MISSION DESIGNER MANUAL - Version 1.2

BORING AND COMPLEX: A NECESSARY EVIL TO MISSION DESIGNER

This manual is here to provide additional information, instruction and "suggested design procedures" made with mission designers in mind.

This is done to address and answer some questions, for example:

- Why you need DSMC
- Install DSMC
- Single player/host and Server/Dedicated server mode
- What exactly is DSMC?
- How does it work?
- What are its limitations?
- How should I setup my host/server to use it?
- How does it relate to mission design?

We hope to answer all those questions in the next pages. Here you're in front of your trusted mechanic to learn how your car works and how to maintain her as it should. Let's start.

If you don't feel the need to dig inside the mod structure, history & geek details, you might jump right to the "What are its limitations" chapter!

WHY YOU NEED DSMC?

Dynamic **S**equential **M**ission **C**ampaign (DSMC) is a tool for mission designers, supporting versions of DCS World 2.5.6 and above[1]. This tool adds persistence to any DCS World mission, done by allowing the user to save the scenario at any moment and generate a new mission file based on the situation at the time of saving, that can be loaded, or edited, using the DCS mission editor. DSMC is required to be installed on the host/server only, and has configuration settings available for both Dedicated server, or Player as Host, via the special options GUI in DCS World.

DSMC creates a new .miz file that includes:

- All original triggers, scripts and embedded files as the original
- Updated unit positions and states (alive, dead)
- Scenery object states, like bridges, houses and structures
- User static object changing coalition when surrounded by the enemies
- Updated airbases, Oil/Gas facilities & FARP ownership
- Updated warehouse contents of fuel, aircraft & ammo
- Modified version of CTLD script by Ciribob with many additional features that will keep functional in the saved mission

DSMC also includes improved features and functions for dedicated server mode, and for servers with desanitized missionscripting.lua settings.

DSMC does not support capturing aircraft or missiles in flight and resume a mission later.

[1] New clouds system is supported only on DCS World 2.7 and above.

WHAT IS DSMC?

First of all, it's D.S.M.C., not DSMC. The acronym means "Dynamic Sequential Mission Campaign"... and to be *really* honest, today it's much more "SMC" than "DSMC". "D" is left out for now: while the final objective is to create a dynamic campaign system, the first strong step is to achieve what DCS doesn't give us: **persistency**.

That is the milestone we're setting now with DSMC: to give the mission designer a tool to reduce the workload to update a mission scenario after a flight with 30 clients outside. Going here and there, destroying units, breaking down bridges, moving resources, etc. etc.... and also updating the position of every ground and sea units out there, with meter precision, tracking every single object spawned in, like units, structures, crates, etc etc.

Ok, so, why? Because some years ago I was a mission designer and I really waited a long time some sort of scripting messiah that creates this thing for me: that will free my soul of hours of works to move those f*****g Urals from one town to another, that tank battalion from those unpronounceable villages to that other, even less pronounceable, place (Caucasus was the main solution at that time).

Well, few years ago, almost 2014, I understood that no Speed, Grimes, Wunderwulf, XCom or Ciribob would do that for me... so I tried to create it on my own. Obviously annoying them a lot in the process, but at least I tried to solve my issues myself.

As I am not a programmer, it required me to learn three times over. My first attempt was named "DAWS", and even if badly coded, it hit some small targets, you know, even today there are people who call this mod DAWS instead of DSMC out of habit. And here we are.

I'm asking you only one thing, and I hope that you will stick to that before even thinking a question: RTFM. Read The Fucking Manual. It is small, written in Arial 16 to help who can't read that much, to fill void spaces and to let you print in A5 format if you like. But it’s a necessary pill to your sickness of trying to use this mod. PS: Don't get me wrong, we both know that if you're a DCS mission designer, you're sick as much as I am.

INSTALLATION

DSMC needs to be installed on the server or mission hosts computer only. It works with both DCS stable & openbeta versions. The following example references the DCS World Open Beta version.

The installation package is a zipped file to be copied “as is”. Installation is performed by copying “DSMC_main” folder content inside your Saved Games\DCS.openbeta\ folder. The mod files are:

- **DSMC** folder
- **Script\Hooks\DSMC_hooks.lua** file
- **Mods\tech\DSMC** folder
- **DSMC_Dedicated_Server_options.lua** file.

To allow faster install/uninstall procedures you may want to use JGSME or OvGME mod managers, which are fully compatible with DSMC.

COMPATIBILITY

DSMC is designed to be fully compatible with;

- Mist
- MOOSE
- LoATC

Special consideration was given to the CTLD script by Ciribob. To add additional features and automation, **CTLD is already included in DSMC** with some additional features, check dedicated chapter in this manual.

SINGLE PLAYER-DEDICATED SERVER MODE

Single player – host with graphics mode

This is the “basic” DSMC mode. It will require no additional modification and allows DSMC to be customized and configured using the special options menu, in the DCS World user interface.

This mode should be used for single players and multiplayer hosts who run the simulation without using a dedicated server. It relies on the 3D interface for configuration.

To do a quick test, follow those steps:

1. Go into the DCS mission editor
2. Open a mission
3. Save the mission renaming with “DSMC_*missionname*.miz”. DSMC will only work if the first letters of the mission name are “**DSMC**”, case sensitive
4. Load the mission, from the editor or from the main menu
5. Enter a client unit or, with Combined Arms, into a tactical commander or game master slot
6. You will see an additional option using F10 radio menu. If you choose “*DSMC - save progress*” it will start a sequence of trigger messages and it will print “*scenery saved!*” in some seconds.

You can now exit the mission, go to the default DCS mission folder (\Saved Games\DCS.*whateverversion*\Missions) and open the and you will see a new mission named “DSMC_*missionname*_**001.miz**”.

Done!

Additionally, DSMC monitors the connected player counts. DSMC will perform an automatic save procedure by itself when the server is empty or only the host is online!

BEWARE: the save process is complex and heavy, **it might takes minutes for complex missions**. Do not run it continuously, or while clients are connected, in order to avoid momentary server disruption!

Dedicated server or "--server" or "--norender" modes active

This mode is made with server administrators in mind, and allows some additional features such as:

- Auto-save process every "n" minutes (1 to 480)
- No screen information messages during the save process
- Extremely light save process (unless F10 menu is used)
- Automatically modify the ServerSettings.lua to make the saved mission the next one on restart

By design, the configuration of this mode is done using the "**DSMC_Dedicated_Server_options.lua**" file instead of the GUI special options menu. Here you will find the DSMC settings you can change.

Dedicated Server mode requires the server administrator to *desanitize* the *missionscripting.lua* file. By default, DCS forbids the interaction with files and folders outside itself when run from within the mission. This process lowers the security level of DCS which, by default, prevents malicious code from being executed onto the Local File System. Therefore we advise researching the impact by searching for the key terms "desanitize" and "missionscripting.lua".

This mode is the only way DSMC will work when using the dedicated server version, headless mode.

In 'dedicated server mode' you will continue to have the radio menu options to begin the "heavy" save process, but you will also have a "light save" process every "n" minutes (frequency defined in the **DSMC_Dedicated_Server_options.lua)** which will create the saved files only when the mission is halted or stopped. This save won't immediately produce the new miz file, so don't look for it before the simulation is closed/stopped.

In addition, DSMC will perform an automatic full save every time the last client leaves the server. DSMC also attempts to save when the dedicated server is closed from Windows, however this method is not supported and requires a 'graceful' process exit.

To help server admin that want to run 24h servers, given that DCS and DSMC don't like that much the windows process killing procedure, DSMC provide an automatic close to desktop command that can be set every "n" hours, from 1 to 24: i.e., If set with 6 hours, after 6 hours of simulation DCS will try to close it to desktop. Try… why? Cause if there are clients connected, it will wait till the last one is out, checking every 5 minutes. Check further explanations in this manual.

BEWARE: DSMC does not support customized or multiple missions' lists. It does require DCS to close and restart any time you save the mission to make things works properly!

DSMC will continuously increment the next mission to run in Dedicated Server mode. DSMC will create an additional duplicate of the saved file named DSMC_ServerReload_*nnn*.miz, where "nnn" is a progressive number beginning with 001. This file will be automatically set as "first mission to run" when the dedicated server is launched. Server Administrators can leverage this mode to have 'hands-off' continuous persistence.

HOW DOES IT WORK?

DSMC is a mod, loaded in DCS World using the inbuilt modding API. It's designed to be as light as possible during simulations, and if everything is ok you won't even notice that it's doing his job. How? Because almost all the functions that run within the simulation are event-related: they don't work continuously; they get in, only if called. And they use the same event recorder that DCS gives us and use for trigger systems.

Also, every function that runs inside the simulation, has as little calculations as possible and the least table traversing as possible. What does that mean? You shouldn't really care. What you should care about is that DSMC is trying its best to do not overload your system while running... at least when it's possible.

DSMC collects data to tables, and when it's the right moment, do all the necessary calculations and pack up a new .miz file. This part is very heavy to your system, so it's never done during the simulation unless you call it by choice using F10 menu options. DSMC does that automatically when the last clients disconnect... that is a crafty way to say that if you are in single player mode, you necessarily have to save the mission using F10 menu because you won't have clients connected.

There is also an additional trick, the autosave function. That is an interesting way to collect the saved data continuously every "n" minutes, and when the simulation closes down. This means both; if you end the mission or if you close DCS to desktop, it will create the saved files. When this option is enabled (or when you're running it in dedicated server mode or in a standard host with no 3D graphic), DSMC will automatically try to desanitize your missionscripting.lua! **This process may fail if you have your main DCS directory protected by windows from writing**.

In the end, when autosave option is enabled, you will also have a DCS crash recover save functionality! As soon as you restart DCS, this mod will check for saved data availability and if it finds what is needed it will create a new autosaved file to be immediately set as current mission for servers & multiplayer.

In the next pages I'll try to add much more details about how the mod behaves and "think", if you like that word. Reading them will let you understand better the design and the workflow, if you like.

STRUCTURE

To add details about how it's done, DSMC is organized in different folders and group of files, which is the reason I suggest installing using a mod manager. Files, all placed in Saved Games\DCS.whatever\ folder, are separated in subdirectories:

- "\Scripts\Hooks\" folder. This contains a single mod file: DSMC_hooks.lua
- "\DSMC\" folder, with "\Docs\" and "\Files\" subfolder. Here you will find all the core files of DSMC
- "\Mods\tech\DSMC\" folder: here there are those files necessary for the in-game options menu

WHY MULTIPLE FILES?

As said, DCS has an inbuilt API mod system, which is utilised by placing a file in the "\Scripts\Hooks\" folder, that is read automatically when you start DCS World.

This file leads the loading of all the other core files, say lua “modules”, which are inside the “\DSMC\” folder. Which files are loaded? Those that are related to the options you choose in the special options menù… we could say that each file other than “UTIL.lua” and “SAVE.lua” are related to a particular option. For example, “SPWN.lua” is the file related about tracking spawned unit, or “MOBJ.lua” is the file related the persistence of map object state.

This is done to reduce the impact of flaws in a single file, which will break that features while keeping functional everything else… and also, this way, any unnecessary code is not even loaded to the simulation. Is this better than loading everything? No, but it’s cleaner… what is not there cannot be an issue when bug fixing.

Regarding the files included in DSMC, only three are mandatory to let it run: SAVE.lua, EMDB_inj.lua and UTIL.lua. Obviously if you delete the others, bad things may happen.

OPTION MENU… OR OPTION FILE?

We talked about the options you could choose to use. Well, there are two ways: one is for standard usage, the other is for dedicated server mode only. The two options do not update each other, they are separate configurations, so please pay attention; if you intend to change from self-hosting and GUI options, to dedicated server later, then, you must edit the options file.

For standard usage DSMC, there is a special options menu inside DCS. Check the User Manual for details.

For dedicated server mode, DCS removed some useful things like mission editor modules... and main menu. Therefore, when you run a dedicated server, DSMC won't be able to read the options menu. To solve this issue I created a file, that is named very obviously:

DSMC_Dedicated_Server_options.lua

That is placed directly inside your Saved Games\DCS.whatever\ directory. If you run a dedicated server, use that to decide what should be activated or not.

WORKFLOW: LOADING THE MOD & DATA TRANSFER

DSMC files are loaded as soon as you enter DCS, but are used only when you load a mission. DCS lua geeks like to say that this mod is run into the “Server environment”. I won’t dig too much into this, but to let you understand better what is going on, you should at least know that DCS World runs two different coding “environments”.

One, the *server env*, is almost unlocked and powerful except for compiled .dll files. You can read, write, modify data as you like and you can use all the functions ED put there to make it work. This env is used in main menu, mission editor, etc etc.

The other, called *mission environment* is where it runs all the lua code used to run the simulation. This environment is heavily protected, so much that is completely separated from the server env. You can’t even read or write (unless you “desanitize” it, as you know).

The relationship between those env are difficult, most like those between you and your cat, if you have one. Server env can send code, functions, info inside the mission env. Like your cat can do with you. But mission env can't even pass information to the server env, except for few on/off parameters (flags) or text strings (chat, radio messages). Like you when you try to teach something to your cat: it works only to the extend he likes… other things simply don't work.

Back to us, DSMC runs in the server env to take advantage of its power but we put some code inside the mission env to build a data collection and exchange interface. I call this "injecting the code".

So, step by step, this happens:

Each time a mission is loaded, DSMC_hooks.lua will check if the .miz file name start with the magic four letter "DSMC": if so, it loads everything you asked in the options menu. If not, it stops immediately letting you run DCS as if DSMC never existed. Also, it creates a temporary folder in the default mission folder used to archive autosave data and information.

Once loaded, the only files that perform actions inside the mission environment are "EMDB_inj.lua" and "TRPS_inj.lua". The first is the data collector for saving activity, the latter is the CTLD modified code that is loaded only if you activate the corresponding options... we will talk about that later.

Our hooks file also set a series of callbacks that will pull the trigger of the code when needed, using DCS modding API system. If you run a "desanitized" server, those triggers are almost completely performed by reading & writing files outside DCS mission environment to the DCS external environment: some call it plugin env, other server env, but it's the same: it's the environment used for main menu code with free access to the main lua functions.

If you run a standard server, the only way to get data from a running mission without desanitizing the environment is by using trigger messages. Collected data is stored into tables that instead of being written out to the mission environment, is "serialized" into text and sent out via trigger messages. Those messages activate a callback, that will check message content to find a table and if so, it will load the message as a string that magically is retransformed into a table.

With autosave options, the EMBD_inj.lua code will collect updated information every “n” minutes and saves them into tables. These are directly written to lua files in the temp directory. That is why when using autosave you need to run the “desanitized” version; else, you won’t be able to write those tables out.

As described before, those things are designed to be “light”, with as low impact as possible on CPU workload on the sim. Data is collected mostly using DCS event handlers (hoggit wiki is your friend if you need a refresh), while the only recurring checks are about position & life points, and they’re called in at every autosave.

WORKFLOW: TRIGGERING THE SAVE PROCEDURE.

In the first pages of this chapter you already had a sneak peek about when DSMC will save your scenario and when not, but here we will dig a little bit more.

The mod implements a very precise save procedure, which can be done “partially” or “completely”. The procedure is also smart: it will check if you run a desanitized environment and adapt itself.

The partial procedure simply collects data from the mission env and stores them into tables. If you run a desanitized environment, DSMC will also proceed by saving those tables into physical files. This is a fast & light process.

The complete procedure does the same things of the partial one but also performs all the operations in the server environment, including creating the new miz file or setting up the server config file. That is an intensive process, that **can literally halt your server for minutes** if run during a game session.

The complete save procedure can be described like this:

- Get’s the “save now” order
- EMBD_inj took all the events populated tables (logistic, deaths, etc) and saves them
- EMBD_inj took all the units’ position & life and saves them
- Hooks file code load these tables into the server env, and triggers SAVE module to do its job

- SAVE module opens the original miz file you loaded, modifies mission, warehouse, dictionary and mapResource file, then repacks everything, naming it “(DSMC_yourmissionname)_001.miz”. If the mission already have the 3 digits sequence at the end of the file, it will add 1 to it. Like if you start a mission called *DSMC_test_024.miz*, the saved file will be *DSMC_test_025.miz*
- Hooks then check if DCS is in server mode: if so, it will open server config files (serversettings.lua) and sets the path of this new copy as the first file to be loaded once DCS run

That's it. Simple, right? Well, **NO**, for goodness sake, it's not simple. The complicated part is about matching data from the mission environment, transforming them with consistency in a format expendable for the server environment, and put every piece in the correct place of the new .miz file that is created. That part is not covered in detail in this manual: really, it's something between 3.5K and 4K lines of code plus 7K+ for the CTLD clone itself (which is 99.9% Ciribob work)... with my ability to synthesize and my language skills I'll probably end up in something like Lord of the Ring capable to kill your grammar teacher in no more than 50 words [sic]. [*EDITOR: I have no idea what the mad person was writing here, so I left it as is for the enjoyment of future generations of scripters*]

So, what is really relevant? To understand WHEN those complicate things happen. The save procedure is "called" in three different moments:

- Manually using the "F10" radio menu[2]
- When the last client in a server has disconnected leaving the server empty (only in dedicated mode)
- When the simulation is closed, as long as the autosave option is enabled
- When DCS is started again after a crash, if the autosave option is enabled.

If you run a server with desanitized environment and autosave, you can:

- Do a manual save
- DSMC will save anyway on last client disconnect
- DSMC will also save again when DCS stops or crashes

[2] The F10 save call can be performed by ANY client.

The purpose is to have the most updated file possible, so **please understand that each save process will overwrite the previous** one: you can only have one saved mission each time. DSMC doesn't care about previous save events: it will always look for the loaded file name and try to update or append the 3-digit code

A simplified matrix of save process call is below. Obviously, we are talking from the player/host/server point of view: clients won't need anything.

		Autosave feature setting	
		Autosave option disabled	**Autosave option enabled** *(desanitized DCS)*
DCS execution mode	**Standard DCS execution; single player**	F10 manual activation	F10 manual activation Autosave
	Standard DCS execution; multiplayer host	F10 manual activation Last client disconnect	F10 manual activation Last client disconnect Autosave
	DCS execution with specials: "--server" or "--norender"	F10 manual activation Last client disconnect	F10 manual activation Last client disconnect Autosave
	Dedicated server executable	F10 manual activation Last client disconnect	F10 manual activation Last client disconnect Autosave

As a dedicated server administrator, you can also choose to disable the F10 manual save option, disabling the corresponding option that is available only in the *DSMC_Dedicated_Server_options.lua*.

The same option isn't available if you run single player or if you host a server using the standard DCS execution, because it is believed that is preferable to let this option always available in those situations.

BEWARE: even if you use F10 option to save the mission, it's strictly necessary that you close the mission and then start the next one: until the simulation is completely stopped, DSMC can't reset and work on another file.

WORKFLOW: WHERE ARE MY SAVED SCENERY FILES?

This time the answer is truly easy. All saved missions are stored in your personal "Saved Games\DCS*.whatever*\Missions" folder. Would you like to change this? Don't. This time it's better if you adapt your habit on this.

Basically, you will have one file saved if you run single player, while you're going to find **two** files if you run a server with autosave enabled.

The file you will always find is **DSMC_*yourmizfilename*_###.miz**, where "###" is a progressive number starting from 001. That will represent the last complete save procedure picture of your mission. It will be always there after a successful save.

So, for example, you want to host a mission with your friend using your dedicated server. You set up everything, ready for launch.

Your mission file is "ExampleMission.miz" what will happen? Nothing, cause you didn't add the DSMC tag: DSC will run like DSMC is not existing.

Ok then, you rename it to "DSMC_ExampleMission.miz". This time DSMC will run: if you quickly connect to the server you might see the "introduction" message of DSMC that confirm correct mod loading. Once all the clients are in, you fly your mission.

At the end of the mission, everyone disconnect from the server. In less than few seconds, a new mission ".miz" file is created, named "DSMC_ExampleMission_001.miz". You can edit it, or not, and fly again.

If you fly another sortie with "DSMC_ExampleMission_001.miz", the new saved file will be named "DSMC_ExampleMission_002.miz", and so on.

The second file is **DSMC_ServerReload_*nnn*.miz**: as said before "nnn" is a progressive number used to enable the possibility for the server to load a mission after the other, without having to touch the Mission Editor.

Why creating other files instead of updating the first one? For two valid reasons:

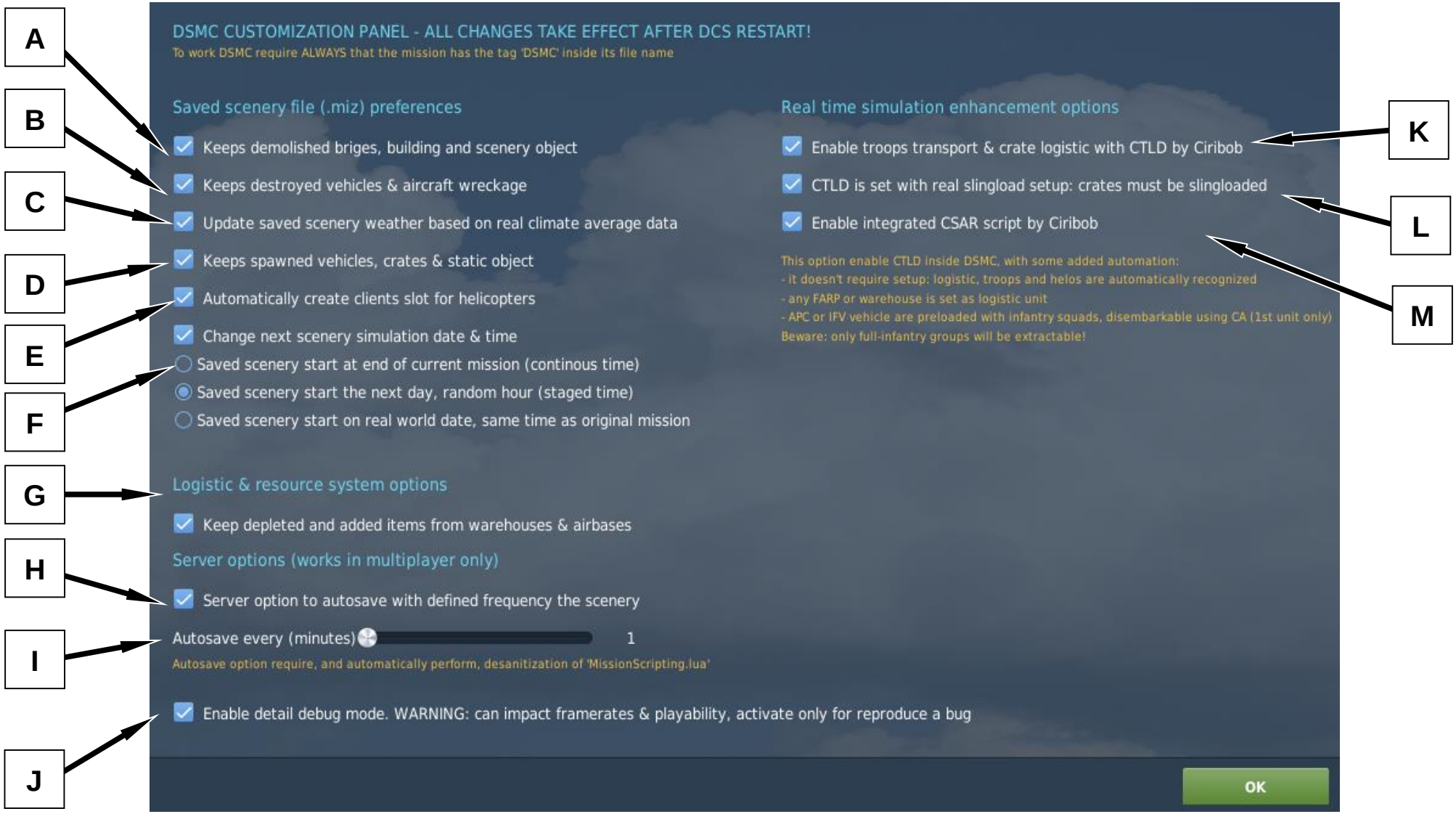
1. Windows won’t let you do that. While DCS is running, the open .miz file is considered “in use”, therefore a VRS error will be prompted in dcs.log and nothing will be saved (that’s a very good reason, you know);
2. Cause if something goes wrong…I bet you would prefer to have the original file to work on.

OPTIONS EXPLANATION

In this paragraph I'll try to explain as simply as possible how you should assume that each option is going to work.

You know, in the testing phase of DSMC (yes, we did one) some of the additional options DSMC provided weren't understood. Also, some of those options will require to be used correctly, else they won't work!

Next page you will have a picture of the in game options menu, with all the options included there highlighted. As said in the previous pages, server options are handled using a lua file that features a little difference: you won't have the updated position of a landed flight while you will get in exchange the possibility to remove the F10 manual save option.


A
B
C
D
E
F
G
H
I
J
K
L
M
DSMC CUSTOMIZATION PANEL - ALL CHANGES TAKE EFFECT AFTER DCS RESTART!
To work DSMC require ALWAYS that the mission has the tag 'DSMC' inside its file name
Saved scenery file (.miz) preferences
Keeps demolished briges, building and scenery object
Keeps destroyed vehicles & aircraft wreckage
Update saved scenery weather based on real climate average data
Keeps spawned vehicles, crates & static object
Automatically create clients slot for helicopters
Change next scenery simulation date & time
Saved scenery start at end of current mission (continous time)
Saved scenery start the next day, random hour (staged time)
Saved scenery start on real world date, same time as original mission
Logistic & resource system options
Keep depleted and added items from warehouses & airbases
Server options (works in multiplayer only)
Server option to autosave with defined frequency the scenery
Autosave every (minutes)
1
Autosave option require, and automatically perform, desanitization of 'MissionScripting.lua'
Enable detail debug mode. WARNING: can impact framerates & playability, activate only for reproduce a bug
Real time simulation enhancement options
Enable troops transport & crate logistic with CTLD by Ciribob
CTLD is set with real slingload setup: crates must be slingloaded
Enable integrated CSAR script by Ciribob
This option enable CTLD inside DSMC, with some added automation:
- it doesn't require setup: logistic, troops and helos are automatically recognized
- any FARP or warehouse is set as logistic unit
- APC or IFV vehicle are preloaded with infantry squads, disembarkable using CA (1st unit only)
Beware: only full-infantry groups will be extractable!
OK

All the options are explained below. Each explanation is also provided with its equal variable name in *DSMC_Dedicated_Server_options.lua*

A. Enable/Disable the feature that keeps track of destroyed scenery objects. A scenery object is a 3D model which is not a static or a unit, like bridges, Buildings, factories, etc.
If this option is enabled, a destroyed bridge will stay destroyed in the saved mission. Also, that works for airbase warehouses... if you destroy the fuel tanks, that airbase won't be able to refuel aircraft, even in subsequent missions.

This is done by keeping track of every "death" event related to a scenery object, which is recognized using a getCategory() function. In the saved mission there will be a single trigger created (it will be a single one even in subsequent missions) that contains a very small "zone" for each object, in which anything is set to "dead". Easy trick. So, you can simply get to the zone list and remove those zones related to the object you want to be alive again. It's harder to explain than to try.

An important side note: if you destroy all ammo or petrol warehouses of an airport, it won't be able to give the corresponding item to anything spawning on that... even in a subsequent saved mission. But, as for a DCS limitation, anything that is spawned in the ME and activated at second "0" of the mission, will be able to load what is needed anyway. That is because the persistency trigger is set at mission start, but mission start is not before the creation of mission object that are not late activation. What does this mean? Two things. Think about this scenario: in mission "003" all fuel tanks of Gudauta are destroyed. Therefore:
- If you want Gudauta to be unable to give fuel to airplane, you must set those airplanes as "**late activation"** (even 1 second);
- If you need something to start immediately from Gudauta, simply add the group and don't touch anything else;
- If you want something to start anyway but not at mission start, you need to set as "**uncontrolled**" and then push the task to start it accordingly to your needs.

Another important side note: destroying bridges could lead to bad things when you try to route convoys... and that's the purpose to break a bridge, to be honest. Therefore, keep an eye on what you task your ground forces after a few missions... you don't really want a Tank to do 3000 kilometers via the Roki tunnel because you wanted to cross a 200 m bridge in Sochi.

Variable name: *DSMC_MapPersistence*

B. Enable/Disable the feature that creates a "dead" static object of the same shape that was the real unit where this unit died. Purpose? almost nothing, it's useful for scenery immersion. Also, those objects are created for downed helos & planes. That might be useful for a CSAR mission.

Variable name: *DSMC_StaticDeadUnits*

C. Enable/Disable the option to track automatically update the weather in the saved mission. The weather will be randomized using real world statistic data, and will work differently in every map.

Randomization will modify, using month/day/hour dataset:

- Temperature;
- Precipitation;
- Wind strength and directions;
- Fog and/or Sandstorm;
- QNH;
- Clouds coverage, minimum altitude and thickness.

So yes, in January over the shore nearby Sochi it’s likely to find a bit of fog in early morning.

An important sidenote: due to the current status of dynamic weather and for the sake of mission planning, DSMC use static weather type.

Variable name: *DSMC_WeatherUpdate*

D. Enable/Disable the option to track spawned objects or not. If false, everything you added into the mission using scripts, won't be saved.

BEWARE: as better explained in the mission design guidelines section, you should be very aware that with this option enabled your persistency will do its best, but also that spawning should be controlled with care: if you have a mission that spawns some groups of tanks each launch, you will have the scenario clogged with hundreds of tanks in very few save iteration… cause each time DSMC will save those tanks, but you keep spawning other at any saved mission start!

DSMC can't know that those "spawned tanks" were the same of the previous mission!

Variable name: *DSMC_TrackSpawnedUnits*

E. Enable/Disable the option to automatically create client helicopter flights in FARPs.

If this option is "on", DSMC will check if FARP warehouse has flyable[3] helicopters available. If so, it will create slots for clients available on parking. This will follow some rules:

- All flights will be 2-ship, no wpt added, unless the warehouse has only 1 or 3 airframes: in that case, a 1-ship or 3-ship flight will be created;
- If the warehouse has more than 4 airframes, only 2 flights (both 2-ship) will be created;
- If the warehouse is set as "unlimited", then it will be created one 2-ship flight for each helicopter model[4].

As some of you could know, airports parking is very very difficult to manage. For the sake of "safety", DSMC will check if all the possibile precondition to create slots are met. If not, it will skip that flight. So,

[3] Currently supports: Ka-50, Mi-8MTV2, UH-1H, all Gazelle variants

[4] For the Gazelle only the 342-L and 342-M variant will be created

currently slot creation in airports is disabled and should be back on in DSMC 1.2 version. You should simply try and see…

Ok… so, what's the advantage of this thing? Easy: imagine a "progressive conquer" campaign: you may have a red farp with no airframe. If you conquer it, and then land helos there (let's say a couple of UH-1). In the saved mission DSMC will:

- Change FARP coalition from red to blue;
- See the 2 UH-1H in the warehouse;
- Create 1 flight of UH-1H, 2-ship, ready to be used or edited.

Also, the same thing apply if Ka-50 where already inside the warehouse: DSMC will create blue flights for those.

This feature is mostly intended for servers' owners who run DCS in a loop, restarting the saved mission each time.

I strongly suggest to use this coupled with limited warehouse system.

Variable name: *DSMC_CreateClientSlot*

F. Enable/Disable the option to modify the starting hour and date of the next mission. Here you have 4 choices:

a. Do not change the start time/date (unclick the checkbox)
b. Change it, using the start time as the exact simulation date & time of the save command
c. Change it, using a randomized hour in the next day relative to the mission flown and saved[5]
d. Change it, using the very same hour of the original mission but the current real world date

Nothing else.

Variable name: *DSMC_UpdateStartTime*
Variable option: *DSMC_UpdateStartTime_mode* (must be 1 or 2)

[5] For lua guru: since DCS has some issue about mission starting daytime and ending nighttime with runway lights, I decided to add a filter: no mission will start after 16:00 and before 04:00. If you want to change-remove this limit, you should manually edit the TMUP.lua file inside DSMC directory… there are a couple of local variable that speaks from themselves: "minHourTime" and "maxHourTime" if you can't understand them, you probably should't modify the file.

G. Enable/Disable the option to track all warehouses items, when they aren't set as "unlimited". That was a very complex task to achieve, and it sticks to **a couple of rules you must adhere**. Resource management in DCS is not accessible by scripting, therefore the only way to track that data is "running" is via parallel code outside, that tracks everything that is removed from a warehouse via takeoff event and also everything that is added via landings into the airbase/FARP.

This system is not designed to work during the mission: as said, I can't track warehouses items during the mission. Instead, everything is tracked at the end.
The main differences you must take into account are two:

- Real time supplies performed by DCS engine will not be tracked;
- All the changes into the items amount you will see during the DCS simulation, is not directly tracked into DSMC, but using takeoff & landing events.

In DCS, items are removed from the warehouse when you spawn and re-added when you de-spawn. DSMC, instead, will track takeoff & landings events.

Also, supplies will be performed accordingly to the removed items from any airbase & FARP, and it will be performed accordingly to the supply chain you set into the mission editor.

So, what changes? Simple: everything will be calculated and performed at mission end. You will find the result into the saved file.

Again, DCS maths is performed real time, during the simulation, therefore it's all about available items & petrol in the running mission. Else, DSMC maths is done after the mission, so it's about the starting items & petrol available for the next mission.

Another important point: DSMC can’t track “fuel tank” & “misc” items such pods, side guns. The POL stored inside the fuel tank is tracked,

the object “fuel tank” is not. This is a limitation DSMC can’t overcome due to DCS getAmmo() function limits.

How to use this properly:
As said this warehouse tracking system runs outside of DCS, using DCS gathered information to add or remove items & petrol. Ok, but how? It uses event handlers:

- When you takeoff from the Airport/FARP/Carrier/Helipad, items are removed.
- When you land, items are added.

If you set anything in the warehouse of the Airport/FARP/Carrier/Helipad as "unlimited", nothing will be added or removed.

It should work on anything that has a warehouse, from airports to FARP to oilrigs to ships to carriers etc etc.

Variable name: *DSMC_TrackWarehouses*

H. Enable/Disable the autosave procedure, we already talked about that earlier. If disabled, autosave won't do its work. If enabled but *missionscripting.lua* is sti ll unmodified (not desanitized), it won't take effect again.

Remember: this mod should "survive" at almost any update except for induced bugs, so you aren't required to do the mod dance disabling it before launching the update and then re-enabling it after update. Else, missionscripting.lua is ALWAYS reverted to its original "desanitized" state each DCS update: remember to desanitize it again.

Variable name: *DSMC_AutosaveProcess*

I. This slider decides how often you want DSMC to take updated data for the autosave procedure, in minutes. You can slide any number from 1 to 480 minutes.

Variable option: *DSMC_AutosaveProcess_min* (from 1 to 480)

J. Enable/Disable the debug verbose log calls. That should be always left off unless you're willing to provide me as much information as possible for debug. Leaving this option as checked not only will produce and enormous amount of data into the dcs.log and DSMC.log files (Saved Games\DCS*.whatever*\Logs folder), but will print a lot of annoying triggered text message during the simulation. Really, leave it off unless needed to reproduce and report a bug.

Variable name: *DSMC_DebugMode*

K. Enable/Disable the CTLD by Ciribob modified code. Enabling this option you will get the fantastic CTLD script working inside your mission, with some small additional features:

a. Weight feeling added for infantries! if you load a squad team in your Huey, you will feel the weight;

b. Reworked spawnable crates & group, now you can spawn FARP support units and also you will be able to spawn combined arms platoon depending on the country you choose: Georgian will have only Georgian vehicles, US only US ones and so on. You will be able to choose any vehicle type in service since 30 years before the mission date (in 2010 you won't have MTLB in Russian units, but instead you will have BTR-80 or BMP-3);

c. Reworked menù, check the ctld guide manual leaflet for more info!

d. You can use any units in a group, you're no more forced to create flights with 1 unit only to comply with DCS limitation. Any pilot in a flight will have commands for each flight member;

e. Ground vehicle group will be able to load/unload troops & crates all units at once, not only the first unit of the group anymore. So you can transport 3 crates at once using a group with 3 trucks inside;

f. Trucks will be able to handle crates & troops by default: one crate or 20 soldiers each. APC will handle squad of troops (6-8 soldiers). IFV will handle fireteams (3-4 soldiers). All APC and IFV will have a soldier team pre-loaded by default;

g. Name coding for pilot units, extractable groups and logistic object is not necessary anymore. DSMC CTLD clone will automatically add to the corresponding tables (still there and editable if you wish) any helicopters, spawned or included in mission into pilots table, and any groups of infantry-only units into extractable groups and warehouses object (static-warehouse category) into logistic units;

h. FOB will include the standard three objects (outpost, beacon and tower) and also a working invisible FARP with ammo dump, fuel dump and windsocks. BEWARE: if resource management is active, the new farp will be set to 0 items of all in the saved mission file!

i. FOB will be recognized in the mission editor, see advanced mission design guidelines for details;

j. JTAC units by default are Stinger and SA-18: those are still controllable by client but also valid troops for helos!;

k. Static units (SAMs) and FOB crates and JTACs can be unpacked everywhere, while instead

l. If you forgot to add beacon.ogg and beaconsilent.ogg files, those are automatically added in a dedicated trigger as soon as you save the scenery

m. If resource management system is active, you will also have some logistic crates available! Those don't need to be unpacked: you will only need to deliver them within 150 m from a FARP or Airbase center and the resources you have brought there will be added to the warehouse in the saved mission. You will have 3 options of fuel quantities to be delivered and some

options for weapons: each weapons crate will add a fixed quantity (i.e. 20) to every item of the choosen category. Category are, for example: rockets, dumb bomb, guided bomb, air to air missiles, and so on.

Additional info: If you have an updated CTLD or a modified CTLD and you want to use that instead of the inbuilt version, no issue! But please remember to uncheck the option in the menu!

Variable name: *DSMC_automated_CTLD*. If you use it in servers, so with the variable option, you will also find some additional customization setup in the “advanced server options” (see mission design guidelines chapter).

L. Enable/Disable the "real slingload" option of CTLD, as it is. Basically if disabled you will be able to "load" crates virtually in your helicopter without the need to sling load them.

Variable option: *DSMC_CTLD_RealSlingload* (true/false)

M. Enable/Disable the CSAR script by Ciribob modified code. Enabling this option you will get a slightly modified version of CSAR script.

BEWARE: inbuilt CSAR script force the “csarMode variable to 0, removing any aircraft disabling when downed thing. That is by design, cause DSMC won’t handle that part of gameplay due to tracking limitation. Also, mist is not needed anymore: the entire script is handled just like the inbuilt CTLD.

Variable option: *DSMC_automated_CSAR* (true/false)
Other variable: *DSMC_DCSR_useCoalitionMessages* (true/false)

The second variabile is only working servermode: if set false, all the messages to coalition, like when an acf is downed, are disabled (this is useful when you have human controllers to handle the scenery).

SERVER & DEDICATED SERVER ONLY OPTIONS

I added some feature that you will find in the dedicated server configuration file and that you won't find in the options menu.

Also, there are some other options that falls into advanced customization. Few of them, related to inbuilt CTLD, can be changed and they are explained in the next chapter: search for "SERVER AND DEDICATED SERVER CTLD CUSTOMIZATION". You shouldn't modify those unless you really checked and tested the effects.

Going back to the options, let's explain them one by one.

DSMC_DisableF10save

This option allow you to remove the F10 menù "save scenery" action. This cause in MP DSMC automatically saves in multiple ways, so it might be safer to prevent a player doing a save action while playing that might result in a violent lag or worse, server crash.

DSMC_WarehouseAutoSetup

This option will make the whole supply net to be rebuilt each time you save a mission, before logistic and resupply calculation takes effect. That is a VERY useful tool if you as mission designer decide to agree to the standard supply net, explained in the Mission design guidelines (next chapter), paragraph "Standard supply net"

DSMC_AutosaveExit_hours

I added this for servers that works 24/7 to automatically close DCS after a defined amount of time, in hours.

It can be set with numeric values from 1 to 25, and they are meant to be "hours". Any number above 24 (but I suggest to keep 25) means that this option won't have any effects and virtually the mission won't stop. Instead, any values between 1 and 24 will command a DCS shutdown after "n" hours.

DSMC_AutosaveExit_time

This option works only if *DSMC_AutosaveExit_hours* is set to “false”. It accept value from 1 to 23, and it will cause the server to save and close right at that time in 00:00 – 23:59 time span, real world. If you want your server to reset every morning at 5:00 am, you set this “5”.

DSMC_AutosaveExit_safe

This option is set true by default: it will make DSMC to check if any client is connected to the server. If so, it will print an advice and then wait 10’ to retry. Else, if false, it will save the server and close regardless of any connected clients.

DSMC_AutoRestart_active

DSMC 1.2.0 add an auto-restart feature with limited functionality. It’s a windows task manager code that checks when DCS stop running and, when so, it automatically restart it. That, coupled with controlled save & close provided by the AutosaveExit options, makes a DCS server 24h/7days proof.

Sadly, **this code has proven to be not perfectly reliable**. It might works or not, you should try & see. Also, it’s extremely important that you consider this: **AUTORESTART IS NOT COMPATIBLE WITH MULTIPLE SERVER ISTANCES**. It works only if you have one.

DSMC_CreateSlotCoalition

This option works only if *DSMC_CreateClientSlot* is enabled. It’s set “all” by default, so that helicopter’ slots are created for both coalitions. You can either set this “blue” or “red” to limit the slots creation to only one coalition.

DSMC_StarTimeHourMin & DSMC_StarTimeHourMax

These options works only when *DSMC_UpdateStartTime_mode* is set to “2”, and define the minimum & maximum hour in the morning when the randomized start time can be set. Minimum can be set from 1 to 14, while

maximum can be set from 15 to 23. Any wrong values will be removed and substituted with defaults (4 & 16).

BEWARE: if autorestart does not work you will still need some kind of support, a software like *Restart On Crash* or a windows task manager script that provide restarting feature.

So… why use this feature instead of closing the simulation with the same task manager script then?

Because this option does a very useful thing: DCS won't go down exactly when the timer goes to zero… *it will try* to go down. If any client is still connected, then it will print a message to everyone and then it will retry in 10 minutes.

So, if one of your clients is going to be back late from a mission, the server will not close "killing" the pilot and the related resources**. It will wait until the player disconnect and then, when the timer arrives to the next 10 minutes step, it will shut down.**

The main positive things about this is to have a controlled and gracefully shutdown: with very complex mission (>2000 units) the save time might be so long (>30s) that windows might recognize DCS as "halted" and kill the process itself. Doing this, there's a probability that the save won't work as expected… So, if you're running a 24/7 server, I strongly suggest you to use this option.

WHAT ARE DSMC'S LIMITATIONS?

Ok, here I'm trying to list some of the limitations of the mod. Basically, some are about its design or "vision" while others are about DCS code. Some, also, are due to my skill limitation.

So, this is going to be a "*No, you can't*" list almost, ordered per feature, as complete as possible. Let's start:

MOD RELATED

- You need your saved games directory and DCS main directory to be not *read-only*, and you should execute DCS as administrator. If you don't comply, the mod could fail.
- If you're using a dedicated server host or a server host with "--nographic" and "--server" custom load string enabled, you must have a desanitized server or the save process will fail
- You can't change anything in the options menu and then immediately start a mission. You always have to close and restart DCS
- DSMC doesn't "freeze" or "pause" a mission: it will save an updated mission scenario
- DSMC does not add any automation or enhanced behaviour of DCS units, groups, controller
- DSMC does not support officially any other mods, except for MOOSE and CTLD (the built in version). In particular, some couple of mods that open & close the file just before mission start (i.e. weather injection tools) have issues sometimes
- DSMC does not keep routes or paths that you created in the M.E. or routes added during the simulation with CA or scripts or otherwise
- Once you choose your special options in the options menu, those are kept as long as you don't uninstall the mod. As said, you shouldn't even when using a server: if you don't want DSMC to work,

simply don't name the miz file starting with "DSMC". After an unistall/reinstall, options are reverted to default

- For dedicated server or server using the "--norender" or "--server" executable, it' required to desanitize *missionscripting.lua*

SPECIAL OPTIONS RELATED

- DSMC do not load the ".ogg" files needed for CTLD for beacons usage in the first mission. Those files are, anyway, kept in subsequent saved files. Those files are stored, to help you, in the "DSMC\Files\" Folder

- DSMC does not edit DCS warehouses in real time (1): you can't add or remove aircraft, weapons or petrol from DCS warehouse using functions, scripting or otherwise

- DSMC does not edit DCS warehouses in real time (2): as I can't check warehouses content in real time, any supply logic provided by DCS won't take effects. Items will be added and removed as if the warehouse didn't have any supplies from other warehouses

- DSMC warehouse tracking can’t track “fuel tank” & “misc” items such pods, side guns. The POL stored inside the fuel tank is tracked, the object “fuel tank” is not. This is a limitation DSMC can’t overcome due to DCS getAmmo() function limits.

- DSMC scenery object persistence can't prevent any AI aircraft spawned at mission start to load fuel & ammo from an airbase which has all the fuel tank and/or ammo buildings destroyed. Else, right after mission start, those items or fuel tons won't be available. My design suggestion is therefore to set any aircraft "late activation" for at least 1 second

- Even with autosave feature enabled, DSMC does not provide code or apps to close & restart a DCS server: you must provide that. If you kill the server with task manger is going to be ok, but I might suggest the software “*RestartOnCrash*”, tested and working fine with a gracetime setting of at least 120 seconds (better 300).

- Ok this one is gonna scare you… it scare me, for sure. About inbuilt CTLD… every 1st unit of a ground group, if It's an APC/IFV vehicle, is "pre-loaded" with a soldier squad. And that's normal. This team is going to stay inside the vehicle unless you enter using CA and use "unload/extract troops" command. And that's normal. BUT… i*f you have a custom kneeboard folder inside the miz file*… well.. all ground squad will immediately extract outside the vehicles at mission start creating a real mess. Why? You tell me: I don't know. God don't know. I mean… the "kneeboard" folder inside the miz file: it has nothing to do with real time CTLD commands… but it's a confirmed bug and behaviour, and you simply have to remove the custom kneeboard to revert back to normal.

MISSION DESIGN GUIDELINE

DSMC is available to some testers since months, something after January or so. This fact allow me to say two things:

If there are bugs, *complain them also*. Ok, you maybe never going to know their name, but at least now you know that they exist, and for sure they let *that* bug intentionally there only for annoying you today.

Thanks to them, and one in particular[6] which really takes fun in running DSMC in any (f*****g) kind of situation and beyond any possible scenario I could imagine, I was able to identify some very useful mission design guidelines.

The very first thing you need to double check is the missionscripting.lua file or user write permission in the main DCS folder. In fact, **when running in a dedicated server or in a server with "--nographic" or "--server" custom string enabled, DSMC needs to have os, lfs and io library available in the SSE**. DSMC itself have a good automatic desanitization check & mod function, but it will run only if your DCS and PC user has write permission. Else, you will need to modify those strings using notepad++ or similar code editor (google about it in the ED forum to understand what I'm talking about).

GROUND GROUPS

The following instructions about ground groups are more about building a future-proof scenario, and they're not a "must" today. But you can take them as good design guidelines.

- Keep ground groups in a size of 2-to-6 units each for vehicle and 2-to-10 units each for infantries.
- Try to use always the same unit type inside the group: it's much better to have 1 group of MBT + 1 group of APC + 1 group of IFV instead of 3 groups made by 1 MBT + 1 APC + 1 IFV.

[6] If you don't know who I'm talking about, doesn't matter. But if you're a tester and think that I'm talking to you, well, it's you.

SHIP POSITION UPDATE & CARRIER GROUP

DSMC update ship position in the same way it does with ground units, **except** from the carrier group (every ship in the group included). That is a necessary compromise to be able to keep flights starting over a carrier in their proper position & parking slots.

So, carrier group won't move from a mission to another, while any other ships will. If you want to move the carrier group, it's obviously possible but will require to edit the saved mission before re-launch it.

WAREHOUSE SUPPLY CHAIN

If you like logistic, this is going to be a key point. As said, all supply operations will be performed at the end of the mission. First thing: **DSMC won't resupply airframe**. It will resupply jet fuel or any type of ammunition, but no aircraft will be resupplied. That is by design decision.

There are 3 ways to resupply an airbase or a heliport:

- By landing there (this will also "move" the aircraft item): everything that is on board the aircraft, fuel & munitions, will be added to the warehouse;
- If inbuilt CTLD is active, by dropping "airlift" crates, which can add fuel or munitions as described in the inbuilt CTLD guide;
- Using the supply chain, described below.

What are the differences between those 3 solutions?

Timing: Landing somewhere will add items & fuel to the warehouse both using DMSC and the Resource Management System of DCS. That means that resources added by landing are available immediately to any other user in that airbase/heliport.

Instead, both airlift crates and supply using the supply chain in DSMC happens at the end of the mission: first the airlift, then the supply chain. Items & fuel will be available in the subsequent run and not real time.

Amount of items & fuel: Both landing and airlift crates adds item and fuel to the warehouse. By adding them, they also "set" new required values of the warehouse. To make it easier: if I land in a heliport that currently has 10 tons of fuel, and I add other 2 tons, in the subsequent mission I'll have

12 tons of fuel… and when depleted, the heliport will try to replenish all the necessary fuel to go back to 12. To 12, not 10! That means that by using airlift crates or landing, you're also increasing the airbase/heliport operational capabilities.

Beware: the "increase capability" take effects only in those cases:

- By landing, **if no items were used** in that airbase/heliport in the same mission;
- By airlift crates, always.

Ok, that said… you know what means to land somewhere (I presume, else… you might probably check flight manual before this one)… and you can easily understand airlift crates reading the CTLD manual.

But WTF is the supply chain?! The supply chain is, by order of magnitude, the most powerful way to replenish used resources in airbase/heliport. Basically, it will check the number of items depleted during the mission and try to find a supplier that has them. If found, those items are removed from the supplier instead of the airbase, that will stay on its initial values. This is also recursive: If the supplier has a supplier itself, it will remove the items from the highest level available.

Then, who is "supplier" of the airbase? Those that are set as supplier in the mission editor. Yes, I made it simple.

But **BEWARE**: since it's not possible to track the RMS of DCS, to be consistent with fuel & items you must accept a compromise: no real time resupply must be used with the RMS. **To be sure that that's not happening, every mission save the timing/size/speed of the RMS resupplies are minimized to prevent them happening.**

Let's dig a bit into how the supply chain works.

Supply mechanics

If 3 aircraft consumed 10 tons of fuel and 6 AIM-120B, then the airport where they operate will check for available suppliers of the same coalition (that you set into the mission editor).

DSMC will check the best suppliers: one that can resupply all the quantities or the biggest part of it. If you set a warehouse (or another airbase) with 40 tons available, then DSMC will transfer the 10 used tons to the airbase, removing them to the supplier. If you set a warehouse with 6 tons, DSMC will take all those 6 and will also remove 4 from the airbase.

If one supplier has fuel, and another the AIM-120, DSMC will take resources from both.

Since DSMC do not track info about “first mission situation”, you surely want to carefully track the initial amount of an airbase resources, because you can supply them only as explained before.

Warehouse objects

DSMC don’t require any particular setup to define a warehouse object. It uses all the inbuilt DCS info: if something is a warehouse in DCS, it will be in DSMC, along with its resources information.

Therefore, any airport/FARP/Oilrig and so on are warehouses. Any ship with helipad or carriers are warehouses. Any FARP resupply object, are warehouses.

Also, any static object of “warehouse” category is a warehouse: you will recognize them because instead of the classic icon, you will see the warehouse icon for fuel or ammo:

Standard Static object

Fuel tank Warehouse object

Ammo depot Warehouse object

Remember, DCS is a sandbox:

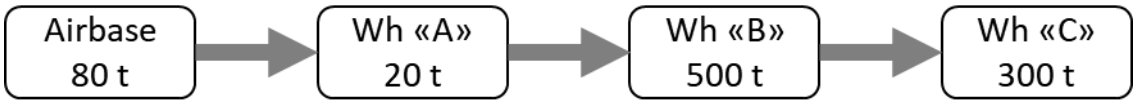
Every time you place a warehouse object on the scenery, it can be everything: a fuel tank (with that icon) can be an ammunition supplier, unless you manually void it. So, as per units’ placement, flight, etc… it’s up to you to set up that correctly.

Sadly, if you copy & paste a warehouse in DCS, you still have to manually set its “warehouse content” table.

Basic vertical supply chain example

In all the examples below, I’ll use fuel (jet fuel) as main resources as example… cause it’s easier to explain the mechanics.

In the mission editor, you can set a supply net, like:



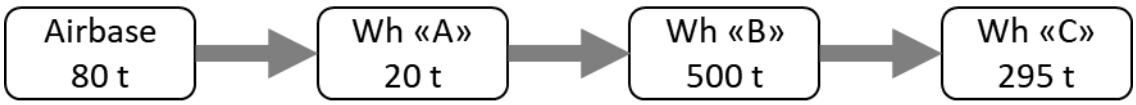
Airbase is the place where your flights consumed resources. The subsequent “Wh” can be another airbase as well as any warehouse object. Let’s say that they are three fuel tank objects where the only resources available is jet fuel (no munition, aircraft are ignored by design).

In our example, Airbase consumed 5 tons of jet fuel at the end of the mission.

Since all of them has sufficient jet fuel, all the tons will be removed from warehouse C cause that is the “mother” of the chain: in fact, at the end of the mission this will happen:

- Airbase require 5 tons from “A”. “A” has them, so they move 5 tons to airbase (you won’t see them depleted in the airbase);
- Warehouse “A” require 5 tons from “B”, same thing;
- Warehouse “B” require 5 tons from “C”, same thing;
- Warehouse “C” does not have suppliers available: the 5 tons will be depleted to it;

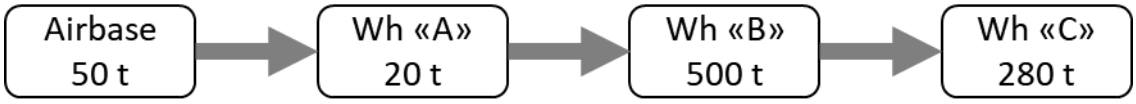
End situation? This:



That is the key of “vertical” side of the chain. But… what happen if the airbase consumes 50 tons?

- Airbase require 50 tons from “A”. “A” do not have them, so they move the maximum amount available (20 tons) to airbase. The remaining amount, it will be depleted from the Airbase.
- Warehouse “A” require 20 tons from “B”, same thing;
- Warehouse “B” require 20 tons from “C”, same thing;
- Warehouse “C” does not have suppliers available: the 20 tons will be depleted to it;

Final situation:

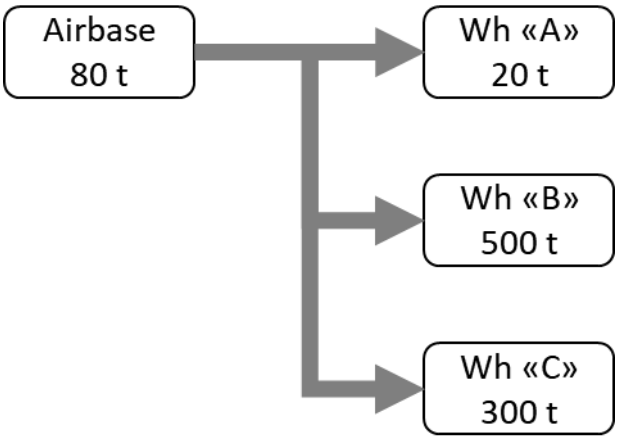


Note the “50 tons” at airbase, which has 80, while warehouse “C” took the 20 tons transferred by warehouse “A”.

What does this mean? That you can set a limit of the expendable fuel per mission, which is given by the warehouse that has less resources. If you use more, this will be depleted from your airbase… and to re-charge that, you will now be forced to land something on it.... which is obviously much harder to do.

Basic horizontal supply chain example

Obviously, you can also set horizontal supply. What does this mean? That an airbase can have multiple warehouses to check for. Let’s say this:



In this case, DSMC will use the better suited warehouse, which is “B” cause it has more fuel than “A” and “C”. This will be the result:

So, what does DSMC not allow? Multiple resupplies to the same base from the warehouses. To explain that easier: DSMC will choose the supplier with more fuel, but only that. In this situation:

DSMC will use full “A” warehouse (cause it’s more than B and C), but will also remove 30 tons of the 50 you asked for from the airbase. It won’t remove anything from B and C. This will be the result:

HOW MANY ITEMS A WAREHOUSE CAN DELIVER?

DSMC makes it simple: any warehouse can deliver an amount of fuel or ammo proportional to the “size” parameters that you can set in the mission editor (standard value is 100, when you create a new object).

Fuel: same as “size”, as jetfuel tons;

Dumb weapons: same as “size”, as items number, for each type (that means 100 S-8 rockets *and* 100 Mk-82, for example);

Smart weapons: 1/5 of the “size”, as items number, for each type (so if size is “100”, it will be 20 of each weapons type);

BEWARE

In the end, there a situation you MUST avoid at all cost. As explained in the vertical chain, DSMC supply chain function will search the best supplier at each step.

But if you accidentally create a loop, like:

Airbase X → Wh A → Wh B → Wh C → Airbase X

That will lead to a DCS & DSMC crash due to endlessy resupply run. So, check this condition very well.

Production sites, coalition depot

So… if you get the basic examples I wrote above, you should easily understand that you can build a couple of interesting warehouse items without having to code anything.

You can create coalition depot of a resource (ammo, fuel) and also create production sites!

Coalition depot

Using fuel as example, to create a coalition depot you can simply create one (or more, in proximity) “tank” warehouse object and place them where you want on the map, filling them with a high value of jet fuel only and no ammunition. Let’s say you place 3 fuel tanks in proximity with 1000 tons of fuel each. Now you set the Tank #001 to be supplied by Tank #002 and Tank #003. Now you have a depot of 3000 tons of fuel.

Then you can set all your airbase to have low fuel amount (let’s say 100 tons) and Tank #001 as supplier. Every consumed fuel in those airbases will be removed from your depot, since it becomes empty.

What’s the point? To have the depot as a valuable target! If enemy destroy the depot, then you will have your resources depleted.

Production sites

To create a petrol production site, you must simply place a warehouse object and set it as “unlimited”. A production sites won’t require suppliers, but it’s still limited! In fact, unlike for

STANDARD SUPPLY NET

First: the standard supply net can be used automatically activating the *DSMC_WarehouseAutoSetup* options in multiplayer, and it’s active by default for single player. That means that if you’re using DSMC in SP, **you must build a scenario that follows these supply rules** or else that is not using warehouse tracking system to prevent unexpected behaviour and supplier changes.

Second: standard supply net, and its activation, **do not change the content of the warehouse**. So, while this option will provide an automatic update of all the supply net without your need to create it in the mission editor using all the possible combination since start, you still have to perform an accurate initial setup of your scenery before launching "mission 000" that comply with the basic scheme explained below.

What autosetup does: this feature works after all units & static objects have already been updated, so it takes action when all the warehouse static objects had changed coalition if necessary.

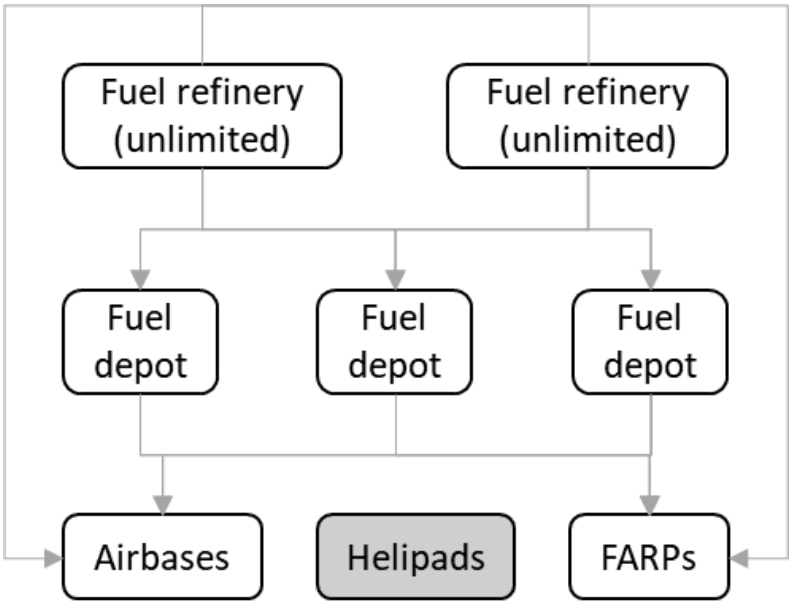
The autosetup use some fixed rules that you need to consider when building a new scenario if you want to take advantage of this option:

- Airports is recognized as "customer" and can only be supplied by depots, but they're not supplier of anyone;
- The very standard FARP object (with 4 helipads) is recognized as "heliport" and therefore will work the same as airports;
- All the others heliport objects (single helipad, invisible, oilrig, etc) won't be set as customer and won't ever have any supplier;
- Warehouses with the shape of a tank will be recognized as fuel depot;
- Warehouses with the shape of an ammunition dump or a standard warehouse will be recognized as ammo depot;
- Warehouses (of any kind) that are set with unlimited fuel or unlimited weapons will be recognized as refineries/ammunition factories.

So, it appears important that setting up the scenario the mission designer will follow these rules:

- All warehouse objects must be set with their content consistent to their shape. It's also important that the other items column will be set as void. For example: set 0 to each aircraft and ammo for the fuel depot, and vice versa for the ammo depot;
- If you set a depot as unlimited in its main item, this will be recognized as production site (with will have unlimited supply over time, but limited supply for each mission!).

The standard supply net for fuel is built following this scheme:



Ammo supply net is the very same as for fuel. The scheme follows these rules:

- Customers (Airbases & FARPs) are supplied by both depots and productions sites;
- Depots (warehouse, not unlimited) are supplied only by production sites;
- Production sites (warehouse, unlimited) does not have suppliers.

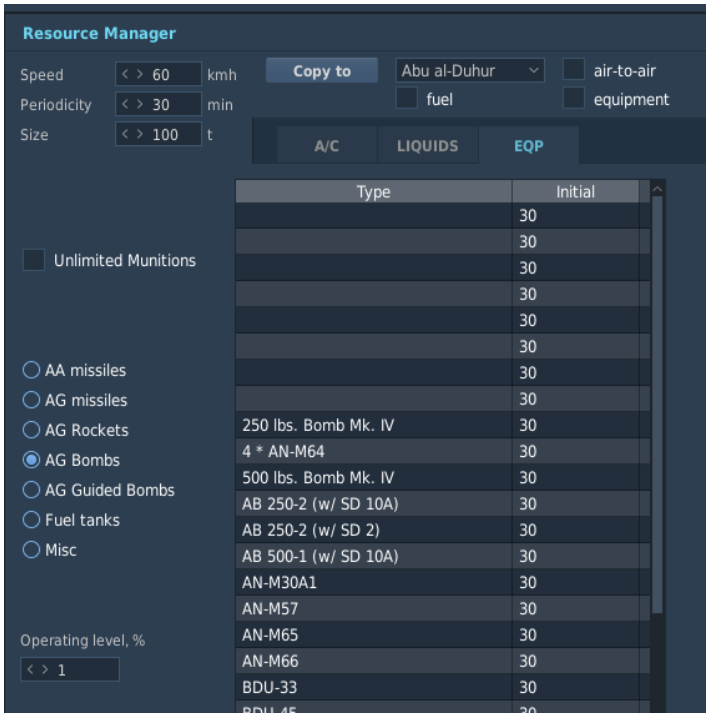
The automatic supply chain setup therefore provides an update in each mission where the available suppliers for each coalition are effectively used in the supply net.

WAREHOUSE FIX FROM UPDATE TO UPDATE

One of the worst thing that I had working with a well defined resource system is that every time DCS update the weapons list or aicfraft list… you have to redo almost everything by set unlimited a warehouse, save, then set it limited… and manually re-entry every single things. Crazy. But necessary. **Until now**.

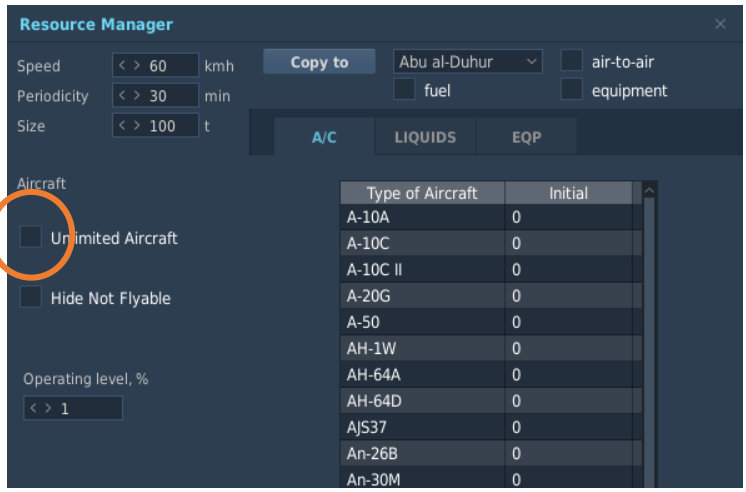
Since DSMC takes a lot of effort in making resource something that is valuable of attentions, it was also necessary to find a way to remove this pain in the ass of every mission designer.

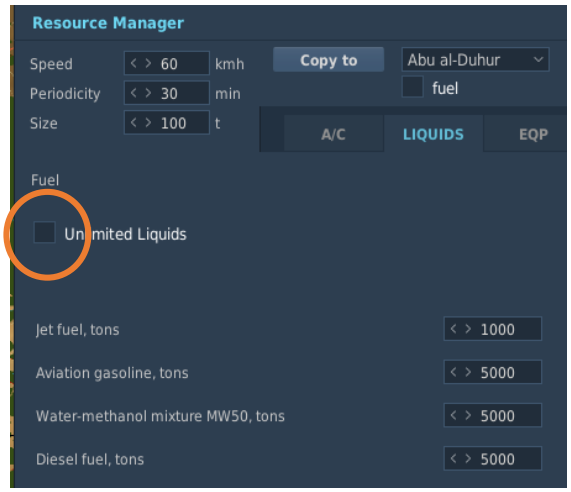
So, every time you update DCS and you see something like this in your weapons or aircraft lists:

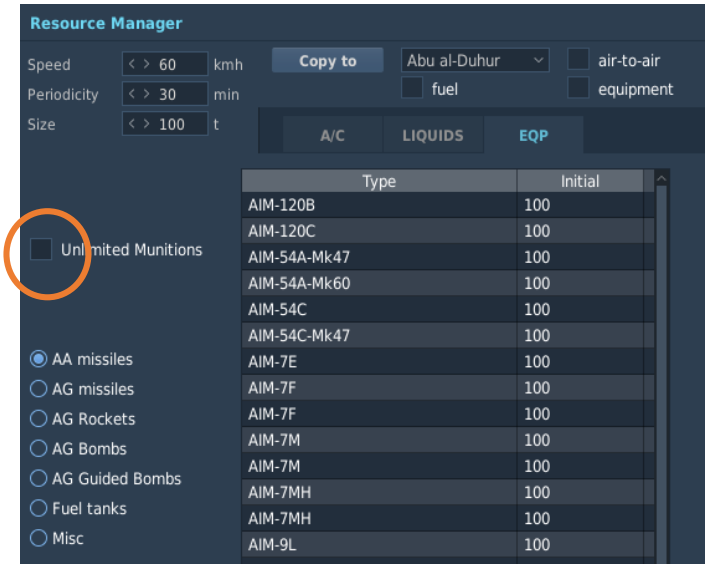


Follow these steps:

Create a new void mission in DCS, any map will be ok. Choose an airport. Enter its warehouse and **set everything to be not unlimited**. Like this:







Then save this mission in the DCS default mission directory and be sure that it has this exact name: **DSMC_fix_warehouse.miz**.

Now launch your “broken” scenario, and save it using DSMC…. Be patient! The fix might take a lot of time, even 2 or 3 minutes, and windows will tell you that DCS might be crashed. It’s not. Wait until the save complete (if it goes more than 5 mins, it’s not ok).

Once done, you should get your DSMC’s saved mission with the warehouse entries fixed. All the new items will be set to “0” as initial amount.

CUSTOM FILE INSIDE MISSION

You can add any external file inside the mission if you want, but I recommend to use winzip and NOT 7zip to pack the .miz file, cause I saw a couple of compatibility error while saving due to the packing procedure, that happen only when 7zip is used for this kind of operation (like adding sound .ogg files)

SPAWNING, IN ANY POSSIBLE WAY (MOOSE OR CODE)

This is a **very important point**. DSMC do not know what you spawn or not, but if you spawn anything and it reaches alive the end of the mission, this thing will be saved and you will find that in the saved mission (if the spawned option is enabled).

What does this mean? That if you have any sort of automation that spawn object at mission start… **you will cumulate those object at any subsequent start**! So, pay particular attention about this.

Since DSMC keeps the group names, this could be easily workaround by checking if that group exist before spawning another one with the same name, or else performing a “world.searchObjects” inside the spawning area to see if something is still there.

AIRBASE RUNNING OUT OF FUEL

If you use functions that spawn aircraft from bases, such as MOOSE dispatcher, you will need to check if that base has enough fuel. Sadly that is not possible in DCS. To prevent issue, with the help of Pikey, I added a small check to DSMC. In the scripting environment you will find a table called *EMBD.airbaseFuelIndex*. This table will be populated with the name of the airports once a birth event happen and the aircraft spawned has no fuel. The aircraft will be immediately removed, you will find a couple of messages into the dcs.log file and the table will be added with the airbase name.

i.e. if a Mig-29S try to spawn at Gudauta, but Gudauta don’t have fuel, the Mig-29 will immediately disappear and *EMBD.airbaseFuelIndex* will be like this:

```
EMBD.airbaseFuelIndex = {
    ["Gudauta]" = true,
}
```

PS: important thing. When you spawn aircraft with fuel tanks, those are fulfilled with fuel even if the airport is at 0 tons.

FARP & AIRBASE COALITION CHANGES

The helicopter landing position save option do not play well with coalition changes. If you plan to land in a conquered airbase or FARP and find your helicopter there in the saved mission, be sure that something else perform the coalition change before you land.

An important point about base/FARP coalition change & helicopter land position update (*DSMC_SaveLastPlanePosition* variable for servers). Imagine this situation:

If you land as a blue helo in a blue farp, then disconnect… so that helicopter group "slot" will be set in that farp in the saved file. But, before the save, some red units come nearby your farp and change its coalition!

In that case, you will see a blue helicopter slot over a red farp. And you can use that helo, only… you won't be able to use the farp services.

To avoid erratic behaviour, I suggest to set all aircraft/helicopter with 0 kg of fuel on start: that way, you may spawn… but you won't be able to load fuel and turn on the aircraft!

CTLD ADDITIONAL CONSIDERATION

CTLD modified version is entirely coded inside the file "TRPS_inj.lua", DSMC directory. You can check inside and customize standard CTLD options as you like, but you should consider this chapter before touching. I can (almost) guarantee a "good behaviour" of the mod if you leave the file as it is.

Crates

All created crates will retain their original "content", unless you change their weight using mission editor. You can complete an SAM installation or FOB in three consequent mission, if you'd like.

Unpack Crates

Here comes something new, pay attention. **You won't be able to unpack any crates where you want**. You will be able to unpack SAM sites and

support vehicles (FARP, rearming) where you like. And that's the same as before.

But any other vehicles, like IFVs, Tanks, APCs can be unpacked only nearby a warehouse object (static object -> warehouse). Once unpacked, you can move them where you need. This will make your warehouse not only a resource production site (if you use warehouse tracking) but also a "military factory", that you will try to protect for sure.

"simulated" sling load action

This option has been disabled by design. You can try to look into TRPS_inj.lua file to modify that, but I strongly suggest to avoid.

FARP support group

A "FARP Support group" has been added as spawnable for blue & red coalition. Will require the proper vehicles as DCS needs, one crates per vehicle. FARP support group won't have anything special: it's a ground vehicle group with the right composition to allow ATC, rearm & refuel when placed in proximity to an heliport.

"In transit" crates

It could happen that the mission will save or close while a crate is ongoing a transport, "inside" an APC or an helicopter (if you don' use slingload mode). In that case, the crates is lost. DMCS won't retain in transit crates.

FOB

Created FOB and its proximity beacon are retained in the saved mission.

JTAC

Due to design reasons JTAC is performed by "manpad" units: Stinger for blue coalition and SA-18 for red coalition. The main reason is that

currently JTAC are unarmed unit that is required for FARP support units, creating bugs.

Anyway, that is a technical feature: “manpad” units can be used by player/client just as vehicles, but they can also be transported by helicopters.

BEWARE: any “single soldier manpad” group will be inspected at mission start, set as JTAC unit, and set ROE to “**return fire**”. Therefore, if you want normal manpads to behave correctly, you simply need a group of at least 2 units. You can add a single soldier, or another manpad, or anything you want. But don’t leave the manpad unit alone!

Logistic zones/object

As said internal CTLD version automatically recognize any previously built FOB & any warehouse object as a logistic object, allowing you to spawn crates, load squads, etc etc.

Also, FOB recognition is not done by adding properties to the object. Instead, it’s done by checking static object in the saved file: each time you place a beacon TTS unit (Fortification class) within 150 meters from an outpost static object (not road outpost, not a fortification), those will be recognized as “FOB” and full ctld functionality will be added to them, including beacon.

PS: that beacon won’t retain the same frequency as the previous mission: it will change each time the mission is restarted.

ADVANCED SERVER CUSTOMIZATION

Inside the DSMC_Dedicated_Server_options.lua file there are also some “advanced server options” that should be used **only** if you clearly know what you’re doing. In particular, you will find:

DSMC_CTLD_JTACenable: If true, this option will enable automatic JTAC feature and also the ability to spawn JTACs crates.

DSMC_CTLD_AllowCrates: This option will enable/disable all the crates transport & building features in inbuilt CTLD. If it's false, you will have only troops transportation.

DSMC_CTLD_AllowPlatoon: This option will enable/disable a new ground unit listing for CTLD crates logic. Instead of having a fixed ground vehicle list that you would be able to create using crates, the list will vary from country to country, regardless of coalition, and depending on everything available in that country units as per DCS database. That won't work with complex SAMs, only for armoured vehicles!

DSMC_CTLD_UseYearFilter: Since for many countries there are A LOT of possible units, but we don't see why you would build a T-55 in 2015, if you enable this filter you will be able to built units that have been created in the last 40 years before the mission start date. If your mission start in 1970s, you won't see T-90 in the CTLD platoons crate list, for example.

*DSMC_CTLD_UnitNumLim*its: this option is done to prevent a server to become unbalanced in ground units availability. If It's set as "true" both coalition won't be able to have more than "n" units category available in the scenery: if that number is more than the limit, you won't be able to use crates to create more vehicle of the same category (you won't be able to unpack the crates, but you can spawn and move these crates).

DSMC_CTLD_Limit_(unit category): This is the number I spoke in the previous option. For example, if you set *DSMC_CTLD_Limit_Tanks = 50* then both coalition will be able to have 50 tanks, no more. If the starting scenery has 200 tank units in the blue coalition, then you will be able to create them using crates only when the alive tanks count goes below 50.

CREDITS

If DSMC works as well as you can see, you have to congrats to all the testers that tried it knowing that it wasn't working yet and tirelessy reports any strange behaviour, bug, or even compatibility issues with MOOSE, CTLD and MIST.

Else, if DSMC doesn't work as expected, here you can find a fairly accurate list of everyone actively helped me or tried to make this work: fell free to blame them. Not me obviously, I'm clearly innocent.

SPECIAL MENTIONS

https://simitaliagames.com/

The day I started asking for "beta testers" I never imagined I could find an entire community of DCS simmers ready to launch the last DSMC built on their 24h/24 servers. SIG_Webber, who manages the servers, was fantastic both in willingness to test but also in reporting every anomalous condition in detail.

One of the most dedicated tester is alt.sanity, he tested DSMC extensively, with particular focus on helicopter operations. He also helped a lot with dedicated tests aimed to reproduce certain undesidered behaviour or bugs.

TESTERS

Besides of the communities above, a lot of effort to make DSMC works has been done by all testers, which tried the mod both in single player and multiplayer and make me aware about many issue that could be solved before the release. Here a summary list of those who directly reported as testers during the “beta” stage, by nickname:

- Dax;
- Enkas;
- ESA_Furia;
- Neon;
- Panthir;
- Paranoid;
- PVI_Eagle;
- Gunny;
- Frosties;
- KaptinKaos;
- [SG] Idefix;

MASTERS

If DSMC even exist is due to the skills that the following people gave to me, directly or even by reading their code. Many of them is always there for sharing knowledge, and that is the core of a working community. From some of them I simply reverse engineer their code to learn, and rebuilt them (or simply modifying them, like for inbuilt CTLD version) as I needed.

So, many many thanks to:

- Rider;
- Ciribob;
- Grimes;
- MBot;
- Pikey;
- Pravus;
- Speed;
- Xcom;

VERY SPECIAL ONES

There are at least two very special person I would like to thank for their patience and dedication to this project:

- My girlfriend, Nicoleta. She always support me with her infinite patience, jokes, fantastic cooking skills, endless love;
- You, cause if you're reading here it means that you care about this small project very much. And even if there will be bugs and sometimes issues to be solved, I know that I can count on you to make DSMC better every day.

DSMC MOD SUPPORT (OR THANKS)

I made this mod for passion, not money. I'm not a software designer, I don't plan to become one, because I love my current job. I put an incredible amount of hours in coding DSMC (in the range of the thousands), but that is mostly cause I started that I didn't knew what means "local", for who could understand, and I had to re-done it at least three time.

I'm already working on the next step... it might take years, maybe less, but I'm not in a hurry. That said, this mod-creation hobby takes hours, usually night hours, so if you'd like to help me completing you can do it by giving me a coffee!

How? I set up a paypal donation and you can add 1 € (the standard espresso here in Milan). If you really liked the mod and you find me to deserve this additional coffe, then click here.