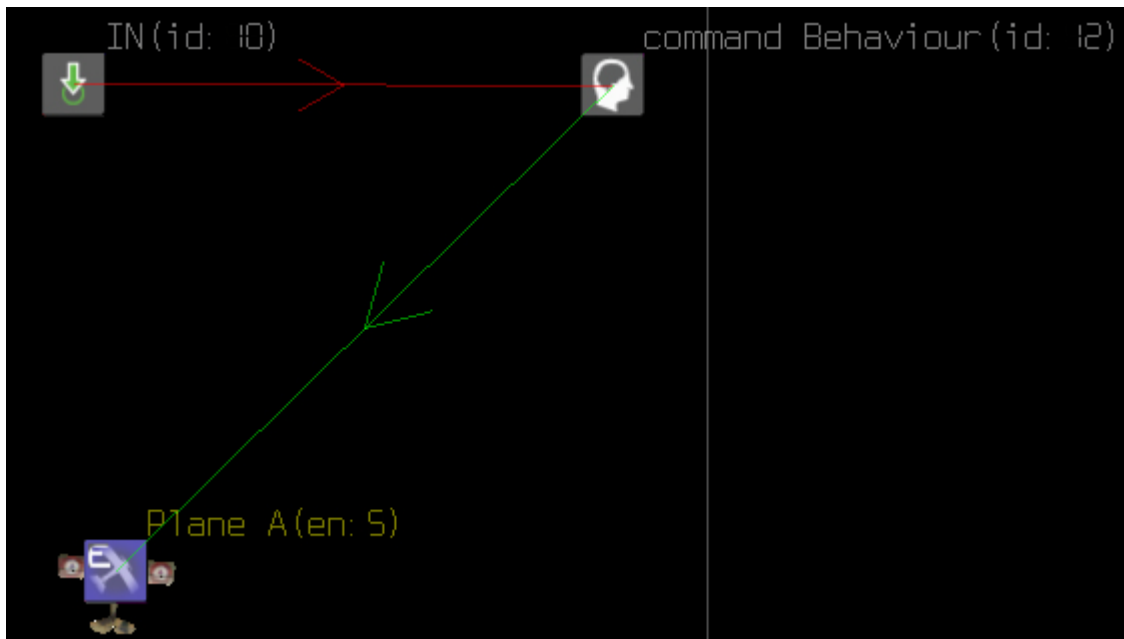
MCU_CMD_AttackArea

```
{
  Index = 4; //ID MCU
  Name = "command AttackArea"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2074.81; // X axis coordinates
  YPos = 113.7; // Y axis coordinates, this is altitude of Attack Area, don't forget to set this altitude since
  AI considers its value
  ZPos = 4018.81; // Z axis coordinates
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation (doesn't have any functions in a mission)
  ZOri = 0; // orientation (doesn't have any functions in a mission)
  AttackGround = 0; // Attack Area type (Attack Ground yes=1/no=0)
  AttackAir = 0; // Attack Area type (Attack Air yes=1/no=0)
  AttackGTargets = 0; // Attack Area type (Attack Ground Targets yes=1/no=0)
  AttackArea = 1000; //radius in meters
  Time = 600; //time of command execution in seconds
  GoalType = 0; //goal type, influence the look of an icon in GUI
  Priority = 1; //command execution priority settings
}
```

7.6.3. Command:Behavior



This command allows dynamically changing such properties as: vulnerability, ammo limit, possibility of being attacked by other AI and AI level.

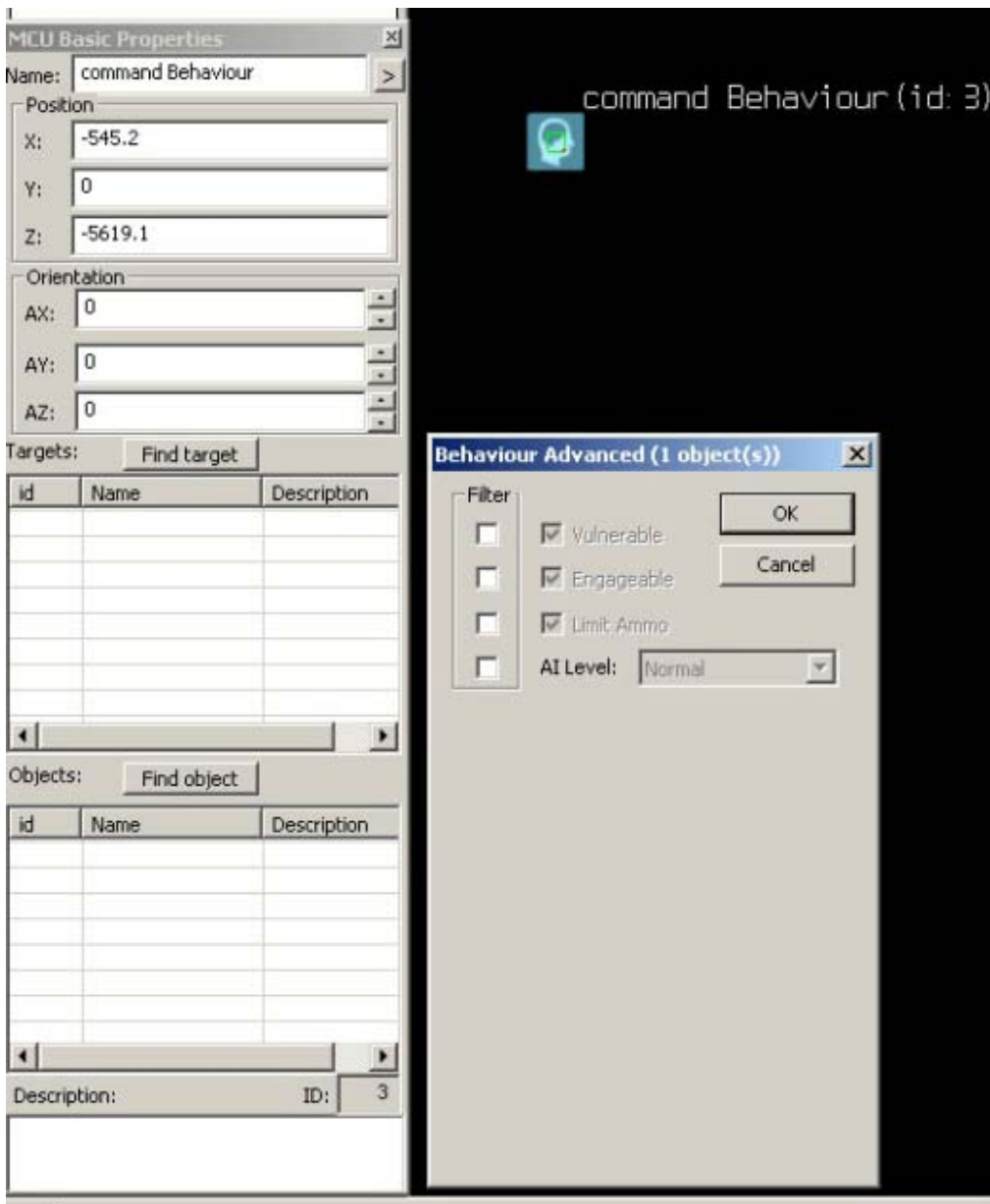


In the given example:

- with Behavior command activation by an outgoing from a mission message (TL from IN) Plane A will change its properties depending on the settings inside Behavior command.

The command can be applied to any object of a mission (except buildings and bridges) if it has the specified dynamic properties.

It can be linked to any object with OL. Double-click command's icon to open Behavior Advanced Properties menu.



Filter checkbox – when these checkboxes are selected object properties will become.

Vulnerable checkbox – object vulnerability to damage, yes/no;

Engageable checkbox – will AI engage the object or not;

Limit Ammo checkbox – unlimited ammo or not;

AI Level field – AI skill settings of the object.

Player – N/A, not used;

AI level can be assigned to any planes (or objects) that are not controlled by player.

Low - low skill;

Normal – medium skill;

High – high skill;

Ace – ace skill.

Text format of the command:

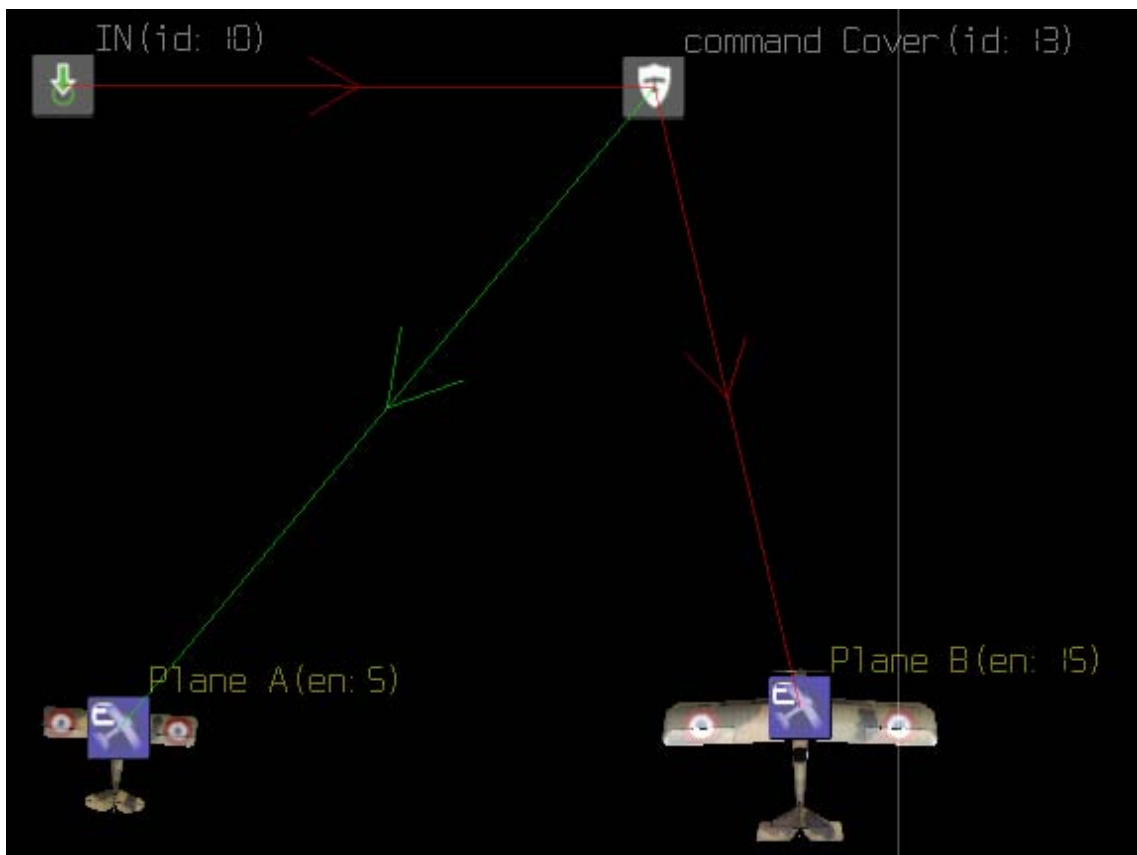
MCU_CMD_Behaviour //MCU type

```
{
  Index = 6; // MCU ID
  Name = "command Behaviour"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2024.18; // X axis coordinate
  YPos = 110.925; // Y axis coordinate
  ZPos = 4164.77; // Z axis coordinate
  XOri = 0; // orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation (doesn't have any functions in a mission)
  ZOri = 0; // orientation (doesn't have any functions in a mission)
  Filter = 0; //settings of the filter for the specified properties
  Vulnerable = 1; //object vulnerability settings
  Engageable = 1; //settings of object interaction with other AI
  LimitAmmo = 1; // ammo limit settings of the object
  AILevel = 2; //AI settings of the object
}
```

7.6.4. Command:Cover

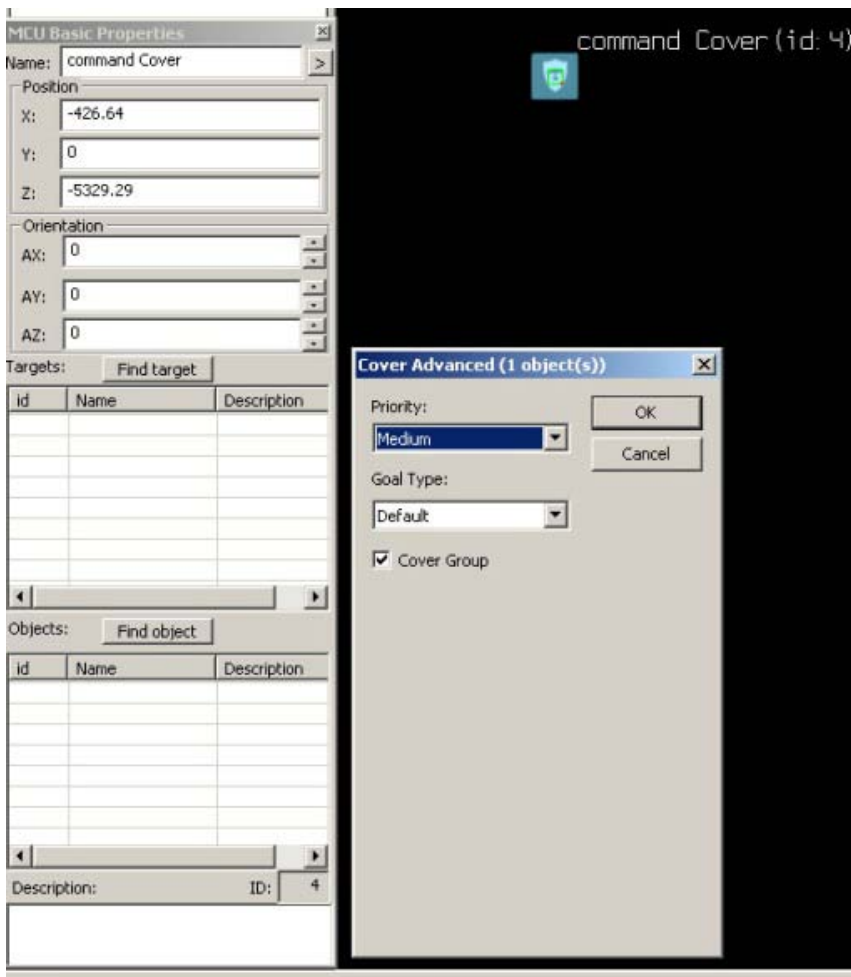


Command to cover (protect) objects or a group of objects. Can be applied ONLY to planes.



In the given example:

- Plane A receives command to Cover Plane B;
- Cover command points at the executing object (Plane A) with OL and to the object to cover with TL;
- IN – any Message coming out of a mission that activates Cover command.



It is linked to the leader of covering group with OL. Double-click commands icon to open Cover Advanced Properties.

Advanced Properties of the command:

Priority – priority of command execution:

Goal Type: Default, Primary, Secondary. Displays team icon assigned to some object during game. This is made for convenience of visual determination of covered object.

Default – icon looks like



Primary – icon looks like



Secondary – icon looks like



Cover Group checkbox: when checked the command is spread to cover the whole group of objects. If disabled the command is directed only to the object that TL points at.

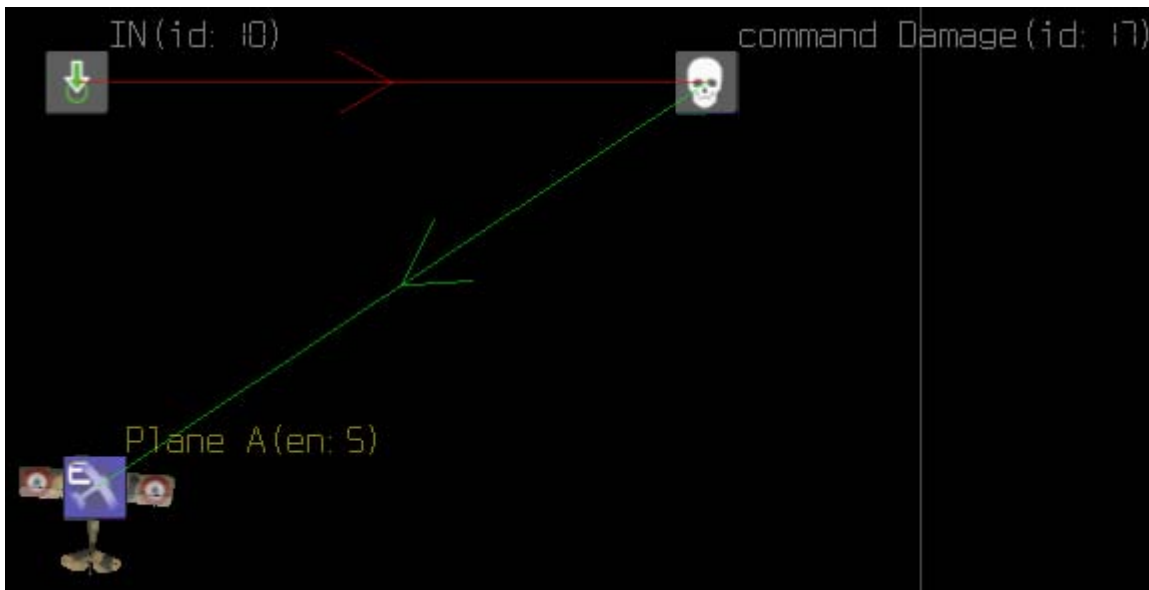
Text format of the command:

```
MCU_CMD_Cover // MCU type
{
  Index = 7; // MCU ID
  Name = "command Cover"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2077.79; // X axis coordinate
  YPos = 110.272; // Y axis coordinate
  ZPos = 4355.4; // Z axis coordinate
  XOri = 0; // orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation (doesn't have any functions in a mission)
  ZOri = 0; // orientation (doesn't have any functions in a mission)
  CoverGroup = 1; //cover the group yes=1/net=0
  GoalType = 0; //goal type, affects icon type in GUI
  Priority = 1; //command priority
}
```

7.6.5. Command:Damage



This command sets damage level inflicted to an object by a command from a mission.



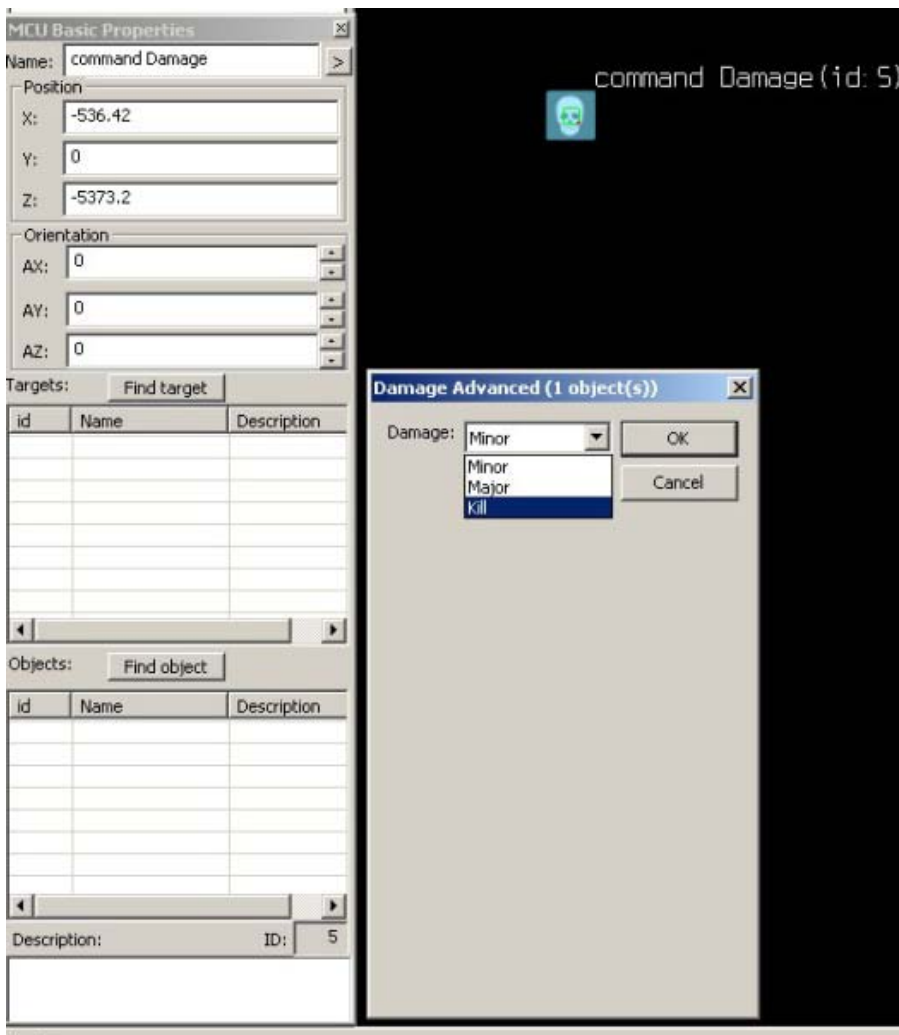
On the given example:

- command Damage – OL from the command to Plane A;
- IN – any command incoming from a mission that activates Damage command that will in turn set specified properties to the assigned object (Plane A).

It is connected to the object with OL.

If it's the group of objects then it's connected to the leader and will affect all objects in the group.

Double-click the icon of the command to open Damage Advanced Properties.



Advanced Properties of the command:

Damage – damage level settings.

Minor – sets low level of the inflicted damage;

Major – sets high level of the inflicted damage;

Kill – sets termination of an object.

Text format of the command:

MCU_CMD_Damage // MCU type

```
{
    Index = 8; // MCU ID
    Name = "command Damage"; // MCU name (not displayed in GUI)
    Desc = ""; // MCU description (not displayed in GUI)
    Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
    Objects = []; // objects' ID that this command points at with OL
    XPos = 2107.58; // X axis coordinates
    YPos = 112.755; // Y axis coordinates
    ZPos = 4272; // Z axis coordinates
    XOri = 0; // orientation (doesn't have any functions in a mission)
```

```

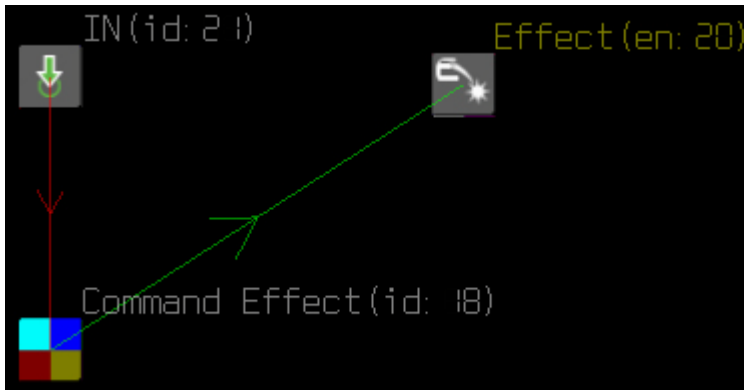
YOri = 0; // orientation (doesn't have any functions in a mission)
ZOri = 0; // orientation (doesn't have any functions in a mission)
Damage = 0; // damage level set to an object 0=Minor, 1=Major, 2=Kill
}

```

7.6.6. Command:Effect



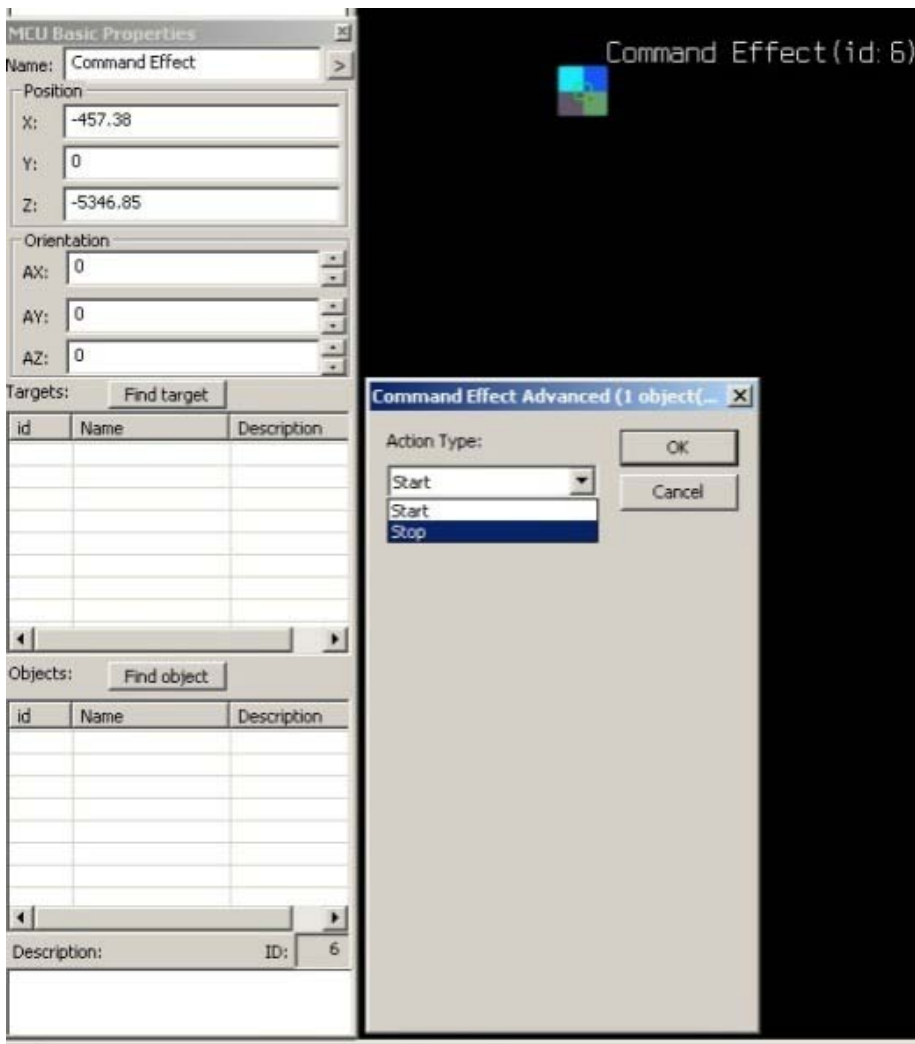
With this command you can activate or deactivate the Effect previously set onto a map.



In the given example:

- IN - any event incoming from a mission that activates Effect command;
- Effect-command connects to Effect-object with OL and depending on the internal settings either activates it (start) or deactivates it (stop).

Double-click command's icon to open Effect Advanced Properties.



Advanced Properties of the command:

Action Type field– select when the Effect activates (start) and deactivates (stop).

Start – Effect is activated;

Stop – Effect is deactivated.

Note: this command works only together with Effects section. This section contains Effects available in the Simple Mission Editor, number and types of which can vary with SME versions.

Using tips:

To start an Effect you need to place it on a map (Object Library=>Effects) and “bring to life” (Create Linked Entity). Then connect Effect-command with the Effect itself with OL. In this case Effect-command is activated by any event coming out of the mission.

Then the Effect will either start playing or stop depending on the assigned action in Action Type field.

Text format of the command:

MCU_CMD_Effect //тип MCU

```
{
  Index = 9; //ID MCU
  Name = "Command Effect"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2116.51; // X axis coordinate
  YPos = 114.073; // Y axis coordinate, for the command itself this position doesn't matter, but for the
Effect that this command launches this position must be specified, as the Effect is a visual (or sound)
object in a mission
  ZPos = 4218.38; // Z axis coordinate
  XOri = 0; // orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation (doesn't have any functions in a mission)
  ZOri = 0; // orientation (doesn't have any functions in a mission)
  ActionType = 0; // start=0 or stop=1 (start or stop)) of the Effect
}
```

7.6.7. Command:Force Complete



Interrupts the last command that the object has received.



In the given example we need to interrupt the Cover command and give the next command to the plane:

- Plane A (in progress of executing Cover command) – is covering Plane B;
- command activated – mission event that had activated Cover command (or any other command in progress);
- IN – event outgoing from a mission that should interrupt the current command (in given case – Plane B cover). The next command for Plane A is triggered by setting Timer 1s (timer set to 1 second) to avoid event collisions (in order not to break the next incoming command occasionally).

Double-click command's icon to open Force Complete Advanced Properties.

Advanced Properties of the command:

Priority – priority of command execution

Using tips:

When using this command you should remember that if there are some other actions (commands) assigned to the object after the current command then between the Force Complete command and the next command there must be Timer trigger set to 1 or 2 seconds. Otherwise all the following commands might not start (see example).

Text format of the command:

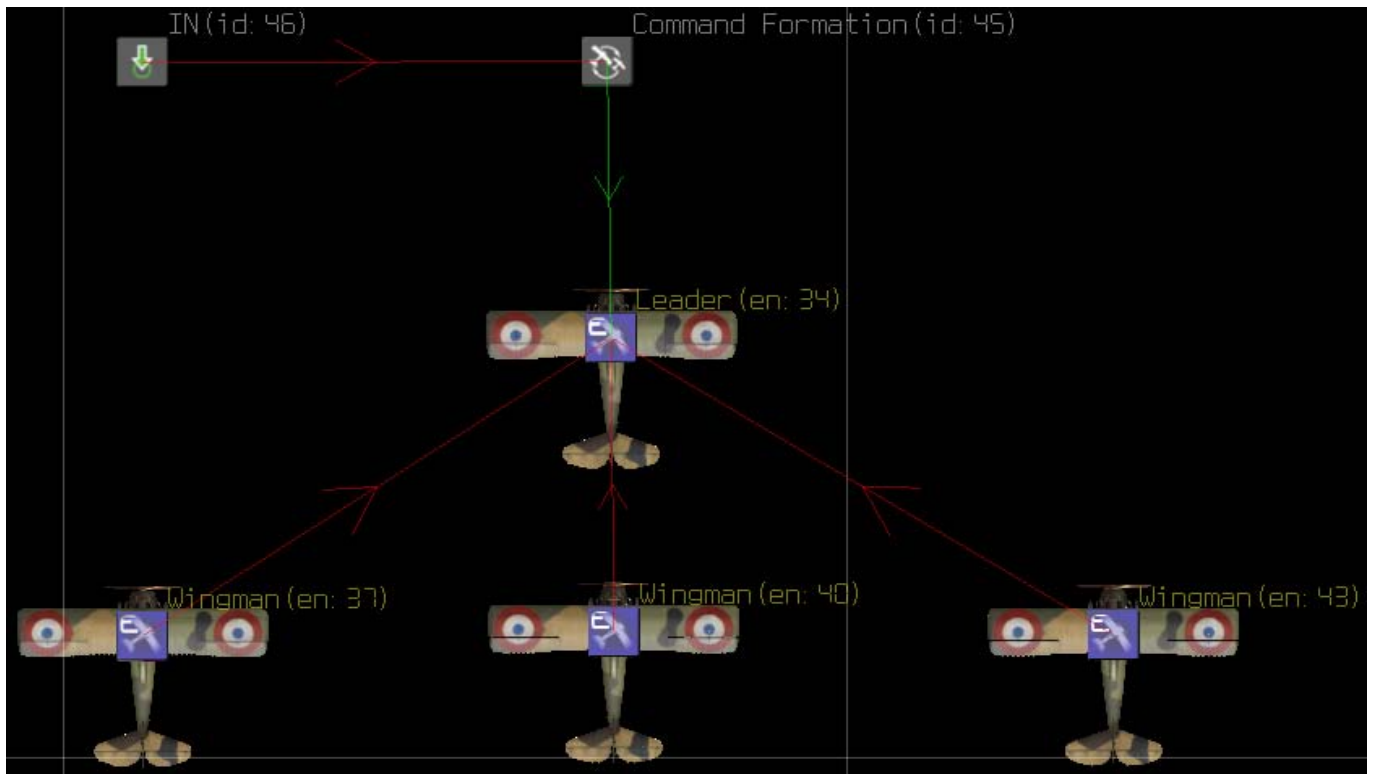
MCU_CMD_ForceComplete // MCU type

```
{
  Index = 10; // MCU ID
  Name = "command Force Complete"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2116.51; // X axis coordinates
  YPos = 115.985; // Y axis coordinates
  ZPos = 4039.66; // Z axis coordinates
  XOri = 0; // orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation (doesn't have any functions in a mission)
  ZOri = 0; // orientation (doesn't have any functions in a mission)
  Priority = 2; //priority settings of the command
}
```

7.6.8. Command:Formation



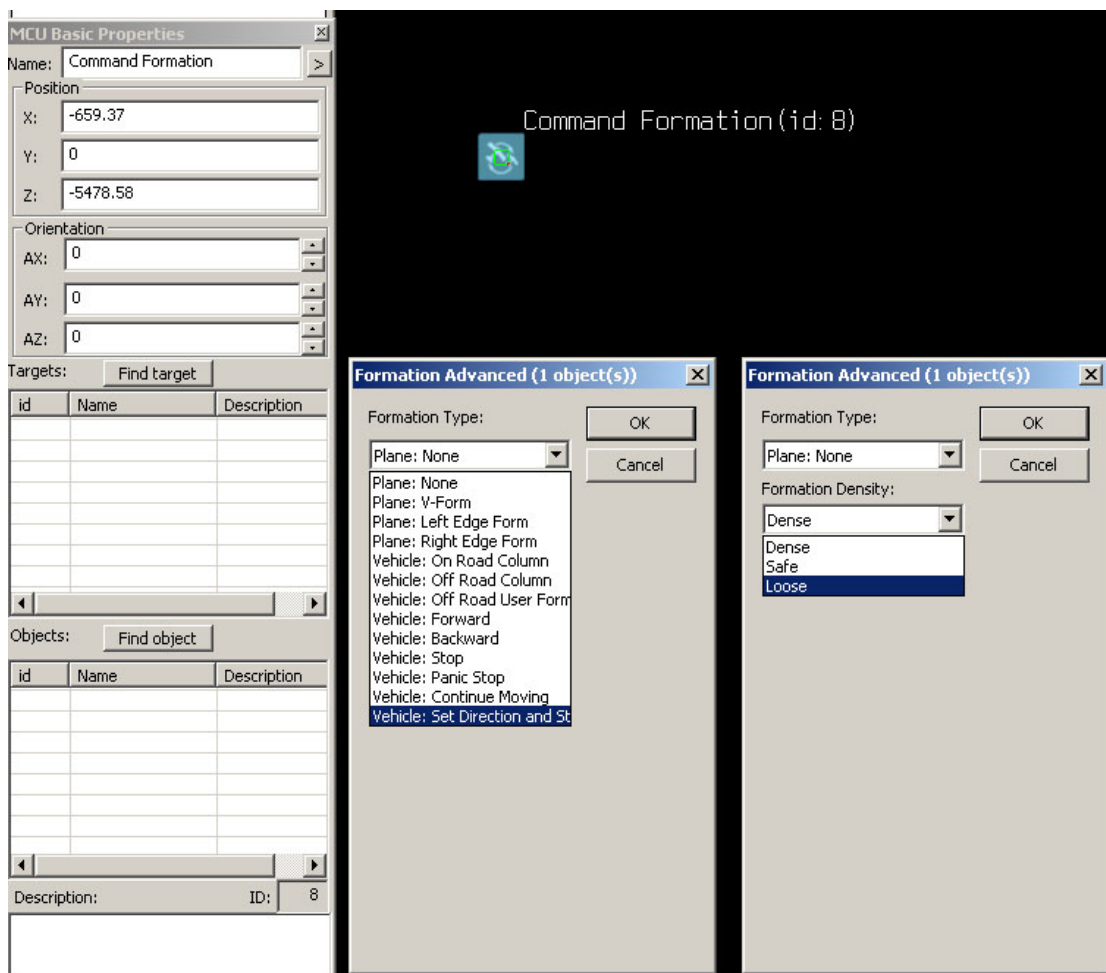
This command specifies the formation that a group of objects will move in.



In the given example:

- Command Formation is connected by OL to the group leader;
- IN – any event outgoing from a mission that activates Formation command, which in its turn gives the specified formation to the group leader.

Double-click command's icon to open Formation Advanced Properties.



Advanced Properties of the command:

Formation Type list – specifies formation type.

Formation types are divided into plane (with Plane: prefix) and vehicle (with Vehicle: prefix).

Plane: None – planes: column.

Plane: V-Form – planes: V образное построение, с ведущим самолетом в вершине.

Plane: Left Edge Form – planes: left echelon.

Plane: Right Edge Form – planes: right echelon.

Vehicle: On Road Column – vehicles moving on road network, in column.

Vehicle: Off Road Column – vehicles moving in column, directly to the point, ignoring road network.

Vehicle: Off Road User Formation – vehicles moving in user assigned order (i.e. in the order that user has placed vehicles in the Editor) directly to the point.

Vehicle: Forward – moving in forward motion.

Vehicle: Backward – moving in reverse motion (only for track vehicles).

Vehicle: Stop – stop on the route.

Vehicle: Panic Stop – stop with spreading around the area.

Vehicle: Continue Moving – continue moving after stop.

Vehicle: Set Direction and Stop – stop with turn to direction, for this you have to set YOri.

Formation Density list – specifies the distance between objects in formation from loose (Loose) to dense (Dense).

Text format of the command:

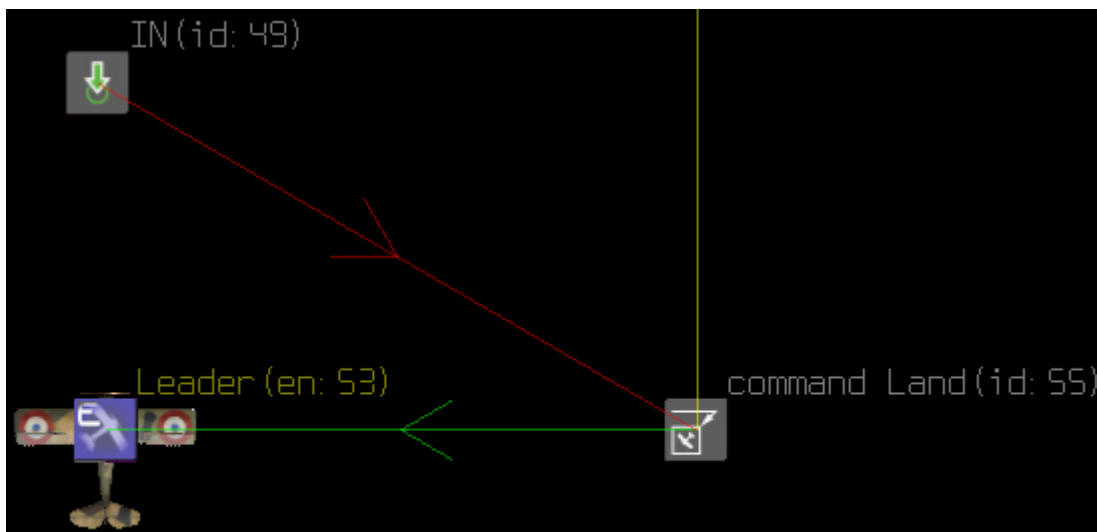
MCU_CMD_Formation // MCU type

```
{
  Index = 11; // MCU ID
  Name = "Command Formation"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2036.09; // X axis coordinates
  YPos = 111.578; // Y axis coordinates
  ZPos = 4209.45; // Z axis coordinates
  XOri = 0; // orientation (doesn't have any functions in a mission)
  YOri = 0; // orientation, used only by Set Direction And Stop
  ZOri = 0; //(doesn't have any functions in a mission)
  FormationType = 0; // formation type 0=Plane: None, 1= Plane: V-Form, 2= Plane: Left Edge Form, 3=
  Plane: Right Edge Form, 4= Vehicle: On Road Column, 5= Vehicle: Off Road Column, 6= Vehicle: Off
  Road User Formation, 7= Vehicle: Forward, 8= Vehicle: Backward, 9= Vehicle: Stop, 10= Vehicle: Panic
  Stop, 11= Vehicle: Continue Moving, 12= Vehicle: Set Direction and Stop
  FormationDensity = 0; //formation density 0=Dence, 1=Safe, 2=Loose
}
```

7.6.9. Command:Land



Command for “plane” type objects to land at the specified point (point where command icon is placed).



In the given example:

- IN - any event outgoing from a mission that activates Land command.

Yellow line shows direction of landing approach for AI, therefore please be careful when setting this command.

The command works with an object through OL (if a group of objects → OL to group leader).

When creating a mission it's important to consider the following:

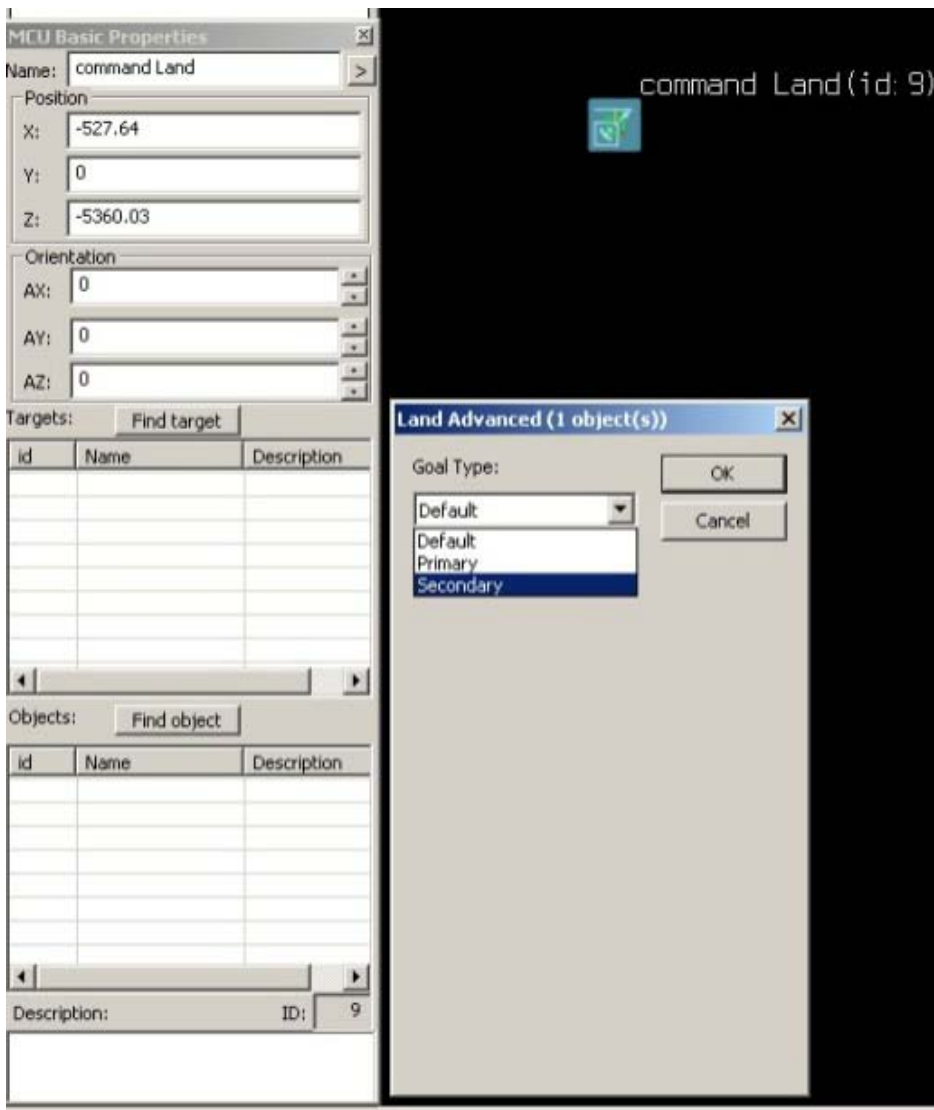
- this command should be given only in the near vicinity of the airfield, because when AI receives Land command it becomes defenseless and if you give this command to it far from an airfield then it won't take any countermeasures when attacked by other mission participants (of the opposite coalition). Basically it'll be a straight flying target.

Plane landing can be made at any point on a map, which surface will allow AI to land safe (if the surface will not be flat or if there will be obstacles on the glidepath then AI may crash).

In case the airfield that Land command points to has Entity then AI will perform landing approach by the scheme assigned to this airfield.

This issue is not critical, but don't forget to lower this command to ground level. If flight route will be displayed in a mission then Land command will be shown at airfield level.

Double-click command's icon to open Land Advanced Properties.



Advanced Properties of the command:

Goal Type: Default, Primary, Secondary. Displays team icon assigned to some object during game. This is made for convenience of visual determination of landing area and doesn't influence mission results..

Default – icon looks like:



Primary and Secondary icons look the same.

Text format of the command:

MCU_CMD_Land // MCU type

```
{
  Index = 12; // MCU ID
  Name = "command Land"; // MCU name (not displayed in GUI)
  Desc = ""; // MCU description (not displayed in GUI)
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
  Objects = []; // objects' ID that this command points at with OL
  XPos = 2036.09; // X axis coordinate, you have to specify this parameter because AI considers position
of this command
  YPos = 111.578; // Y axis coordinate, you have to specify this parameter because AI considers position
of this command
  ZPos = 4283.91; // Z axis coordinate, you have to specify this parameter because AI considers position
of this command
  XOri = 0;
  YOri = 0; // direction of landing approach – considered by AI
  ZOri = 0;
  GoalType = 0; //goal type, affects icon type in GUI
}
```

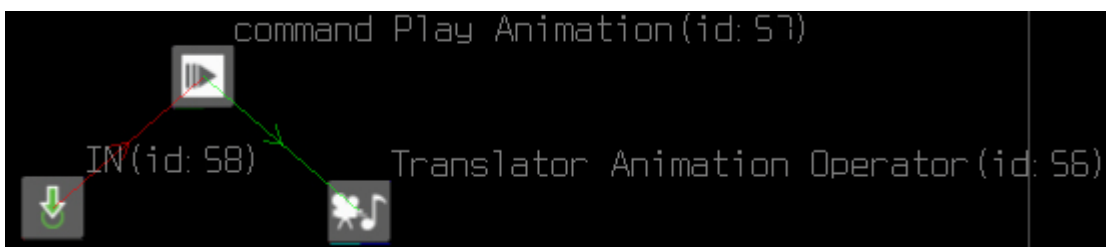
7.6.10. Command:Move

This command is not used.

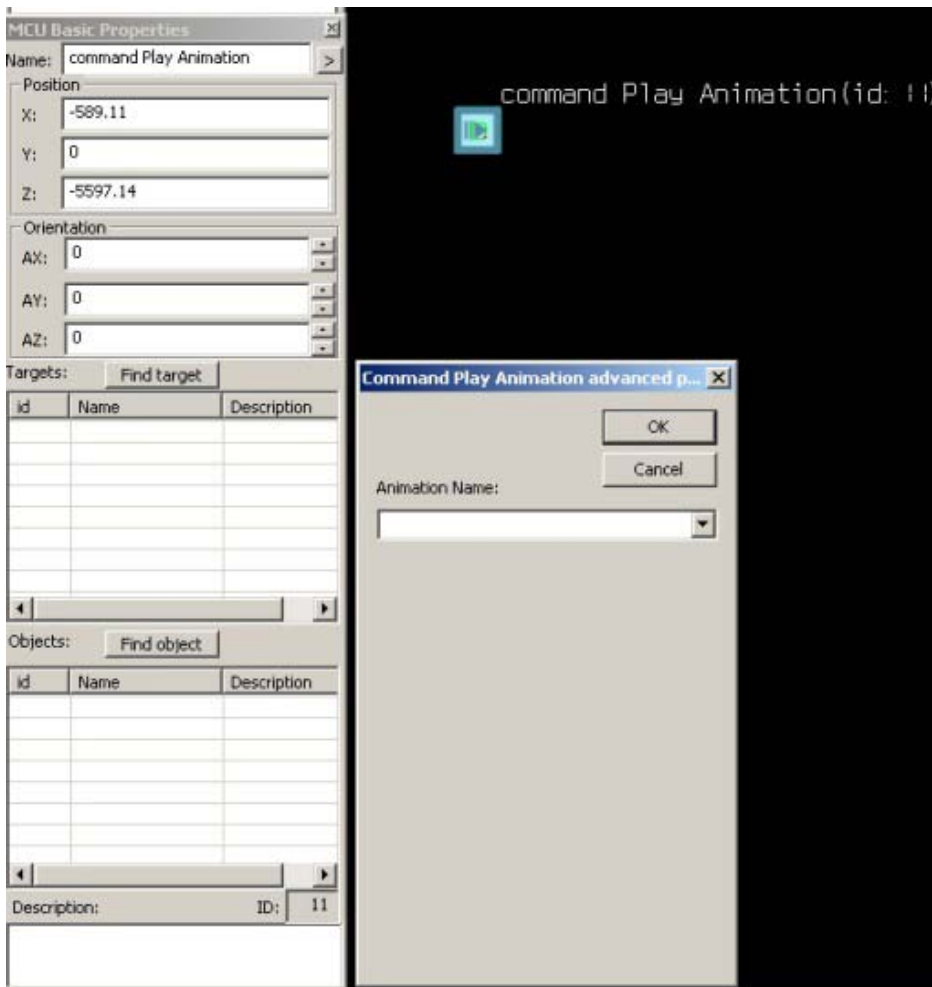
7.6.11. Command:PlayAnimation



This command orders Translator:Animation Operator (see 7.7.1.) to play one of its animations by the given name.



Double-click command's icon to open Play Animation Advanced Properties.



Advanced Properties:

Animation Name field: - animation name that should be played. List of animations is taken from Animation Operator translator that is connected to the command by OL.

Note: this command works only together with Animation Operator translator (see 7.7.1).

Using tips:

To start animation you need to connect Play Animation to Animation Operator with OL and select the type of animation to be played in PlayAnimation. PlayAnimation command is activated by any event from a mission. For more see Sections 9.7 and 9.8.

Text format of the command:

```
MCU_CMD_PlayAnimation // MCU type
```

```
{
```

```
  Index = 14; // MCU ID
```

```
  Name = "command Play Animation"; // MCU name (not displayed in GUI)
```

```
  Desc = ""; // MCU description (not displayed in GUI)
```

```
  Targets = []; // objects' ID that this command points at with TL, this command can't have any TL
```

```
  Objects = []; // objects' ID that this command points at with OL
```

```

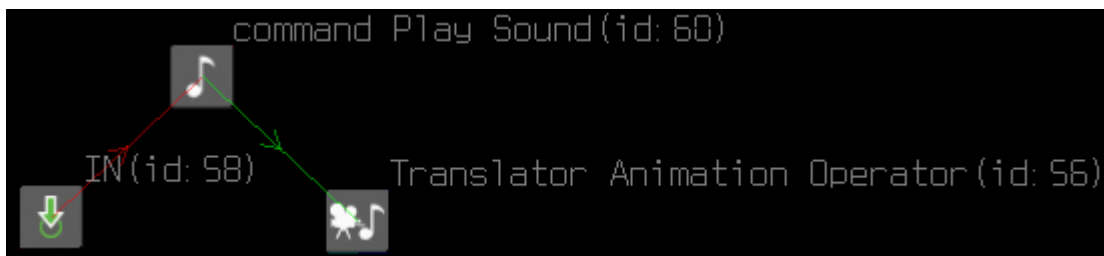
XPos = 2283.32; // X axis coordinates
YPos = 2107.58; // Y axis coordinates
ZPos = 115.08; // Z axis coordinates
ZPos = 4197.53;
XOri = 0; // orientation (doesn't have any functions in a mission)
YOri = 0; // orientation (doesn't have any functions in a mission)
ZOri = 0; // orientation (doesn't have any functions in a mission)
AnimationName = ""; // selected animation to be played
}

```

7.6.12. Command:PlaySound



This command tells Translator:Animation Operator (see 7.7.1.) to play a sound..

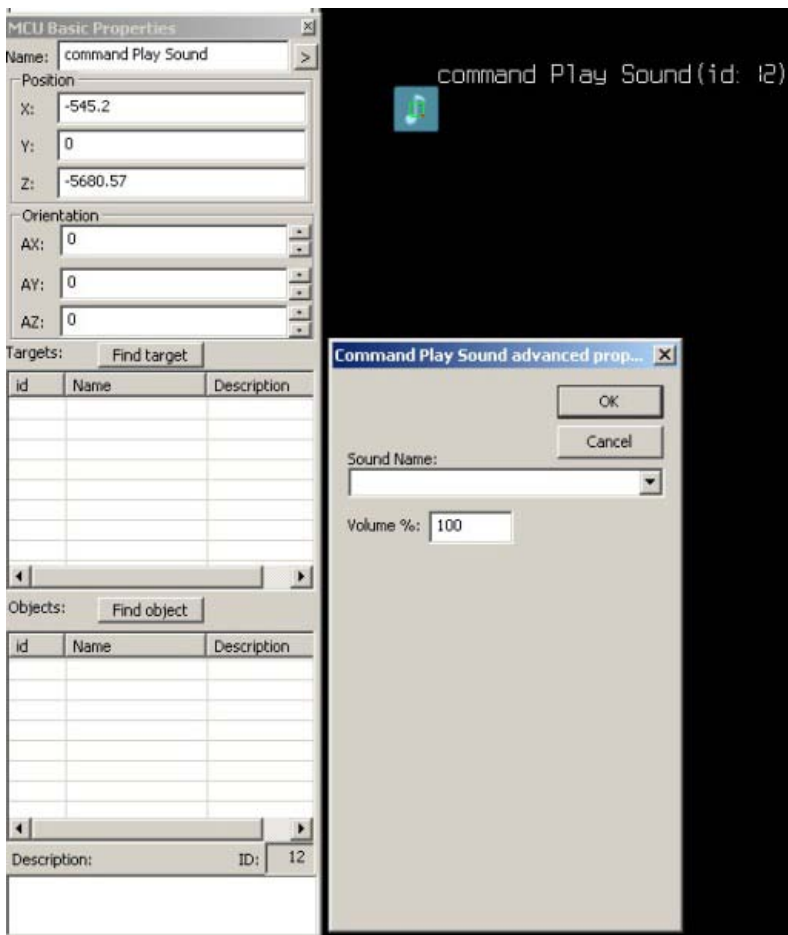


In the given example:

- Translator Animation Operator – in properties of this translator you specify configuration file (sound scheme)/ or sound sample (see 7.7.1);

- IN – in outgoing from a mission event, which starts Play Sound command by TL (or Message).

Double-click command's icon to open Play Sound Advanced Properties.



Advanced Properties:

Sound Name: - name of the sound to be played. List of sounds will correspond to the list in Animation Operator that is connected to the command by OL.

Volume %: - volume of played sound in %.

Note: this command works only with Animation Operator translator (see 7.7.1.).

Using tips:

To start any Sound you need to put Animation Operator translator, connect PlaySound command and Animation Operator translator with OL, set the desired sound in the command or simply start the command if the desired sound is a sample. PlaySound command can be started by any outgoing mission event. For more see 9.7 and 9.8.

Text format of the command:

MCU_CMD_PlaySound // MCU type

{

Index = 15; // MCU ID

Name = "command Play Sound"; // MCU name (not displayed in GUI)

Desc = ""; // MCU description (not displayed in GUI)

Targets = []; // objects' ID that this command points at with TL, this command can't have any TL

Objects = []; // objects' ID that this command points at with OL

XPos = 2012.26; // X axis coordinate

YPos = 107.672; // Y axis coordinate

```

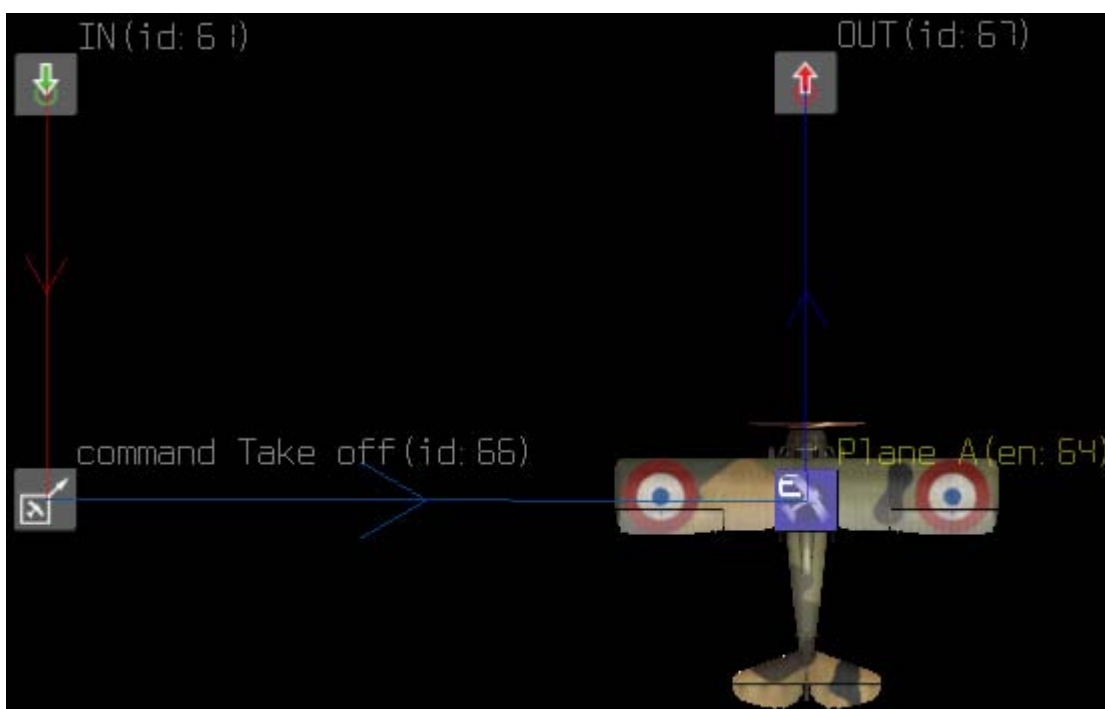
ZPos = 4385.19; // Z axis coordinate
XOri = 0; // orientation (doesn't have any functions in a mission)
YOri = 0; // orientation (doesn't have any functions in a mission)
ZOri = 0; // orientation (doesn't have any functions in a mission)
SoundName = ""; // sound to play. The sound can be not specified if sound type is selected as "config is sample"
Volume = 100; // sound volume
}

```

7.6.13. Command:Take off



This command orders a plane to take-off.



On the given example:

- Take off command is connected to Plane A with OL;
- IN – any outgoing mission event that activates Take off command by TL;
- Plane A – plane to take off. In plane properties you should set On Take off Message (see 2.3.), which has Take off command as source and any following action the plane has to perform after taking off as Target (OUT).

In general for a plane to take off it's enough to simply trigger the Take off command that has this plane as OL. But usually that's not enough because the plane has to do the following action after it has taken off.

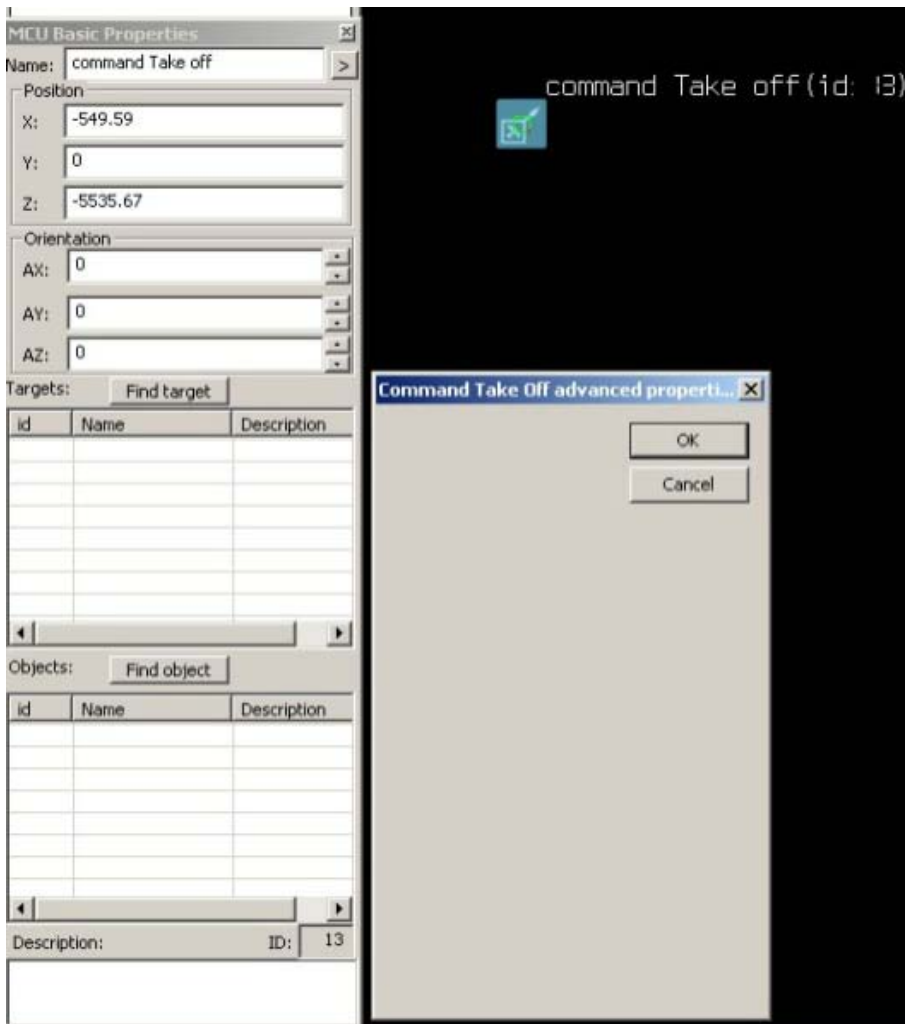
Most often it's enough to specify the Target only (i.e. the action after AI reports about taking off, see 2.3) in On Take off Message but sometimes you need to specify the Source too, i.e. which Take off command exactly (report of execution) set the next action (Target). Time from time it's needed when there are

several Take off commands in a mission (according to mission plot) so that the plane knows the exact sequence of Messages.

Usually the first action specified after take off is proceeding to the first waypoint. So the plane after taking off flies to this waypoint (see 9.1.).

A plane can take off from any point on a map, which surface will allow AI to perform safe take off (if the surface is not flat – AI can crash).

Double-click command's icon to open Take Off Advanced Properties:



There are no additional properties for this command.

Text format of the command:

MCU_CMD_TakeOff // MCU type

{

Index = 16; // MCU ID

Name = "command Take off"; // MCU name (not displayed in GUI)

Desc = ""; // MCU description (not displayed in GUI)

Targets = []; // objects' ID that this command points at with TL, this command can't have any TL

Objects = []; // objects' ID that this command points at with OL

XPos = 2095.66; // X axis coordinates

YPos = 114.842; // Y axis coordinates

ZPos = 4197.53; // Z axis coordinates

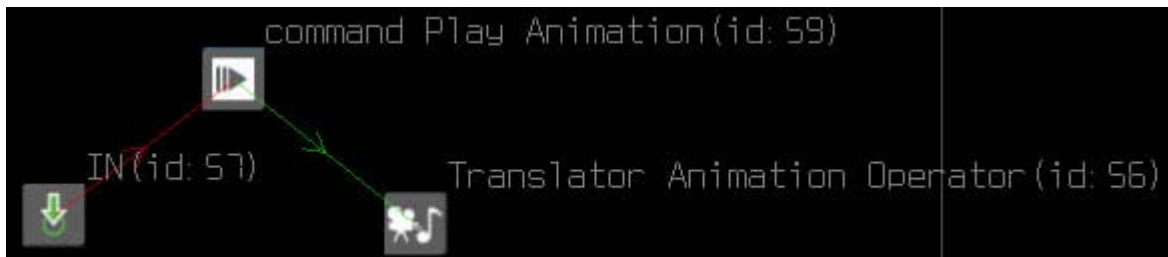
```

XOri = 0; // orientation (doesn't have any functions in a mission)
YOri = 0; // orientation (doesn't have any functions in a mission)
ZOri = 0; // orientation (doesn't have any functions in a mission)
}

```

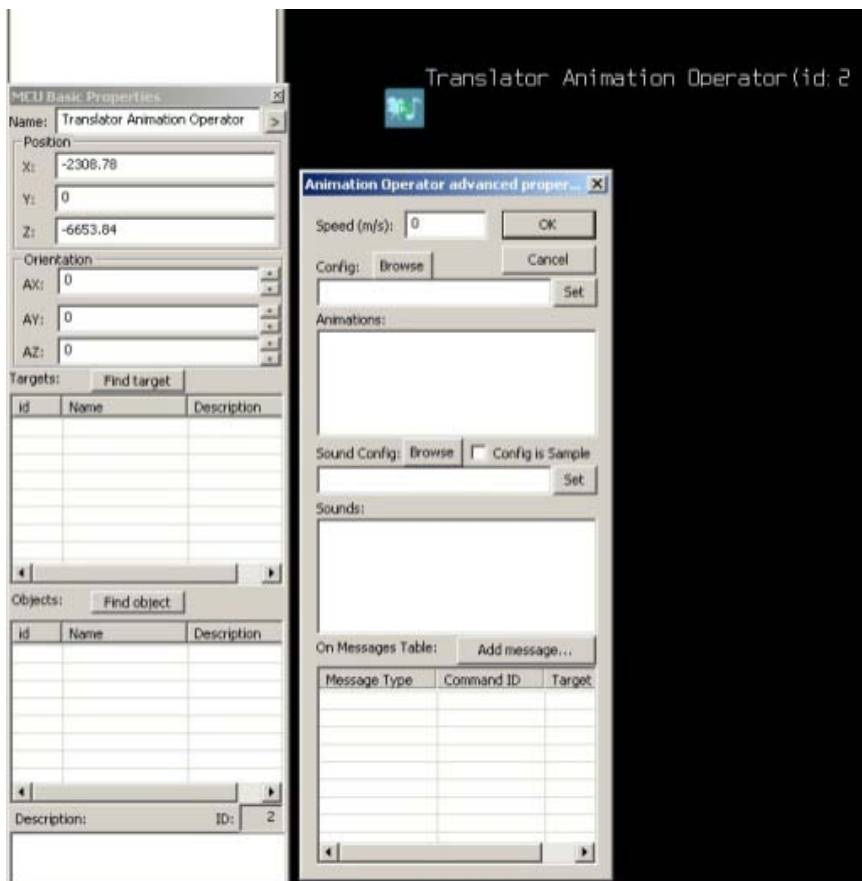
7.7. MCU Translator.

7.7.1. Translator:Animation Operator



In the given example:

- Translator:Animation Operator is connected to Play Animation command (see 7.6.11) by TL;
- IN – any outgoing mission event that starts Play Animation command;



This translator plays animation scenes. If this translator is in a mission then animation is played in idle (initial) state.

You should also remember that if there exists an animation scene (the scene is described in Translator:Animation Operator config) the same scene in idle mode will appear in the mission right from its start. Therefore if you need the scene to appear at a specific moment and not from mission start you need to deactivate the scene by Deactivate trigger. Later when you want this scene to start you need to activate it by Activate trigger (the scene will appear in game world) and only after that you should start animation or sound of the scene.

To play animation use Play Animation command. This command is linked to this trigger by OL. Double-click translator's icon to open Play Animation Advanced Properties.

Advanced Properties:

Speed (m/s): - sets initial speed of scene movement in the game world.

Config: specifies path to the file describing scene animation. The file has *.chr extension.

Set button – confirms file path.

Animations list: in case everything is selected correctly (*.chr file specified correctly) a list of animations will appear in this window that can be played by Play Animation command.

Sound Config: - specifies the path to a script file describing scene sounds. The file has *.lua extension.

In most case when you want to play a sound in a scene it's more convenient to use a sound sample (in *.mp3 format). For this you should select Config is Sample checkbox and specify sound file path (*.mp3). After that you should use PlaySound command to set the moment when the sound will start playing (see 7.6.12).

Sounds list: - if everything is selected correctly there will be a list of sounds in the list field, that can be played by PlaySound command (see 7.6.12.), except for those cases when the sound is sample.

On Message Table field: - allows assigning OnAnimationStop message after animation will stop playing.

Add message... button – sets message type.

There is a way to play the specified scene in the Editor. For that in the already prepared scene (animation is set in "Animation Name" field) right-click Play Animation command, Select Object Menu =>Play.

Text format of the translator:

```
MCU_TR_AnimationOperator //MCU type
{
  Index = 17; //MCU ID
  Name = "Translator Animation Operator"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL
  Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
```

```

XPos = 2089.71; // X axis coordinate
YPos = 114.516; // Y axis coordinate, this one is important to set as this parameter specifies the altitude
where an object or sound will appear
ZPos = 4182.64; // Z axis coordinate
XOri = 0; // turn angle around X axis
YOri = 0; // turn angle around Y axis
ZOri = 0; // turn angle around Z axis
Enabled = 1; //is not used through the Editor interface
Speed = 0;
Config = "";
SoundConfig = ""; //specified sound config
ConfigIsFile = 0; //checkbox, if sound file is separate sound sample (*.mp3)
OnMessages //list of assigned messages
{
}
}

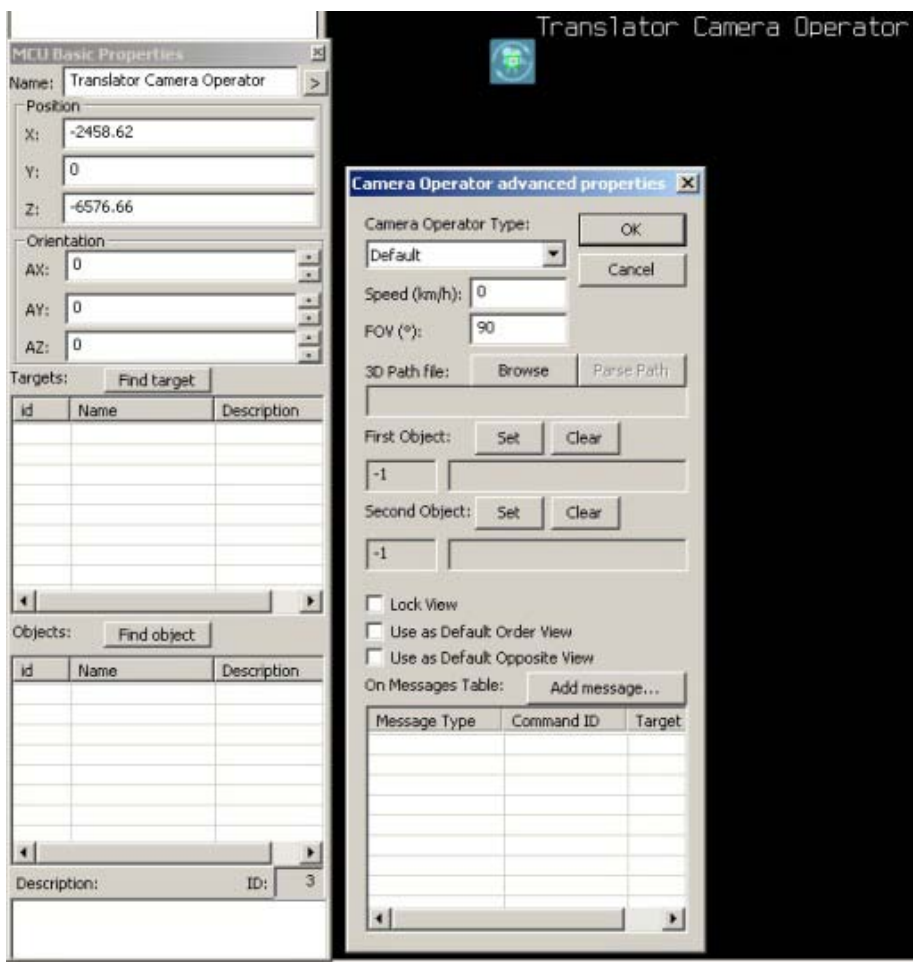
```

7.7.2. Translator:Camera Operator



In the given example:

- Translator: Camera Operator – object-camera in mission, which is activated by any mission event;
- IN – any outgoing mission event that starts Camera Operator by TL (or Message);



This translator allows placing a camera in the game world. Double-click the translator's icon to open Camera Operator Advanced Properties.

Advanced Properties:

Camera Operator Type list: - select camera type.

Camera types:

Default – the view during mission start or the view from cockpit. Also used as default camera after mission start (see below).

Cockpit – N/A

External Free – camera that looks at a plane (or an object) selected in First Object group. Doesn't depend on its orientation in space.

External Fixed – camera that looks at a plane (or an object) selected in First Object group. This camera is fixed at a certain view to a plane and changes position when the plane changes position.

Flyby – camera that show a plane (or object) selected in First Object group flying by.

External Chase – camera that has 2 objects in its view angle, selected in First Object and Second Object groups.

Free – camera that can be freely moved by a player.

Self – camera that points at the place specified in its basic properties or a camera that moves along CameraPoint or according to 3D path script. This camera is a first person view camera.

Padlock – tracking camera, that follows the object specified in Second Object group from the Camera Operator position. At this the translator has to be specified in First Object group.

Speed field– camera movement speed in km/h.

FOV (°) field: - camera view angle in degrees.

3D Path file: - possibility to “attach” the camera trajectory that was unloaded from the 3D Editor (*.cam files). For more see 9.7 and 9.8.

First Object group: sets “first object” for the camera. Set button brings a list of translators to select from. Clear button cancels translator selection.

Second Object group: sets “second object” for the camera. Set button brings a list of translators to select from. Clear button cancels translator selection.

Lock View checkbox – when checked disables the possibility to switch to another camera. If a camera with such setting is activated in a mission then the player won’t be able to switch to another camera until the next camera is activated.

Use as Default Order View checkbox – sets the camera as start up camera for players of Order Coalition. If there are several such cameras then start up camera is selected randomly from them at mission start.

Use as Default Opposite View checkbox - sets the camera as start up camera for players of Opposite Coalition. If there are several such cameras then start up camera is selected randomly from them at mission start.

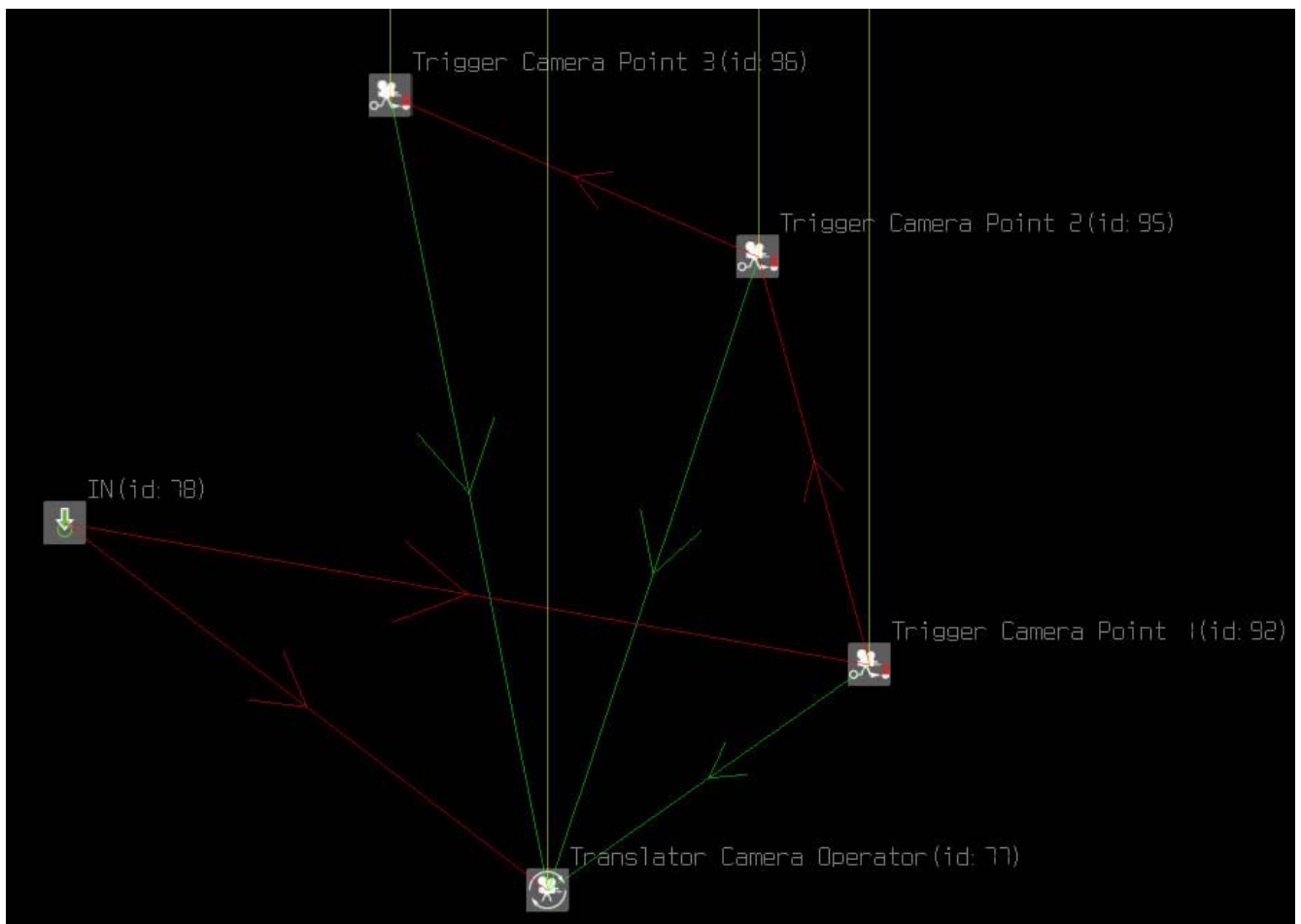
In case there are several cameras of Default type you can switch between them in mission with F12 key (F12 – order, ctrl+F12 – opposite),

If you select Translator:Camera Operator you will be able to see camera view angle (view angle will be shown depending on the FOV set in properties) and view axis.

Also if you need to set camera frame precisely (for example in case of Self camera, i.e. first person view camera) then select the camera, right-click on Select Object Menu => Look from This Object. After you’ve set the desired frame of a camera right-click the Editor’s working space Select Object Menu => Copy Camera Position and Orientation to Object. After that camera position will be copied to the translator.



In cases when Translator:Camera Operator is used together with Trigger:Camera point (see 7.8.2.), i.e. camera will have to move along the trajectory specified by Camera point triggers, there is a possibility to place Camera points automatically.



For this select Translator:Camera Operator, hold left Shift and place the waypoints.

Don't forget to specify FOV for the camera, since Trigger:Camera point will automatically set properties for the waypoints by reading them from Translator:Camera Operator. Camera speed in Camera point will be taken from the first waypoint of the camera.

You should remember that if according to mission plot you need to start the camera going through waypoints instantly then Translator:Camera Operator can't "start" Trigger:Camera point, since the translator doesn't have any outgoing TL itself.

That's why an incoming message (IN) should simultaneously start the operator itself and its first waypoint. With this the camera will be moving from the operator initial position to the specified point with the speed specified in the first camera waypoint.

When camera will be moving its speed and orientation will be changing smoothly from point to point according to the settings and positions of the points.

Text format of the translator:

```
MCU_TR_CameraOperator //MCU type
{
    Index = 18; //MCU ID
    Name = "Translator Camera Operator"; //MCU name (not displayed in GUI)
    Desc = ""; //MCU description (not displayed in GUI)
    Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL
    Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
    XPos = 2012.26; // X axis coordinate, is set in the Editor – where the camera looks
    YPos = 110.272; // Y axis coordinate (altitude), is set in the Editor – where the camera looks
    ZPos = 4069.45; // Z axis coordinate, is set in the Editor – where the camera looks
    XOri = 0; // is set in the Editor – camera frame tilt (bank)
    YOri = 0; // is set in the Editor – camera turn angle in horizontal plane
    ZOri = 0; // is set in the Editor – camera turn angle in vertical plane
    Enabled = 1; //is not used through the Editor's interface
    Speed = 0; //camera initial speed
    FOV = 90; //camera view angle
    FirstId = -1; //ID of the first object that the camera refers to (-1 – object not assigned)
    SecondId = -1; //ID of the second object that the camera refers to (-1 – object not assigned)
    LockView = 0; //switching from the camera is blocked
    CameraType = 0; //camera type
    DefOrderView = 0; //is used as default camera for Order
    DefOppositeView = 0; // is used as default camera for Opposite
    Config = ""; //camera config file name (*.cam)
    OnMessages //list of assigned messages
    {
    }
}
```

7.7.3. Translator:Icon

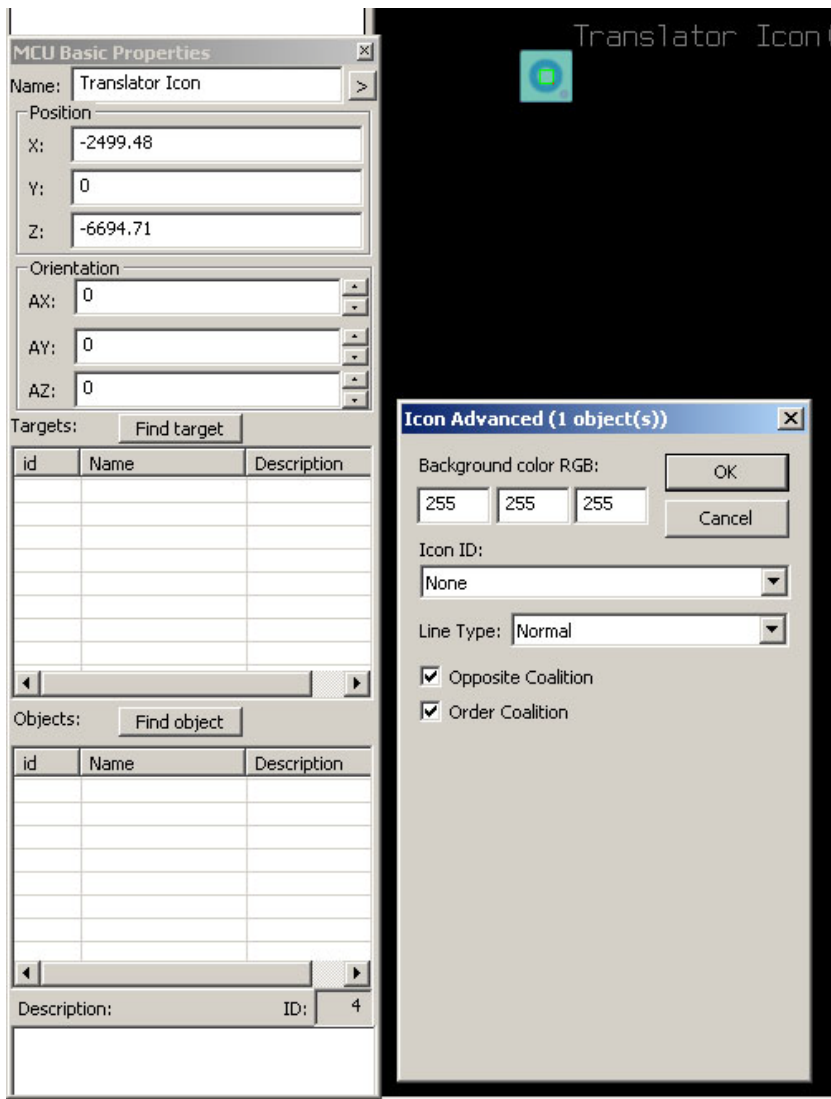


Translator:Icon is used for visualization of flight map.

If you need to draw a flight route, attack area or patrol area on the map you should do it with this translator, since such mission elements as Trigger:Waypoint or other won't be displayed on the map on their own.

This mechanism is aimed at the following targets:

- mission designer can visualize any element of a mission on the mission map;
- ROF visualization system will take all names and descriptions of a map to localization files (see 5.2.);
- possibility to show the route individually for each side (in multiplayer missions);
- possibility to visualize a map differently (color of lines, their thickness, icon types).



Advanced Properties:

Background color field: - defines the color of route line outgoing from an icon on the flight map in standard RGB palette. Sets intensity of red, green and blue components of a line color.

For the lines to be displayed icons have to be connected by TL.

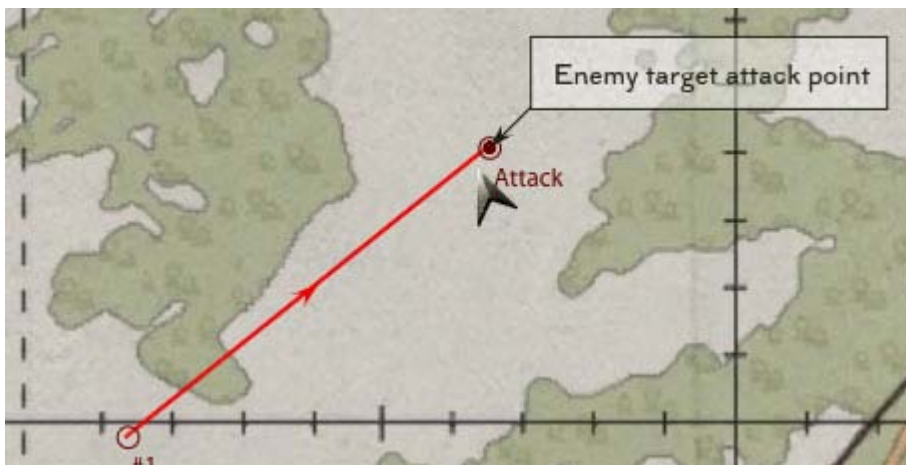
If when setting an icon on the map you enter some text into Name field then it'll be displayed on the map. For example: «№1» or «Attack position».

If you enter text into Description field this text will be displayed when pointing a mouse over this point on the flight map. For example, if you drag a mouse over the «№1» or «Attack» points then a pop-up window (tooltip) will appear on the map where you can type full description of the point. For example «Waypoint №1» or «Enemy target attack point» respectively.

Editor view:



Flight map view (GUI):



Mission designer can also make various shapes with this translator. To do that select icon type as “None”. For example:

Editor view:

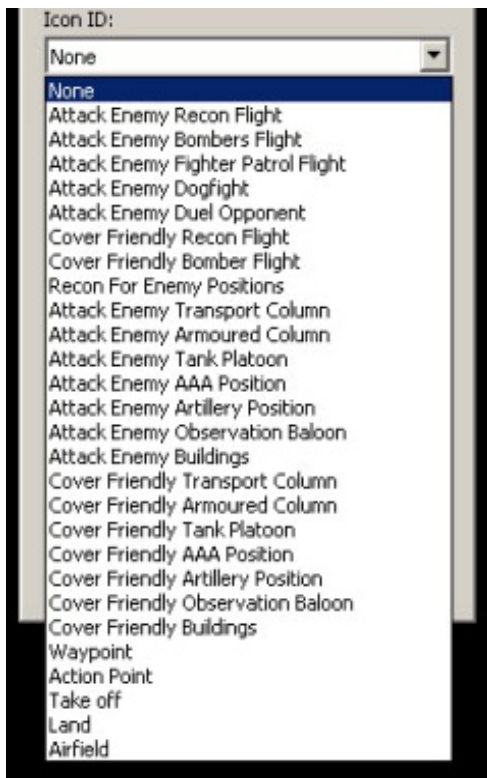


Flight map view (GUI):



An aerial photograph of a forested area with a network of red lines connecting yellow circular nodes. The nodes are labeled with IDs: (id: 3), (id: 4), (id: 5), (id: 6), (id: 7), (id: 8), (id: 9), (id: 10), (id: 11), (id: 12), (id: 13), (id: 14), and (id: 15). The red lines represent a network structure, possibly a road network or a data flow network, connecting the nodes across the landscape. The background is a satellite-style image showing green fields and dark forest areas.

Icon ID field: - specifies icon type on the map.



Types:

None – there is no icon and its visualization on the map;



Attack Enemy Recon Flight – attack enemy recons;



Attack Enemy Bombers Flight – attack enemy bombers;



Attack Enemy Fighter Patrol Flight – attack enemy fighters;



Attack Enemy Dogfight – assist friendly planes in dogfight with enemy fighters;



Attack Enemy Duel Opponent – dogfight - duel;



Cover Friendly Recon Flight – cover friendly recons;



Cover Friendly Bomber Flight – cover friendly bombers;



Recon For Enemy Position – enemy positions recon area;



Attack Enemy Transport Column – attack enemy transport column;



Attack Enemy Armoured Column – attack enemy armored column;



Attack Enemy Tank Platoon – attack enemy tanks;



Attack Enemy AAA Position – attack enemy AAA positions;



Attack Enemy Artillery Position – attack enemy artillery positions;



Attack Enemy Observation Balloon – attack enemy observation balloon;



Attack Enemy Buildings – attack enemy buildings;



Cover Friendly Transport Column – cover friendly transport column;



Cover Friendly Armoured Column – cover friendly armored column;



Cover Friendly Tank Platoon – cover friendly tanks;



Cover Friendly AAA Position – cover friendly AAA positions;



Cover Friendly Artillery Position – cover friendly artillery positions;



Cover Friendly Observation Balloon – cover friendly observation balloon;



Cover Friendly Buildings – cover friendly buildings;



Waypoint – flight route waypoint;



Action point – point of main action on the flight route;

Take off - ;



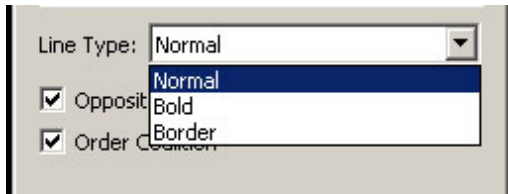
Land – landing point;



Airfield - airfield.

Line Type: - determines type of lines on the map of the flight route.

Types:

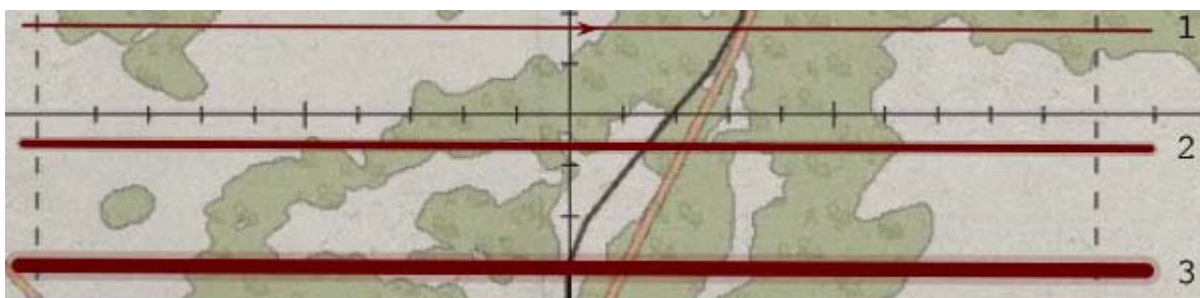


Normal (1) – thin line with an arrow pointing the direction of flight. An arrow pointing the direction “from” and “to” will be displayed, which is usually used for visualizing of flight route;

Bold (2)– thick line;

Border (3)– thick line with a border.

View of lines of different type on the flight map in the game:



Opposite Coalition and Order Coalition checkboxes: - when checked they specify the side which the icon on the map belongs to and also the side that will see the route on the map.

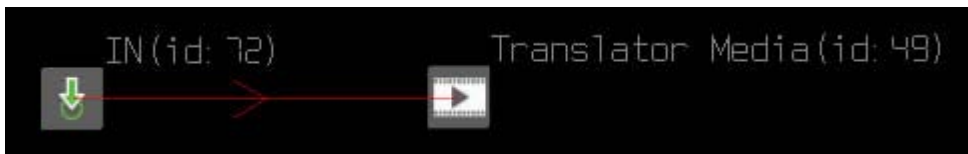
It's possible to place icons that are interconnected by TL automatically. To do this select an icon, hold Shift and place the desired number of icons onto a map.

Text format of the translator:

```
MCU_Icon //MCU type
{
  Index = 19; //MCU ID
  Targets = []; // Objects' ID that this translator points at with OL
  Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
  XPos = 2045.03; // X axis coordinate
  YPos = 112.067; // Y axis coordinate
  ZPos = 4283.91; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Enabled = 1; //is not used through the Editor's interface
  LCName = 3; //index of localization line in localization file of the mission. Will be displayed in GUI as
icon name
  LCDesc = 4; //index of localization line in localization file of the mission. Will be displayed in GUI as
icon description (window will pop-up when you move mouse over it)
  OppositeAirplanes = 1; //will be visible only for Order team
```

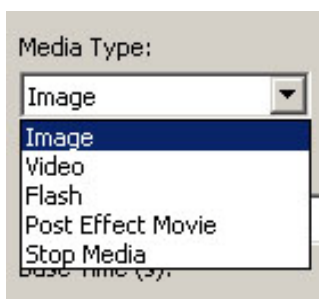
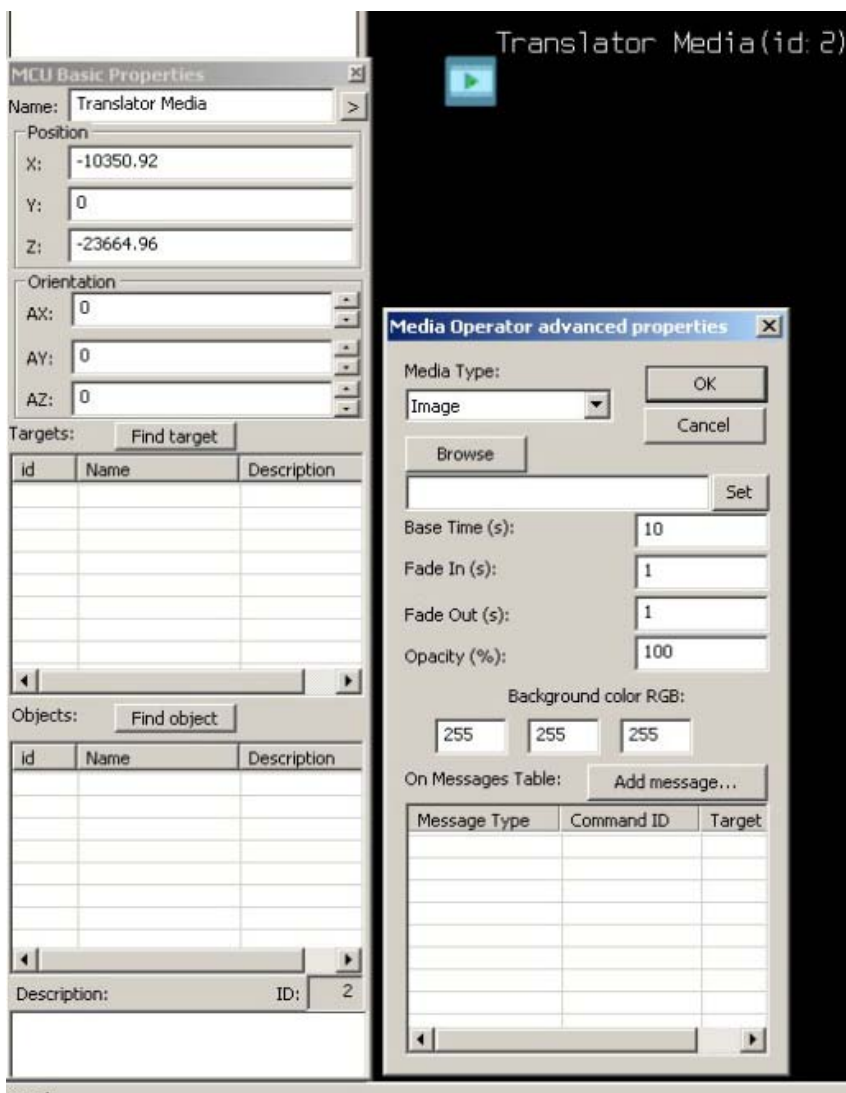
```
OrderAirplanes = 1; // will be visible only for Opposite team  
IconId = 0; //icon type  
RColor = 255; //red color component  
GColor = 255; // green color component  
BColor = 255; // blue color component  
LineType = 0; //line type  
}
```

7.7.4. Translator:Media



In the given example:

- Translator: Media – media insertion into a mission;
- IN – any incoming from a mission event that activates Translator: Media by TL (or Message);



Various types of Media: graphic, video or flash insertions into a mission.

Double-click translator icon to open Media Advanced Properties.

Advanced Properties:

Media Type list: available media types to be played in the mission.

Image – a picture (supports *.jpg, *.dds files);

Video – a video clip, supports blink video (*.bik);

Flash – flash inclusion (*.swf);

Post Effect Movie – apply post-effect to a given picture. “Old movie” effect is applied to the screen;

Stop Media – stop playing any Media.

Browse button – specify the path to the Media file.

Set button – confirm Media selection.

Base Time (s) field: - time that the Media is displayed on the screen, excluding time of emersion and gradual disappearance of the picture.

Fade In (s) field: - time of gradual emersion of the Media, i.e. time needed for Media saturation to change from 0 to the value specified in Opacity (%) field.

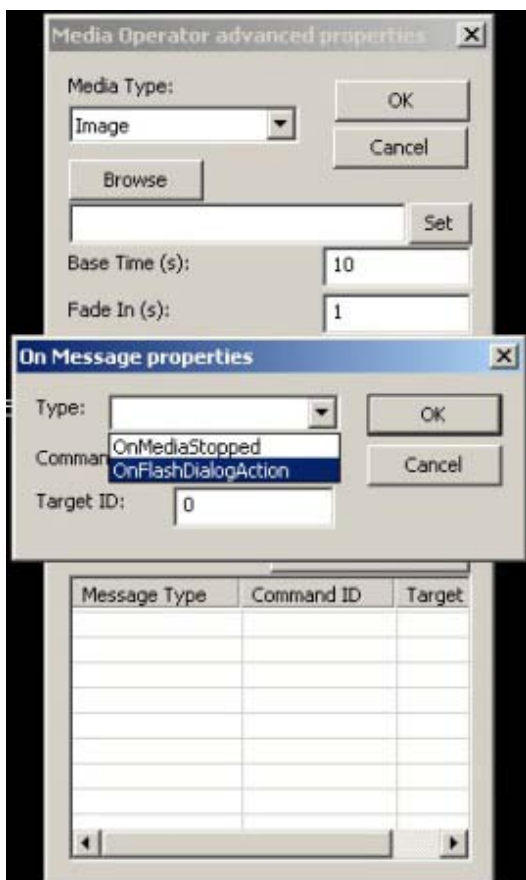
Fade Out (s): - time of gradual disappearance of the Media, i.e. time needed for Media saturation to change from the value specified in Opacity (%) field to 0.

Opacity (%) field: - sets saturation value (opacity) of Media in % during Base Time.

Background color RGB group: - adjusts background color by setting intensity of red, green and blue components of the color.

On Message Table field: - allows setting OnAnimationStop message after the animation finished playing, i.e. after the Media has stopped playing you can send a Message into the Mission.

Add message... button – works only with Stop Media Type and sets message type.



OnMediaStopped – automatically starts the next mission event (through Message) when the Media has stopped playing.

OnFlashDialogAction – «manually» (on pressing a certain button or a field in the mission) starts the next mission event.

For example, you can assign Flash to Skip button and then when the Media will be playing if you click Skip button the translator will launch the next event.

ATTENTION!: in the current Editor version only OnMediaStopped works with functions of OnFlashDialogAction. For example – by clicking Skip in training missions.

ATTENTION!: - after any Translator:Media starts time acceleration won't be available in the mission. Therefore after Translator:Media stops working you need to “pull” Translator:Media with settings: Media Type: Stop Media (in its properties).

ATTENTION!: - at the current stage of project development game mechanism doesn't control the time inside Flash (the whole visualization mechanism is built on this engine), therefore while playing Media the game CAN'T BE PAUSED (this is done as a fuse mechanism).

Therefore if you are using Media right after mission start (and during the start the game pauses in about 1-2 seconds after the mission has loaded) – remember about this issue. For an example of using Media in such a case see Section 8.8 with a detailed example.

Text format of the translator:

MCU_TR_Media //MCU type

```
{
  Index = 20; //MCU ID
  Name = "Translator Media"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL
  Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
  XPos = 2024.18; // X axis coordinate
  YPos = 109.304; // Y axis coordinate
  ZPos = 4355.4; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Enabled = 1; //is not used through the Editor's interface
  Config = ""; //translator config file path
  TotalTime = 10; //total time of translator action (Bas Time)
  ExpandTime = 1; //translator media fade in time (Fade In)
  FadeTime = 1; // translator media fade out time (Fade Out)
  Opacity = 100; //opacity of translator's media
  RColor = 255; //red component of background color
  GColor = 255; // green component of background color
  BColor = 255; // blue component of background color
```

```

MediaType = 0; //media type, 0=Image, 1=Video, 2=Flash, 3=Post Effect Movie, 4=Stop Media
OnMessages //list of assigned messages
{
}
}

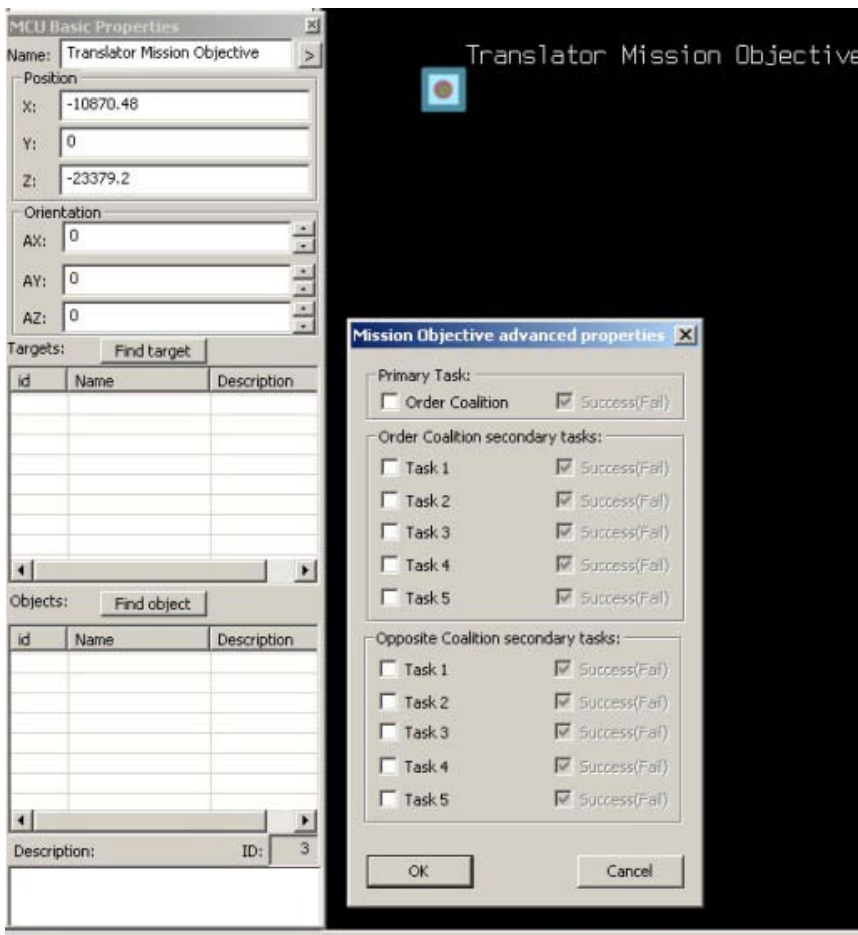
```

7.7.5. Translator:Mission Objective



On the given example:

- Translator:Mission Objective – translator that determines success/fail of mission tasks;
- IN – any incoming mission event that by TL (or Message) activates Translator: Mission Objective.



This translator defines primary (main) or secondary (additional) mission tasks for the coalitions. Double-click translator icon to open Mission Objective Advanced Properties.

Advanced Properties:

To assign the primary task:

Primary Task group: determines the primary task of the coalition (Order) and its result (Success/Fail). Order Coalition checkbox must be turned on all the time! Otherwise the translator won't work.

For example if you select Success/Fail checkbox it will mean that Order Coalition has accomplished the primary task. If you unselect this checkbox it will mean that Order Coalition has failed. By checking the specified conditions mission designer determines what exactly work and in which case.

Groups: Oder Coalition secondary tasks and Opposite Coalition secondary tasks, - are not used in this case.

To assign the secondary task:

Primary Task group: is not used in this case.

Oder Coalition secondary tasks group: defines the secondary tasks(s) for Order side.

Task 1 – 5 checkboxes when selected allow specifying up to 5 secondary tasks;

Success(Fail) checkboxes when selected (Success) in case of successful accomplishment will be written into the stats as accomplished and displayed on the flight map. In unselected state – will be written into the stats as failed and displayed on the flight map.

Opposite Coalition secondary tasks group: defines secondary task(s) for Opposite side.

Task 1 – 5 checkboxes when selected allow specifying up to 5 secondary tasks.

Success(Fail) checkboxes when selected (Success) in case of successful accomplishment will be written into the stats as accomplished and displayed on the flight map. In unselected state – will be written into the stats as failed and displayed on the flight map (see below).

Restriction: you can't use the same translator to define more than one task simultaneously.

Number of secondary tasks can't exceed 5 for each side.

In case at least one Mission Objective Fail has worked – this task will be considered failed not depending on the number of Mission Objective Success that have worked.

There can be any number of Translator:Mission Objective set in a mission – depending on the active branch of the mission plot.

Name field – if you type a name into this field then when translator will work the text will be shown on a map. Also if you enter some text into Description field then when a mouse will be over the name of the accomplished mission on the map (in debriefing) this text will appear in the pop-up window.

This translator is displayed differently in GUI (in the game, on the flight map) depending on the type of the succeeded/failed condition defined in the properties. Moreover it should be (but not necessarily) used together Translator:Icon – if it's a target's icon and by implication the task (target) is approximately at the place of the icon in this case you can combine both translators in the mission. Visually it'll look like an icon with different substrates on the background.

Text format of the translator:

```
MCU_TR_MissionObjective //MCU type
```

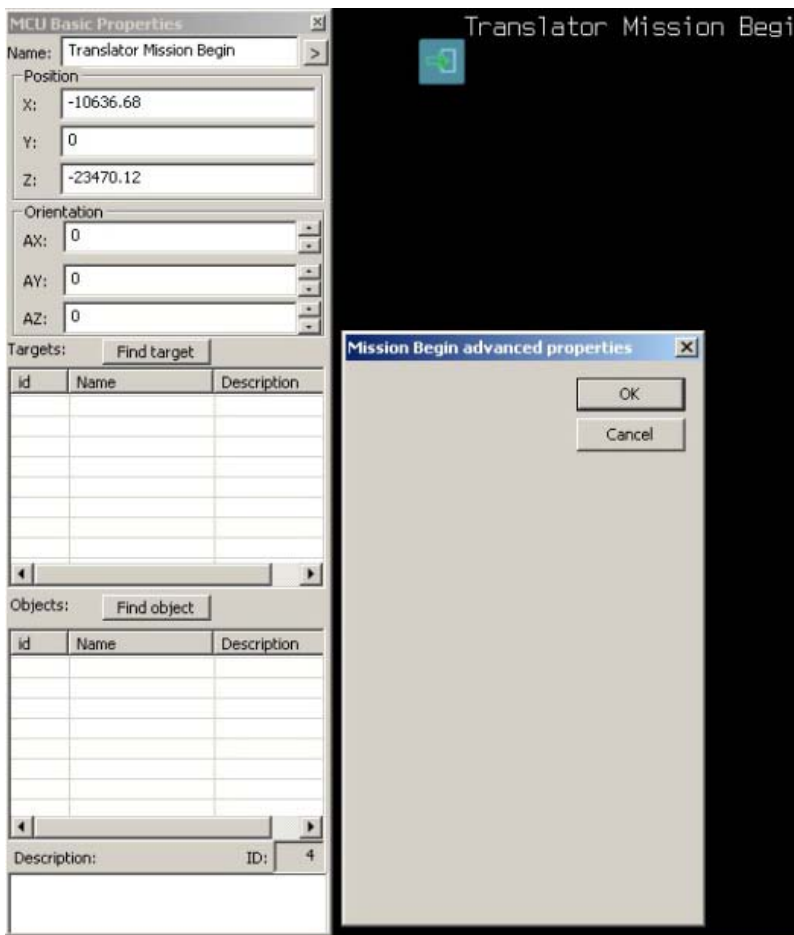
```
{  
  Index = 21; //MCU ID  
  Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL  
  Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL  
  XPos = 2113.54; // X axis coordinate  
  YPos = 116.999; // Y axis coordinate  
  ZPos = 4257.11; // Z axis coordinate  
  XOri = 0; //orientation (doesn't have any functions in a mission)  
  YOri = 0; //orientation (doesn't have any functions in a mission)  
  ZOri = 0; //orientation (doesn't have any functions in a mission)  
  Enabled = 1; //is not used through the Editor's interface  
  LCName = 3; //localization line index in mission localization file, will be displayed in GUI as the name  
of activated task  
  LCDesc = 4; // localization line index in mission localization file, will be displayed in GUI as the  
description of activated task  
  TaskArray //task accomplishment settings array (different combination with different checkboxes  
selection in the translator)  
  {  
  }  
}
```

7.7.6. Translator:Mission begin



In the given example:

- Translator:Mission begin – translator determining the initial action in a mission, in fact – the starter of all events in a mission;
- OUT – any initial event in a mission that is activated by: Translator:Mission begin.



This one starts mission events after it has loaded. Transmits “mission begin” event from the game engine. Double click translator icon to open Mission Begin Advanced Properties.

There can be any number of such translators in a mission. If there are several of them they will all work at mission start.

This feature can be used when importing mission blocks with already set Mission Begin.

Text format of the translator:

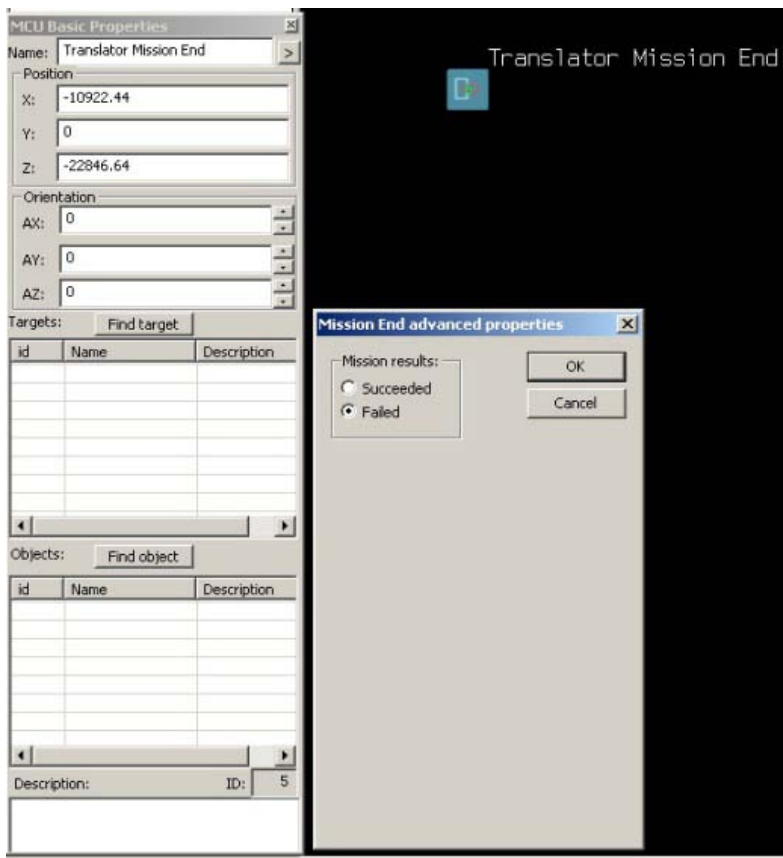
```
MCU_TR_MissionBegin //MCU type
{
    Index = 22; //MCU ID
    Name = "Translator Mission Begin"; //MCU name (not displayed in GUI)
    Desc = ""; //MCU description (not displayed in GUI)
    Targets = []; // Objects' ID that this translator points at with TL
    Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
    XPos = 2101.62; // X axis coordinate
    YPos = 115.08; // Y axis coordinate
    ZPos = 4272; // Z axis coordinate
    XOri = 0; //orientation (doesn't have any functions in a mission)
    YOri = 0; //orientation (doesn't have any functions in a mission)
    ZOri = 0; //orientation (doesn't have any functions in a mission)
    Enabled = 1; //is not used through the Editor's interface
}
```

7.7.7. Translator:Mission End



In the given example:

- Translator: Translator: Mission End – translator defining mission end, switching to mission statistics menu;
- IN – any incoming mission event that by TL (or Message) activates Mission End.



When activated finishes mission mandatory without player interference. Double-click the translator icon to open Mission End Advanced Properties.

Advanced Properties:

Mission Results field: - has 2 settings Succeeded and Failed meaning respectively successful mission accomplishment and failure. In current ROF version this option in Advanced Properties of the translator Mission End DOESN'T WORK.

Text format of the translator:

MCU_TR_MissionEnd //MCU type

```

{
  Index = 23; //MCU ID
  Name = "Translator Mission End"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL
  Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL
  XPos = 2050.98; // X axis coordinate
  YPos = 112.394; // Y axis coordinate
  ZPos = 4209.45; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Enabled = 1; //is not used through the Editor's interface
  Succeeded = 1; //is not used through the Editor's interface, n/a
}

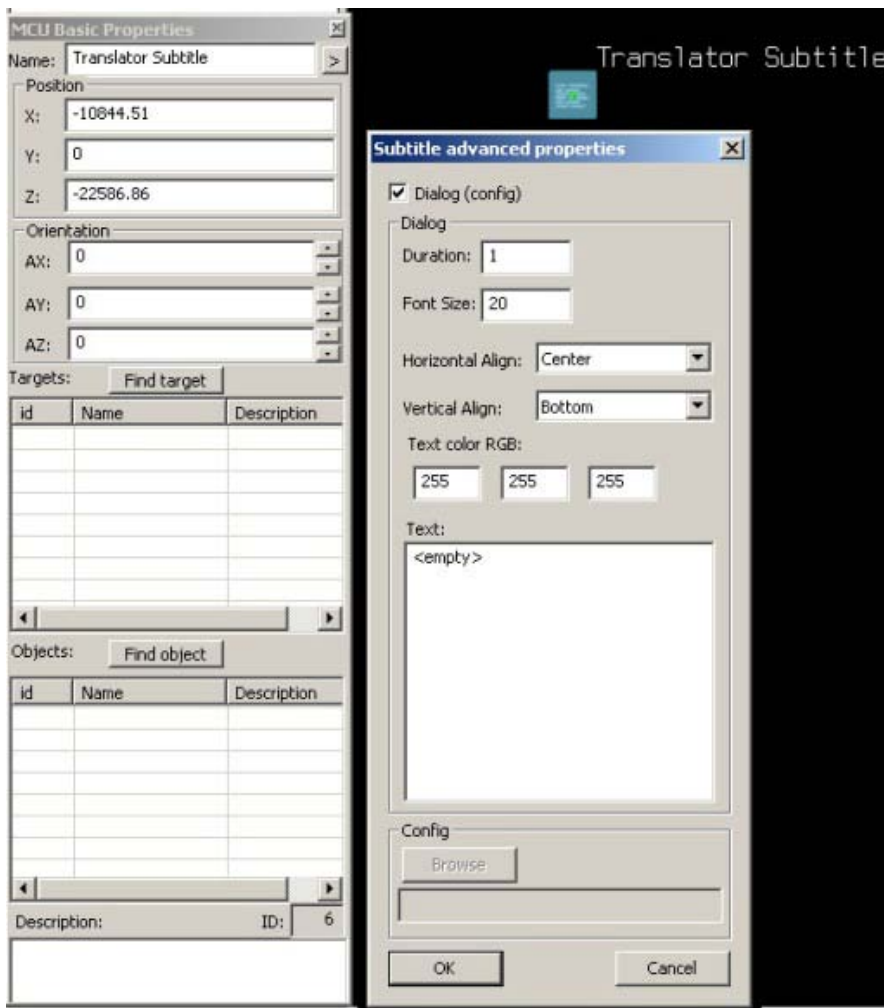
```

7.7.8. Translator:Subtitle



In the given example:

- Translator: Subtitle – inserts subtitles in a mission;
- IN – any incoming mission event that by TL (or Message) activates Translator: Subtitle;



This translator allows displaying subtitles in a mission. Double-click translator icon to open Subtitle Advanced Properties.

Advanced Properties:

Dialog (config) checkbox – must be selected to be able to adjust subtitles (selected by default).

Dialog field.

Duration – time of displaying of a subtitle on user's screen in seconds.

Font Size – subtitle font size.

Horizontal Align – subtitle horizontal alignment (left, center, right).

Vertical Align - subtitle vertical alignment (top, center, bottom).

Text Color RGB – subtitle color in RGB palette, defines red, green, blue components of the color.

Text – enter subtitle text here.

Translator:Subtitle allows using html tags:

 - line break;

<u>...</u> - text underlining;

<i>...</i> - italics;

... - bold.

For example: MISSION DESCRIPTION
<u>Date:</u> 03.09.1918.

In the game window this text will look like:

—

MISSION DESCRIPTION

Date: 03.09.1918.

—

There can be any number of Translator:Subtitle in a mission.

To display a subtitle in a mission you need to activate it by TL.

When Translator:Media of Stop Media type is being activated (see 7.7.4) the current subtitle will also stop.

ATTENTION!: - at the current stage of project development the game mechanism doesn't control time inside Flash (the whole visualization mechanism is built on this language, including subtitles). Therefore if you pause the game during playing subtitles then Subtitles WILL NOT PAUSE and will continue to play independently (according to their internal time).

That's why if you're using Subtitle right after mission start, for example as an introduction text (after loading the game pauses for about 1-2 seconds) – remember about this issue and launch subtitles with timer delay (~3 seconds). As an example of using Subtitle in such a case see Section 8.8.

Text format of the translator:

MCU_TR_Subtitle //MCU type

{

Index = 24; //MCU ID

Name = "Translator Subtitle"; //MCU name (not displayed in GUI)

Desc = ""; //MCU description (not displayed in GUI)

Targets = []; // Objects' ID that this translator points at with TL, this translator can't have any TL

Objects = []; // Objects' ID that this translator points at with OL, this translator can't have any OL

XPos = 2158.22; // X axis coordinate

YPos = 115.53; // Y axis coordinate

ZPos = 4194.55; // Z axis coordinate

XOri = 0; //orientation (doesn't have any functions in a mission)

YOri = 0; //orientation (doesn't have any functions in a mission)

ZOri = 0; //orientation (doesn't have any functions in a mission)

Enabled = 1; //is not used through the Editor's interface

OnMessages //N/A

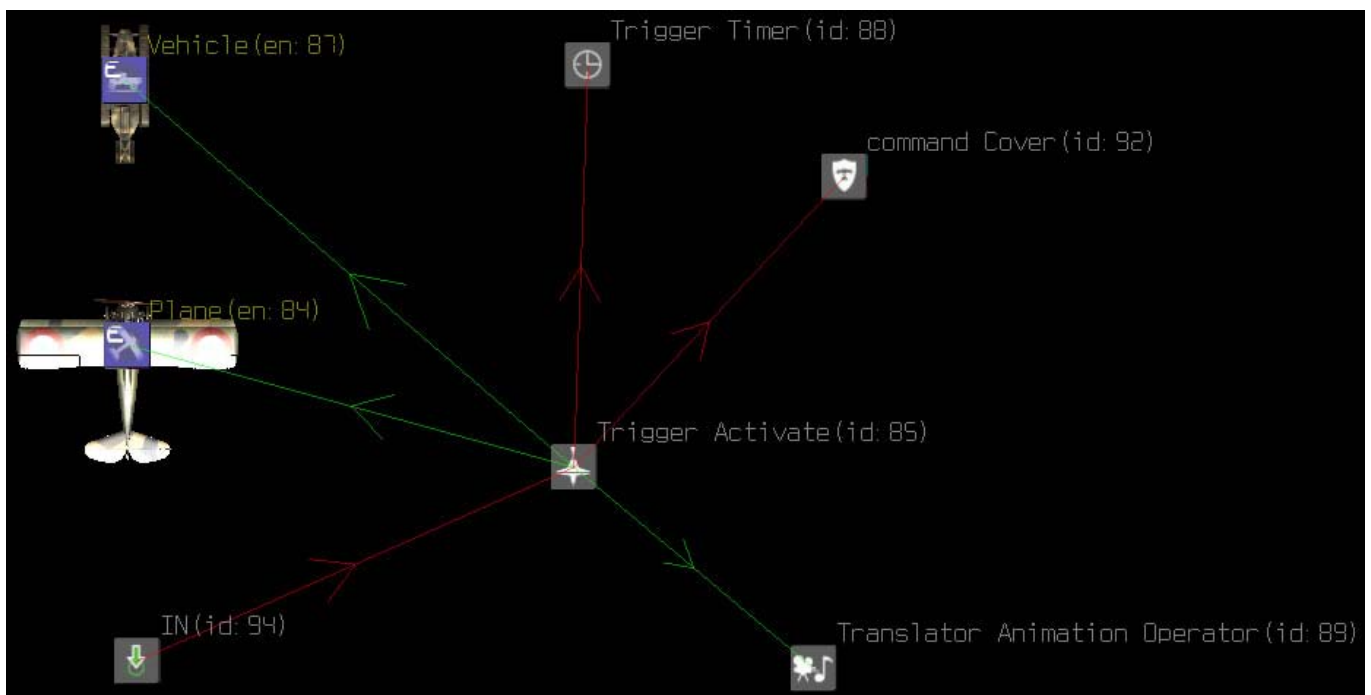
```

{
}
Dialog
{
    Duration = 1; //duration of subtitles
    FontSize = 20; //subtitle font size
    HAlign = 1; // subtitle horizontal alignment 0=left, 1=center, 2=right
    VAlign = 2; // subtitle vertical alignment 0=top, 1=center, 2=bottom
    RColor = 255; //red component of subtitle text color
    GColor = 255; // green component of subtitle text color
    BColor = 255; // blue component of subtitle text color
    LCtext = 3; //line index in localization file
}
}

```

7.8. MCU Trigger.

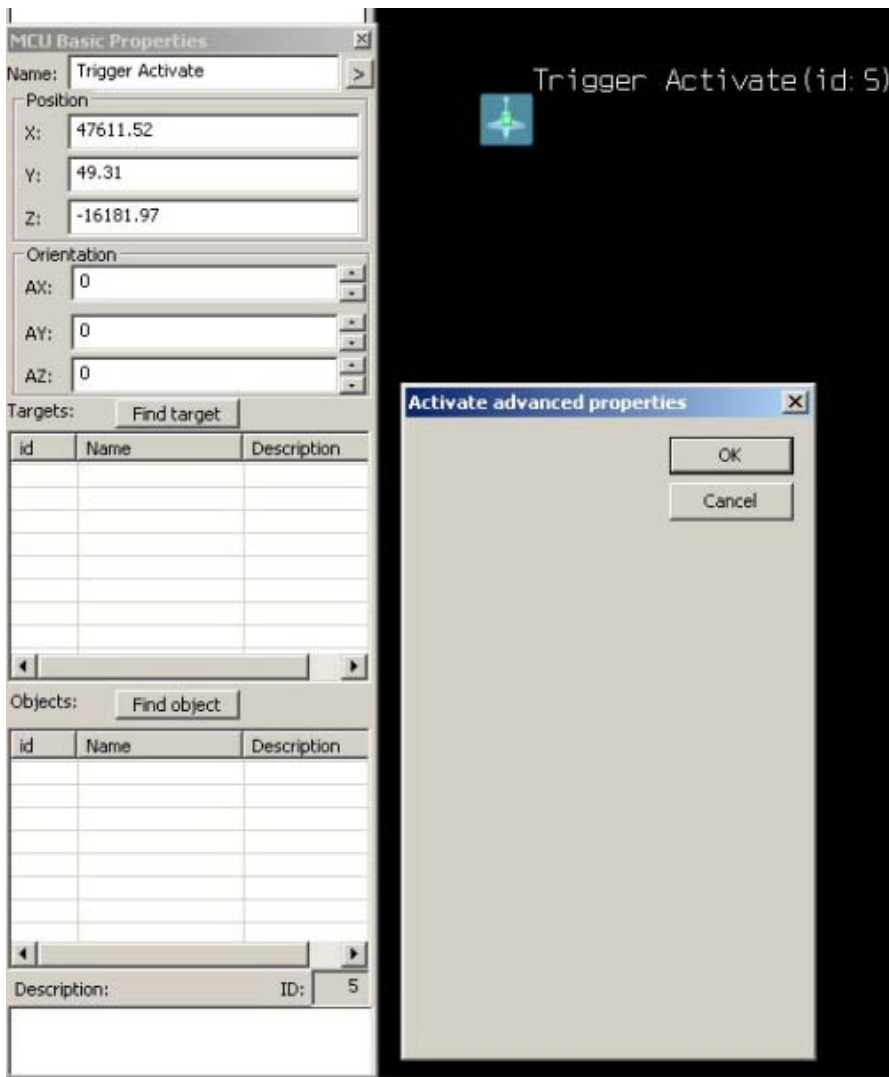
7.8.1. Trigger:Activate



In the given example :

- Trigger:Activate – trigger that activates mission objects if they were previously deactivated or not active (se 7.2.1) by OL (planes/vehicles and translators) and by TL (triggers/commands).
- IN – any incoming mission event that activates Trigger:Activate.

It's important to remember that an object that wasn't activated doesn't respond to commands from the mission and doesn't interact with mission objects.



This trigger activates objects or MCUs that were previously deactivated in a mission.

Double-click trigger icon to open Activate Advanced Properties (none).

This trigger is often used together with Deactivate trigger (see 7.8.5.).

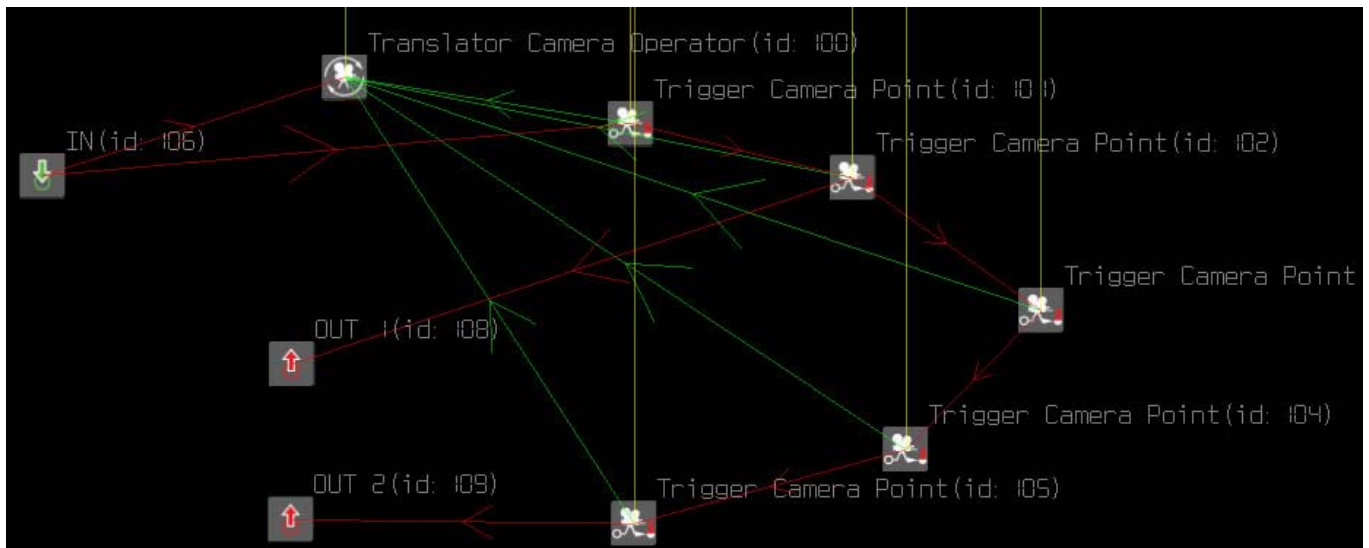
Text format of the trigger:

```
MCU_Activate //MCU type
{
  Index = 25; //MCU ID
  Name = "Trigger Activate"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2107.58; // X axis coordinate
  YPos = 112.755; // Y axis coordinate
  ZPos = 4292.85; // Z axis coordinate
  XOri = 0;
  YOri = 0;
  ZOri = 0;
}
```

7.8.2. Trigger:Camera point



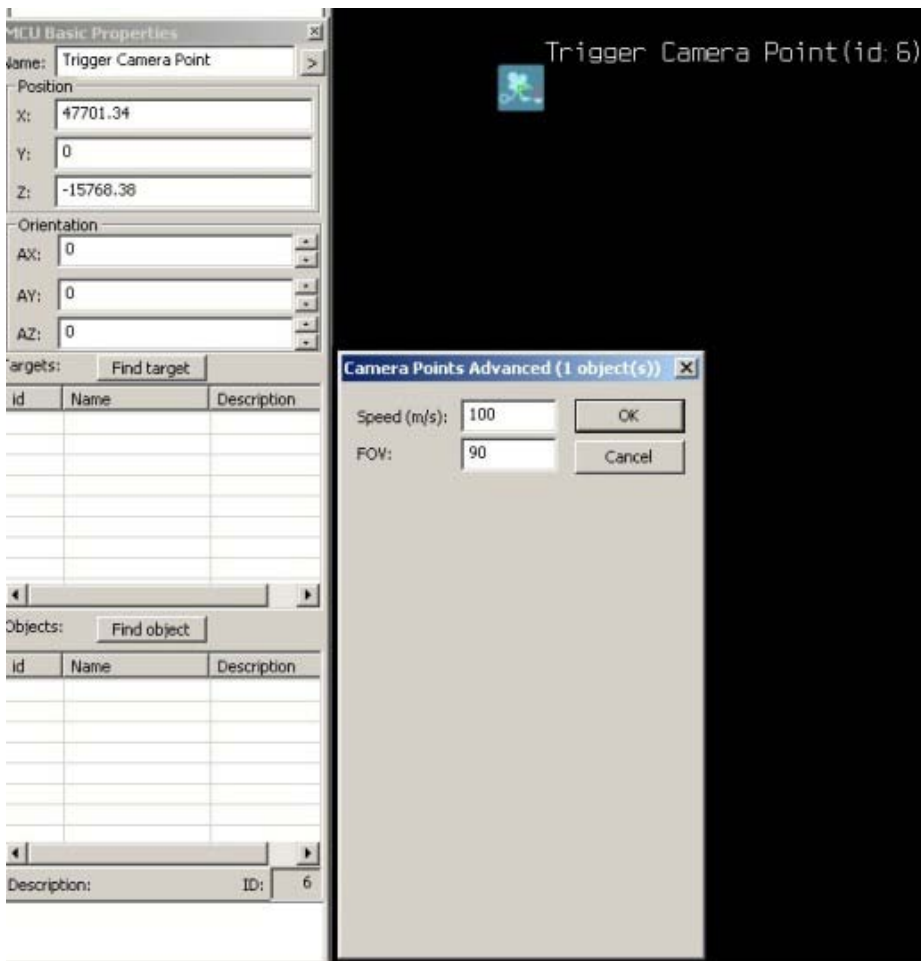
See Section 7.7.2.



In the given example :

- Trigger:Camera point – camera waypoint trigger. By itself Trigger:Camera point has no sense as it in fact is a slave object to Translator:Camera Operator and thus is tied to it by OL. I.e. Camera Operator sets camera properties and its type in the mission and Camera point sets camera movement scheme;
- IN – any incoming mission event starting Trigger:Camera point and Translator:Camera Operator;
- OUT 1 & OUT 2 - Camera point is a trigger therefore when activated it can “pull” by TL other mission objects.

During camera movement through waypoints FOV will smoothly change from point to point depending on the setting at each Camera Point.



This trigger is activated by a camera (camera as a moving object) and at the same time gives a command to the camera to move to the specified point with specified speed and also to take the position and orientation specified in Basic Properties of the trigger.

Double-click trigger icon to open Camera Points Advanced Properties.

Camera Point Advanced Properties:

Speed (m/s) field: - specifies the speed (in meters per second) the camera has to be moving at to the point.

FOV field: - camera field of vision.

Camera speed must not exceed 200 m/s!

Text format of the trigger:

MCU_CameraPoint //MCU type

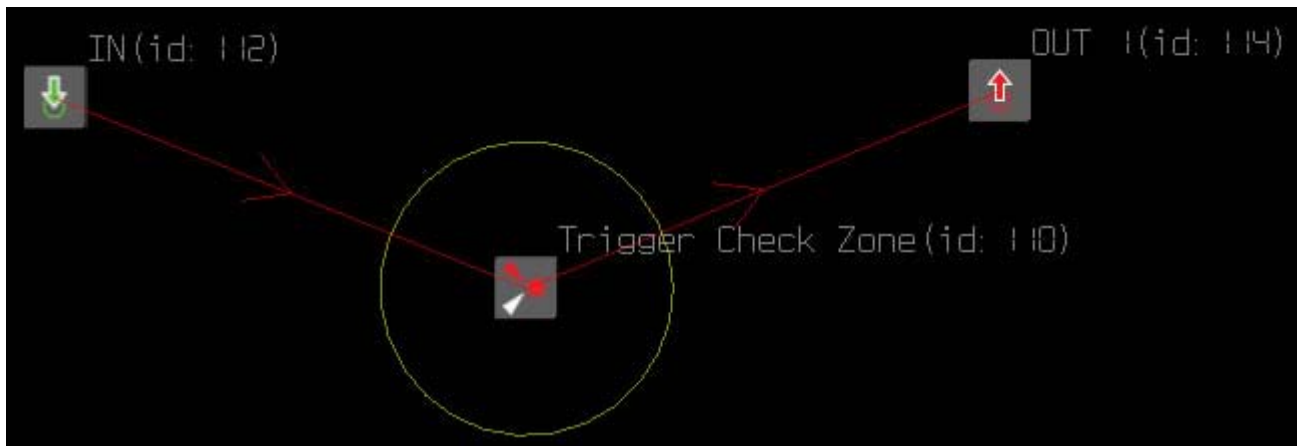
```
{
  Index = 26; //MCU ID
  Name = "Trigger Camera Point"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2104.6; // X axis coordinate
  YPos = 113.408; // Y axis coordinate
  ZPos = 4367.32; // Z axis coordinate
  XOri = 0; // orientation by X axis – camera tilt (bank) in horizontal plain
```

```

YOri = 0; // orientation by Y axis – camera turn angle in horizontal plane
ZOri = 0; // orientation by Z axis – camera tilt in vertical plain
Speed = 100; //camera speed
FOV = 90; //camera FOV angle
}

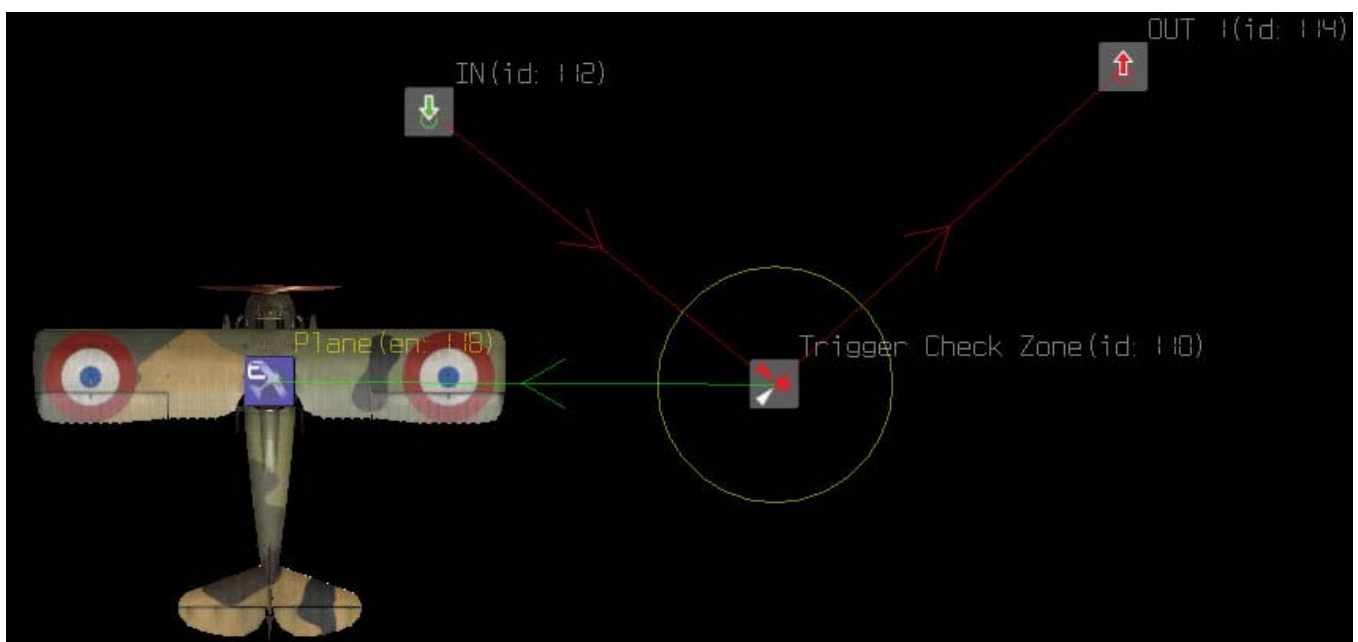
```

7.8.3. Trigger:Check zone



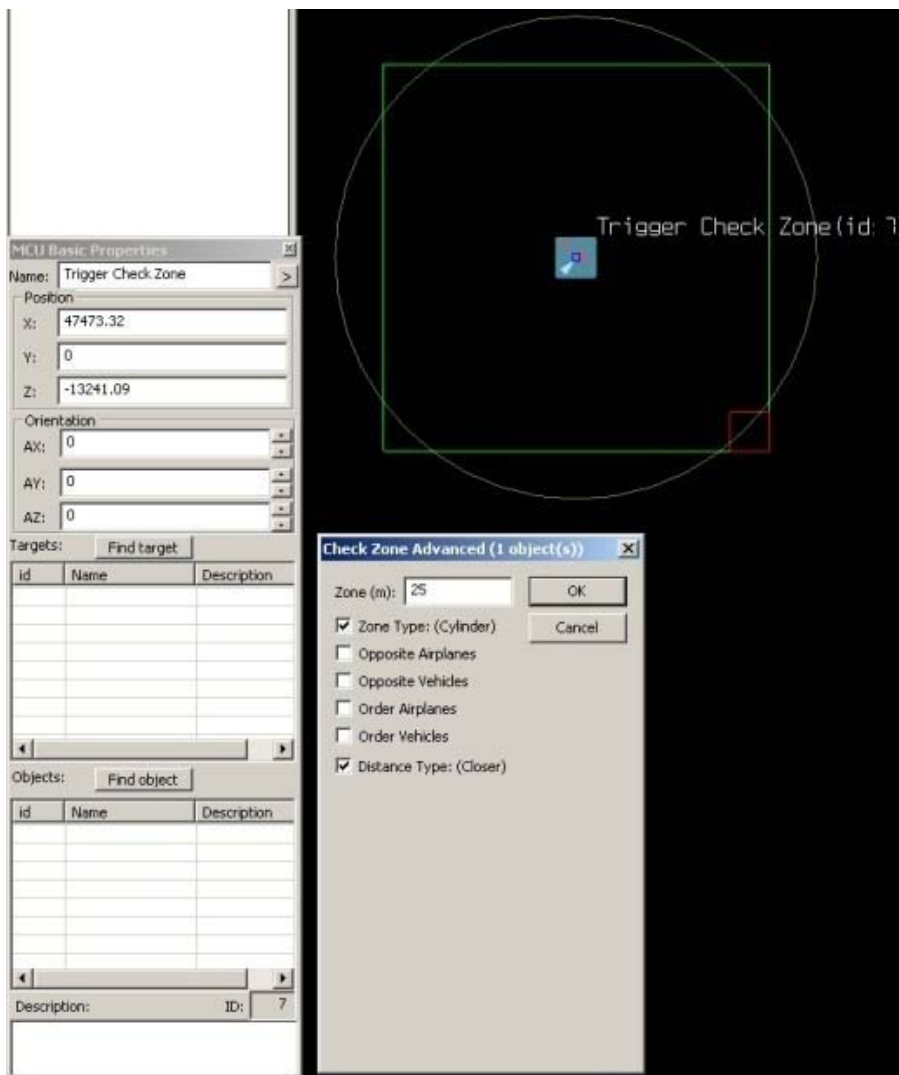
In the given example (triggering conditions are set in Properties):

- Trigger: Check zone – this trigger checks if the objects of given type are in or out of the specified zone (the zone is marked by a yellow circle);
- IN - any incoming mission event that by TL (or Message) activates Trigger:Check zone;
- OUT – any mission event that is activated by Check zone trigger after it has been enabled.



In the given example (triggering condition by a linked object):

- Trigger: Check zone – the trigger checks if the object (here – a plane) is in or out of the zone. The trigger refers to the plane by OL and is activated only by it;
- IN - any incoming mission event that by TL (or Message) activates Trigger:Check zone;
- OUT – any mission event that is activated by Check zone trigger after it has been enabled



There are two modes: checking objects presence inside the given action radius (close) and outside of it (further).

Double-click trigger icon to open Check Zone Advanced Properties.

Advanced Properties:

Zone (m) field: - specifies the radius of the zone where trigger works, in meters.

Zone Type:Cylinder (Sphere) checkbox – sets zone shape as a cylinder (when enabled) or a sphere (when disabled). When specifying Sphere option, don't forget to set trigger altitude (YPos).

Opposite Airplanes checkbox – activating (on)/not activating (off) the trigger for Opposite Coalition planes.

Opposite Vehicles checkbox – activating (on)/not activating (off) the trigger for Opposite Coalition vehicles.

Order Airplanes checkbox – activating (on)/not activating (off) the trigger for Order Coalition planes.

Order Vehicles checkbox – activating (on)/not activating (off) the trigger for Order Coalition vehicles.

Distance Type:(Closer) checkbox – when enabled this option means that the trigger will get activated when an object approaches the center of the zone to a distance, specified in Distance field.

Distance Type:(Further) checkbox – when enabled this option means that the trigger will get activated when an object moves from the center of the zone to a distance, specified in Distance field.

Don't forget to “turn on” the trigger by TL from some event (Mission Begin, fly-by Way Point etc.), otherwise the trigger won't work!

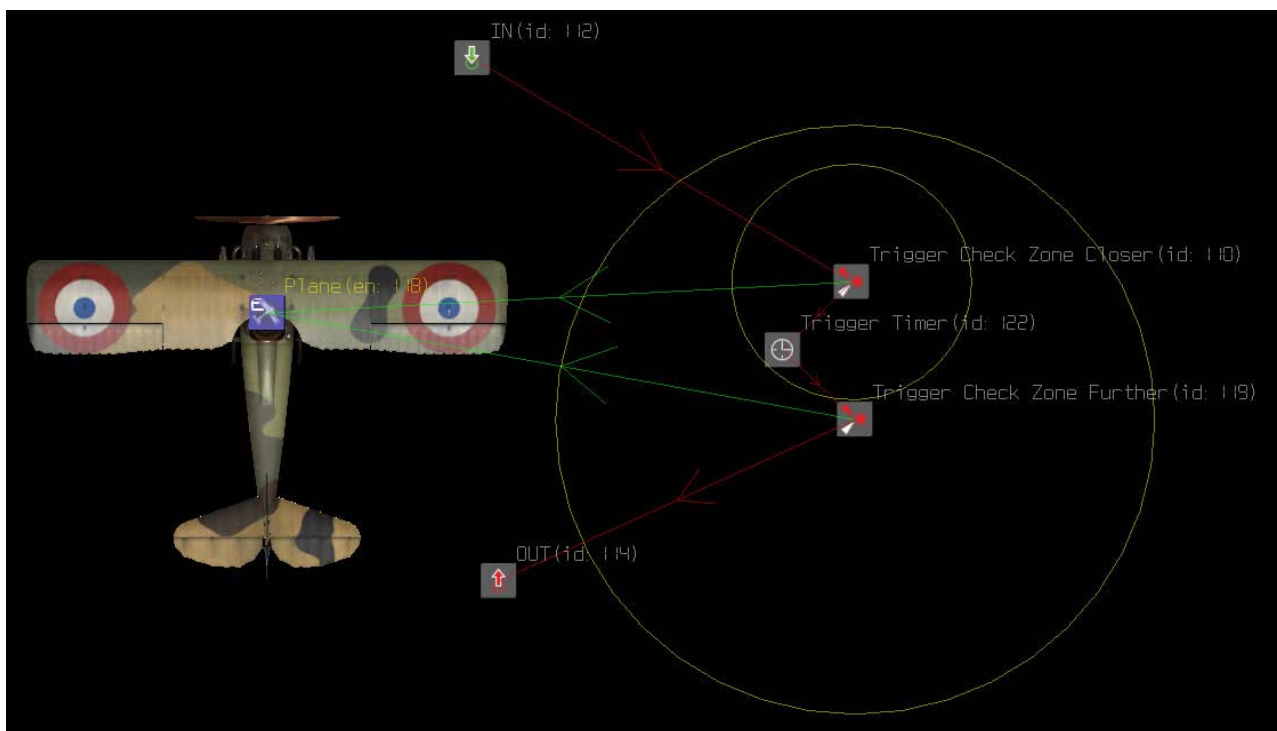
In case when for one object there are to check (closer and further) being performed there must be some hysteresis (difference in “close” and “further” radiuses).

Moreover “closer” is being checked first and only when this checking is complete you can start (by timer or event) the checking of “further”. Otherwise the logics of checking will be ruined.

For example: if the checked object is born inside the given radius of Check Zone (further) – all will work correctly, i.e. the checking won't start, since the object in fact is inside the specified radius of the zone and Check Zone (close) isn't needed in this case.

If the checked object is born outside of the given radius of Check Zone (further) then you should first enable checking “close” so that the trigger understands that the object has entered the checking zone and only then enable checking “further” to control leaving of the zone.

If Check Zone deactivated by “Deactivate” trigger is activated it will work again with the settings set in Advanced Properties.



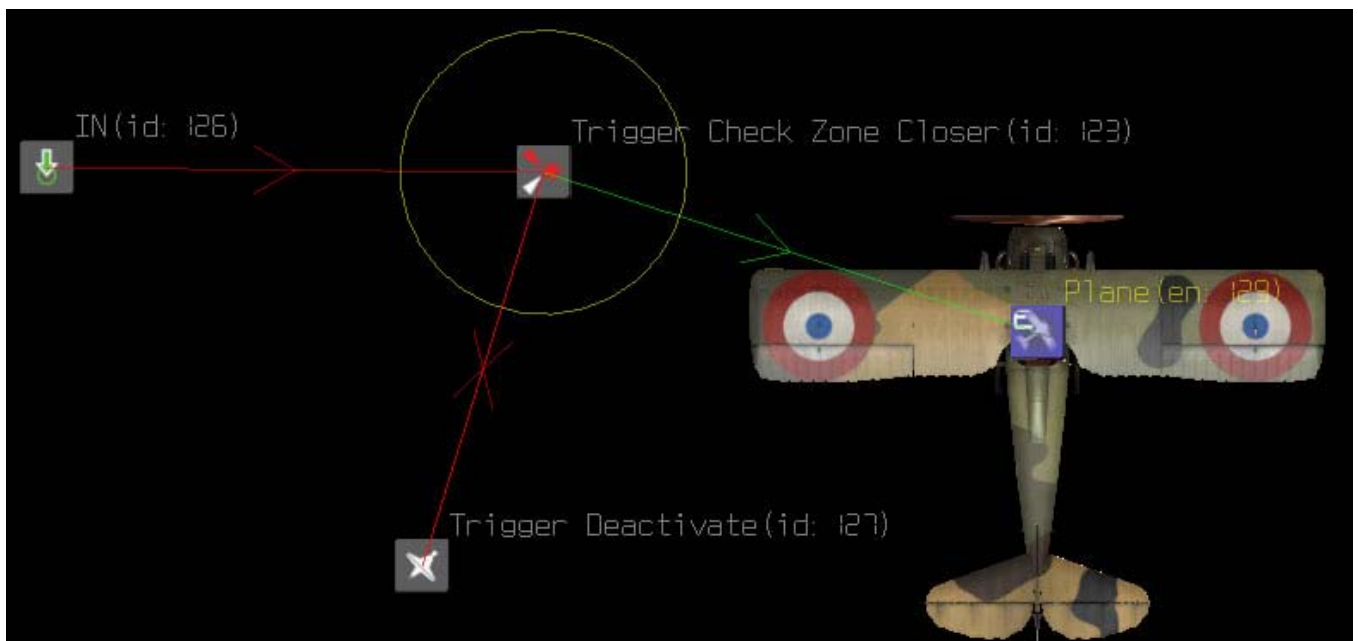
In the given example (triggering condition by a linked object):

- Trigger: Check zone Closer – the trigger checking if the object (here – a plane) is inside the zone. The trigger refers to the plane by OL and is activated only by it;
- Trigger: Check zone Further – the trigger checking if the object (here – a plane) is outside the zone. The trigger refers to the plane by OL and is activated only by it;
- Trigger Timer – this trigger starts Check zone Further after Check zone Closer was activated;
- IN - any incoming mission event that by TL (or Message) activates Trigger:Check zone;
- OUT –any mission event that is triggered by Check zone Further after it has been activated.

So, the abovementioned example checks the plane that has entered the specified zone and the left it. After that OUT can activate any mission event.

Also please note that Check Zone trigger has been activated it will be active for the whole time of mission progress.

Usually only single activations of the trigger are needed (if you don't need to check the zone continuously) therefore we advise to deactivate unused Check Zone triggers in a mission:



In this example Check Zone Closer will work once on the given object then deactivate itself by Deactivate trigger (see 7.8.5.). It should be done so that there are no active Check Zone triggers left in the mission then can accidentally get activated again (if it's not needed by mission plot of course).

This trigger has FIXED coordinates in a mission unlike the Proximity trigger (see 7.8.6.).

Text format of the trigger:

```
MCU_CheckZone //MCU type
{
  Index = 27; //MCU ID
  Name = "Trigger Check Zone"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
```

```

Objects = []; // ID of the objects that this trigger refers to by OL
XPos = 2095.66; // X axis coordinate
YPos = 112.102; // Y axis coordinate
ZPos = 4307.74; // Z axis coordinate
XOri = 0; //orientation (doesn't have any functions in a mission)
YOri = 0; //orientation (doesn't have any functions in a mission)
ZOri = 0; //orientation (doesn't have any functions in a mission)
Zone = 1; //radius of triggering zone in meters
Cylinder = 1; //zone type, 1=cylinder, 0=sphere
OppositeAirplanes = 0; //trigger activated by Opposite planes, 0=off/1=on
OppositeVehicles = 0; // trigger activated by Opposite vehicles, 0=off/1=on
OrderAirplanes = 0; // trigger activated by Order planes=off/1=on
OrderVehicles = 0; // trigger activated by Order vehicles, 0=off/1=on
Closer = 1; //zone types, 1=closer, 0=further
}

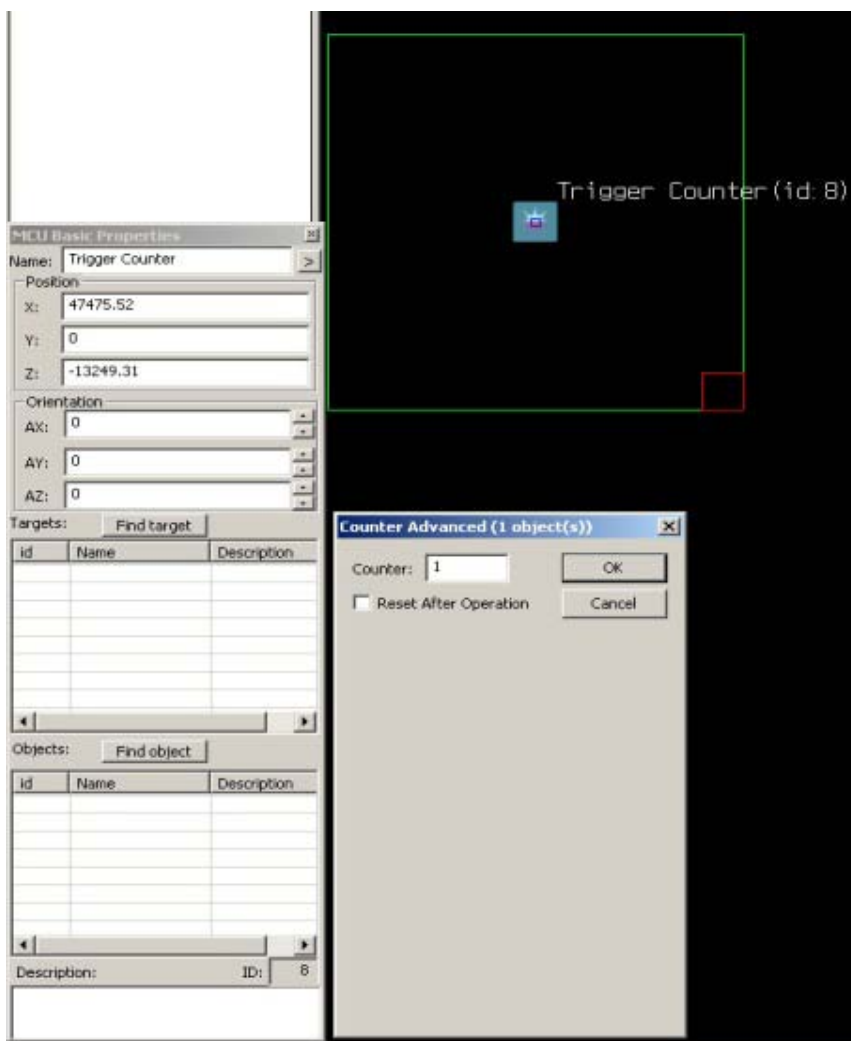
```

7.8.4. Trigger:Counter



In the given example:

- IN - any incoming mission event that by TL (or Message) activates Trigger:Counter, i.e. each time when this event happens Counter will calculate it (+1);
- OUT – any mission event that is activated by Counter trigger after it was enabled.



Trigger – event counter (with each event adds +1 to the counter).

Double-click trigger icon to open Counter Advanced Properties.

Advanced Properties:

Counter field: - limit value, by reaching it the counter will activate all of its TL.

Reset After Operation checkbox – resets the counter when it reaches the limit. This option is used if you need to use the conditions set in this trigger repeatedly.

When this trigger is deactivated by the Deactivate trigger (see 7.8.5.) all counted events are not erased but are “frozen” and stay in trigger “memory”. If you activate it again the trigger will continue to count events considering the data that was left in its “memory” and will trigger when it reaches the limit value.

Text format of the trigger:

MCU_Counter //MCU type

{

Index = 28; //MCU ID

Name = "Trigger Counter"; //MCU name (not displayed in GUI)

Desc = ""; //MCU description (not displayed in GUI)

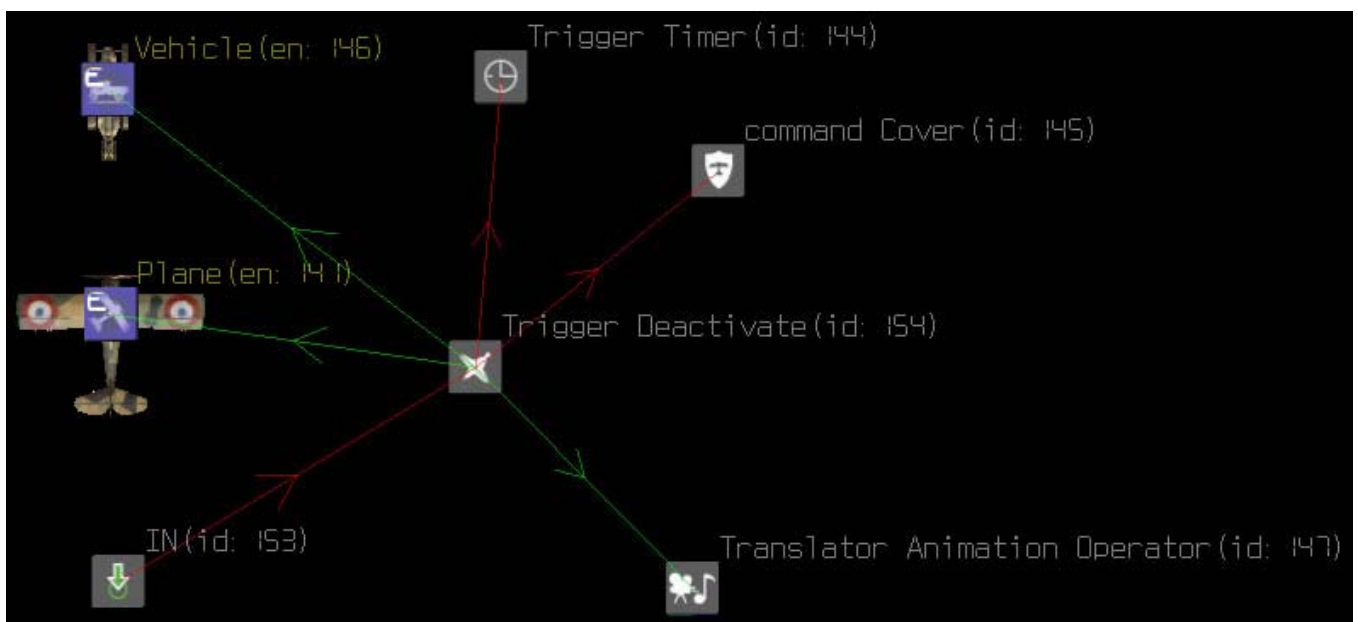
Targets = []; // ID of the objects that this trigger refers to by TL

```

Objects = []; // ID of the objects that this trigger refers to by OL, this trigger can't have any OL
XPos = 2080.77; // X axis coordinate
YPos = 114.026; // Y axis coordinate
ZPos = 4170.72; // Z axis coordinate
XOri = 0; //orientation (doesn't have any functions in a mission)
YOri = 0; //orientation (doesn't have any functions in a mission)
ZOri = 0; //orientation (doesn't have any functions in a mission)
Counter = 1; //trigger limit value
Dropcount = 0; //erasable (1) /inerasable (0) to 0 after trigger is enabled
}

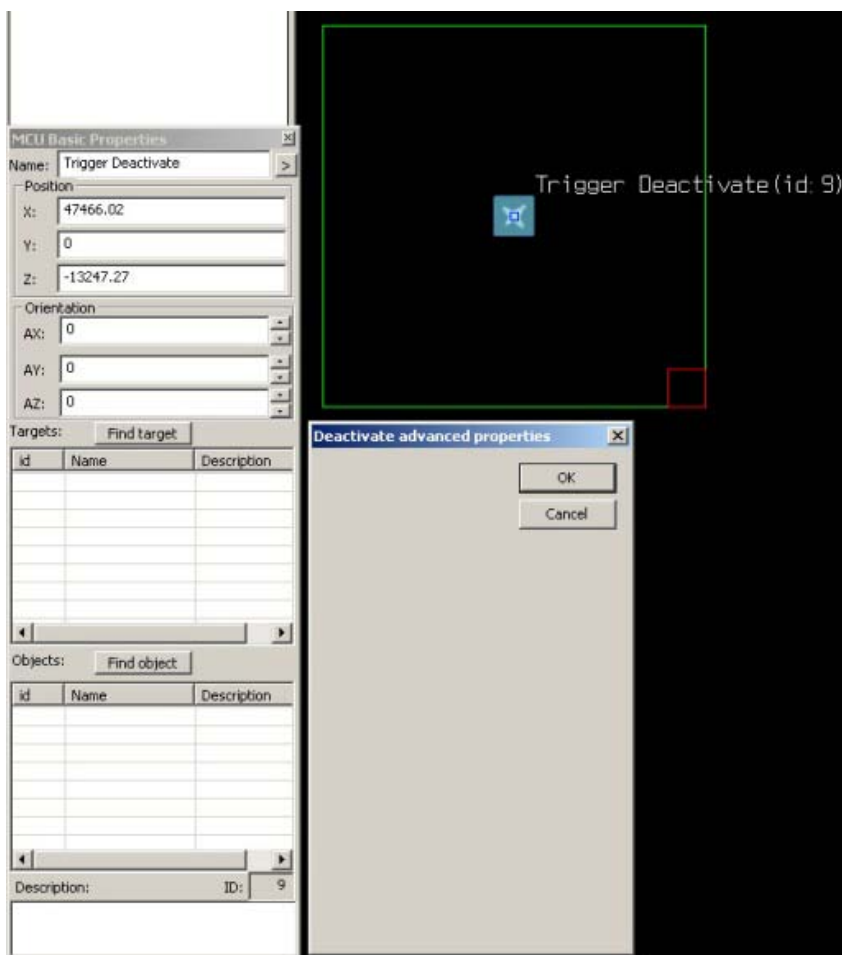
```

7.8.5. Trigger:Deactivate



In the given example:

- Trigger:Deactivate – this trigger deactivates mission objects by OL (planes/vehicles and translators) and by TL (triggers/commands);
- IN - any incoming mission event that by TL (or Message) activates Trigger:Deactivate.



Double-click trigger icon to open Deactivate Advanced Properties.

This trigger is often used together with Activate trigger (see 7.8.1).

PAY ATTENTION – TL of this trigger DEACTIVATES objects, i.e. by TL from this trigger you can't enable further mission events.

PAY ATTENTION – if the object is deactivated then it doesn't respond to commands from a mission and doesn't interact with other mission objects.

This trigger acts differently with MCU and objects in a mission:

- after being deactivated planes, tanks, vehicles, locomotives, artillery and balloons these objects stay in the mission in NOT active state and don't respond to mission events until they are activated;
- objects from Blinks and Battlefield sections when deactivated stay visible in the game;
- when Timer trigger (see 7.8.8.) is deactivated then the time inside of it will be erased to the initial value;
- when deactivating Check Zone trigger and the activating it again the trigger will work again (see 7.8.3.) with the specified inside of it conditions;
- when deactivating Counter trigger the current value will "freeze";

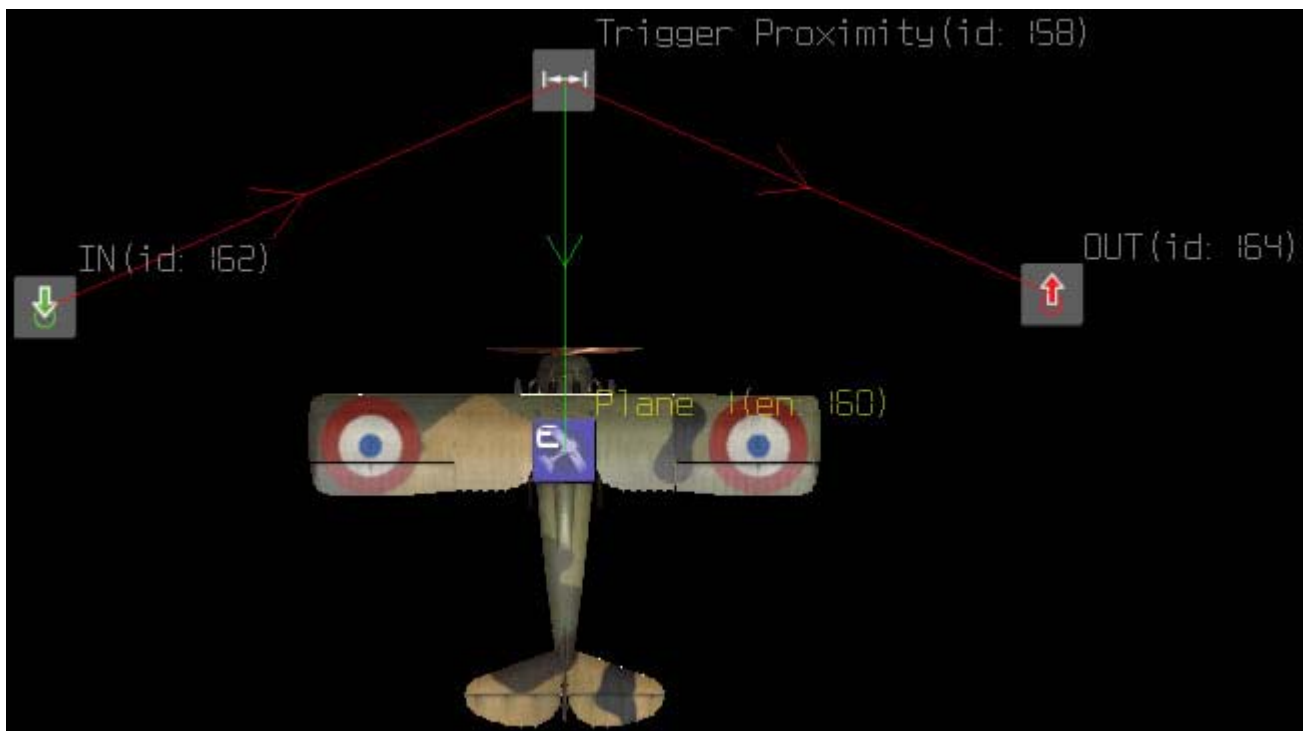
Text format of the trigger:

```
MCU_Deactivate //MCU type
{
  Index = 29; //MCU ID
  Name = "Trigger Deactivate"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2021.2; // X axis coordinate
  YPos = 110.762; // Y axis coordinate
  ZPos = 4149.87; // Z axis coordinate
  XOri = 0;
  YOri = 0;
  ZOri = 0;
}
```

7.8.6. Trigger:Proximity



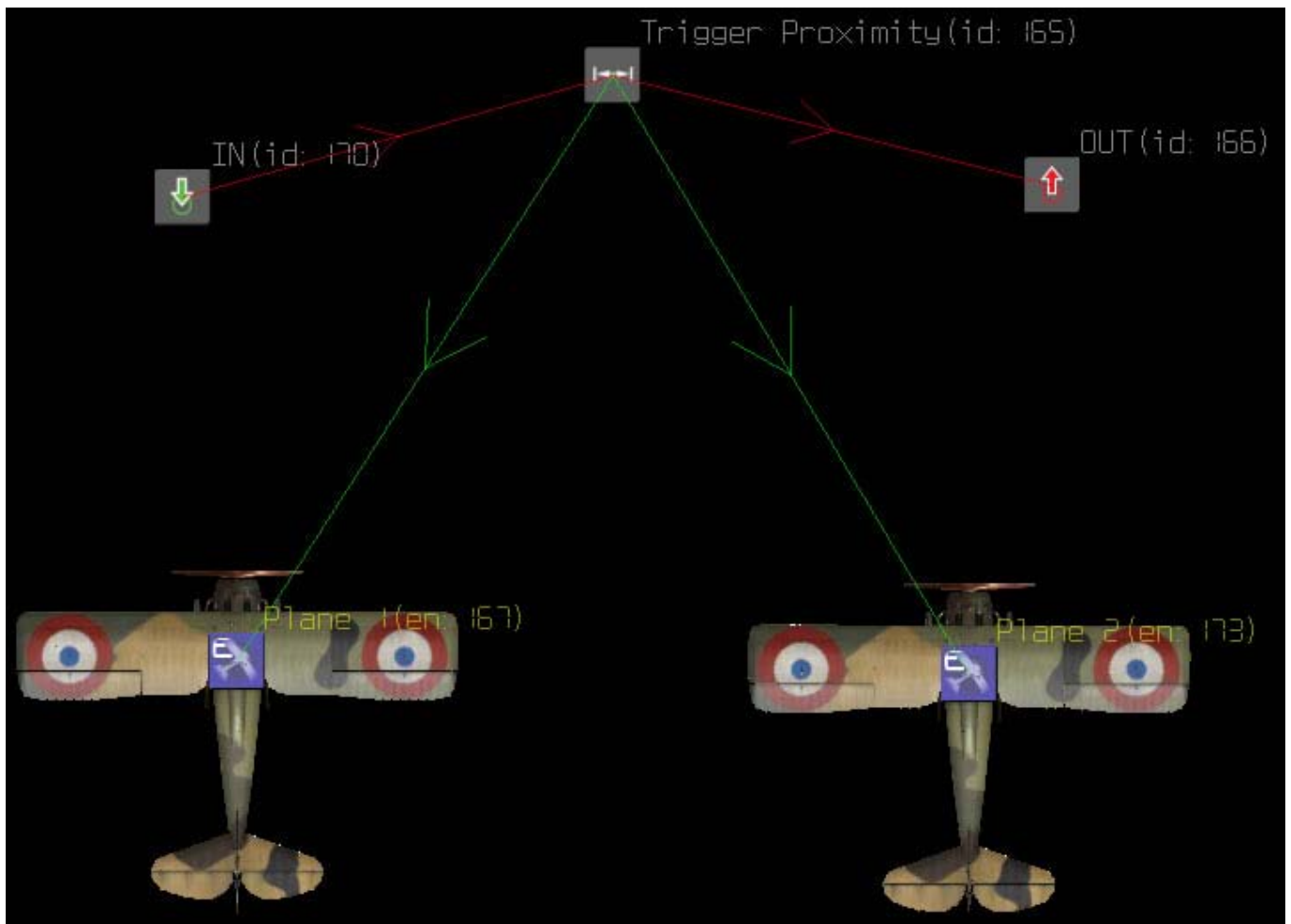
Example of one object and objects specified in Properties:



In the given example:

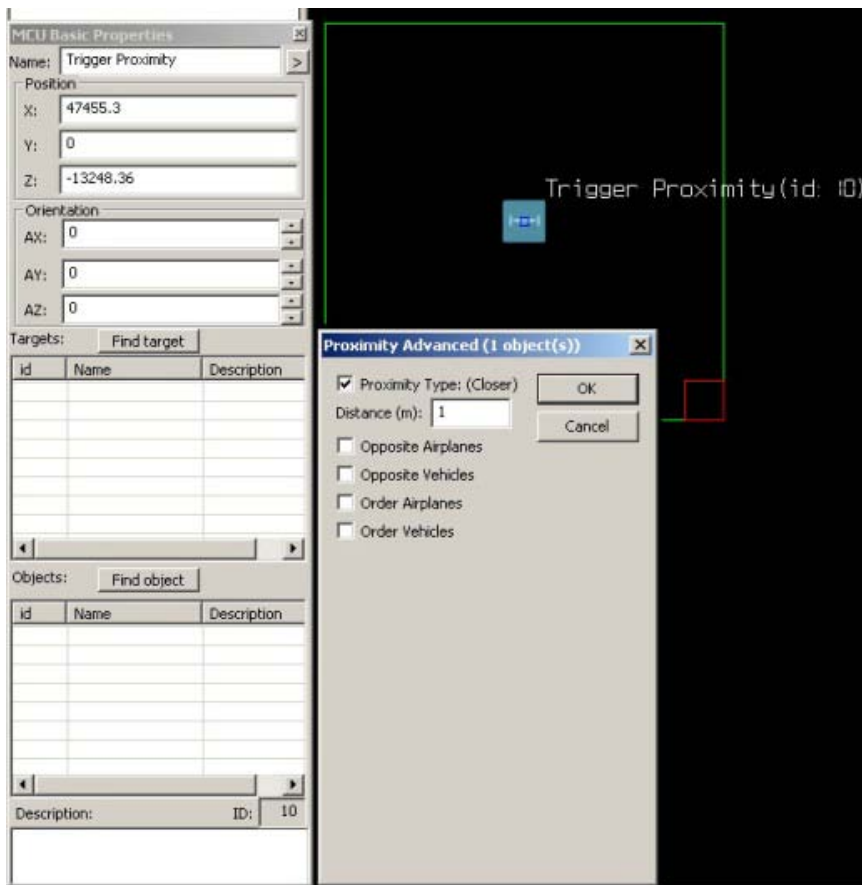
- Trigger: Proximity – trigger checking the condition if there are objects closer/farther then the specified in the Properties distance;
- IN - any incoming mission event that starts Proximity trigger.
- OUT – any mission event that is activated by Proximity trigger after it has been enabled.

An example of Proximity between two given objects:



In the given example :

- Trigger: Proximity – the trigger checks the distance condition (closer or further) of the given objects, in this case: Plane 1 and Plane 2;
- IN - any incoming mission event that by TL (or Message) activates Proximity trigger.
- OUT – any mission event that is activated by Proximity trigger after it was enabled.



This trigger dynamically measures distance between the assigned through OL objects that are closing in or moving away. Two modes: measures closing distance of objects (closer) and measures distance of objects moving away from each other (further). Double-click trigger icon to open Proximity Advanced Properties.

Advanced Properties:

Proximity Type: (Closer) checkbox – the trigger will work when objects are closing in to a distance equal or smaller then the one specified in Distance field.

Proximity Type: (Further) checkbox – the trigger works when objects are moving away from each other to a distance equal or bigger then the one specified in Distance field.

Distance (m): field – specifies trigger radius of action in meters.

Opposite Airplanes checkbox – trigger is activated (on)/not activated (off) by Opposite Coalition planes.

Opposite Vehicles checkbox – trigger is activated (on)/not activated (off) by Opposite Coalition vehicles.

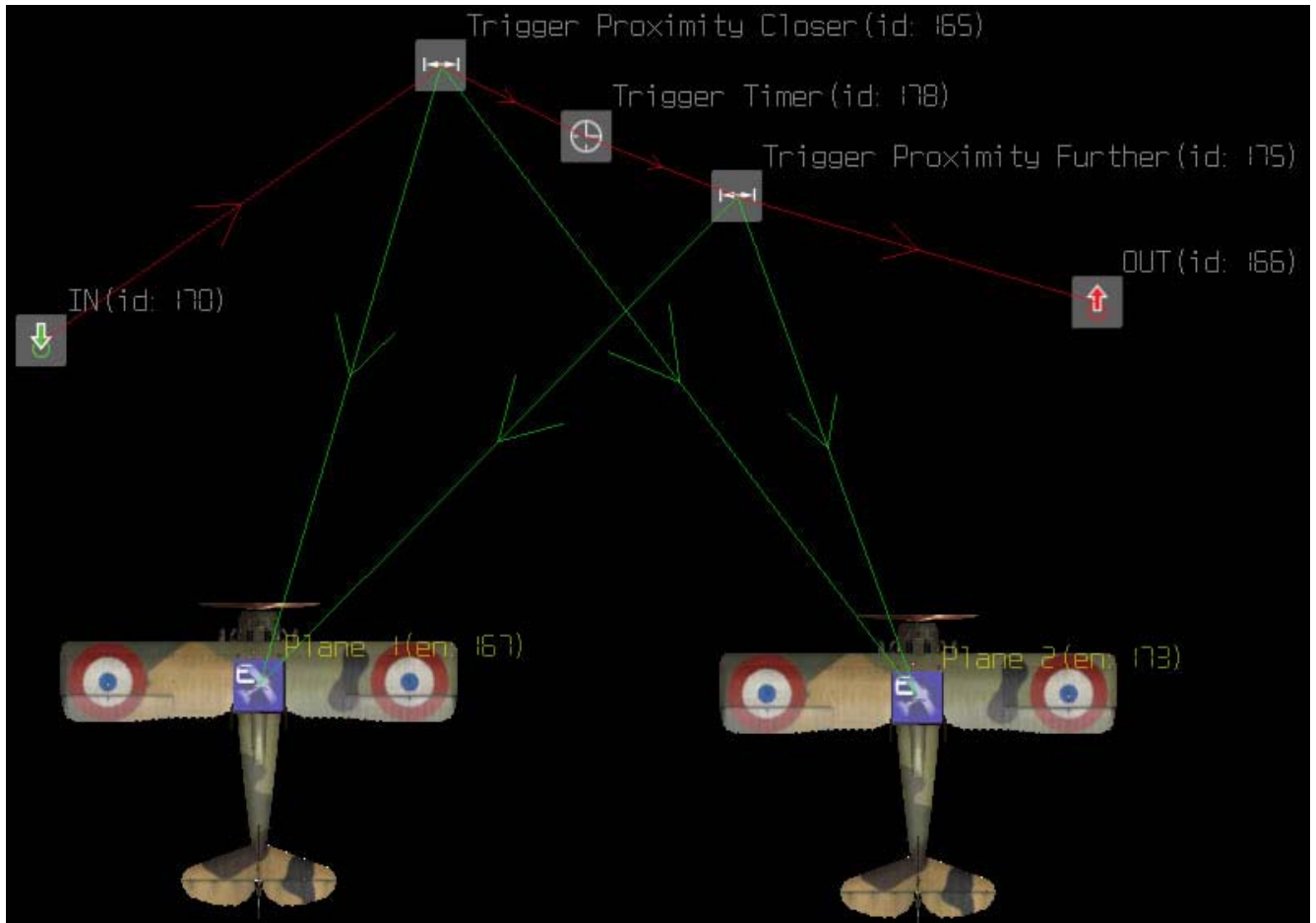
Order Airplanes checkbox – trigger is activated (on)/not activated (off) by Order Coalition planes.

Order Vehicles checkbox – trigger is activated (on)/not activated (off) by Order Coalition vehicles.

In case if the distance is measured for objects by “closer” and by “further” then after “closer” is enabled “further” will be activated in 1-2 seconds by Timer trigger.

This Timer inclusion should be made since Proximity doesn’t have and defined coordinates mission objects can be anywhere.

Proximity Further can get activated earlier then Proximity Closer (Further initially checks the distance between the moving away objects, but during initialization in a mission in the first moment it has (0;0) coordinates). So for Proximity Further to have time to measure the distance between the objects (to have time to analyze the distance between the objects considering their coordinates in the game world) by further condition you need to place a delay by this timer.



Text format of the trigger:

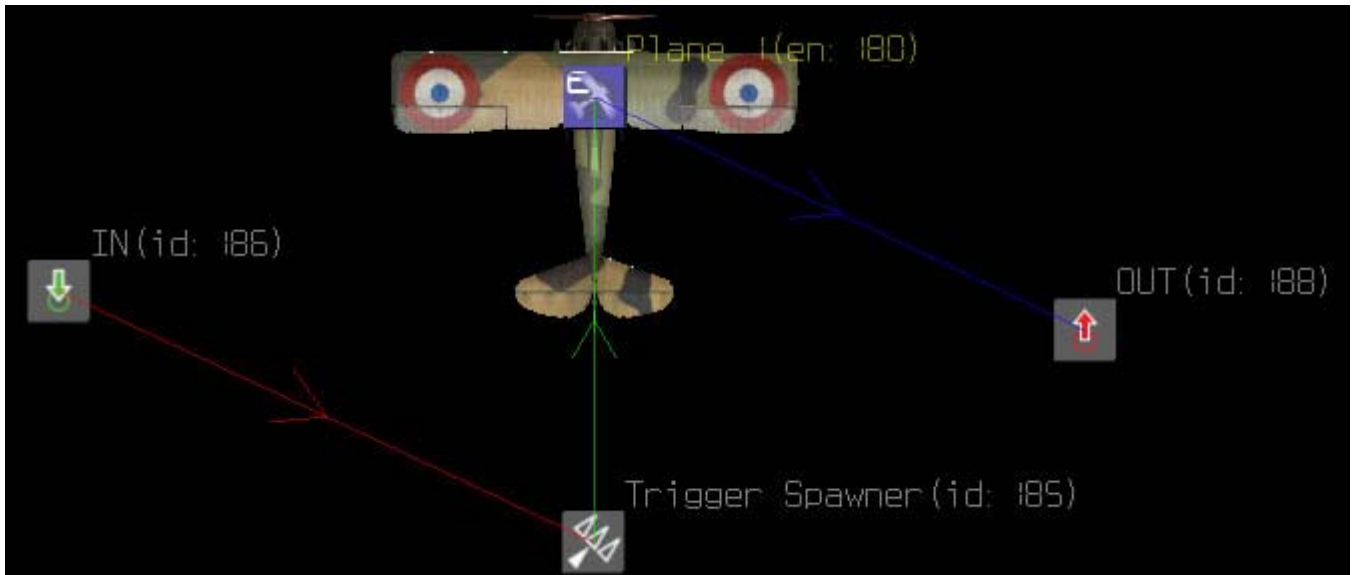
```
MCU_Proximity //MCU type
{
  Index = 2; //MCU ID
  Name = "Trigger Proximity"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2177.36; // X axis coordinate
  YPos = 112.34; // Y axis coordinate
  ZPos = 4197.41; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Distance = 1; //distance settings for trigger activation
  OppositeAirplanes = 0; //measures distance relatively to Opposite planes
  OppositeVehicles = 0; // measures distance relatively to Opposite vehicles
  OrderAirplanes = 0; // measures distance relatively to Order planes
```

```

OrderVehicles = 0; // measures distance relatively to Order vehicles
Closer = 1; //trigger type, 1=closer, 0=further
}

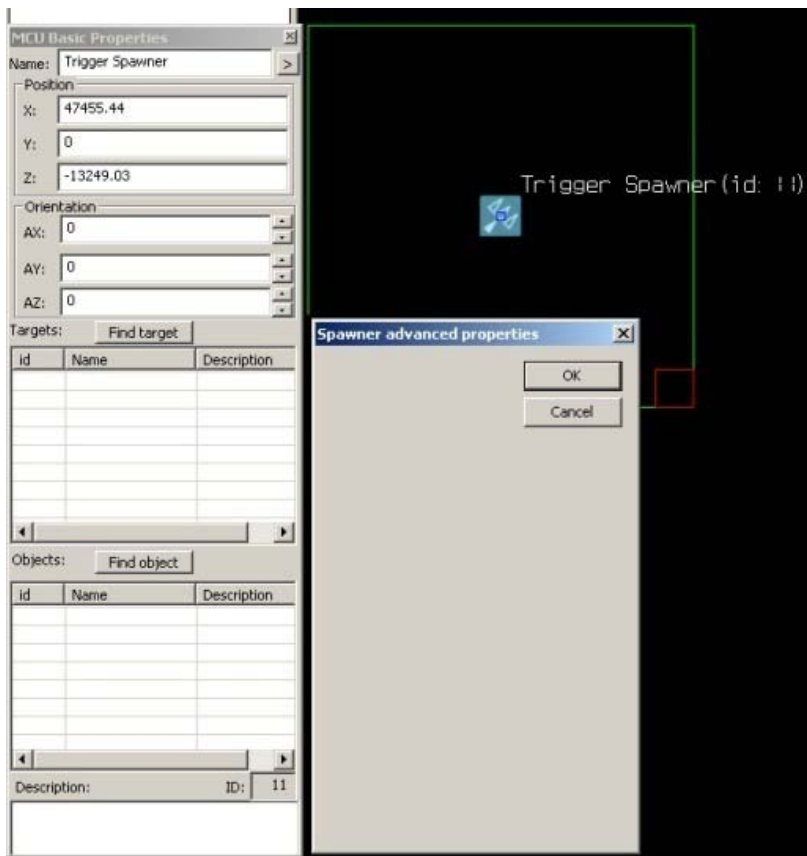
```

7.8.7. Trigger:Spawner



In the given example:

- Trigger:Spawner – trigger that spawns “gives birth to” a plane (in this example) in the game world;
- IN - any incoming mission event that by TL (or Message) activates Spawner trigger.
- OUT – any mission event that is activated by On Spawned Message, i.e. after “birth” of this object in a mission.



This trigger “spawns” copies of objects with the same properties into the game world.

Double-click trigger icon to open Spawner Advanced Properties.

To “spawn” an object it’s enough to set OL from the trigger to the object and start the trigger.

If after being “spawned” the object has to do something (for example: fly or drive to a waypoint) then you need to set “Message” link from the object to the next command or trigger of this object (On Spawned Message: Target).

Spawner can spawn an object any number of times.

ATTENTION! – You can spawn only single objects. You can’t spawn a player or a group of objects.

ATTENTION! – If this trigger is connected by OL to an object in a mission then this object won’t appear in the mission until you start the Spawner trigger.

ATTENTION! – Mission script can’t distinguish copies of an object in a mission (those that were spawned by this trigger).

Text format of the trigger:

MCU_Spawner //MCU type

```
{
  Index = 31; //MCU ID
  Name = "Trigger Spawner"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL, this trigger can't have any TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2152.26; // X axis coordinate
```

```

YPos = 115.08; // Y axis coordinate
ZPos = 4254.13; // Z axis coordinate
XOri = 0; //orientation (doesn't have any functions in a mission)
YOri = 0; //orientation (doesn't have any functions in a mission)
ZOri = 0; //orientation (doesn't have any functions in a mission)
}

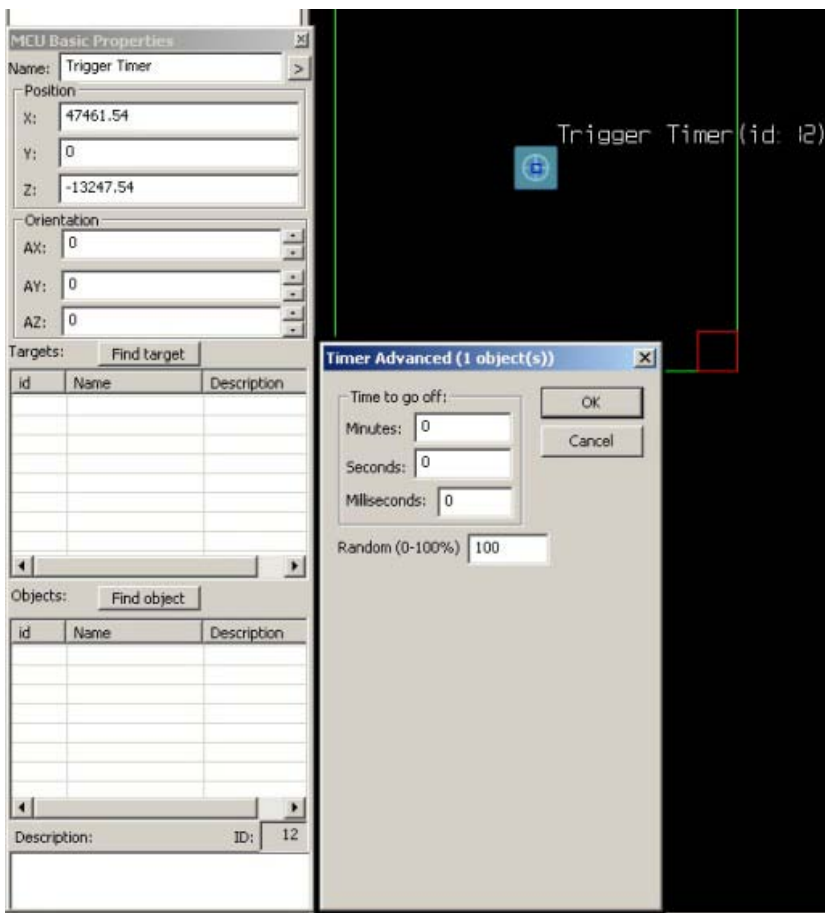
```

7.8.8. Trigger:Timer



In the given example:

- Trigger: Timer – trigger that is activated by the specified time;
- IN - any incoming mission event that starts Timer trigger.
- OUT – any mission event that is activated when Timer trigger is enabled.



This trigger is activated the specified time runs out.

Double-click trigger icon to open Timer Advanced Properties.

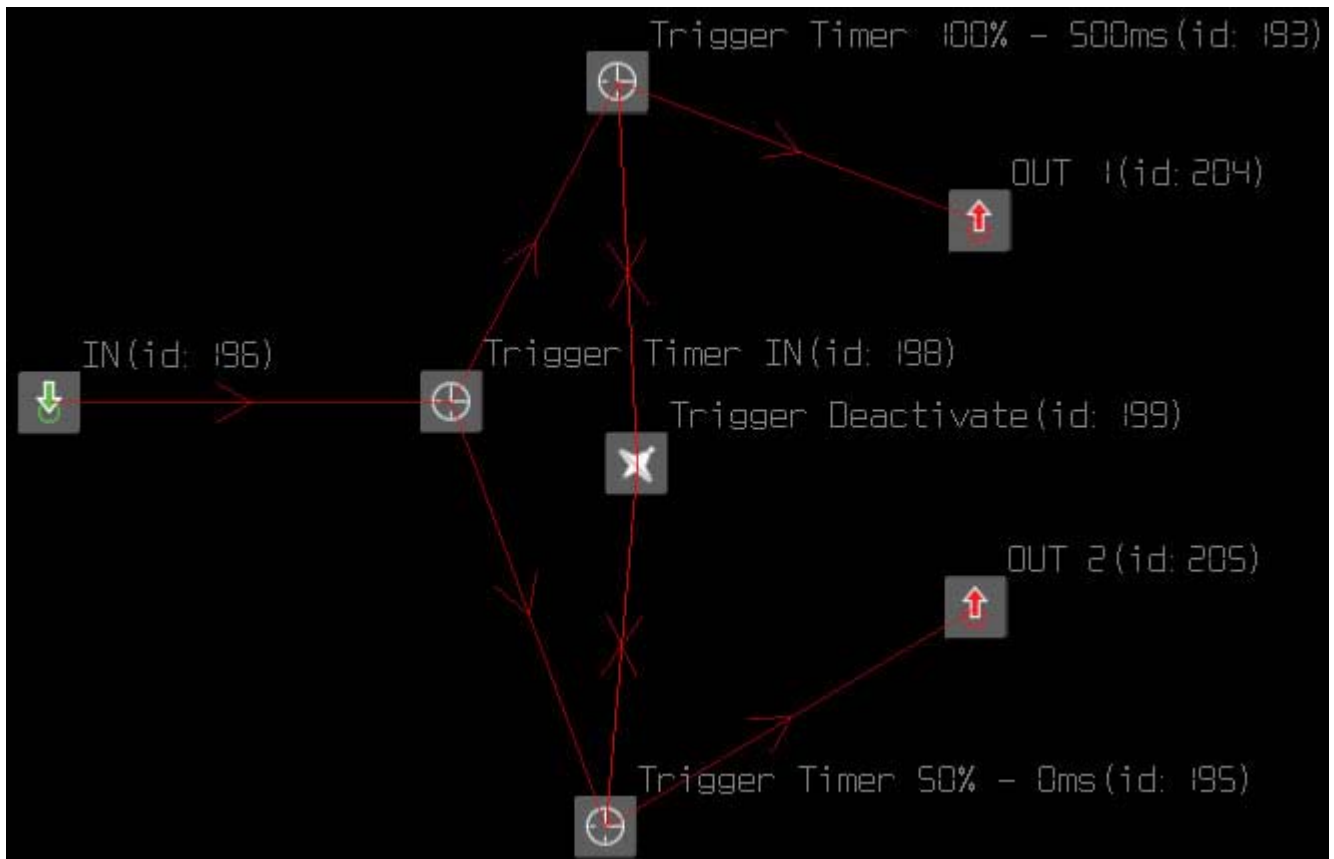
Advanced Properties:

Time to go off: Minutes, Seconds, Milliseconds – sets time in minutes, seconds, milliseconds. After this time the trigger will be activated.

Random (0-100%) – sets probability of trigger activation in %.

7.8.9. Random Timer scheme.

This scheme allows starting any random events in a mission depending on the specified probabilities with the help of Trigger:Timer:



In the given example you can see “random 50/50” scheme:

- IN - any incoming mission event that by TL (or Message) activates Timer IN trigger.
- Timer IN – buffer timer (actually a blank inclusion made for convenience);
- Trigger:Timer 100% - 500ms – timer with settings: Random=100% and time=500ms;
- Trigger:Timer 50% - 0ms - timer with settings: Random=50% and time=0ms;
- Trigger:Deactivate – trigger that deactivates either “Trigger:Timer 100% - 500ms” or “Trigger:Timer 50% - 0ms” – depending on which timer will first trigger the deactivator;
- OUT 1 – any mission event that is started when “Timer 100% - 500ms.” Trigger is activated;
- OUT 2 – any mission event that is started when “Timer 50% - 0ms.” is activated.

The scheme works in the following way:

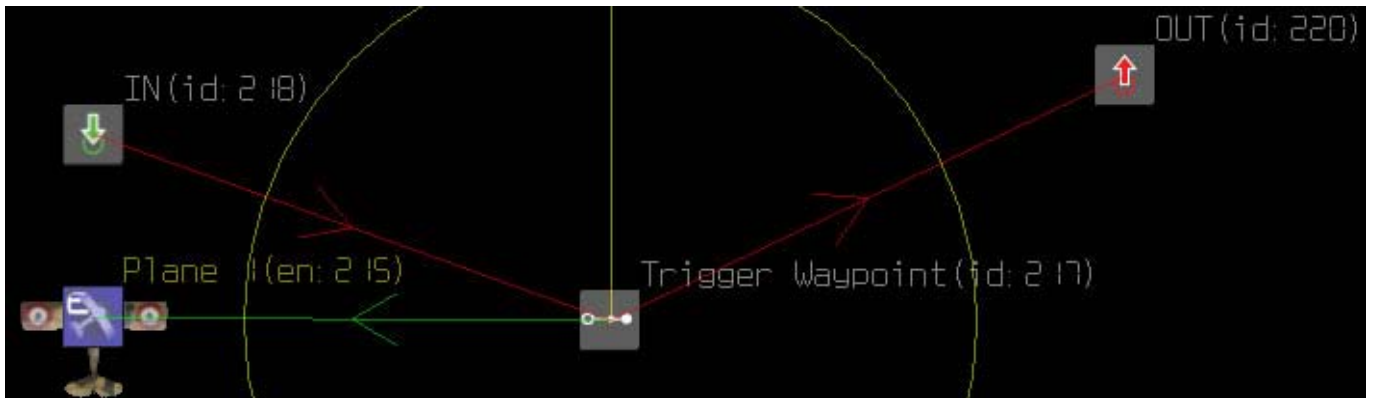
- if “Timer 50% - 0ms.” is activated (if 50% timer will work before the 100% timer) then it deactivates “Trigger:Timer 100% - 500ms” and will activate the mission event by “OUT 2” link;
- if “Trigger:Timer 100% - 500ms” is activated then it deactivates “Trigger:Timer 50% - 0ms” and will activate the mission event by “OUT 1” link.

This way you can balance activation conditions depending on Random settings of the Timer.

Text format of the trigger:

```
MCU_Timer //MCU type
{
  Index = 32; //MCU ID
  Name = "Trigger Timer"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL , this trigger can't have any OL
  XPos = 2164.17; // X axis coordinate
  YPos = 115.08; // Y axis coordinate
  ZPos = 4280.93; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Time = 0; //time settings
  Random = 100; //probability of trigger activation in %
}
```

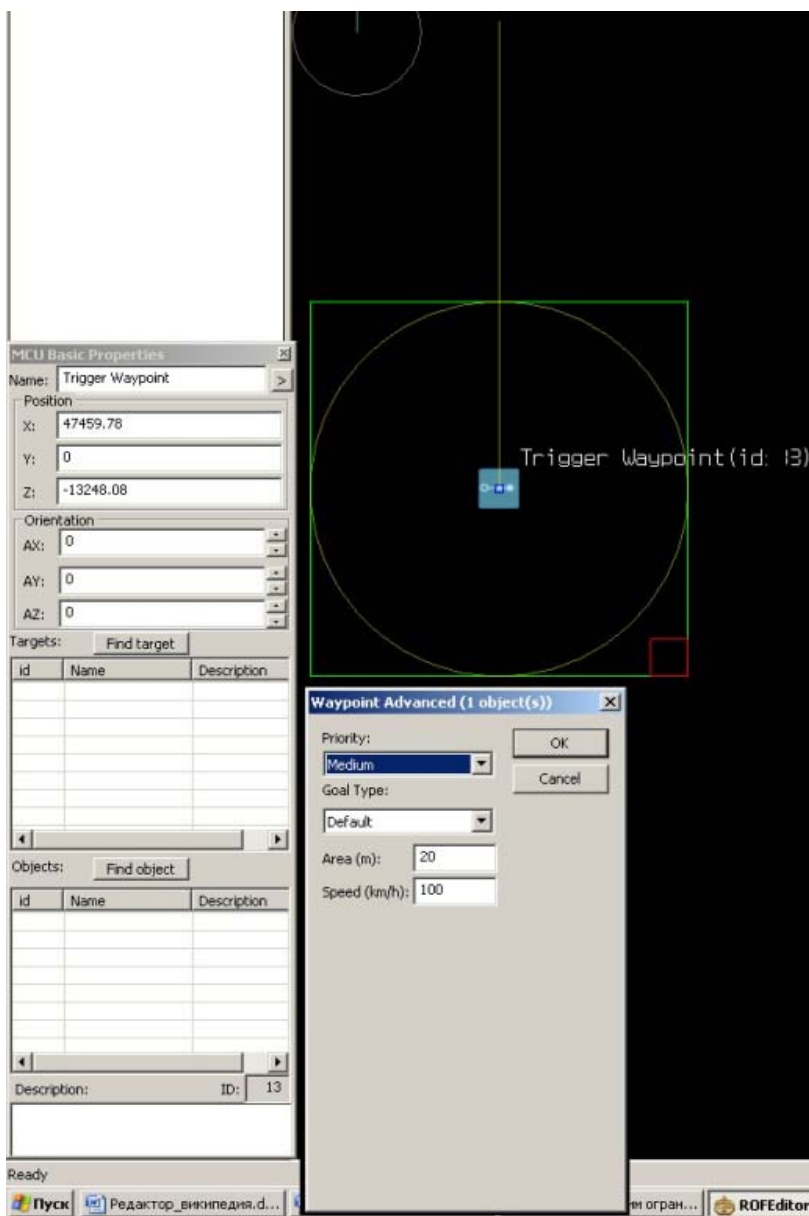
7.8.10. Trigger:Waypoint



In the given example :

- Trigger: Waypoint – this trigger gives command to an object to move to this trigger and works only for this object. **In fact Waypoint is at the same time a command for moving objects;**
- IN – any incoming mission event that by TL (or Message) activates Waypoint trigger.
- OUT – any mission event that is started by Waypoint trigger.

This trigger **MUST** have a linked object (or objects), i.e. must be connected to them by OL.



Double-click trigger icon to open Waypoint Advanced Properties.

Advanced Properties:

Priority: High, Medium, Low – AI priority of moving to this point.

Low – when flying to the point with this priority AI will be searching for any available targets and attack them where possible;

Medium – when flying to the point with this priority AI will detour from the route or perform any actions only if it will feel danger (if a plane attacks from behind – AI will defend);

High – AI fly to the point ignoring all dangers until it reaches it (executes command).

Goal Type – Default, Primary, Secondary. Three type of visualization of a waypoint icon. View on the flight map and in game. Made for convenience of visual determination of waypoints and doesn't affect mission results.

Default – icon looks like 

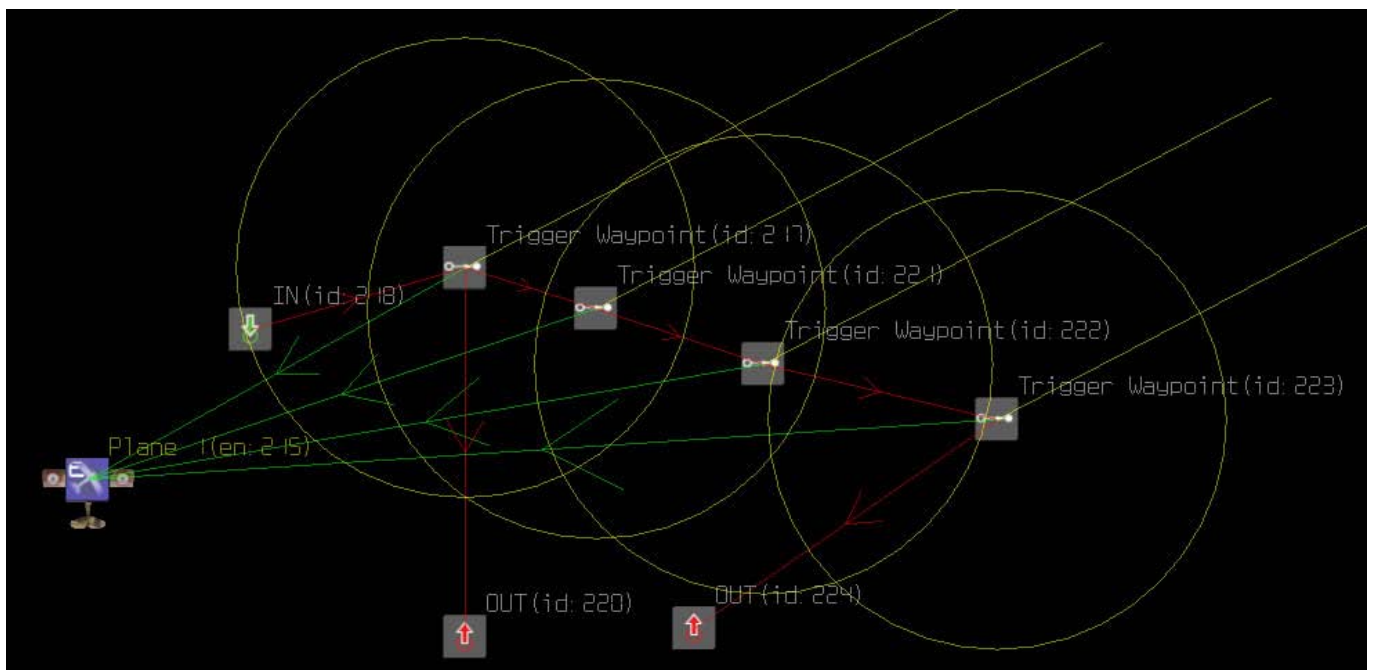
Primary – icon looks like 

Secondary – icon looks like 

Area (m): - size (radius) of the area where trigger is activated. In fact Waypoint space is a endless cylinder;

Speed (km/h): - specifies the speed (in km/h) that an object has to move to the trigger with.

When placing waypoints for the selected object it's possible to do that automatically by holding Left Shift+left-click, and then all OL and TL along the route will be placed automatically.



Therefore, since Waypoint is functionally a trigger, then when activated it can activate further Waypoints or other mission objects.

When setting Waypoint (Area) radius, try not to set it too small, since AI can miss the Waypoint and will continue attempts until finally reaching the specified area.

Don't forget to set altitude (YPos) of a waypoint, since AI will be flying at the altitude specified at this Waypoint.

If a single Waypoint is assigned to different objects (objects are not in group) then the trigger will work only for the first object to enter the Waypoint Area. The second object entering this Waypoint will stay over it unless there is another command specified for this object.

There is a possibility to "retarget" objects to another route. For example: an object is moving to a specified point and receives another Waypoint – then it will move to the newly received point, not depending on the priority of the new waypoint. In this case all unused (not activated) Waypoints must be

deactivated by Deactivate trigger, otherwise if this object will occasionally fly through the not activated Waypoint it will be activated, since it is stored in AI command buffer.

ATTENTION! – if during executing any current command AI receives Waypoint command then this command will overrule any other current commands ignoring their priority and type. Therefore always give a command AFTER (for example insert a Timer) you have given Waypoint command.

Text format of the trigger:

```
MCU_Waypoint //MCU type
{
  Index = 33; //MCU ID
  Name = "Trigger Waypoint"; //MCU name (not displayed in GUI)
  Desc = ""; //MCU description (not displayed in GUI)
  Targets = []; // ID of the objects that this trigger refers to by TL
  Objects = []; // ID of the objects that this trigger refers to by OL
  XPos = 2092.68; // X axis coordinate
  YPos = 111.939; // Y axis coordinate, in fact it's the altitude of the Waypoint
  ZPos = 4319.66; // Z axis coordinate
  XOri = 0; //orientation (doesn't have any functions in a mission)
  YOri = 0; //orientation (doesn't have any functions in a mission)
  ZOri = 0; //orientation (doesn't have any functions in a mission)
  Area = 20; //radius of command execution
  Speed = 100; //AI flight speed
  Priority = 1; //priority of command execution for AI
  GoalType = 0; //specifies icon view in GUI
}
```

8. Building a cooperative mission.

In fact a coop mission is similar to single-player mission except for the following conditions:

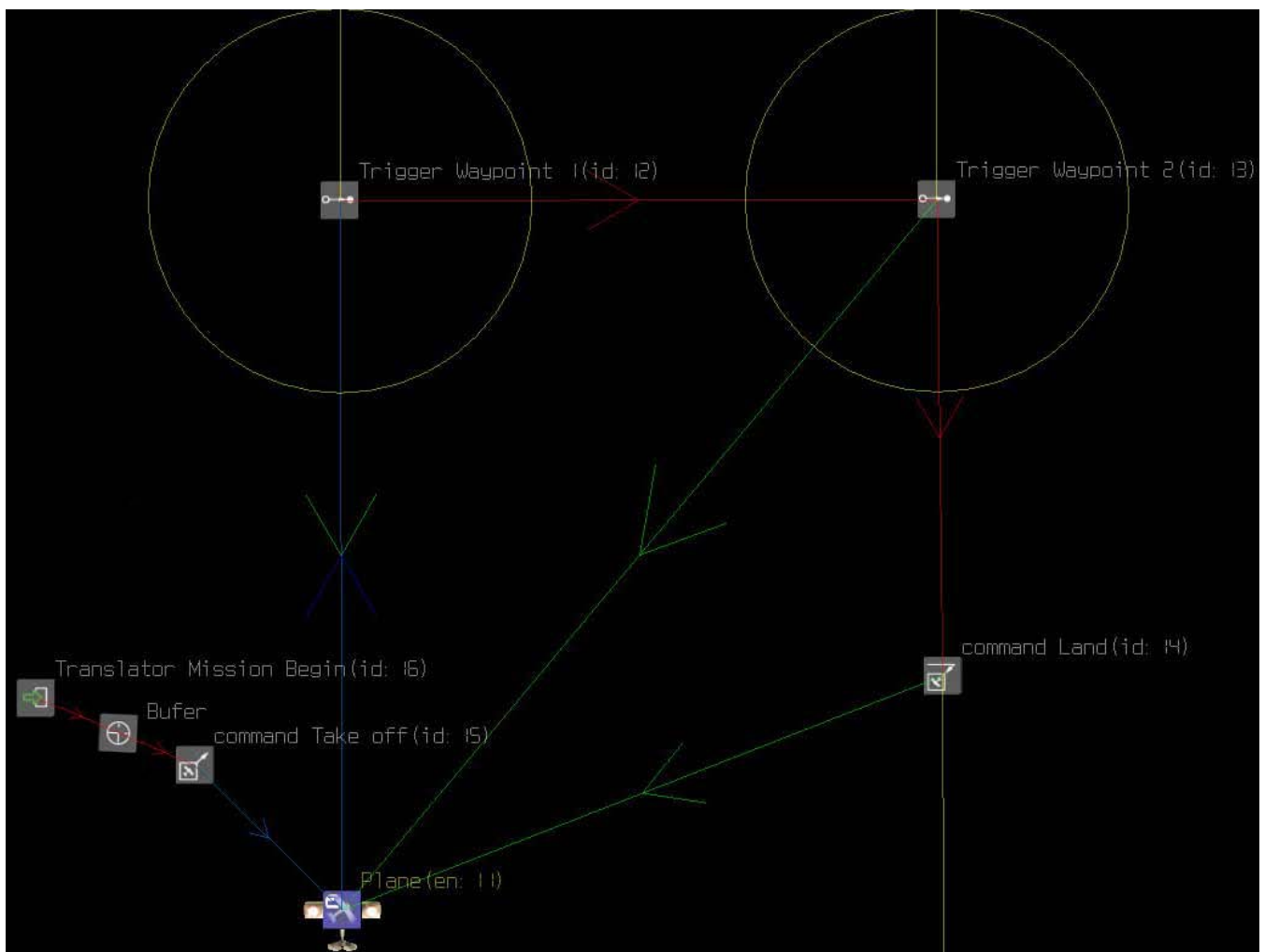
- mission type (see 5.1.) must be “cooperative”;
- for the coop mission to work properly it needs to be saved to \data\Multiplayer\Cooperative\. There must be mission list-file (*.list) placed in the same folder;
- Only one primary task (see 7.7.5) can work (will be accounted as successfully accomplished) in a cooperative mission with settings: Primary Tasks/Order coalition =>Success, since with mission instruments in coop mode you can set only one winner – Order Coalition. Therefore if during a mission Translator:Mission Objective/ Order coalition =>Success doesn't occur then Opposite Coalition wins automatically. The same restrictions apply also to Secondary tasks, therefore be careful, if there are several “Mission Objective/Order coalition =>Success” in a mission then the mission will count only one – the last one that occurred;
- if after Order coalition accomplishes its task there will be no alive players left in Order coalition then Lost will be automatically awarded to Order coalition;
- if there are no live players left in the mission then draw will be stated;

- for a plane to be available for selection in coop mission lobby “cooperative start” setting must be enabled in plane’s properties (see 7.3.1). This way you can set as airstart, so that ground start also;
- be careful when setting Default Camera Operator for coalitions (see 7.7.2). Don’t forget to set them – one for each coalition, so that players who’ll participate in the mission as spectators appear in the desired place of map after mission loading;
- don’t forget to set Translator:Icon properties correctly (see 7.7.3.);
- in coop mission there can be no plane with AI=Player (see 7.3.1.);
- if in coop mission a plane is not occupied by any player then after mission start this plane will fly as AI with AI level set as in mission settings.

The rest: the whole logical mechanism of mission (operation of MCU, AI etc.) will work just as in single-player mission.

9. Frequently used mission blocks, examples and building tips.

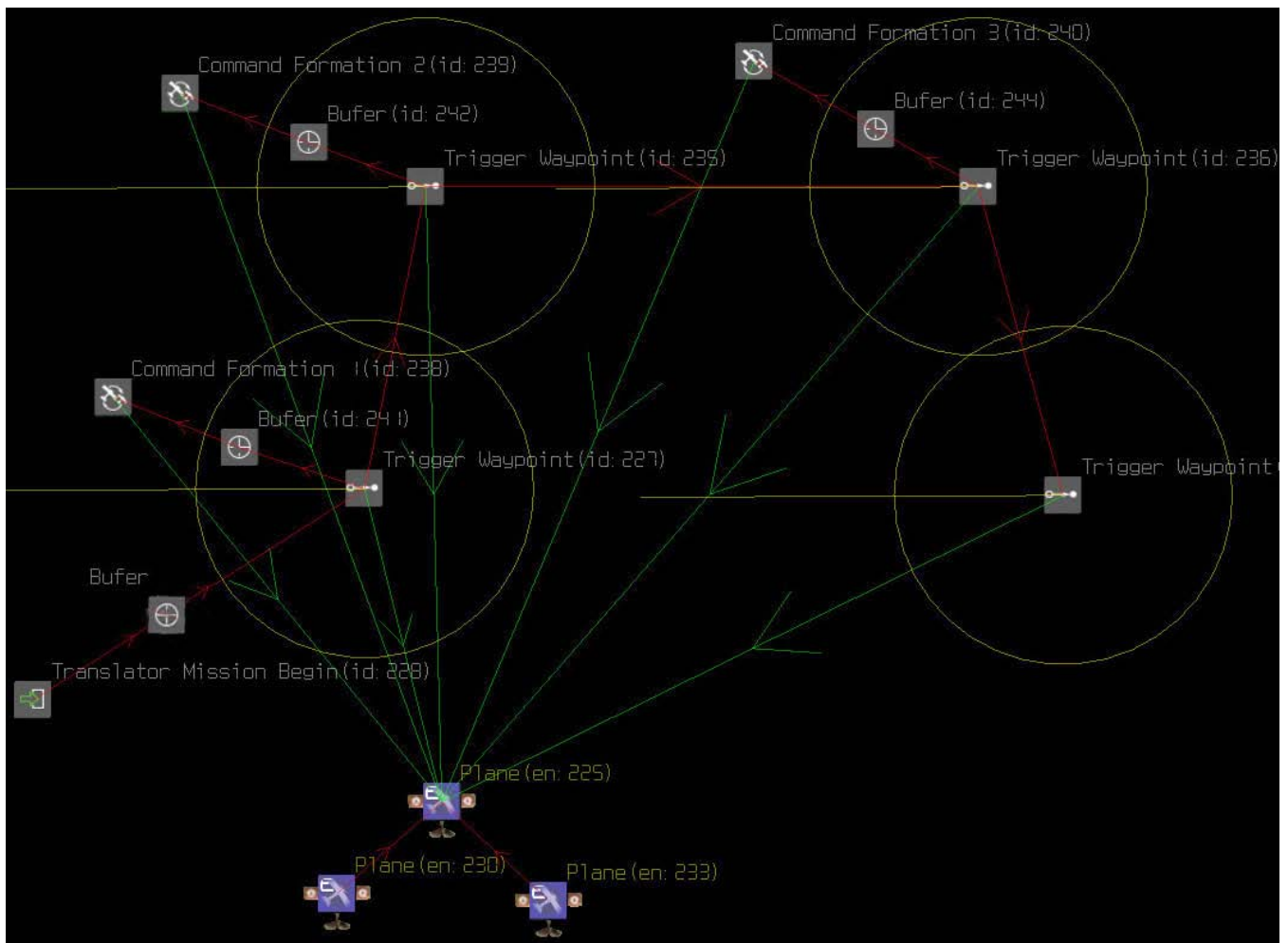
9.1. Taking off, flying through waypoints and landing.



In the given example :

- Take Off command (see 7.6.13) is connected by OL to a plane and is started Translator:Mission begin. I.e. to a plane that knows this command refers to him and when this command is activated – executes it;
 - Waypoint (see 7.8.10.) 1 and 2 – plane waypoints that in turn refer to the plane by OL. The first Waypoint is activated by On Take off Message of the plane (see 2.3.) and by TL activates the next Waypoint en route. So, when the plane executes take off command (actually takes off) will activate the first waypoint. In Advanced Properties of the plane in “On Messages table” menu (see 7.3.1.) will be the following picture: [Message Type: Take Off; Command ID:15; Target ID 12];
 - Land command refers to the plane by OL and is activated by the last waypoint of the route;
- This is an example of how a plane will take off, fly through two waypoints and land.

9.2. Formation and changing formation in flight.

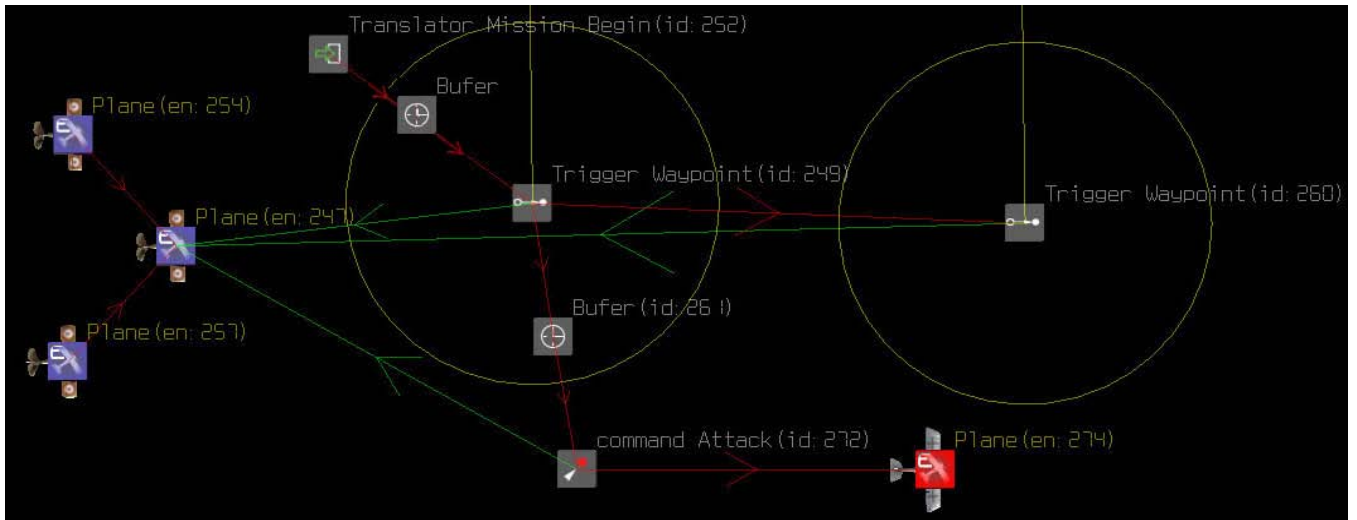


In the given example :

- three planes are united into a wing. All commands are tied to the leader (ID:225) through OL (see 6.5.);
- the wing flies through waypoints, at this, waypoints are activating commands to change formation (see 7.6.8.) with a delay by Timer to avoid one plane (here: wing leader) receiving multiple commands simultaneously (here: waypoint and formation);

- Formation commands refer to the wing leader by OL this way passing him the command to change formation.

9.3. Attack command en route.

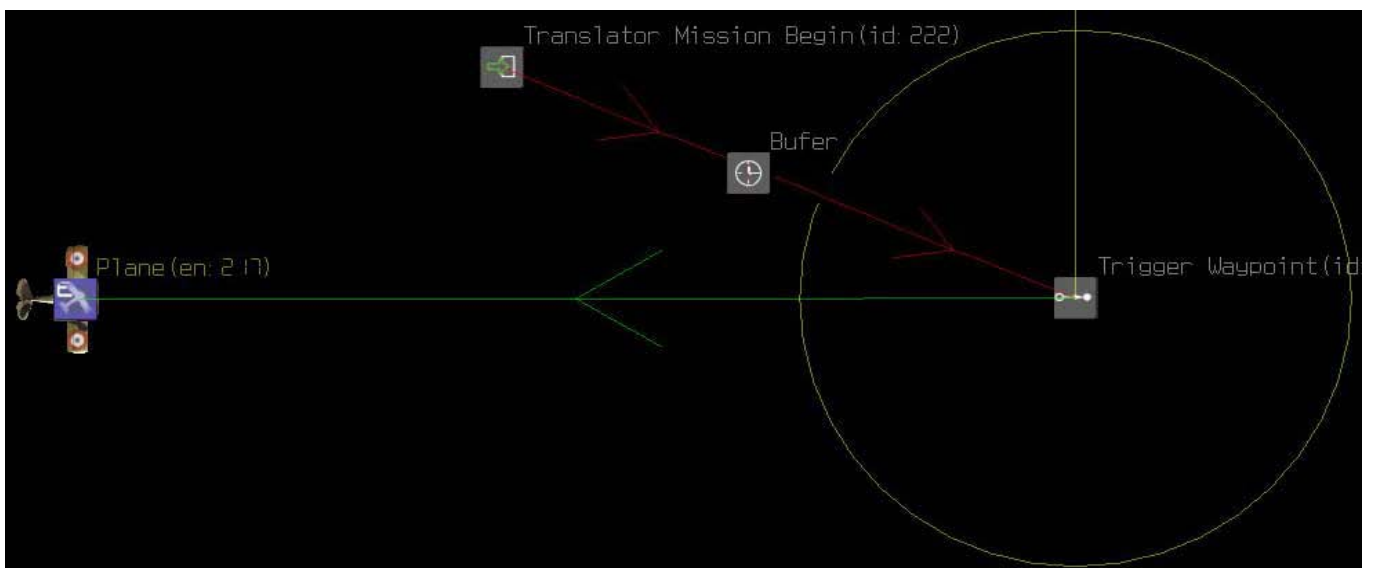


In the given example:

- three planes are united in wing. All commands are tied to the leader (ID:247) through OL (see 6.5.);
- the wing flies through waypoints, at this, Attack command is activated that by OL is connected to the wing leader and by TL points at the attack object.

In this example the group will receive two commands consequently: at first the next waypoint and in a time specified by Timer – a command to attack. The group will execute Attack command (for example: destroy a target) and after it's finished will fly to the waypoint that is left in AI “memory”.

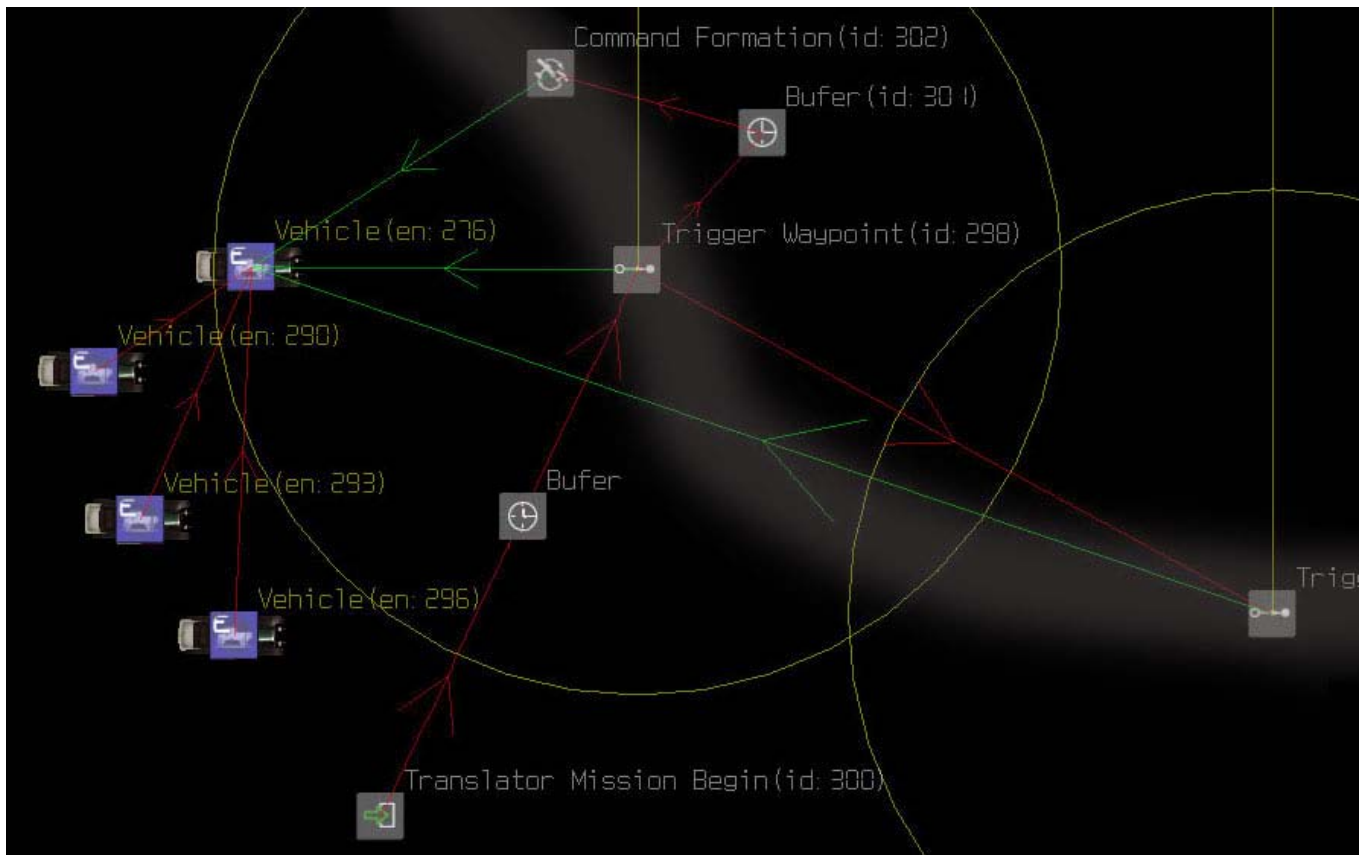
9.4. Airstart.



In the given example a plane after appearing in the air will receive a command to move to a waypoint (Waypoint).

For Airstart of a plane it's basically enough to select "Start in the Air" (see 7.3.1.) in its properties. Don't forget to set the altitude (Position: Y) that it has to appear at (see 7.2.1.). The same applies to building cooperative missions with Airstart.

9.5. Vehicle column on a road.



In the given example:

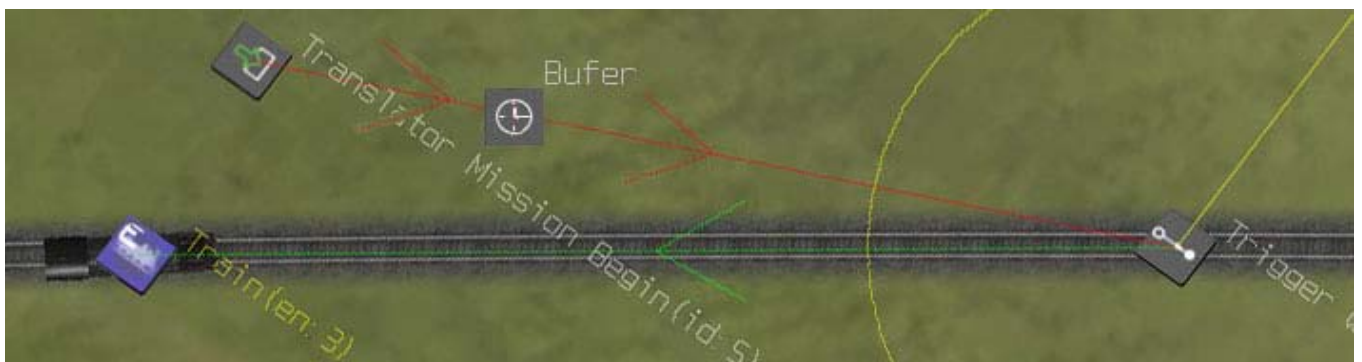
- vehicles are united into a group and will start moving after they receive a command to move to a waypoint (Waypoint);
- upon reaching the waypoint the group will receive a command to change formation after a delay by timer (see 7.6.8.) and if in Formation properties "Vehicle: On Road Column" will be specified then the vehicles will find the closest road (unless mission designer haven't already placed them on a road) and aligning in a column will move to the next waypoint.

Basically, to make any mobile object (Vehicle or Train type) move you need to activate its Waypoint.



In the abovementioned example a Vehicle (or Train) is moving on a road through several waypoints. Upon reaching the first waypoint (id:208) the Vehicle receives a command to move by road (id:213) (in case of a train this command is not necessary) and proceeds to the next waypoint and so on by the sequence.

9.6. Train movement.

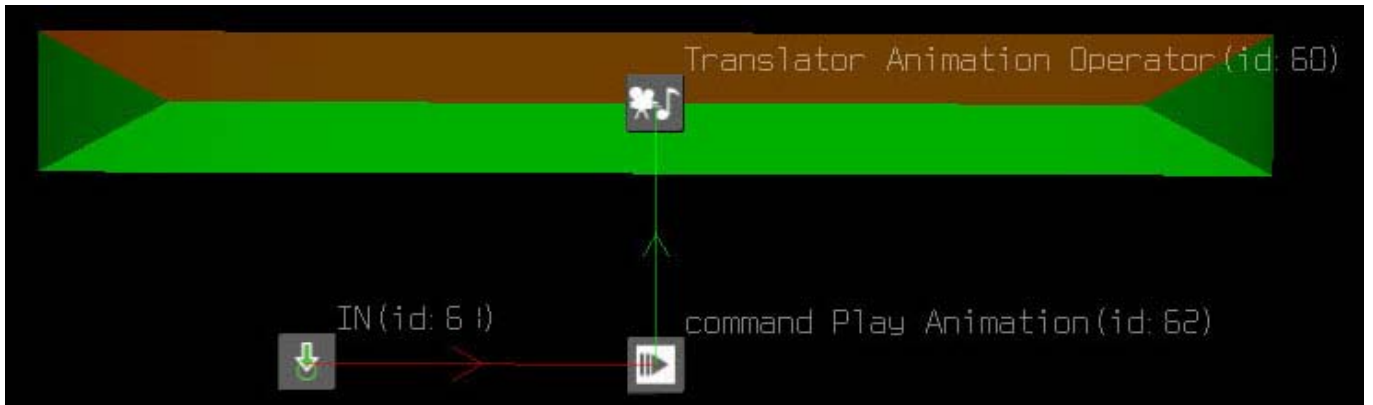


To launch a train (see 7.2.2.) it's enough to place it on the rails and activate its first waypoint.

MANDATORY – place a train and its waypoints on a rail-track.

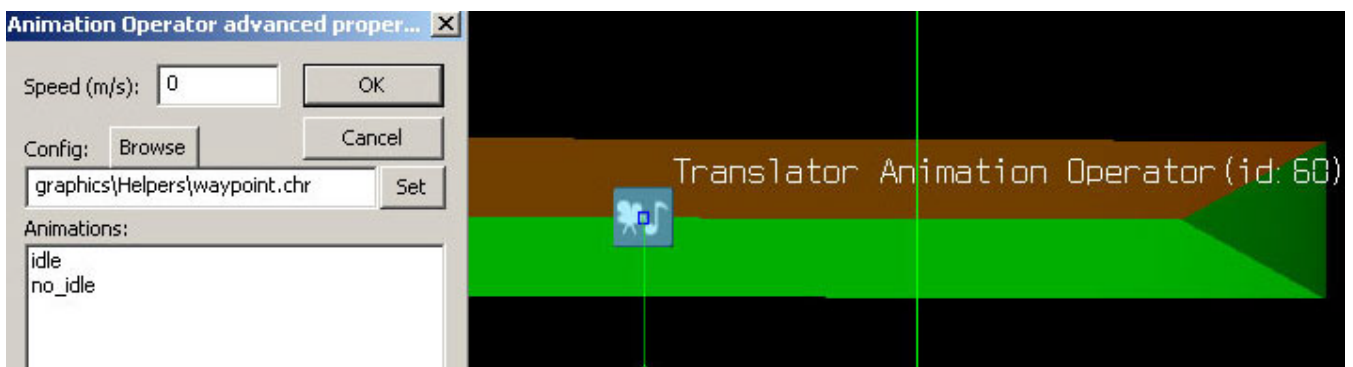
9.7. Creating an animated scene on the example of Helper (a shape in the air).

9.7.1. Animated object is visible right after mission start.



In the given example:

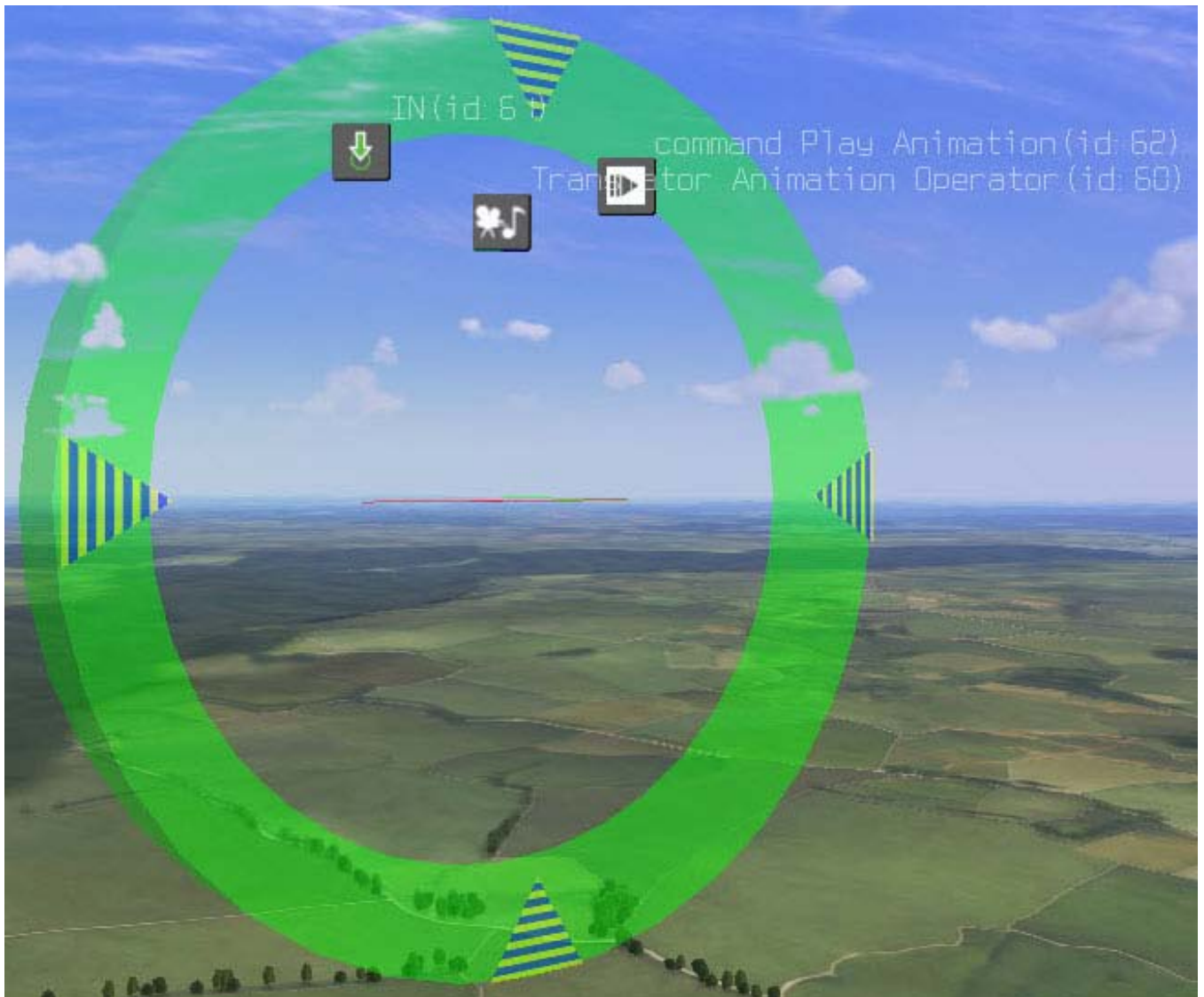
- setting Translator:Animation Operator (see 7.7.1.) and in its properties selecting the scene we want (in this case waypoint.chr or any other *.chr file).



- command:Play Animation (see 7.6.11.) is connected by OL to Animation Operator; inside Play Animation either idle or no_idle is set (i.e. in this case – to play/ not to play).

- IN – any incoming mission event starting Play Animation.

3D view of this shape:



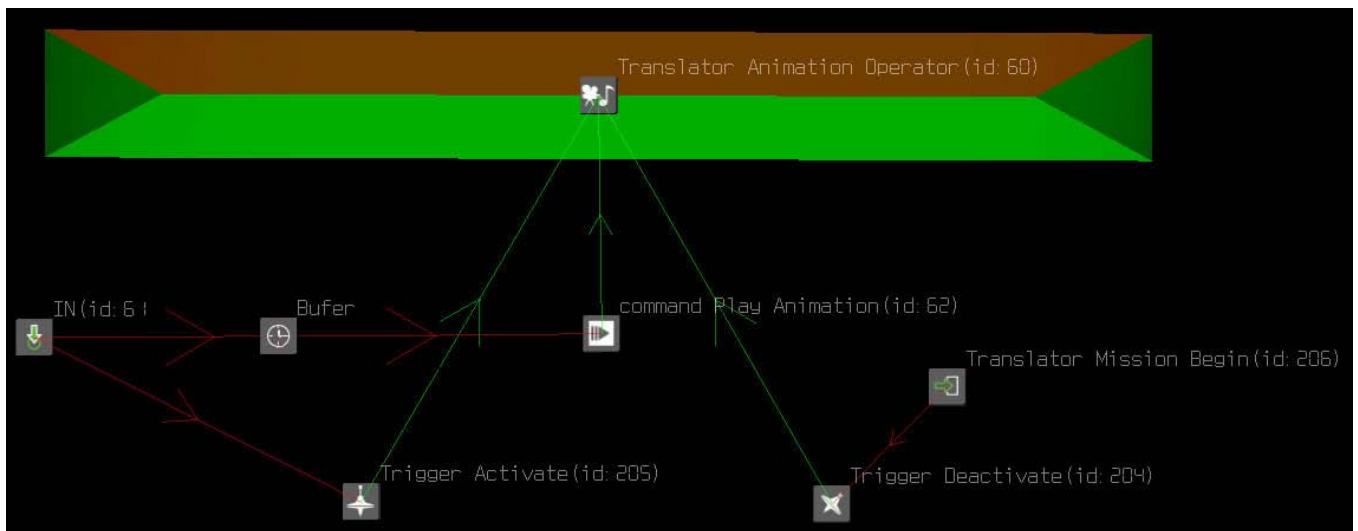
Therefore after Play Animation command is launched the animated parts of this shape will start to move (in this example – striped arrows).

Don't forget to set the altitude of Animation Operator and its orientation.

Also in the Editor you can test-launch animation. For this:

- select Play Animation, right-click and select “Select Object Menu =>Play”;
- after this the view will automatically change to 3D and mouse pointer will change its type;
- it might happen that the view will be pointed not at the Animation Operator, if it is so then simply find the object on the map (see 4.3.).

9.7.2. Animated object is visible NOT right after the mission start.



In the given example:

- Mission Begin deactivates Animation Operator from the very mission start (so the shape won't be visible);
- IN – any incoming mission event that activates Animation Operator and starts its animation (by Play Animation).

Therefore this example differs from the previous one only by its initial realization.

You can also repeatedly turn off (deactivate) or turn on (activate and start Play Animation again) any similar animated object, set Animation Operator.

9.8. Creating an animated scene on the example of a scene from Training Campaign.

To create a scene of such type you need:

- *.MGM file – unloaded 3D scene (the Editor doesn't use it, but it's indirectly subdued to *.chr file);
- *.chr file – file describing scene animation;
- *.cam file – file describing flight trajectory of a camera in the scene;
- *.mp3 file – file containing sound track synchronized with the scene.

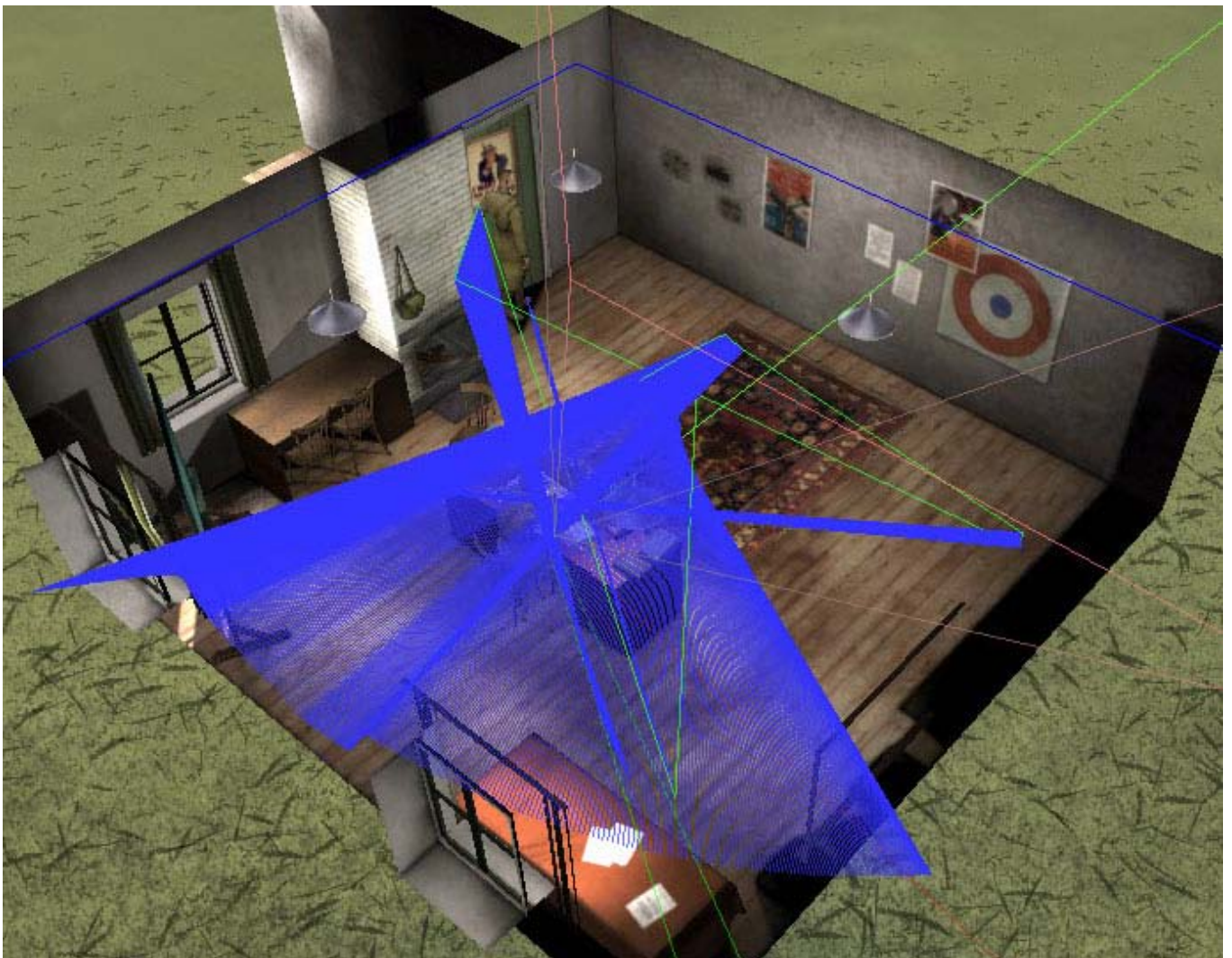
To create such scene you need to use:

- Translator:Animation Operator (see 7.7.1.);
- Translator:Camera Operator (see 7.7.2.);
- Command: Play Animation (see 7.6.11.);
- Command:Play Sound (see 7.6.12.).



In the given example:

- Animation Operator is launched by Play Animation. In Animation Operator properties there is the scene (*.chr – specifying it in “config” menu) and the sound track (*.mp3 – selecting option “config is sample” and in “sound config” menu specifying sound file path);
- Camera Operator is with the adjusted camera config file (*.cam file – specifying path in “3D path” menu, with this not forgetting to select camera type as “self”). The file is started by timer (or any mission event). If camera path is specified properly you’ll see visualized camera trajectory (blue color – camera direction, green – its trajectory).



In the example above Animation Operator and Camera Operator must have the same coordinates, set in Basic Properties (see 7.4.1.). This has to be done since the scene and the camera act together and have the same initial coordinates (in Position menu all three coordinates: X, Y and Z, must be the same).

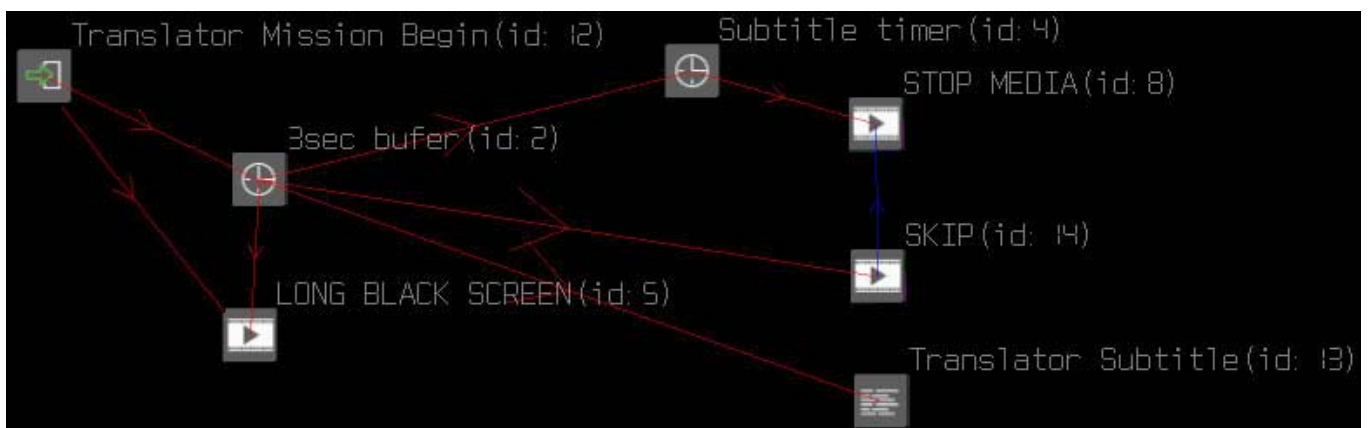
After that you have to launch all scene elements simultaneously (as in this example it's required by design):

- Play Animation (for mission start) and Play Sound (for sound track start) launch Animation Operator (are connected to it by OL);
- Camera Operator is started directly by TL.

9.9. Screen picture during mission start.

The task is to display text on black background during mission start. At this there should be SKIP menu located at the bottom of the screen so that if a player doesn't want to read the text he can skip it (or any other screen picture).

While keeping in mind those restrictions mentioned in paragraphs 7.4.7 and 7.7.8, you can use the following scheme:



- 3sec buffer – timer that launches all mission events with a delay for not to start Subtitle earlier then needed, before the mission pauses after it has loaded (see 7.7.8);
- LONG BLACK SCREEN (Translator Media) with settings: Media Type:Image \ [black picture, min 1 x 1 pixel].jpg and “base time” 10000 seconds, with black background (Background color RGB) [0;0;0], - is started right after mission start (after launch, Mission Begin). In other words, since we, because of the restrictions, can't control time in Media inclusions (after the start mission will pause and Media won't) we will launch very long black background from the mission start.
- Translator Subtitle – mission subtitles that are launched in 3 seconds after mission start;
- STOP MEDIA (Translator Media) – with “Media Type:Stop Media” settings will start with a delay by “Subtitle timer”, which has time delay settings equal to those in Subtitle;
- SKIP – (Translator Media) – with “Media Type:Flash” settings and specified path to a Flash element of “Skip” dialog (swf\skip.swf). Also you need to set the same time in “base time” as in Subtitle to synchronize them between each other.

- SKIP by using On Media Stopped Message (see 7.7.4) «pulls» STOP MEDIA. In other words, if during a mission a player clicks Skip dialog the SKIP will stop all Media – LONG BLACK SCREEN and Subtitle and the player will be able to start playing;
- if the player won't click Skip dialog, then after the time inside Subtitle timer runs out this timer will stop all Media in the mission.

9.10. Basic mission – all towns, airfields and bridges on the map.

Actually, all objects on a map are objects of the Editor, therefore to create such mission you'll need groups of files (*.Group see 6.6.).

All objects of a map are stored in the following *.Group files:

- Base-no-trunc.Group – all large historic cities and airfield;
- Base-for-trunc.Group – all small towns, villages and bridges.

Therefore, in order to get such mission you are going to need to open a blank mission and gradually import these groups. Save the mission after you have finished.

You can also import objects into the already made mission, but be careful – don't import objects twice so that not to create "object inside object".

Since map size is large and the number of objects on the map is even bigger, we advice you to make a template for the mission, which will place objects near the targets (if by mission plot the player isn't supposed to fly far). Such solution will allow you to save PC resources and will considerably lower mission loading time.

9.11. Files that are used in Media and Animation Operator.

9.11.1. Media files in game compilation:

- Campaigns\Training\Images\BlackFade.jpg – black Image (*.jpg) 1x1px
- swf\skip_ratio.swf – SKIP button to be used with 3D engine (considers different screen aspect ratios);
- swf\skip.swf – SKIP button to be used with video/flash clips;

9.11.2. *.chr files for Animation Operator:

- graphics\Helpers\waypoint.chr – round ring with animated arrows;
- graphics\Helpers\arrow.chr – big green 3D arrow;
- graphics\Helpers\descent30.chr – 3D trajectory – 30 degree descent
- graphics\Helpers\descent45.chr – 3D trajectory –45 degree descent
- graphics\Helpers\loop.chr – 3D trajectory – loop
- graphics\Helpers\poi.chr – 3D trajectory – animated waypoint

- graphics\Helpers\turn180.chr – 3D trajectory –180 degree turn
- graphics\Helpers\turn360.chr – 3D trajectory –360 degree turn
- graphics\Helpers\turn90.chr – 3D trajectory –90 degree turn

10. General information

Author of the Manual: Sechnoy Viktor

Designer of the Mission Editor: Michael Lepakhin

Compilers: Sechnoy Victor, Michael Levin, Michael Lepakhin, Daniel Tuseyev

Editing of the Manual: Sechnoy Viktor, Michael Lepakhin, Michael Levin, Adamov Arseniy, Daniel Tuseyev.

Current version: ROF v1.007



© “neoqb” LLC, 2008—2009. All rights reserved.