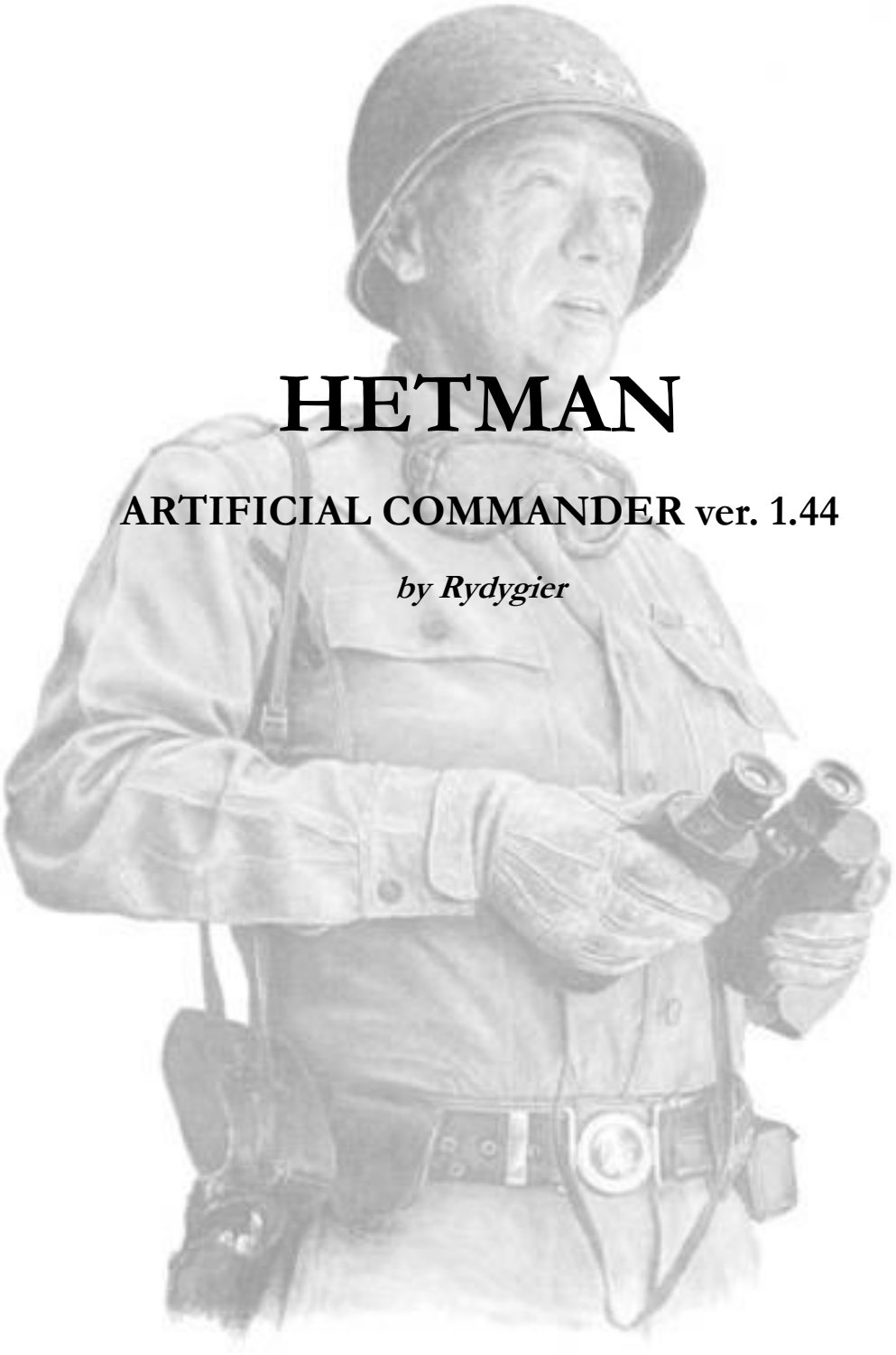


HETMAN

ARTIFICIAL COMMANDER ver. 1.44

by Rydygier

TABLE OF CONTENTS

1. INTRODUCTION

The addon "HAC" is intended to enliven the battlefield the same way a human leader would operate. HAC does not deal with the manner in which orders are executed (unit level), but deals with the issuing of orders. In other words, this addon gives one or both sides of a conflict a field-commander level AI.

One of most important goals of this addon is to keep high compatibility level with other addons, in particular those that extend the possibilities of AI for units and groups. HAC control is mostly based on issuing waypoints, and as much as possible HAC avoids interfering with low-level AI. In other words, HAC focuses on giving orders, rather than on the way in which orders will be executed.

HAC gives new, higher level AI, rather than changing the existing AI. There are only a very few exceptions to this rule, and generally these are optional. HAC should therefore enrich the gaming experience with new features, without conflicts with other addons. HAC is intended to complement their effects rather than to compete with them for control over units (at least for those addons that do not mess with waypoints).

HAC can serve as an instant battle generator as well as a general player opponent, or as base for complex missions and/or gameplay modes. Discussion of these can be found in the HAC thread in the BI Forums:

http://forums.bistudio.com/showthread.php?t=129003

1.1 Sample recipes with HAC ingredient

1.1.1 Pizza & cola 1: the lazy mix

Needed: - 1 piece of HAC for both sides; some units on map; neutral player at position with good view; pizza; cola*.*

How to play: sitting in front of your monitor to eat pizza, drink cola and watch the show.

** By default. Optional. May be replaced with other food.*

1.1.2 Pizza & cola 2: the very lazy mix

Needed: - 1 piece of HAC and 1 piece of some spawning script for both sides; some units on map; neutral player at position with good view; more pizza and cola.*

How to play: sitting in front of your monitor to eat pizza, drink cola and watch the show.

** By default. Optional. May be replaced with other food.*

Example of spawning script: ***Dynamic AI Creator (DAC)*** *by* **Silola***: http://www.armaholic.com/page.php?id=10621*

1.1.3 Battle Chess

Needed: - 1 piece of HAC for AI opponent side and 1 piece of High Command extension for player's side; some units on map; pizza and cola.*

How to play: create a bloody inferno for the opponent… if you can, Napoleon.

** Caution: pizza may cool down and the cola may go flat.*

Example of HC expander addon: ***High Command Extensions*** *by* **DomZ***: http://www.armaholic.com/page.php?id=14961*

1.1.4 Private Player: alone in warzone

Needed: - 1 piece of HAC for both sides and 1 private for player; some units on map; pizza and cola.*

How to play: follow orders, do your best and try to survive. Enjoy being under HAC control. No need to think and plan - what a relief…

** Caution: better to eat before the game, to avoid clogging your keyboard with sauce.*

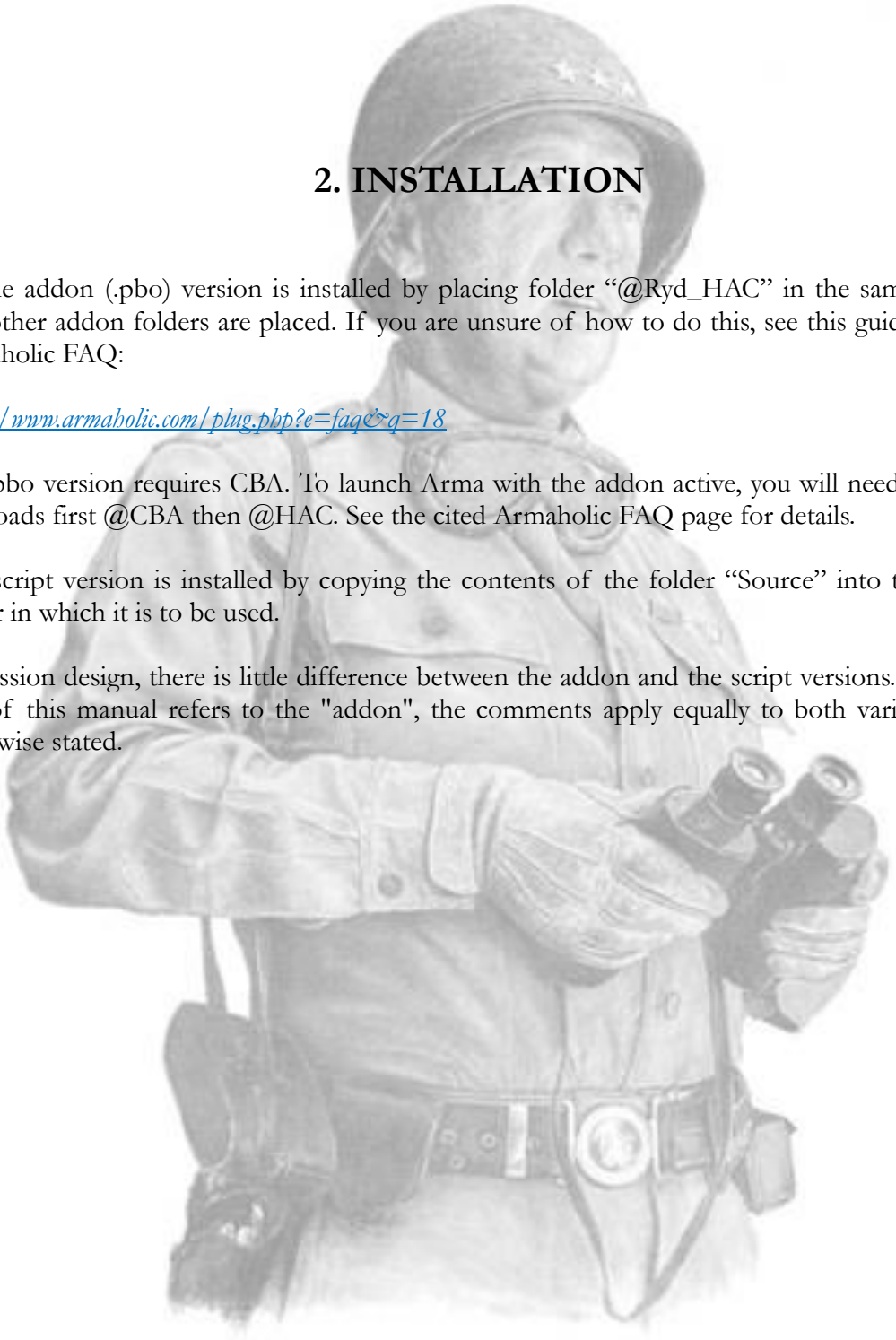
1.2 Some recommended co-addons

It is worth combining HAC with one of available great low-AI level boosters. For only few examples, see:

ASR AI Skills *by* **Robalo***: http://www.armaholic.com/page.php?id=12105*

A2SLX & COSLX *by* **Günter Severloh** *based on* ***SLX mod*** *by* **Solus***:*
A2SLX: *http://www.armaholic.com/page.php?id=15378*
COSLX: *http://www.armaholic.com/page.php?id=15379*

Automated Medic *by* **Demonized***: http://www.armaholic.com/page.php?id=14769*



2. INSTALLATION

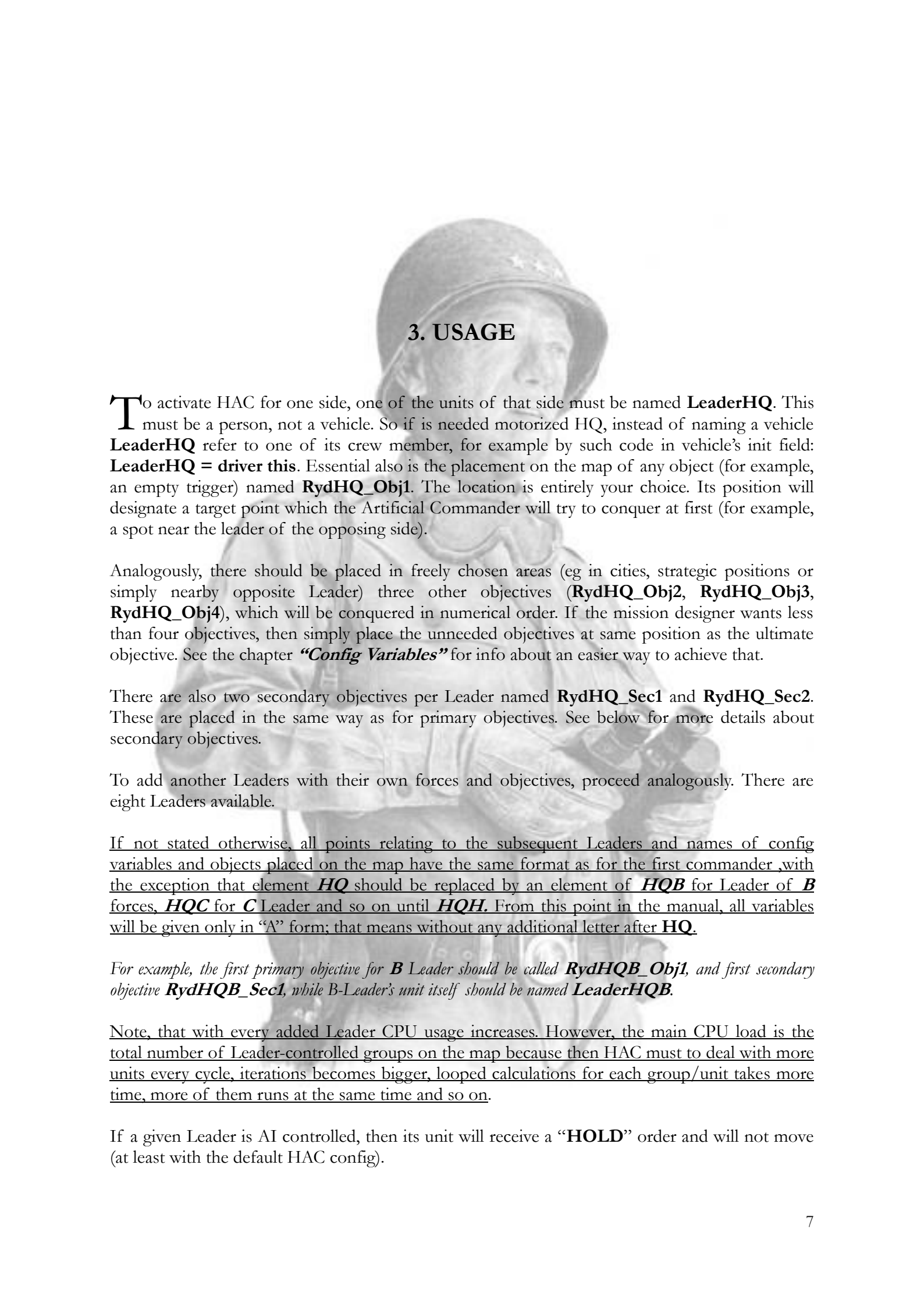
The addon (.pbo) version is installed by placing folder "@Ryd_HAC" in the same way that other addon folders are placed. If you are unsure of how to do this, see this guide from the Armaholic FAQ:

http://www.armaholic.com/plug.php?e=faq&q=18

The pbo version requires CBA. To launch Arma with the addon active, you will need a shortcut that loads first @CBA then @HAC. See the cited Armaholic FAQ page for details.

The script version is installed by copying the contents of the folder "Source" into the mission folder in which it is to be used.

In mission design, there is little difference between the addon and the script versions. Where the rest of this manual refers to the "addon", the comments apply equally to both variants unless otherwise stated.



3. USAGE

To activate HAC for one side, one of the units of that side must be named **LeaderHQ**. This must be a person, not a vehicle. So if is needed motorized HQ, instead of naming a vehicle **LeaderHQ** refer to one of its crew member, for example by such code in vehicle's init field: **LeaderHQ = driver this**. Essential also is the placement on the map of any object (for example, an empty trigger) named **RydHQ_Obj1**. The location is entirely your choice. Its position will designate a target point which the Artificial Commander will try to conquer at first (for example, a spot near the leader of the opposing side).

Analogously, there should be placed in freely chosen areas (eg in cities, strategic positions or simply nearby opposite Leader) three other objectives (**RydHQ_Obj2**, **RydHQ_Obj3**, **RydHQ_Obj4**), which will be conquered in numerical order. If the mission designer wants less than four objectives, then simply place the unneeded objectives at same position as the ultimate objective. See the chapter ***"Config Variables"*** for info about an easier way to achieve that.

There are also two secondary objectives per Leader named **RydHQ_Sec1** and **RydHQ_Sec2**. These are placed in the same way as for primary objectives. See below for more details about secondary objectives.

To add another Leaders with their own forces and objectives, proceed analogously. There are eight Leaders available.

If not stated otherwise, all points relating to the subsequent Leaders and names of config variables and objects placed on the map have the same format as for the first commander ,with the exception that element ***HQ*** should be replaced by an element of ***HQB*** for Leader of ***B*** forces, ***HQC*** for ***C*** Leader and so on until ***HQH.*** From this point in the manual, all variables will be given only in "A" form; that means without any additional letter after **HQ**.

*For example, the first primary objective for **B** Leader should be called **RydHQB_Obj1**, and first secondary objective **RydHQB_Sec1**, while B-Leader's unit itself should be named **LeaderHQB**.*

Note, that with every added Leader CPU usage increases. However, the main CPU load is the total number of Leader-controlled groups on the map because then HAC must to deal with more units every cycle, iterations becomes bigger, looped calculations for each group/unit takes more time, more of them runs at the same time and so on.

If a given Leader is AI controlled, then its unit will receive a "**HOLD**" order and will not move (at least with the default HAC config).

For the script version only, to initialize HAC the following code should also be executed in some way, e.g., by placing the following script in the init field of any object (for example, in the activation field of an empty trigger or waypoint):

nul = [] execVM "RydHQInit.sqf";

*(note: pay attention to the type of quotes, these must be double - " - **not** similar quotation mark like - ”).*

Alternatively, that line may be placed at the end of a mission's init.sqf.

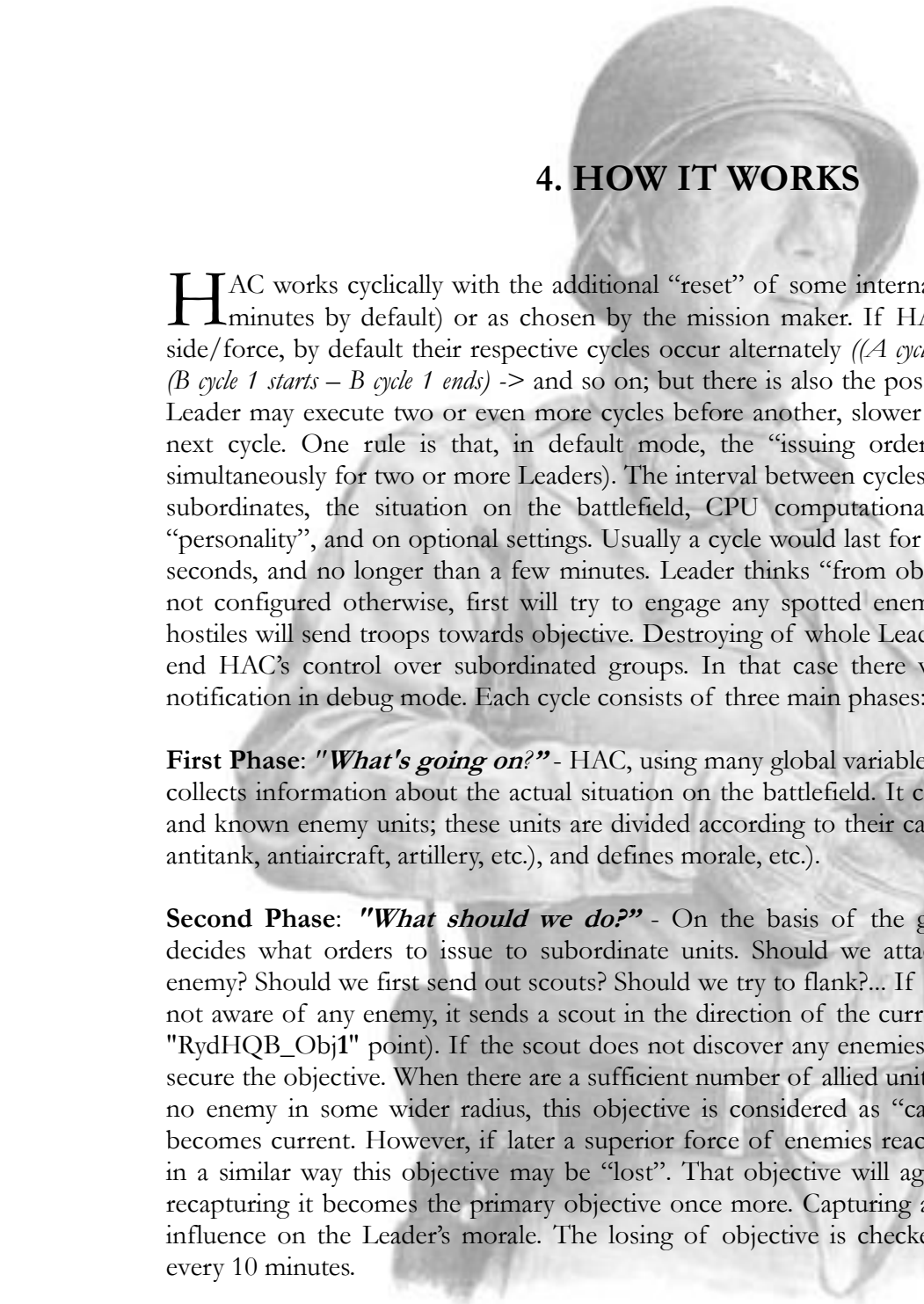
HAC will start working by default after about 15-30 seconds from the start of the mission, and will control all units of given side and their allies (except civilians) in default mode; or only defined groups in “limited control” mode (see below). “Limited control” is particularly useful, when you need more than one Leader for one side. HAC will take command over both fighting and support units (ambulances, ammo trucks, etc.) and will automatically assign them tasks appropriate to the situation on the battlefield.

If a Leader’s unit dies, the next most senior unit from its group will become a new Leader, but with worsened commanding attributes. The death of a last member of Leader’s group means the end of HAC's control of the mission for given force - that Leader's army has lost its "brain” (so this “staff” group may be compared to a “King” piece in chess).
Many HAC settings are optional and may be easily customized by the user in the init config, and executed in a similar way as for the code for the initialization of the script version. This will be discussed in chapter ***“Config Variables”.***

Additionally user has to his disposal three (per each Leader) special empty trigger objects, so called “decoys” for idle (except support), land support and resting missions (see 5.6.8, 5.6.10 and 5.6.11). Depending on their settings idle, support (when idle) or resting groups will choose marked by this triggers areas as mission destination areas for sure or with some probability. Simply on the map may be placed in chosen spot a trigger and named **RydHQ_IdleDecoy, RydHQ_SupportDecoy** or **RydHQ_RestDecoy**. Chance of choosing marked such way areas by each relevant group can be customized via init variables (**RydHQ_IDChance, RydHQ_SDChance** and **RydHQ_RDChance** see chapter ***“Config Variables”***), that by default are equal to 100 (percent). Bigger of the trigger’s axis (recommended are same values for both axis, must greater than 0) will determine max placement distance in meters from the trigger’s position (square area). Shape (ellipse/rectangle) of the trigger will determine, if known nearby enemy presence should affect (forbid) choosing of marked area by relevant groups. Rectangle means “should affect”, ellipse – on the contrary.

HAC is primarily intended for SP game play and there is no any guaranty of MP compatibility, however should be fully functional as server-side AI opponent. To make it compatible with client-side human controlled team leaders (visible over the net communication with HQ and briefing task entries) should be placed on map function module. Fulfilled tasks in MP are not deleted, but marked as done. There is also sever check at the init to avoid multiple launches per each client.

Important note for Big Boss mode in MP: to make this mode working in MP environment, must be used described in other part of this manual customized battlefield area via placing on map a square trigger named **RydBB_MC**.



4. HOW IT WORKS

HAC works cyclically with the additional "reset" of some internal values at a given time (10 minutes by default) or as chosen by the mission maker. If HAC controls more than one side/force, by default their respective cycles occur alternately *((A cycle 1 starts – A cycle 1 ends)* -> *(B cycle 1 starts – B cycle 1 ends)* -> and so on; but there is also the possibility that a quick-thinking Leader may execute two or even more cycles before another, slower Leader will be ready for its next cycle. One rule is that, in default mode, the "issuing orders" phase will never occur simultaneously for two or more Leaders). The interval between cycles depends on the number of subordinates, the situation on the battlefield, CPU computational power, the commanders' "personality", and on optional settings. Usually a cycle would last for no less than several tens of seconds, and no longer than a few minutes. Leader thinks "from objective to objective", but, if not configured otherwise, first will try to engage any spotted enemy, and only after defeating hostiles will send troops towards objective. Destroying of whole Leader's group will permanently end HAC's control over subordinated groups. In that case there will be displayed on screen notification in debug mode. Each cycle consists of three main phases:

First Phase: ***"What's going on?"*** - HAC, using many global variables (mostly of the array type), collects information about the actual situation on the battlefield. It creates a list of subordinates and known enemy units; these units are divided according to their categories (infantry, tanks, air, antitank, antiaircraft, artillery, etc.), and defines morale, etc.).

Second Phase: ***"What should we do?"*** - On the basis of the gathered information, HAC decides what orders to issue to subordinate units. Should we attack? How to attack a given enemy? Should we first send out scouts? Should we try to flank?... If the Artificial Commander is not aware of any enemy, it sends a scout in the direction of the current objective point (e.g., the "RydHQB_Obj**1**" point). If the scout does not discover any enemies, a group of units is sent to secure the objective. When there are a sufficient number of allied units near (at) the objective, *and* no enemy in some wider radius, this objective is considered as "captured"; the next objective becomes current. However, if later a superior force of enemies reaches that captured objective, in a similar way this objective may be "lost". That objective will again be the primary one; i.e. recapturing it becomes the primary objective once more. Capturing and losing objectives has an influence on the Leader's morale. The losing of objective is checked every "reset", by default every 10 minutes.

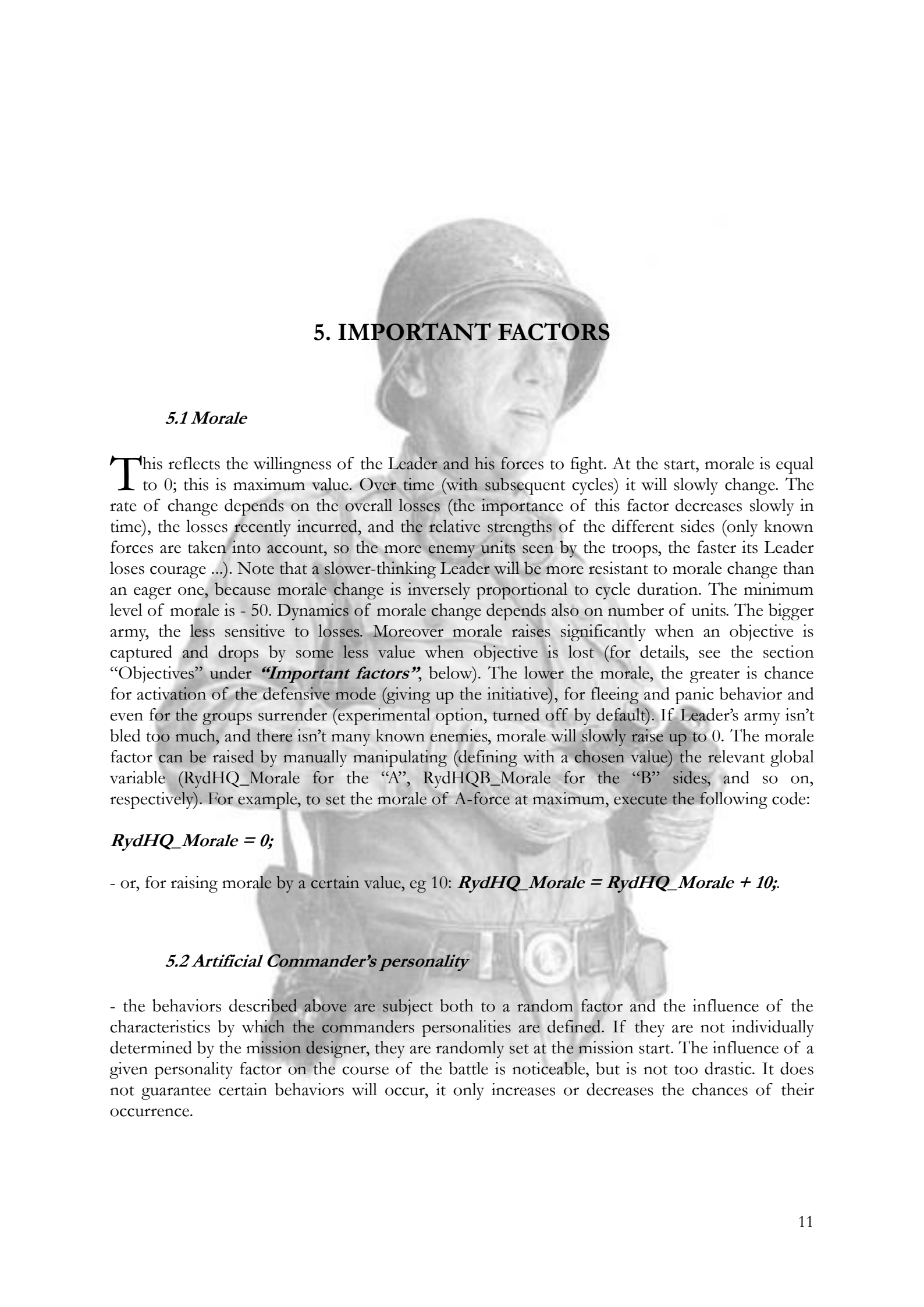
When enemy forces are detected, HAC will attempt to destroy them by sending the appropriate number of groups of the appropriate type(s). By default three infantry groups and/or two armored and/or one air against one enemy group of any nature. So, newly detected enemies will likely result in HAC ordering three *INF*, two *ARM*, and one *AIR* to the location in an attempt to maintain at least a 3:1 strength ratio. These will try to outflank and destroy the enemy group. In

addition, HAC will retain a certain amount of forces as a reserve. At an appropriate moment part of this reserve may be used for large-scale encirclement of the entire known enemy's lines. The end of this phase means the end of the cycle.

- Note that there is a good chance that the Artificial Commander will not send armored units into the attack if he knows that enemy anti-tank weapons (such as ATGM units) are present in the area. Similarly, HAC will be reluctant to send air units if the presence of AA weapons was detected. HAC will also not be too willing to send units into areas where they perform poorly, e.g., armored units in cities or forests.

- If HAC concludes that enemy has a sufficiently large advantage, and when his own morale has become sufficiently low, the chances are high that the army will go into defensive mode. Units will be deployed in selected positions between the Leader's position and the enemy, or, if the enemy forces are too close or when their locations are unknown, directly around the leader unit. The Commander may decide to place them in positions on the near reverse slope, i.e. hidden in ambush. HAC will behave similarly if a defensive mode is manually set.

Third Phase: ***"Yes sir!"*** - Now scripts are activated that send troops to appointed positions. Waypoints are assigned, usually the 'Search and Destroy' type, in front of and at the sides of the enemy position(s) to be attacked. The distance of those waypoints from the target position depends on the type of the friendly attacking unit. The cycle ends when the last command is given (note: not when the last one is completed).



5. IMPORTANT FACTORS

5.1 Morale

This reflects the willingness of the Leader and his forces to fight. At the start, morale is equal to 0; this is maximum value. Over time (with subsequent cycles) it will slowly change. The rate of change depends on the overall losses (the importance of this factor decreases slowly in time), the losses recently incurred, and the relative strengths of the different sides (only known forces are taken into account, so the more enemy units seen by the troops, the faster its Leader loses courage ...). Note that a slower-thinking Leader will be more resistant to morale change than an eager one, because morale change is inversely proportional to cycle duration. The minimum level of morale is - 50. Dynamics of morale change depends also on number of units. The bigger army, the less sensitive to losses. Moreover morale raises significantly when an objective is captured and drops by some less value when objective is lost (for details, see the section "Objectives" under ***"Important factors"***, below). The lower the morale, the greater is chance for activation of the defensive mode (giving up the initiative), for fleeing and panic behavior and even for the groups surrender (experimental option, turned off by default). If Leader's army isn't bled too much, and there isn't many known enemies, morale will slowly raise up to 0. The morale factor can be raised by manually manipulating (defining with a chosen value) the relevant global variable (RydHQ_Morale for the "A", RydHQB_Morale for the "B" sides, and so on, respectively). For example, to set the morale of A-force at maximum, execute the following code:

RydHQ_Morale = 0;

- or, for raising morale by a certain value, eg 10: ***RydHQ_Morale = RydHQ_Morale + 10;***.

5.2 Artificial Commander's personality

- the behaviors described above are subject both to a random factor and the influence of the characteristics by which the commanders personalities are defined. If they are not individually determined by the mission designer, they are randomly set at the mission start. The influence of a given personality factor on the course of the battle is noticeable, but is not too drastic. It does not guarantee certain behaviors will occur, it only increases or decreases the chances of their occurrence.

5.3 Surrender of groups

- this may occur for groups in panic, when morale is low, and advantage of the enemy is large enough. When group surrenders, units give up their weapons and are switched into the "captive" state. Vehicles are unassigned and abandoned. Foot soldiers will raise their hands for a while. Surrendering is experimental feature, turned off by default.

5.4 Cargo System/infantry movement of long distances

- this is an optional feature, turned off by default, that gives to HAC the handling of cargo vehicles (all that are able to carry passengers). With this system, HAC may use for transport purposes not only vehicles already grouped with infantry, but also any vehicles placed on the map as separate groups (e.g., a truck with driver) or as empty vehicles.

If some infantry group without its own vehicle receives orders for an attack/flank/capture mission far away (1000 meters or more), they will look for some sufficiently capacious empty vehicles nearby (the search radius is customizable). If such a vehicle is found, it will be permanently assigned to that group. If there is no empty vehicle available, HAC will look for a temporary vehicle with sufficient cargo space for that group (land cargo vehicles should be 1500 meters or closer to given group; for choppers this radius is equal 4500 meters). If found, such a vehicle will drive to the group's position, the group will get in, then the vehicle will drive/fly towards the group's target area. The group is disembarked to follow with its mission, and the cargo vehicle returns to its initial position. Once returned, it becomes available for other groups. There are some additional rules with distant movements: if there is no available transport, and the target point is distant enough, the group will move with limited speed in column formation or (when enemies are known to be nearby the route, the target position or the Leader position) with normal speed but with some rest both at the destination point and midway to it. Withdrawal movements are always conducted with normal/full speed. If enemies are known to be near the target point, unarmed cargo vehicle will disembark group at the halfway point. Unarmed land transports will also disembark the group if enemies are encounter en route. This system's code is quite complex, and the system itself is sensitive to a given unit's AI behavior; so effects may sometimes prove less than perfect, especially with aerial transport. For that reason, there is an option to turn whole system off entirely, or turn it off selectively for either land or aerial cargo vehicles. To exclude given empty vehicle from HAC's auto-assigning, put following into its init field:

this setVariable ["Hired",true];

5.5 Medical and Logistical Support System

- another optional feature, turned on by default, that gives to HAC the handling of support vehicles (medevac, ammo, fuel and repair). With this system HAC will send support units of appropriate kind into areas where such support is needed. This cooperates with generic Arma support handling – if a support unit is within a 500 meter radius from a unit, that needs assistance, that unit's leader will be aware of available support and will send units towards the support vehicle(s), and also the support vehicle will relocate towards those units. Such support

vehicles will with time also gather behind the front line as some kind of temporary support base. Into that area will also be gathered combat ineffective-groups withdrawn from the fight.

Support vehicles are used with some rules: usually these will not be sent to points near known enemy positions, nor will they be sent to areas where support of given kind is already present (subject to the radius values). There are some possibilities for customization. These will be described in the chapter ***"Config Variables"***, but generally there is the possibility turn off each kind of support separately, or to make chosen groups be unsupported. Further, there is an option of "extended support mode". In this mode support units may be called to any location on the map regardless of the situation; this is also based on some generic Arma features. Note: because units without fuel or otherwise immobilized are unable to move to a support vehicle, and support vehicles often stop too far from immobilized vehicles, there is a small 'cheat' incorporated that makes immobilized vehicles able to move after some slight auto-repair, and/or be auto-supported with a minimal amount of fuel if a support vehicle is within a given radius (about 50 meters). In this way, support runs smoothly.

Infantry medic units' support is not handled by this system.

5.6 Possible Missions

HAC will issue orders as required to every controlled group. There are many possible kinds of orders, and these are used depending on current needs:

5.6.1 Recon mission

if at the start there is no known enemy, these will be the first missions ordered by as Leader. A group sent with recon mission orders will search a designated area near the current objective and will avoid combat. After reconnaissance it will return. External cargo will be not assigned. An infantry group will not have any transport assigned to it; if a light vehicle is sent on recon, it will not have troop cargo assigned to it, unless recon infantry group has its own vehicle. More recon missions can be issued at night;

5.6.2 Attack mission

the basic offensive mission, ordered when there is a known enemy unit on the map which has not yet been attacked with sufficient force. The group will move towards a point situated in front of the enemy or on one flank; it then is given a "search & destroy" waypoint directly at the enemy position. Parameters, way of execution and possible path of approach varies depending on the kind of group (infantry, armored/mobile, snipers (no charge, quick withdraw after few shots - only when group consist no more, than two members), special forces (wide flanking, valuable targets only), aerial (without flanking movement)). Transport may be assigned;

5.6.3 Capture mission

this kind of mission may be assigned when a recon finishes after the last "reset", and there is no or few known enemy at an objective. The attacking group will flank the objective area, then receive a "SAD" waypoint at the objective point. If successful, the group will stay there for some minutes (customizable; 10 minutes by default) or till

objective status change to taken, before it will be available for another mission. Transport may be assigned.

5.6.4 Flanking mission

some reserve groups may be sent to outflank the entire known enemy formation. This mission consist of four waypoints: three "move" and the fourth, "SAD", set on the center of the enemy formation. "Move" waypoints are calculated in a way that creates a route providing a good chances of outflanking enemies without being detected/engaged, and brings the groups to a position suitable for an attack from behind. Transport may be assigned.

5.6.5 Defend mission

when "defend" mode becomes active, groups are ordered to take defensive positions in a defensive perimeter area. This area's size is dependent on the number of available groups. Its location and shape are dependent on the situation (distance between Leader's unit and enemy, knowledge about enemy). Land combat groups will take positions on reverse slopes, near the Leader's unit, or near closest taken objective and hold them until new orders arrive. Air units will provide aerial cover from above, and recon units will be placed at positions with good fields of view towards the enemy (or random direction, if enemy positions are unknown). Unarmed cargo and support units take up positions behind the Leader's unit. Some units may take guarding positions at nearby roads. Transport will be not assigned.

5.6.6 Defensive reserve mission

Some of the groups in "defend" mode will take position near the Leader's unit, but when the enemy attacks on some point becomes sufficiently threatening, some of these reserves will be sent in as reinforcement. Transport will be not assigned.

5.6.7 Garrison mission

Chosen groups may form a garrison. For infantry, if there are some empty static weaponry nearby, some group members (except snipers/recon) will get in. Some other members will look for suitable positions inside nearby buildings. Rest (at least team leader) will patrol randomly chosen nearby buildings, including interiors. By default motorized and mechanized groups will only get sentry waypoint at their current position. Same, if there is no any buildings in about 200 meters radius. After each fight garrison should be re-deployed. Garrisoned units are excluded from any other missions except attack missions near to its garrison area (500 meters by default). After any such mission, they will return to their initial position. Transport will be not assigned.

5.6.8 Support mission

Only for support vehicles. They will receive "move" waypoints (or "support" waypoints in extended support mode) in areas where such support is needed, and where there is not already a support vehicle of the same kind; usually providing that there is no known enemy near (not always, so situations may occur in which support will be sent dangerously close to the fighting). The vehicle will stay there for future needs in that area. Troop cargo will be not assigned to support vehicles. Very special kind of support mission is ammo drop. To prepare such missions Put group name of chosen (any) air unit(s) into

RydHQ_AmmoDrop config init array, set on map circular trigger named **RydHQ_AmmoDepot**, and within its radius place chosen number of ammo boxes, that should be used during this mission. This aerial groups will be used only to support tasks.

5.6.9 Cargo mission

Certain kinds of vehicles, those able to carry passengers, may receive such order when transport is needed somewhere. See "***Cargo System/infantry movement of long distances"*** section for more details.

5.6.10 Rest mission

If group becomes "combat ineffective" due to lack of ammo or fuel, due to damage, or due to casualties and injuries, then it withdraws from the combat zone with full speed to "rest positions" for regrouping. If (optimal) near support bases, smoke is used to cover the withdrawal movement (optional, active by default). This mission will end when the group becomes combat effective again. There is an option, inactive by default, that allows for pairs of such groups, when without vehicles and under 50% of nominal strength, and if close enough, to combine together into one group. Transport will be not assigned.

5.6.11 Idle mission

If a group has no any other mission assigned, it will receive temporary "idle mission" orders and be held in reserve. This mission will have "guard" or looped "patrol area" waypoints either near the current group position, near the Leader's unit (as all special force groups), near one of the secondary objectives, or at some nearby road. This kind of mission can be interrupted at any time by other mission order (with special rules for SF groups). Transport will be not assigned.

5.7 Objectives

- the presence of objectives on the map makes Leaders act dynamically and aggressively even if there is no known enemy on the map. For each Leader ,there are four primary objectives and up to two (optional) secondary objectives (see **USAGE** chapter). The goal of each Leader is to capture all its primary objectives in numerical order (i.e., the numbers included in the names of the objectives.. Thus the Leader will start with attempts to capture its first objective (e.g., **RydHQ_Obj1**). If that objective is attained, the Leader will then try to conquer the area near next objective (e.g., **RydHQ_Obj2**) and so on. The conditions that must be met in order for a given objective to be classed as "captured" are customizable. By default there must be at least 25 allied units in 300 meter radius from the objective, and very few or no enemy units in a 500 meter radius. If so, the objective becomes "captured", the next one becomes current and morale increases by 20. To recognize a given, previously captured, objective as "lost" (with all others following, so if the second objective is lost, so also are the third and four no longer "captured", if they were), there must be more enemies in 300 meters radius than allied units in 500 meters radius from the objective. If an objective is lost, morale drops by 5.

The status of objectives is checked every "reset"; by default this occurs every ten minutes. There is an option, turned off by default, that the Leader's unit will move towards the last captured objective with limited speed (and retreat with normal speed, if the objective is lost). This gives to the Leader better control and the possibility of quicker responses to situational changes (reserves

are moved forward with the Leader's unit - closer to the front). However, this may prove risky for Leaders. By default Leader units hold their initial positions all the time, so are safer; but as the offensive progresses, reserves get stuck far from the front line.

Secondary objectives may becomes targets for some "idle missions" (patrols and guarding for units without orders), so these may be placed at positions where the mission maker wants to create some random chance for arrival/relocation of some units during the game, and hence a random chance of encounters.

5.8 Mission Notifications/missions for player-controlled groups

- when the team leader of a given group is controlled by a human player, with every new mission assigned the player receives notifications: a special marker on the map with name of the team leader and the type of mission; a journal entry with a short mission description; and an on-screen message with some sound. There is also an experimental option that extends this with some radio "mini-dialog" between HQ and team leader. This is inactive by default, and is recommended only for USMC faction, although is available for all factions. Generally, HAC expects the player controlled group to follow orders, like any other AI controlled group, but of course, the player may have another plans… An uncompleted mission is called off when groups stay without moving long enough (there is speed test conducted every about six seconds for team leader units). So if you want to receive another mission, just stay still for some minutes. This test also works as a backup system for AI groups, if a group gets stuck for some reason.

5.9 Custom defense

- HAC has ability to dynamically pass from offensive to defensive stance and vice versa. Mission maker however sometimes may want to set up mission, where one side will be constantly in defense mode. In such case can be useful ability of customization defense perimeters. Line of defense may be constructed based on five areas: area around Leader, and four areas around each considered as taken objective. By using of relevant config variables described in chapter 7. mission maker can to adjust, how many groups should be the closest for objective to make it defense center and in which direction (North-South, East-West and intermediate) front of given perimeter should be faced (instead of normal way, where front is set up towards known enemies or randomized if no enemy was spotted). This direction control works also for normal, autonomously set defense. Thanks to this is possible for example to set up a round defense around Leader's position. See example config for details. NOTE: objective must be considered as taken. To achieve that for custom defense missions, use internal HAC's variable, that indicates progress in taking objectives: **RydHQ_NObj = 1**. Value range there is 1 to 5, where 1 means, that HAC is about to conquer his first objective, and 5 – that all objectives are already taken.

5.10 Limited Control Modes/several commanders per side/fronts

- in normal/default mode the Leader will control all allied units except civilians. However, sometimes mission makers may want to add additional Leaders for one side; or may simply want that the Leader controls will be only part of the allied forces. There are two ways for exclude chosen groups from the Leader's control; these are described in the chapter "***Config Variables***".

There are three, non-exclusive ways (which may be combined) of defining units that are subject to the control of HAC in selective mode:

1. Using synchronization of the Leader's unit with selected team leaders
2. By naming chosen team leaders: "**Ryd***(Side letter)*" + number (1-100 default, the maximum number can be changed. For example "**Ryd10**" or, for B side, "**RydB35**")
3. By placing a **group name** in the "included" array (there is also the reverse, an "excluded" array, which works with all modes and must also contain **names of groups**).

The variables that limit the scope of control are discussed in the section ***Config Variables***" , below.

HAC also gives the possibility to divide the map into so-called "fronts". To achieve that, mission makers should place in the editor empty triggers with the desired size, shape, direction, and position. Their boundaries will define front boundaries. The name of such a trigger will assign the given front to a given Leader: **HET_FA** means a front for A-Leader, **HET_FB** for B-Leader and so on, until **HET_FH** for H-Leader. Fronts may share some of an area; also there may be areas on map not covered by any front. It is necessary to turn on the "front" mode variables (see config variables list, below). But how does this mode work? Simple. A Leader with an assigned front will pay attention only to enemy forces located inside the assigned front area. However, groups controlled by him may be located outside the front area, and/or that Leader may assign to 'his' groups waypoints outside the front area, for example to flank an enemy located within the front. Forces controlled by the Leader may sometimes attack enemies outside the front, but only if when carrying out their mission come across such enemy groups by chance and decide to engage.

5.11 Artillery fire missions

- Leader will use vanilla artillery pieces added under his control for most kinds of fire missions available in Arma 2 (HE, WP, SADARM, SMOKE, ILLUM).

For HE, WP or SADARM missions there is separate set of debug markers to show on map, which battery is firing, where aims, how big is salvo's drift and dispersion (so where is final impact area). Drift and dispersion are dependent on several factors, as battery crew (leader) skills and condition, weather, kind of artillery or distance. There is also shown kind of used ammo and current time of flight (TOF). HAC will choose most tempting target for each used battery, will take into consideration target's movement vector and other factors, will try to avoid "friendly fire" (still some risk exist, also Leader will know only about friendly groups, that he controls) and will pick appropriate kind of ammo depending on type of target and available ordnance. Of course target must be located within artillery range. Artillery by default will not fire at targets, that are located too close to the friendly forces. Safe zone radius is dependent on setting, kind of guns and weather. Default for mortars for best weather is 100 meters. Number of ammunition per battery may be customized (see chapters ***7*** and ***9*** for arty config variables). Each artillery group (battery) must consist of guns of same kind. Guns of given battery should be placed not so far to each other (eg 30-60 meters should be fine). Here you can find useful info about artillery ranges and available for given kind ammunition:

http://community.bistudio.com/wiki/Artillery_Module

Aside throwing/shooting smoke grenades group, for withdrawal concealment, will now often call arty smoke mission, if available. During the night, defending troops, aside using 40mm flare grenades, will know call for artillery flare illumination, when enemy is spotted. For smoke arty mission howitzers (M119,D-30) in range are needed, for illumination – howitzers or mortars (M252, Podnos). See ***"Config Variables"*** for init details.

Internal HAC's artillery handler is optional. Can be disabled via config variable and, for example, some external handler may be used instead.

5.12 Debug Mode

- when activated, it shows the course of events on the battlefield through entries in the .rpt file or on screen (commanders attributes, cycles, and morale changes info about surrendering the army); and by markers (dots) on the map. There are two debug levels for regular HAC and another two for BB. The markers are as follows:

Red dot: points of attack. ***Inf*** - infantry and cars; ***air*** - air force; ***arm*** - armor; ***rec*** - scout; ***snp*** - snipers; ***cap*** – target points for groups sent to occupy objective(s)

Orange dot: shows stages of the routes of flanking forces;

Black dot: in defensive mode, denotes good observation points for reconnaissance and for forces set in ambush

Brown dot: shows occupied defensive positions. ***Rec*** - reconnaissance; ***air*** – air units; ***LMCU*** - "land mobile combat units"

Green dot: indicates center of defense perimeter

Color change to ***blue***: indicates completed order

Small, half-transparent red dot with "Res" mark – indicates idle mission;

Small, green triangle with side mark only – shows rest mission;

Small, brown square with "Garr" mark – garrison;

Red, half transparent dot with "DRes" mark – defensive reserve;

Small, green dot only with side mark – observation direction for groups in defensive positions;

Khaki cross – medevac support mission;

Green/orange/red dot with number – mark on each group leader's position with numerical "danger" factor. If high enough, also color will change;

Khaki rectangle – indicates front boundaries for each Leader;

Khaki circle with arrow – other support mission. If an arrow is above the circle – refuel; if beside – reammo; if beneath – repair;

Yellow dot with Leader's letter and "!" sign – indicates positions, where a group has a panic status;

Pink (purple) dot with Leader's letter and "!!!" sign – indicates positions, where a group surrendered;

Red/blue/yellow circle with number – part of BB debug, This will mark strategic objectives areas on map, where radius and number indicates value, and color – status of objective for this BB (taken by OPFOR/by BLUFOR/not taken);

Black squares with numbers linked by pink lines – part of BB debug, current offensive path for each BB-controlled Leader;

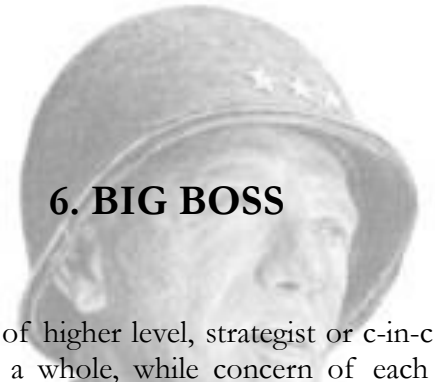
Blue/red triangle with Leader's name – part of BB debug, current target on Leader's offensive path;

Blue triangle with Leader's name and "RDA" text – part of BB debug, current target area for reserve Leader (Reserve displacement Area);

Artillery debug – empty, black square on battery position permanently visible after first fire mission and temporary markings to show fire mission execution: target area, final impact area (after dispersion calculations, auxiliary lines and text with salvo status (Time Of Flight/Splash));

BB "Simple Dynamic Debug" – (BETA) additional debug mode for BB, that for chosen side(s) will show with arrows divisions movements and spotted enemy concentrations. Width of the arrow indicates amount of forces, length is proportional to movement distance. Size of enemy concentration (battle) markings depends of enemy's forces size. Also will be displayed current position of each Leader for chosen side(s)

Hint: if there is a High Command module present on the map, the player can enable with ***Ctrl+Space*** *the additional HC markers representing chosen routes and waypoints.*



6. BIG BOSS

Big Boss is simply a simulation of higher level, strategist or c-in-c HQ staff for one or both sides. His concern is map as a whole, while concern of each Leader is only set of his objectives. Without BB, setting up of objectives is done manually by mission maker. In this mode BB will autonomously and dynamically assign objectives and fronts for each subordinated to him Leader this way, so whole army should act like consistent organism in coordinated attempt to conquer whole map (to take control over every point on map, that BB will consider as strategic). This is theory of course, in practice, as any war, also this process can be described shortly as "organized chaos". To achieve his goal, BB will cyclically (every 20 minutes, or earlier in urgent situations) make several strategic decisions, where some of many important factors are: topography, size of map, presence of enemy, number and characteristic of own divisions, reserves, important, key areas on map and so on. Initially BB will check, which of found strategic areas should be considered as already taken due to own forces presence and some other factors.

To set up mission with Big Boss mission maker needs for each Leader, that will be subordinated to BB:

1. All four primary objectives anywhere on map;
2. Rectangular front trigger of any size, direction and position;
3. Secondary objectives should be not used (BB ignores them);
4. For BB's own debug and for setting up custom strategic objectives as addition to autonomously chosen, should be placed game logic of name: **RydBBa_SAL** (for Big Boss "A") or **RydBBb_SAL** (for Big Boss "B"). To add custom strategic objectives set in chosen points of map another game logics (without name), with such init: ***this setVariable ["AreaValue",(positive integer)];***, where the greater number is, the greater value/importance of objective. Usually value's range is between 1 (some village) to 10 (capital city), but can be also greater, if needed. Any of such placed game logic objects should be synchronized with one (or both) of previously set and named "SAL" game logics. Check the chapter ***"Config Variables"*** to know, how to add custom objectives also via init config. Each side has own, independent set of objectives;
5. If **RydBB_BBOnMap** variable is turned on, on map should be also placed a "physical representation" of BB for each used BB's side, named **RydHQBBaHQ** (**RydHQBBbHQ** for "B"). Destroying/killing of this object will terminate BB for that side.

Obviously for each Leader should be turned on one of limited control modes, so each will command only certain part of forces of given side. In addition some init config should be prepared: arrays for each BB, that consist Leaders to be subordinate (**RydBB(a/b)_HQs**),

optionally debug (**RydBB_Debug**), additional objectives' arrays (**RydBB(a/b)_Str**) and if BB should look for objectives on his own (**RydBB(a/b)_CustomObjOnly**). Also should be turned **off** for each subordinated Leader variable **RydHQ_LRelocating** (as it is by default). To switch on/off whole feature use **RydBB_Active = true/false**. See also ***"Config Variables"*** chapter, config examples and special demo mission for details.

Note, that in BB mode all subordinated Leaders will relocate towards recently taken strategic objective.

7. CONFIG VARIABLES

The functioning of HAC can be modified by manipulating the appropriate global variables. Such optional manipulation is easy and is conducted in the same way regardless of the HAC version (pbo or script). At the end of list will be some examples; for now I say only that the variables listed below should be executed at the mission start, or during the game by any effective method (init and act fields of units, waypoints and triggers in editor, other script files executed at beginning or activated in certain circumstances; there is the possibility to make whole scripts that will change these variables according to the mission maker needs; simply, or complex…). Effects will of course appear after such variables are executed, so if you input some variable into a trigger's act field, this will not take effect until the trigger conditions will be met, and similarly with waypoints. These variables give mission makers a wide variety of ways to customize and adjust HAC to their own purposes; or even to improve its activity, if needed.

*(remember: each Leader has its own set of variables. The examples listed below are only for the "A" side, if not otherwise stated. To customize other Leaders, simply add it's letter mark after "HQ", eg. RydHQ**D**_Debug for D-Leader instead of RydHQ_Debug. After "=" usually the default value is given)*

RydHQ_Wait = 15 - time in seconds before HAC will start working. This variable is shared by all Leaders;

RydHQ_Fast = false - if "true", Leader will not wait with new cycle for the end of the opposing sides cycles or even for the end of its own previous cycle. This creates a risk of "clogging" the computer with calculations, so caution is recommended.

RydHQ_CommDelay = 1 - this is the multiplier of the pause between the end of one cycle and the start of a new one for the given Leader. I advise caution with values smaller than 1; the flow of information can possibly be significantly perturbed by low values. Values higher than 1 are for realism fans who would like the delivery of orders to take as much time as it would in reality.

RydHQ_Debug = false - if true, debug mode is activated;

RydHQ_DebugII = false - if true, some additional markings will be visible on map;

RydHQ_DbgMon = true – shared by all Leaders. If true, additional info about morale and percentage of losses for each active Leader will be displayed on screen, if also **RydHQ_Debug** is true for given Leader;

RydHQ_Order = "DEFEND"; – sets army in defensive mode. Units will maintain positions near their commander. Re-defining this variable with any other value will set army back in attack mode *(note: pay attention to the quotes, these must be double - ")*;

RydHQ_Excluded = [] – array that contains names of the team leaders whose groups shall not be controlled by HAC. E.g., ***RydHQ_Excluded = [LeaderA, LeaderC, LeaderG];*** to place a given team leader's group under HAC control at some later point, just redefine this array (e.g., using a trigger) without the chosen team leader's name: ***RydHQ_Excluded = [LeaderA, LeaderG];*** which will include LeaderC's group; or: ***RydHQ_Excluded = []*** for include all previously excluded that way groups; note that to exclude a player leader, the player's unit **must** be given a name in the editor that meets the rules for naming, e.g.. ***LeaderP, LeutenantJack*** or ***Some_guy***.

RydHQ_ExInfo = false - if true - Leader receives information about the enemy also from excluded units;

RydHQ_UnlimitedCapt = false - if true, the commander will not stop sending further units to the objective, if false, will stop, once there is a certain number of them. Note: if true, objectives never will be considered as "captured";

RydHQ_CaptLimit = 10 - the number of allied units (not groups!) near an objective that commander will consider as sufficient to "capture" that objective. The lesser value the more dynamic progress.

RydHQ_ResetTime = 600 - interval in seconds between "resets" (see above);

RydHQ_ResetOnDemand = false - if true HAC will not "reset" cyclically (default each 10 minutes), but only when ***RydHQ_ResetNow*** variable becomes true; this may be useful to control the dynamics of situation. For example to make objectives status checks more often or vice versa (only after such check status of objectives may be changed to "captured" or "lost" and only then new one is assigned as current), to maintain strict control over interval between recon missions, between relocations of garrisons and reserve/idle/defending groups etc. Generally the longer interval between resets the more static situation.

RydHQ_ResetNow = false - if true and **ResetOnDemand** ("ROD") mode is on, reset occurs and this variable is auto-reset back to "false";

RydHQ_CargoFind = 0 - radius in meters for search for empty vehicles. Not recommended to enter too large values - that can cause lags. 100-200 meters should be enough. When this variable is equal or less than 0, the whole cargo transportation system is turned off (default) and groups will use only their own (grouped) vehicles. To make the transport system active without any searching for empty vehicles, set this variable to a minimal positive value - more than 0, up to 1.

RydHQ_NoAirCargo = false - if true, air units will not be assigned to transport tasks;

RydHQ_NoLandCargo = false - as above, for land cargo units;

RydHQ_SubAll = true - default control mode - if true, all allied groups on the map will be controlled by HAC (minus "excluded");

RydHQ_SubSynchro = false - when "SubAll" control mode is false, and this is true, HAC will control all groups whose leaders are synchronized with the Leader's unit (non-exclusive relative to other methods of choosing);

RydHQ_SubNamed = false - when "SubAll" control mode is false, and this is true, HAC will contol groups whose leaders are properly named (see above chapter 5.10, not exclusive relative to other methods of choosing);

RydHQ_Included = [] – array; when "SubAll" control mode is false, HAC will control groups whose names are included in this array (not exclusive relative to other methods of choosing);

RydHQ_ExcludedG = [] - Another exclusion method, working with both limited and unlimited control modes. As above, but on the contrary;

RydHQ_NameLimit = 100 - maximum number of names used in "subNamed" method;

RydHQ_ReSynchro = true - if true, with "SubSynchro", HAC will re-synchronize with the Leader's unit any group that has lost its team leader;

RydHQ_SMed = true - if true, medical support handling is on (ambulances/medevacs only, not group medic units);

RydHQ_SFuel = true - if true, fuel support handling is on;

RydHQ_SAmmo = true - if true, ammo support handling is on;

RydHQ_SRep = true - if true, repair support handling is on;

RydHQ_SupportWP = false - if true, support missions are extended by SUPPORT waypoints for support units (see earlier section about support system);

RydHQ_ExReammo = [] - groups included in this array will not call for ammo, however they will still use ammo support units if nearby;

RydHQ_ExRefuel = [] - as above for fuel;

RydHQ_ExRepair = [] - as above for repair;

RydHQ_ExMedic = [] - as above for medical aid;

RydHQ_Smoke = true - if true, withdrawing groups should try to use smoke grenades before withdrawal. Controls also calling arty smoke missions for same purpose;

RydHQ_Flare = true - if true, defending groups will use flare illumination when enemy is spotted. Controls also calling arty illum missions for same purpose;

RydHQ_KnowTL = true - makes Leader's knowledge about units be visible on the map for human controlled team leaders (cyclically reveals positions on map of all units known to the Leader);

RydHQ_Garrison = [] - array for names of groups that will become "garrison". Recommended mostly for infantry or land vehicles;

RydHQ_GarrR = 500 - radius in meters around a given garrisoned group team leader's initial position, within which this group should be available for attack missions;

RydHQ_GarrVehAb = false - if true, garrisoned motorized groups will disembark their vehicles and act as non-motorized garrison group. Otherwise motorized garrisons will only get sentry waypoint at its current position;

RydHQ_AOnly = [] - groups whose names are in this array will be used only for attack/defense/reserve/capture/flank missions (no recon);

RydHQ_ROnly = [] as above, but with recon and without attack missions;

RydHQ_NoFlank = [] - groups included there will be not used for flanking;

RydHQ_NoDef = [] - as above, but disables defense missions for included groups;

RydHQ_FirstToFight = [] - groups included in this array will be not considered as reserve. These groups will be constantly "attack available"; this does not mean however, that these groups always and immediately receive attack mission. This means only that such groups will be always in the pool from which a Leader takes groups for offensive missions as needed;

RydHQ_VoiceComm = true - voice notifications for human controlled team leaders with dialogs if true, and only with beep&text if false. This variable is shared by all Leaders and sets VoiceComm option for all of them;

RydHQ_Smoke = true – if true, AI groups will use smoke grenades to cover withdrawal maneuver;

RydHQ_Front = false - if true and a "front area" trigger has been prepared, then the assigned Leader will pay attention only for enemies inside this area. *Note: do **not** set this "true" before a front trigger has been prepared!*

RydHQ_ObjHoldTime = 600 - how long (in seconds) should a group with "capture" orders must stay in the objective area before the group becomes available again for other tasks;

RydHQ_ObjRadius1 = 300 – inner radius for "captured"/"lost"-objectives checks. For "captured", a certain number of ally units must be closer to objective than this value; for the "lost" check, hostiles inside this radius are counted;

RydHQ_ObjRadius2 = 500 - external radius for "captured"/"lost"-objectives checks. For "captured", a certain number of enemy units will be counted, for the "lost" check, allied units inside this radius are counted;

RydHQ_LRelocating = false – if true, Leader will be relocated to last captured objective (and will retreat if this objective become "lost". See earlier section about objectives);

RydHQ_IdleOrd = true - if false, "idle" orders will be not issued;

RydHQ_CivF = ["CIV","CIV_RU","BIS_TK_CIV","BIS_CIV_special"] – factions included in this array will be ignored by Leader;

RydHQ_Flee = true - if false, morale changes will affect only Leader's decisions, but not subordinated groups behavior;

RydHQ_Surr = false - if true, there is checked every cycle 50% chance for each group in panic for surrendering (permanently stop fighting, set as captive and abandoning vehicles and arms);

RydHQ_Muu = 1 – associated with morale multiplier of tendency to panic and flee when morale is low. The higher, the bigger chance for such behavior, however multiplied by this variable value will never be higher than 1. Can be used for setting of side's "determination". Set to 0 means total invulnerability on morale drop;

RydHQ_Rush = false – variable shared by all Leaders. If true, groups will never move slowly, but always with normal speed and aware behavior instead of safe. One of "dynamising" variables;

RydHQ_NoRec = 1 – Percentage chance for omission recon stage – another "dynamising" variable. NOTE: is multiplied by ***(RydHQ_Recklessness + 0.01)***;

RydHQ_RapidCapt = 10 – Percentage chance for capturing objectives "for all cost", means immediate and regardless of enemy presence capturing missions. Can be rised, when taking land should be of higher priority than defeating known enemy. Another "dynamization". NOTE: is multiplied by ***(RydHQ_Recklessness + 0.01)***;

RydHQ_DefendObjectives = 4 – In "DEFEND" mode this variable controls, how many groups should have already taken objective as closest, to consider it as additional defense perimeter. If set to 0, only Leader's position is considered as perimeter center/reference point. This variable allows to avoid situations, when alone group or too few of them are defending given perimeter;

RydHQ_Withdraw = 1 – multiplier of needed "danger level" (indicated in "debugII" mode as number at group's position) for tactical withdrawal overwhelmed groups. The higher value, the bigger danger is needed for withdrawing. If set to 0 – withdrawal is turned off;

RydHQ_Berserk = false – if true, Leader will keep offensive stance regardless of circumstancies. This will overwrite also **RydHQ_Order = "DEFEND"** setting;

RydHQ_DefFrontL, RydHQ_DefFront1, RydHQ_DefFront2, RydHQ_DefFront3, RydHQ_DefFront4 = ["N",""] – an array of two strings, not defined by default. Useful mostly in forced "DEFEND" mode. Each variable corresponds to one of possible defense perimeter point – Leader and each of taken objectives positions. First string (primary direction) must contain one of ***N***, ***S***, ***E*** or ***W***. Second string (secondary direction) always can be empty. If first string contains ***N*** or ***S***, second may contain ***E*** or ***W***. If defined such way, this variables will replace default system of choosing direction of defense front line (towards known enemy positions or randomized, if there is no known enemy). Thanks to this is possible to set forced defense perimeter of custom shape by setting defense front direction separately for all perimeter points;

RydHQ_ArtyShells = 120 – how many shells will be available per battery (howitzer/mortar/rocket artillery group) per kind (for rare SADARM ammo it is 10% of defined value). If set to 0 – internal arty handling become inactive;

RydHQ_PathFinding = 0 – this variable is shared by all Leaders. If set with positive value, experimental path finding based on terrain is turned on for non-motorized infantry. It is on waypoints level. Value controls "resolution" in meters – on how long sections route will be splitted for assigning mid-waypoints. Recommended values are 100 - 400. Not motorized groups will choose path preferably through good covered areas, as forests, urban areas or mountains, if near to route. NOTE: use with caution, this is experimental feature and can make troubles sometimes;

RydxHQ_SynchroAttack = false – shared by all Leaders. If true, Leaders will try to coordinate sent attacks – infantry or motorized/mechanized/armor groups dispatched to attack given target will wait as long, as all of them will be on flanking positions before final attack will be initiated. This is experimental feature, turned off by default, as sometimes may cause problems and not always work, as expected;

RydHQ_TimeM = false – when set as true, additional action menu options will be shown for each switchable unit on map, that will allow to customize simulations speed;

RydxHQ_GPauseActive = false – when set as true, game will be paused with on screen message each time player receive new order;

RydHQ_IDChance = 100 – chance (in percent) of choosing by each idle (except land support) group as its destination area position marked by **RydHQ_IdleDecoy** empty trigger;

RydHQ_SDChance = 100 – chance (in percent) of choosing by each idle land support group as its destination area position marked by **RydHQ_SupportDecoy** empty trigger;

RydHQ_RDChance = 100 – chance (in percent) of choosing by each resting/withdrawn group as its destination area position marked by **RydHQ_RestDecoy** empty trigger;

RydHQ_AmmoDrop = [] – aerial groups contained in this array will perform, when needed (there is some unit without any magazine on map), ammo drop missions, if there are some available ammo boxes inside **RydHQ_AmmoDepot** circular trigger radius. Also will be treated as support groups;

RydHQ_SFBodyGuard = [] – SpecFor groups contained in this array will always stay as HQ guard (excluded from SF attack missions);

RydHQ_LZ = false – If true, at landing zones of cargo missions for choppers will be found safe LZ with temporarily placed on it invisible helipad to help choppers avoid collision with trees and buildings during (un)loading cargo troops. Particularly helpful in difficult terrain. Can cause sporadic micro lags;

RydHQ_OALib = false – Shared by all Leaders. If true, classnames from OA will be added (can be used instead of regular RHQ set for OA units);

RydHQ_BAFLib = false – same for BAF classnames;

RydHQ_PMCLib = false – same for PMC classnames;

RydHQ_ACRLib = false – same for ACR classnames;

RydHQ_CargoOnly = [] – groups contained in this array will be used only for cargo missions;

RydHQ_NoCargo = [] – groups contained in this array will be not used for cargo missions;

RydHQ_AirDist = 4000 – idle aerial groups located further, than this value from their Leader will relocate close to the Leader's position, if **RydHQ_LZ** is active. Avoid very low values. To "disable" this, set variable with distance not possible on used map;

RydHQ_DynForm = false – if true, groups will on the fly change their formation, behavior and speed according to the danger factor calculated every minute;

RydHQ_ReconReserve = 0.3 * (0.5 + RydHQ_Circumspection) – multiplication of total number of available in given cycle recon capable groups by this value will return number of such groups, that HAC will keep as recon reserve;

RydHQ_AttackReserve = 0.5 * (0.5 + (RydHQ_Circumspection/1.5)) – multiplication of total number of available in given cycle combat effective groups by this value will return number of such groups, that HAC will keep as main reserve (part of them may be used for main flanking maneuver);

RydxHQ_AIChatDensity = 10 – shared by all leaders. Chance in %, that given event (support request, success, failure, sitrep, danger, hostile presence discovered, HQ decisions, low morale etc.) will generate appropriate to the situation radio message audible for all units of the sender's side. Tested every event for every groups under HAC control. Recommended values are 10-30, dependent also on number of groups. Set to 0 will turn off this feature;

RydxHQ_NEAware = 0 – shared by all Leaders. When set with positive value, each group will be warned about each enemy group known to the HQ inside radius of that value in meters around group's position. Such warned group will act like in danger. Note, that knowledge about any other unit is automatically set back to 0 outside view distance value. Values bigger than 800-1000 meters not recommended;

RydxHQ_MARatio = [-1,-1,-1,-1] – shared by all Leaders. By default HAC will send against each enemy group up to three infantry/soft, two armored, one air and two sniper groups. Each number in this array is corresponding to one of that kinds of forces in same order. Setting any of that numbers with not negative value will replace for that kind described default allocation with alternative one: total number of non-reserve and not busy groups of that kind will be multiplied by set corresponding value. Result rounded up means maximal number of groups of given kind, that HAC will try to send against one enemy group. So for example, if there are 6 infantry groups and 8 armored groups not set as reserve and currently idle, for [0.15,0.5,-1,-1] against next spotted enemy group in that cycle HAC will send up to 6 * 0.15 = 1 infantry group and 8 * 0.5 = 4 armored groups. Air and sniper groups will be allocated in default way;

RydHQ_GetHQInside = false – if true, and Leader's group relocating is active, HAC will try to find some position inside random enterable building in 100 meters radius around relocation waypoint and will send Leader's group there. Works only, if there is no vehicle assigned to that group;

RydBBa_SimpleDebug (RydBBb…) = false – if true, dynamic arrows for chosen side's divisions movement along with battle\enemy presence and each Leader position markings will be dispalyed;

RydBB_Active = false – if true – Big Boss system is active;

RydBBa_HQs = [] – array, that contain names of Leaders, that should be subordinated to Big Boss "A";

RydBBb_HQs = [] – same, for Big Boss "B";

RydBB_Debug = false – if true, some related to Big Boss info will be displayed on screen (needs logic named ***RydBBa_SAL*** on map) and in RPT file. For BB "A" red and blue circles will mark objecitves (color means status, number - value);

RydBBa_Str = [] – contains additional strategic areas for Big Boss "A" in form ***[[ATL position1,number1,boolean1], [ATL position2,number2,boolean2],…]***, where first value means position ([x,y,0]) of strategic area, second means how valuable area is (usual values are 1 (eg village) to 10 (capital city), but may be greater, if needed) and third value – status: if is currently considered as taken by "A" Big Boss', or not. Values may be changed, if occurs fusion of nearby strategic objectives;

RydBBb_Str = [] – same for Big Boss "B";

RydBB_CustomObjOnly = false – if true, Big Bosses will not study the map for self-determining of strategic locations. In this mode BBs will pay attention to custom added by mission maker objectives only;

RydBB_MC = *(undefined)* – this variable may be used for customizing of size and placement of area, that Big Boss of both sides will study for topographical data (square) and objectives choosing (circular, so square's corners will be excluded). May be useful for BB initialization quickening, when map is big, and only its part will be used as battlefield. Works in two ways. **First**, mission maker can name as RydBB_MC a trigger, which position will mark center of customized battlefield area and trigger's width multiplied by two will indicate side length of square battlefield area. This length should be divisible by 500. Optimally trigger should be not twisted (angle = 0) and square. This way its border will indicate exact border of customized battlefield area. **Second**, may be defined as an array, 2D position coordinates of custom battlefield's center point. NOTE: if not used, should stay undefined (nil);

RydBB_MapLng = *(undefined)* – used only, when **RydBB_MC** is used as position array. Then this variable should be defined as number divisible by 500, that will indicate side length of custom battlefield square area;

RydBB_BBOnMap = false – if true, Big Bosses will get a "physical" representations on map. With this setting mission maker should set on map for each BB any object/unit of appropriate name: **RydBBaHQ** for BB "A" and **RydBBbHQ** for BB "B". Removal or destroying/killing of this object will permanently terminate BB's control over given side;

RydBB_LRelocating = true – BB uses own HQs relocating routine. If this is set to false – that routine will be turned off and subordinated Leader's group will stay on place, not touched by BB. Changing this value is not recommended, especially for scenarios on big maps;

RydBB_MainInterval = 20 – Shared by both BBs. Value in minutes. Interval between each BB cycle (except for "urgent" situations). Should be positive integer. Minimal possible value is 1. Lowering this value may speed up map conquering at the cost of bigger CPU load.

7.1 RHQ and RHQs Arrays

HAC by default recognizes types (classes) of units from Arma 2 1.10. To be able to include other units, e.g.., those from OA, or units from addons, you must define the corresponding global array variable by entering the class names of the new unit types into the appropriate categories. This can be done - regardless of HAC version (pbo/script) - using prepared, external so-called "RHQ arrays", which are a kind of config value (and are initialized in the same way as other config variables). Note: categories are not exclusive; for example, a soldier with an RPG will, or at least should, belong to both the InfAT and the Inf categories. Note also, that eg AT assistant (and similar units of other kinds) haven't by itself AT ability, so shouldn't be added to InfAT.

In general, the category or categories to which a unit is assigned determines for what sort of tasks **whole group** of that unit will be used by HAC, so it is important to be careful when adding new classes. Thus, a truck class mistakenly entered to the armor category will be treated by HAC as if it were an armored vehicle. Some categories (e.g., naval) are used only so that HAC knows which units should be ignored either always, or under certain circumstances. Here is an example of a defined category with two new classes of crew unit:

RHQ_Crew = ["US_Soldier_Crew_EP1", "CZ_Soldier_Pilot_EP1"];

In the same way can be defined arrays for subtraction included by default classnames from given category. Only difference is, that such array name contains **RHQs_** instead of **RHQ_**. For example this code in init config:

RHQs_Cars = ["GRAD_RU"];

will make, that HAC will not recognize Russian BM-21 GRAD as Car category vehicle, but only as artillery piece, what will have an impact on kind of orders issued by HAC for such unit.

The available category arrays are:

RHQ_SpecFor – special forces group. Will be used for constant guarding of HQ if idle or chosen for it. If there is known valuable target (enemy HQ, arty or static), there is a chance dependant on Leader's personality (bigger at night, much lower, when target is armored), that group(s) will be send to eliminate that target with wide flanking route;
RHQ_Recon - reconnaissance units; note, that all classes included here should be also included to Inf category;
RHQ_FO – forward observer units; note, that all classes included here should be also included to Inf category;
RHQ_Snipers - snipers, sharpshooters; note, that all classes included here should be also included to Inf category;
RHQ_ATInf - infantry and unarmored vehicles with antitank weapons; note, that all infantry classes included here should be also included to Inf category and unarmored vehicles to RHQ_Cars;

RHQ_AAInf - infantry and vehicles with antiaircraft weaponry; note, that all infantry classes included here should be also included to Inf category, unarmored vehicles to RHQ_Cars and armored to RHQ_LArmor categories;
RHQ_Inf - infantry in total;
RHQ_Art - artillery;
RHQ_HArmor - tanks;
RHQ_MArmor – medium armor IFVs, APCs with significant firepower or aged tanks like T-34; note, that all classes included here should be also included respectively to LArmor or HArmor category;
RHQ_LArmor – all light armored vehicles, included medium armor category, APCs, etc. (excluding self propelled artillery like MRLS). Light armor will sometimes be used for recon;
RHQ_LarmorAT - as above, with weapons effective against armored vehicles (such as mounted ATGM
launchers); note, that all classes included here should be also included to LArmor category;
RHQ_Cars - vehicles without armor;
RHQ_Air – all helicopters and planes;
RHQ_NCAir – all unarmed helicopters and planes; note, that all classes included here should be also included to Air category;
RHQ_BAir – bombers. Note, that all classes included here should be also included to Air category. This category is empty by default, because some units are effective both as fighter and bomber CAS, so with this array mission maker may choose which generic or non-generic planes should behave as bombers. Planes added to this category will receive precise "destroy" waypoint attached to the target group's team leader or team leader's vehicle instead of usual "SAD" waypoint. However, this will not make the given unit drop bombs at target, for this additional external code is needed. Without such external code (which perhaps will be provided someday in HAC's BIS Forum thread…) such planes will use only guns or rockets, as non-bombing CAS;
RHQ_RAir – recon aerial units, mostly for unmanned UAVs (generic UAVs added); the recon of first choice; note, that all classes included here should be also included to Air category;
RHQ_Naval - boats;
RHQ_Static - static weapons including MG nests;
RHQ_StaticAA - as above, anti-aircraft; note, that all classes included here should be also included to Static category;
RHQ_StaticAT - as above, anti-tank; note, that all classes included here should be also included to Static category;
RHQ_Support - logistics and medical support units;
RHQ_Med = [] – medevac support vehicles; note, that all classes included here should be also included to support category;
RHQ_Ammo = [] – reammo support vehicles; note, that all classes included here should be also included to support category;
RHQ_Fuel = [] – refuel support vehicles; note, that all classes included here should be also included to support category;
RHQ_Rep = [] – repair support vehicles; note, that all classes included here should be also included to support category;
RHQ_Cargo - all vehicles, armored and not, able to carry passengers; ; note, that classes of unarmed vehicles included here should be also included to Cars category, and armored to LArmor;
RHQ_NCCargo - as above, unarmed;
RHQ_Crew - only pilots and designated vehicle crew members; note, that all classes included here should be also included to Inf category.

Each Leader shares same RHQ arrays set, so only RHQ_ are needed, and no RHQB_ and subsequent are needed. All categories of non-generic "vanilla" Arma 2 (from user-made addons, official DLC's, expansions, etc.), both units and vehicles, present on the map should be included in RHQ arrays, regardless of faction, that uses the given unit/vehicle, unless mission designer wants to make a given Leader not able to recognize & control units of selected classes. The necessary class names may be found in given addon's readme, on BIS Forums, or on DevHeaven. Alternatively they may easily be determined by an in-editor method: add following code to init field of any unit whose class name you want to know: ***diag_log (typeOf this);*** and check RPT file after "preview", or ***hint (typeOf this);*** and watch the on screen message.

7.2 HAC's personality variables

Leader personality is defined by several attributes. The value of a given attribute determines the tendency for a behavior of the given type. The influence of these variables is not huge, but it is noticeable. Typically, these values are set randomly (values will be in the range of 0 to 1, average 0.5), but you can set them manually if you choose. There are two ways to do this.

RydHQ_MAtt = false - switches on/off manually defining personality. If false - personality is randomized taking into account the whole package of attributes.

Manual attributes setting:

1. Each attribute separately:

RydHQ_Recklessness (and **RydHQB_ ...**) - the higher value, the greater tendency for risky orders. For example, the Leader will be more likely to send a helicopter to an area where the opponent has deployed AA weapons, or to send tanks into urban areas. If known enemies are averagely closer to Leader, that allies, and close enough (av. < 2000), or any enemy is very close (< 600), this value will be temporarily vastly risen.

RydHQ_Consistency - the higher value, the less likely will be a transition from attack to defense and vice versa.

RydHQ_Activity – the higher value, the greater the tendency to attack, and to remaining in this mode. If the value is low, the commander will easily give up initiative, and often will choose the defensive mode.

RydHQ_Reflex - the higher value, the faster commander reacts to changes on battlefield and will issue new orders more frequently.

RydHQ_Circumspection - the higher value, the greater percentage of forces commander would retain in reserve and for flanking maneuver.

RydHQ_Fineness - the higher value, the greater chance of flanking maneuvers, and a greater propensity to set ambushes on the reverse slope when in defense mode.

2. Selection of entire package of attributes with a single variable:

RydHQ_Personality = "GENIUS";

Here are in bold the possible values of this variable, and in brackets the pre-defined attribute values assigned to them:

"GENIUS" *(RydHQ_Recklessness = 0.5; RydHQ_Consistency = 1; RydHQ_Activity = 1; RydHQ_Reflex = 1; RydHQ_Circumspection = 1; RydHQ_Fineness = 1).*

"Genius" does not actually make the artificial commander a military genius, but in my opinion it sets optimal values.

"IDIOT" *(RydHQ_Recklessness = 1; RydHQ_Consistency = 0; RydHQ_Activity = 0; RydHQ_Reflex = 0; RydHQ_Circumspection = 0; RydHQ_Fineness = 0).*

As above, but to the contrary ...

"COMPETENT" *(RydHQ_Recklessness = 0.5; RydHQ_Consistency = 0.5; RydHQ_Activity = 0.5; RydHQ_Reflex = 0.5; RydHQ_Circumspection = 0.5; RydHQ_Fineness = 0.5).*

Average values, similar to default randomized.

"EAGER" *(RydHQ_Recklessness = 1; RydHQ_Consistency = 0; RydHQ_Activity = 1; RydHQ_Reflex = 1; RydHQ_Circumspection = 0; RydHQ_Fineness = 0).*

Reckless; acts faster than he thinks. Prefers to regret what he did, rather than hold back. The corpses of his subordinates may have entirely different opinions on this matter.

"DILATORY" *(RydHQ_Recklessness = 0; RydHQ_Consistency = 1; RydHQ_Activity = 0; RydHQ_Reflex = 0; RydHQ_Circumspection = 0.5; RydHQ_Fineness = 0.5).*

Still has doubts; lingers as much as possible. Would rather avoid errors resulting from haste, but his tardiness could mean defeat.

"SCHEMER" *(RydHQ_Recklessness = 0.5; RydHQ_Consistency = 1; RydHQ_Activity = 0; RydHQ_Reflex = 0; RydHQ_Circumspection = 1; RydHQ_Fineness = 1).*

Tricks are his forte. Devises complicated plans and implements them consistently.

"BRUTE" *(RydHQ_Recklessness = 0.5; RydHQ_Consistency = 1; RydHQ_Activity = 1; RydHQ_Reflex = 0.5; RydHQ_Circumspection = 0; RydHQ_Fineness = 0).*

Finesse? What is that? Ruses? What for? Most important is: fight to the death! Enemy death, of course... However, he is not mindless; he will do anything to keep the initiative.

"CHAOTIC" *(RydHQ_Recklessness = 0.5; RydHQ_Consistency = 0; RydHQ_Activity = 1; RydHQ_Reflex = 1; RydHQ_Circumspection = 0.5; RydHQ_Fineness = 0.5).*

He acts quickly and easily changes his mind. He is unpredictable, even to himself.

"OTHER" *(personality available only in random mode, means each attribute is randomized separately with rather average values)*

7.3 *Config examples*

1. Simple config for two Leaders with debug and cargo system turned on from init.sqf file (recommended way) for script version:

```
RydHQ_Debug = true;
RydHQB_Debug = true;
RydHQ_CargoFind = 100;
RydHQB_CargoFind = 1;

//usually always last:
nul = [] execVM "RydHQInit.sqf";
```

2. Complex config for three Leaders, with some useful tips commented (green, //):

```
//Debug for all

RydHQ_Debug = true;
RydHQB_Debug = true;
RydHQC_Debug = true;

//Manual personality setting on for A and C

RydHQ_MAtt = true;
RydHQC_MAtt = true;

//Choice of personalities for A and C

RydHQ_Personality = "GENIUS";
RydHQC_Personality = "EAGER";

//Cargo system for all

RydHQ_CargoFind = 150;
RydHQB_CargoFind = 150;
RydHQC_CargoFind = 150;

//Fronts for A and B

RydHQ_Front = true;
RydHQB_Front = true;

//Unlimited control off for A and B

RydHQ_SubAll = false;
RydHQB_SubAll = false;

//"Synchronized only" limited control mode on for B, used for placing, e.g., some
guerilla groups under leaderHQB control

RydHQB_SubSynchro = true;

//Six garrisoned groups for C (names set in init fields of team leaders g1 to g6)

RydHQC_Garrison = [g1,g2,g3,g4,g5,g6];

//Quick method for adding under Leader's control all groups of a given faction present
on map at start, shown here for A and B

RydHQ_Included = [];
RydHQB_Included = [];
```

```
{
if ((faction (leader _x)) == "USMC") then {RydHQ_Included = RydHQ_Included +
[_x]};
if ((faction (leader _x)) == "CDF") then {RydHQB_Included = RydHQB_Included +
[_x]};
```

```
        }
foreach AllGroups;

//making some A-side groups nearly passive may be good for groups that should always
concentrate on idle missions (?)

RydHQ_AOnly = [g7,g8];
RydHQ_ROnly = [g7,g8];
RydHQ_NoFlank = [g7,g8];
RydHQ_NoDef = [g7,g8];

//Player's group excluded

RydHQ_ExcludedG = [P1];

//RHQ array set for all Leaders

RHQ_Crew = ["US_Soldier_Crew_EP1","CZ_Soldier_Pilot_EP1"];

//Let's play! :) (for script version only)

nul = [] execVM "RydHQInit.sqf";
```

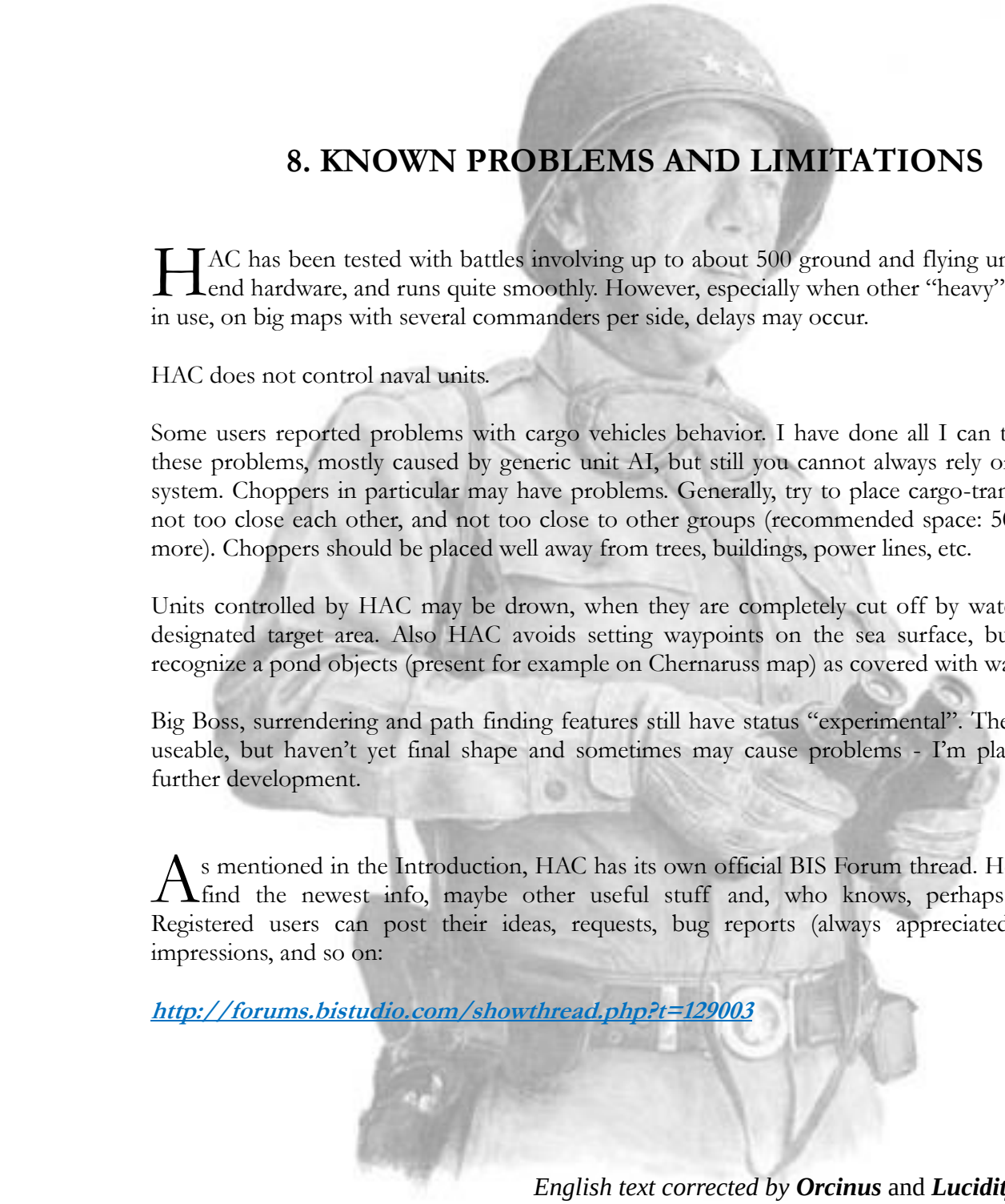
3. Variant of Big Boss config for all Leaders of both sides:

```
RydHQ_Debug = true;
RydHQB_Debug = true;
RydHQC_Debug = true;
RydHQD_Debug = true;
RydHQE_Debug = true;
RydHQF_Debug = true;
RydHQG_Debug = true;
RydHQH_Debug = true;

RydHQ_DebugII = true;
RydHQB_DebugII = true;
RydHQC_DebugII = true;
RydHQD_DebugII = true;
RydHQE_DebugII = true;
RydHQF_DebugII = true;
RydHQG_DebugII = true;
RydHQH_DebugII = true;

RydHQ_SubAll = false;
RydHQB_SubAll = false;
RydHQC_SubAll = false;
RydHQD_SubAll = false;
RydHQE_SubAll = false;
RydHQF_SubAll = false;
RydHQG_SubAll = false;
RydHQH_SubAll = false;

RydHQ_SubSynchro = true;
RydHQB_SubSynchro = true;
RydHQC_SubSynchro = true;
RydHQD_SubSynchro = true;
RydHQE_SubSynchro = true;
RydHQF_SubSynchro = true;
RydHQG_SubSynchro = true;
RydHQH_SubSynchro = true;

RydHQ_Surr = true;
RydHQB_Surr = true;
RydHQC_Surr = true;
RydHQD_Surr = true;
RydHQE_Surr = true;
RydHQF_Surr = true;
RydHQG_Surr = true;
RydHQH_Surr = true;

RydHQ_LRelocating = true;
RydHQB_LRelocating = true;
RydHQC_LRelocating = true;
RydHQD_LRelocating = true;
RydHQE_LRelocating = true;
```

```
RydHQF_LRelocating = true;
RydHQG_LRelocating = true;
RydHQH_LRelocating = true;

RydHQ_PathFinding = 100;

RydHQ_Rush = true;

RydBBa_HQs = [leaderHQ,leaderHQB,leaderHQC,leaderHQD];
RydBBb_HQs = [leaderHQE,leaderHQF,leaderHQG,leaderHQH];

RydBB_Active = true;
RydBB_Debug = true;

//RydBBa_Str = [];
//RydBBb_Str = [];

//RydBB_CustomObjOnly = false;

nul = [] execVM "RydHQInit.sqf";
```

4. Exemplary config for custom, round defense:

```
//to keep Leader in defend mode

RydHQ_Order = "DEFEND";

//only, if at least 6 groups will be closest to given objective, that objective become
a perimeter point

RydHQ_DefendObjectives = 6;

//to make all objectives "taken"

RydHQ_NObj = 5;

//directions of each perimeter, where objectives are placed around Leader

RydHQ_DefFrontL = ["N","E"];//North-East
RydHQ_DefFront1 = ["N",""];//1st perimeter facing North
RydHQ_DefFront2 = ["E",""];//2nd perimeter facing East
RydHQ_DefFront3 = ["S",""];//3rd  perimeter facing South
RydHQ_DefFront4 = ["W",""];//4th perimeter facing West

nul = [] execVM "RydHQInit.sqf";
```



8. KNOWN PROBLEMS AND LIMITATIONS

HAC has been tested with battles involving up to about 500 ground and flying units on low-end hardware, and runs quite smoothly. However, especially when other "heavy" addons are in use, on big maps with several commanders per side, delays may occur.

HAC does not control naval units.

Some users reported problems with cargo vehicles behavior. I have done all I can to eliminate these problems, mostly caused by generic unit AI, but still you cannot always rely on the cargo system. Choppers in particular may have problems. Generally, try to place cargo-transport units not too close each other, and not too close to other groups (recommended space: 50 meters or more). Choppers should be placed well away from trees, buildings, power lines, etc.

Units controlled by HAC may be drown, when they are completely cut off by water from it's designated target area. Also HAC avoids setting waypoints on the sea surface, but does not recognize a pond objects (present for example on Chernaruss map) as covered with water.

Big Boss, surrendering and path finding features still have status "experimental". These are fully useable, but haven't yet final shape and sometimes may cause problems - I'm planning their further development.

As mentioned in the Introduction, HAC has its own official BIS Forum thread. Here you can find the newest info, maybe other useful stuff and, who knows, perhaps even me? Registered users can post their ideas, requests, bug reports (always appreciated), describe impressions, and so on:

http://forums.bistudio.com/showthread.php?t=129003

English text corrected by ***Orcinus*** and ***Lucidity*** *– thanks!*

Online version: ***HAC's manual***

9. APPENDIX – INTERNAL VARIABLES

A list of chosen (potentially important, not all) internal, global variables, that can be useful for more advanced mission makers and for modders.

RydHQ_Cyclecount - holds number of current cycle. Used for time coordination of some calculations and actions and in morale formula (long-term losses factor);

RydHQ_NCVeh - holds classes of all "no combat" vehicles;

RydHQ_ReconDone - become true, when after recon mission, back to false every reset;

RydHQ_Inertia = (+/-) 30 * (0.5 + RydHQ_Consistency)*(0.5 + RydHQ_Activity) - reflects tendency to keep current stance (offensive/defensive);

RydHQ_Morale - (-50 to 0) - force's morale. The lower is, the bigger chance for fleeing, panic, surrender and changing stance to defensive. Calculated every cycle;

RydHQ_KnEnPos – holds last 100 known enemy positions. Used for determining direction of defensive perimeter. Actualized every cycle;

RydHQ_AirInDef – used in defensive stance. Holds air groups currently assigned to defense;

RydHQ_Init – is true only at first part of first cycle. Some things are calculated only, when this is true;

RydHQ_CInitial – initial number of own soldiers. Calculated once, used in morale formula (long-term losses factor);

RydHQ_CLast – refreshed every cycle. Holds number of own soldiers in past cycle. Used in morale formula (short-term losses factor);

RydHQ_CCurrent – refreshed every cycle. Number of own soldiers in current cycle. Used in morale formula;

RydHQ_Surrender – vestige of old surrender routine. Is set to false. If set to true – Leader's control will end, like when Leader's group is destroyed;

RydHQ_Progress – initially equal to 0. Will become 1 after each next objective takeover and -1 after each objective loss. Activates Leader's relocating routine if turned on.

RydHQ_AAthreat – holds all known AA-effective enemy groups. Used in risk calculations. Refreshed every cycle;

RydHQ_ATthreat – same for AT-effective enemy groups;

RydHQ_Airthreat – similar, but holds all known enemy aircraft groups. Used in air cargo mission risk calculation;

RydHQ_Exhausted – holds all currently combat ineffective subordinated groups (wounded/damaged, lack of fuel or ammo);

RydHQ_Done – indicates status of the cycle. If is not used fast mode, all Leaders will wait with their next cycle, until all currently pending cycles are done;

RydHQ_Friends – holds all subordinated groups. Refreshed every cycle;

RydHQ_Enemies – same of all enemy groups;

RydHQ_KnEnemies – holds all currently known enemy units;

RydHQ_KnEnemiesG – same, for known enemy groups;

RydHQ_FValue – hold number indicating current value of own forces, where each of infantry and static units has value 1, cars and artillery – 3, light armor – 5, heavy armor – 10, air units – 15;

RydHQ_EValue – same for known enemies. Both variables are used for stance choosing by Leaders and Big Boss;

RydHQ_NObj – initially equal to 1. It is positive integer of range 1 to 5. Indicates, which objective is currently a target. 5 means, that all objectives are taken;

RydHQ_Obj – refers to current objective's object (usually a trigger (empty detector));

RydHQ_Boxed – array containing groups, that was supported via ammo drop feature, to keep HAC from supporting each group that way more than once;

RydxHQ_AllLeaders – array containing all present on map Leaders;

RydART_FMType – arty handler config variable. Refers to mode of executed mission of HE, WP or SADARM type. Default is "IMMEDIATE", may be set as "TIMED";

RydART_Amount – arty handler config variable. For IMMEDIATE fire missions indicates total number of fired during mission shells per battery. For TIMED indicates amount of time in second of battery fire per mission. Default is set to 6 for HE and WP and 1/3 (rounded up) of that for SADARM;

RydART_Rate – arty handler config variable. Rate of fire – indicates time gap between each shot in fire mission. Default 0, means battery will shoot as fast, as possible;

RydART_Disp – arty handler config variable. Multiplier of salvo dispersion value. Default 0.4;

RydART_Acc – arty handler config variable. Multiplier of salvo drift value. Default 2;

RydART_Safe – base value of minimal distance in meters between planned impact and nearest known friendly group leader to allow HE, WP or SADARM fire mission executing. Default 100. This value will be multiplied by factor dependent on kind of artillery (to reflect accuracy and danger radius differencies) and by weather factor (the worse weather, the bigger safety distance), so final value will be bigger;