

Wargame: Red Dragon Internal Mechanics Manual

Resident Mario

November 6, 2016

Contents

1	Introduction	7
2	Getting Started	8
3	TUniteAuSolDescriptor	10
3.1	DescriptorId	10
3.2	_ShortDatabaseName	10
3.3	_ClassNameForDebug	10
3.4	StickToGround	10
3.5	ManageUnitOrientation	10
3.6	HitRollSizeModifier	10
3.7	DeathExplosionAmmo	11
3.8	IconeType	11
3.9	PositionInMenu	11
3.10	NameInMenuToken	11
3.11	AliasName	11
3.12	Category	11
3.13	AcknowUnitTpye	12
3.14	TypeForAcknow	12
3.15	Nationalite	12
3.16	MotherCountry	12
3.17	ProductionYear	12
3.18	MaxPacks	13
3.19	Factory	13
3.20	ProductionPrice	13
3.21	MaxDeployableAmount	13
3.22	ShowInMenu	13
3.23	ProductionTime	13
3.24	CoutEtiole	14
3.25	TextureForInterface	14
3.26	TextureTransportForInterface	14
3.27	TextureMotherCountryForInterface	14
3.28	UnitTypeTokens	14
3.29	UnitMovingType	14
3.30	VitesseCombat	15
3.31	UpgradeRequired	15
3.32	IsPrototype	16
3.33	Key	16
3.34	HitRollECMModifier	16
3.35	Modules	16

4	TTypeUnitModuleDescriptor	17
4.1	ControllerName	17
4.2	TypeUnitValue	17
4.3	TypeUnitHintToken	17
4.4	NameInMenuToken	17
4.5	GenerateName	17
4.6	Filters	17
4.7	MotherCountry	18
4.8	UnitInfoJaugeType	18
4.9	Training	18
4.10	CIWS	18
4.11	Sailing	19
5	TDebugModuleDescriptor	19
6	TStateEngineModuleDescriptor	19
7	TFlagsModuleDescriptor	19
8	TCriticalEffectModuleDescriptor	19
9	TTargetCoordinatorModuleDescriptor	19
10	TPositionModuleDescriptor	19
10.1	ControllerName	19
10.2	InGeoDb	19
10.3	ClampInWorld	20
10.4	GfxDescriptorPorteur	20
10.5	Radius	20
10.6	AddToHexagonMap	20
10.7	PorteurMustBeVisible	20
10.8	RelativeScanningPosition	20
10.9	CameraFollower	20
10.10	MustAllowZoneIndice	20
10.11	LowAltitudeFlyingAltitude	20
10.12	NearGroundFlyingAltitude	20
10.13	ClampOutMap	20
10.14	HasNearlyNullBBox	21
10.15	ShortDatabaseName	21
11	TInfammableModuleDescriptor	21
12	TLinkTeamModuleDescriptor	21

13 TModernWarfareExperienceManagerDescriptor	21
13.1 ControllerName	21
13.2 CanWinExperience	21
13.3 ExperienceGainBySecond	21
13.4 KillExperienceBonus	21
14 TIAStratModule	21
15 TModernWarfareCadavreModuleDescriptor	22
16 TTurretSkeletonModuleDescriptor	22
17 TMissileCarriageModuleDescriptor	22
18 TCommandManagerModuleDescriptor	22
19 TGhostManagerModuleDescriptor	23
20 TScannerModuleDescriptor	23
21 TScannerConfigurationDescriptor	23
21.1 ControllerName	23
21.2 UnitType	23
21.3 DetectionTBA	23
21.4 PorteeVision	23
21.5 OpticalStrength	24
21.6 OpticalStrengthAltitude	24
21.7 SpecializedOpticalStrengths	24
21.8 SpecializedDetections	24
21.9 PorteVisionTBA	24
21.10OpticanStrengthAntiradar	24
21.11_ShortDatabaseName	24
22 TFuelModuleDescriptor	24
22.1 ControllerName	24
22.2 FuelCapacity	24
22.3 FuelMoveDuration	25
23 TMovementHandlerLandVehicleDescriptor	25
23.1 ControllerName	25
23.2 Maxspeed	25
23.3 UnitMovingType	25
23.4 SpeedBonusOnRoad	25
23.5 TempsDemiTour	25
23.6 MaxAcceleration	25
23.7 MaxDeceleration	25
23.8 VehicleSubType	26

23.9	CriticalEffectModule	26
23.10	TerrainsToIgnoreMask	26
24	TMouvementHandlerHelicopterDescriptor	26
24.1	ControllerName	26
24.2	Maxspeed	26
24.3	MaxAcceleration	26
24.4	MaxDeceleration	26
24.5	UnitMovingType	26
24.6	CyclicManoeuvrability	26
24.7	GFactorLimit	27
24.8	LateralSpeed	27
24.9	Mass	27
24.10	MaxInclination	27
24.11	RotorArea	27
24.12	TorqueManoeuvrability	27
24.13	UpwardsSpeed	27
24.14	CriticalEffectsModule	27
24.15	TempsDemiTour	27
25	TMouvementHandlerAirplaneDescriptor	27
25.1	Maxspeed	27
25.2	UnitMovingType	28
25.3	FlyingAltitude	28
25.4	MinimalAltitude	28
25.5	PhysicsConfiguration	28
25.6	CriticalEffectsModule	28
25.7	GunMuzzleSpeed	28
25.8	LandingGearOutPhysicalPropertyName	28
25.9	LandingGearSubDescriptionName	28
25.10	FrontLandingGearMeshNodeName	28
25.11	BackLandingGearMeshNodeName	28
26	THaloModuleDescriptor	29
27	TGroupeCombatModuleDescriptor	29
28	TTransportableModuleDescriptor	29
28.1	ControllerName	29
28.2	Categories	29
28.3	SuppressDamageRatioIfTransporterKilled	29
28.4	TransportListAvailableForSpawn	29
28.5	TransportedTexture	30

29 TModuleModernWarfareSupplyDescriptor	30
29.1 ControllerName	30
29.2 SupplyCapacity	30
29.3 DeploymentDuration	30
29.4 WithdrawalDuration	30
29.5 SupplyPriority	30
29.6 SupplyDescriptor	30
30 TWeaponManagerDescriptor	31
31 TModernWarfareDamageModuleDescriptor	31
32 TAppearanceModelDescriptor	31
33 TVisibilityModuleDescriptor	31
33.1 ControllerName	31
33.2 UnitStealthBonus	31
33.3 _ShortDatabaseName	32
34 TCompanyUnitDescriptor	32

1 Introduction

This document lays out the structure and internal mechanics of the units in RTS video game Wargame: Red Dragon (and thereof, of the entire Wargame series). Over the course of its release the series has garnered a sizable modding community, one which has done much great work tinkering with the game and exploring the boundaries it lays out. Much credit goes in particular to Enohka, whose **Wargame Modding Suite** makes studying and patching the game's files easy and intuitive (or at least as easy and intuitive as modifying a game of this complexity can be).

This document was prepared and published in support of an initiative in the **extraction and datafication** of the 1,800+ units in Wargame: Red Dragon. In its attempt to summarize in detail all that is known about the game units' internals it has many precursors. Although there are other aspects to modding, unit creation and modification is easily the most heavily travelled and easiest aspect to modding the game, and it is the explicit focus of this text.

All numbers and counts in this guide refer to Wargame: Red Dragon circa late-2016, version 510049986.

2 Getting Started

This guide is written from the perspective of the Wargame Modding Suite. If you have not done so already, download this wonderful tool.

When you open up the Modding Suite for the first time, it should be able to find a reference to your most recent copy of Wargame: Red Dragon. If it does not, or you want to switch versions, click on "Open" in the taskbar, then navigate to your game's copy of the data file you would like to open.

Wargame database files are backed up, with the structure and contents of each previous version of the game backed up in its own folder on disk. The practical reason for this additional use of space is that it enables the in-game replay viewer, which obviously relies on the gamestate being what it was at the time that the game took place. It does take up additional space, but the files are relatively small compared to the game's art and physics assets, and having the game's history also lets us explore what it looked like in the past (something most other games don't allow).

As a consequence of this organization, files related to various patch versions of the game are stored as subdirectories of a top-level data folder: `C:/Steam/steamapps/common/Wargame Red Dragon/Data/WARGAME/PC` on my disk. If you open this folder you will see a descending list of folders, each of which referring to a specific patch version of the game. The higher the number, the more recent the version. Naturally the highest number corresponds to the most recent version, and the lowest number to the first release version.

The most recent version as-of-writing is 510049986. Each version of the game will be a similar 9-digit string. The first two elements are the major version number, while the last five (last three in older versions) is the patch number.

Each patch is accompanied by a Eugen changelist in the forums, so to find out what changes were applied in which patches, search that number in their forums. The major content patches are described in **series of forum posts**; go there to see them.

Every folder contains `NDF.Win.dat`, which is the primary database file for that version of the game. Information internal to Wargame follow an ascending schema: information and assets introduced in the first version of the game which do not receive any patches along the way (like almost all the art assets, for example) continue to live wherever they were first introduced, and do not get copied "up" the patch list. When Wargame initialized, the Loading Screen is actually the game working to link all of these changes up together to get a working schema of the current version of the game.

Within `NDF.Win`, things are organized in terms of files of the "ndfbin" type. The most important of these is "everything.ndfbin", which contains nearly all easily modifiable attributes of the game. This file, decompiled by the Modding Suite, in turn consists of a couple hundred tables. These tables, or "modules", are linked to one another within a complex hierarchy.

The most important table, for our purposes, is `TUniteAuSolDescriptor`. This table is top-level object describing all of the purchasable and playable units in

the game (as well as a variety of deprecated and occasional "special" ones):

The screenshot shows the Ndf Editor interface. The main window displays a table of unit descriptors. The table has columns for Name, Instances, and Instance. The Properties panel on the right shows the details for the selected instance, including fields like DescriptorId, Guid, Modules, ShortDatabaseName, ClassnameForDebug, StickToGround, ManageUnitOrientation, HitRollSizeModifier, DeathExplosionAmmo, IconType, PositionInMenu, NameInMenuToken, LocalisationHash, AliasName, Category, AcknowUnitType, TypeForAcknow, Nationalite, MotherCountry, ProductionYear, MapPacis, Factory, ProductionPrice, MaxDeployableAmount, ShowInMenu, ProductionTime, CoutEtoile, TextureForInterface, TextureTransportForInterface, TextureMotherCountryFoot, UnitTypeTokens, UnitMovingType, VitesseCombat, UpgradeRequire, IsPrototype, Key, and HitRollECMModifier.

Name	Instances	Instance
95 TModuleSelectorFilter	1858	14672
101 TCompanyUnitModuleDescriptor	1837	14673
81 TUnitEuiSolDescriptor	1818	14674
194 TWargameUnitModuleDescriptor	1815	14675
89 TAmmunition	1756	14676
103 TWeaponManagerModuleDescriptor	1661	14677
137 TTurretTwoAxisDescriptor	1587	14678
207 TMissileCarriageSubDepictionMis	1568	14679
325 TWargameUnitDeck	936	14680
219 TResourceAnimationMapHolder	915	14681
148 TTurretUnitDescriptor	668	14682
165 TCompositeHappening	601	14683
145 TTurretInfanterieDescriptor	592	14684
179 TMissileCarriageWeaponInfo	589	14685
357 TDeckConstraints	556	14686
200 TActionDescriptorLaunchEffectOn	421	14687
144 TMissileCarriageConnoisseur	404	14688
114 TMissileCarriageModuleDescriptor	404	14689
364 TStrategyMapEventPopupDescr	365	14690
350 TPoolElement	335	14691
176 TMissileCarriageSubDepictionGe	319	14692
209 TMultipleResourceAnimationMap	318	14693
185 TUnit_Odc_AnimationFastAccess	317	14694
57 TSequencingActionHappening	317	14695
229 TOfDescriptorAnimation	309	14696
60 TActionCall	306	14697
147 TWeaponModuleHolder	296	14698
146 TUnitBehaviourDescriptor	296	14699
115 TGroupCombatModuleDescriptor	296	14700
186 TSubDepiction	265	14701
331 TIAGeneralStrategyTransition	260	14702
235 TGoFastAccessKey	254	14703
107 TFuelModuleDescriptor	246	14704
87 TTransportableModuleDescriptor	242	14705
30 TIAcknowHQUnitDescriptor	239	14706
153 TAirplanePhysicsConfiguration	234	14707
121 TMovementHandlerAirplaneDes	229	14708

Name	Type	Value
437 DescriptorId	Guid	00000000-0000-0000-0000-0000e803000
438 Modules	List	Collection[25]
439 ShortDatabaseName	TableString	Descriptor_Unit_152mm_SpGH_Dana
440 ClassnameForDebug	TableString	Unit_152mm_SpGH_Dana
441 StickToGround	Boolean	True
442 ManageUnitOrientation	Boolean	True
443 HitRollSizeModifier	Float32	0.05
444 DeathExplosionAmmo	ObjectReference	89 : 16533 (True) - TAmmunition
445 IconType	Int32	2
446 PositionInMenu	Int32	4
447 NameInMenuToken	LocalisationHash	8E14D5902C0F8600
448 AliasName	WideString	DANA
449 Category	Int32	3
450 AcknowUnitType	Int32	32
451 TypeForAcknow	Int32	119
452 Nationalite	Int32	1
453 MotherCountry	TableString	TCH
454 ProductionYear	Unit32	1977
455 MapPacis	Unit32	2
456 Factory	Int32	13
457 ProductionPrice	List	Collection[5]
458 MaxDeployableAmount	List	Collection[5]
459 ShowInMenu	List	Collection[5]
460 ProductionTime	Int32	10
461 CoutEtoile	Int32	2
462 TextureForInterface	ObjectReference	78 : 16534 (True) - TUJResourceTexture
463 TextureTransportForInterface	ObjectReference	78 : 16535 (True) - TUJResourceTexture
464 TextureMotherCountryFoot	TransTableReference	Typeface3D_TitleMono_CompensatePost
465 UnitTypeTokens	List	Collection[2]
466 UnitMovingType	Int32	3
467 VitesseCombat	Float32	4160
468 UpgradeRequire	Unset	null
469 IsPrototype	Unset	null
470 Key	Unset	null
471 HitRollECMModifier	Unset	null

A sister table, TUnitDescriptor, describes both these units and additionally missiles, leading to a first, mildly amusing observation: missiles in the game are treated as their own units by the engine.

This table is our starting point.

Note that the first entry in this table is the 90-point Czech DANA artillery piece, which happens to, arbitrarily, have the lowest table ID in the game. This is followed by a Polish KUB-M anti-air unit, and then by a Russian Tunguska anti-air unit, as good a starting point as any other.

3 TUniteAuSolDescriptor

3.1 DescriptorId

This is a multi-part hex hash that is used internally by the engine. It is a unique key for the table. Don't touch it.

3.2 _ShortDatabaseName

Prepended version of _ClassNameForDebug.

3.3 _ClassNameForDebug

An always-present non-localized unit name. Sometimes not entirely serious: for instance, Li Jian are given the moniker "Chinese Swords", while South Korean elite spec ops are "Black Berets". Not the name displayed in-game.

3.4 StickToGround

True if the unit is a ground unit, else null.

3.5 ManageUnitOrientation

Null for infantry units, True otherwise.

It is believed that this controls whether or not the unit can be given a position orientation command (Shift+Drag) in-game.

3.6 HitRollSizeModifier

The effect that the size of the unit has on the chance-to-hit of other units firing at it. Larger-than-average units have a high HitRollSizeModifier, smaller-than-average ones have a low HitRollSizeModifier. This statistic is one of many calculations that factors into chance-to-hit.

The "Size" statistic in the armory screen is a direct translation of this variable. May be set to any float, but the values used in-game are:

Value	Armory	Applies To	Example	Instances
-0.2	Very Small	Scout helos	AH6C Little Bird	37
-0.15	Very Small	Infantry	Morskaya Pehota	296
-0.1	Very Small	Light helos	MI2URPG	20
-0.05	Small	Light vehicles	Scorpion Light Tank	20
null	Medium	Most things	BMP-1K	958
0.05	Big	Tanks, Heavy AA	M11P Abrams	237
0.1	Very Big	Large helos	Mi-26	9

3.7 DeathExplosionAmmo

A reference to a TAmmunition module which explodes in-place when this unit dies. This is a clever secondary use of TAmmunition, which is covered elsewhere in this guide.

At present time, units explode with one of just 11 animations. This includes null (e.g. no animation) which probably in error applies only to the Polish Star 266 supply truck.

3.8 IconeType

Unknown. Options are 1 (default), 2 (artillery and anti-ship missiles), and 3 (anti-air).

3.9 PositionInMenu

Does nothing.

3.10 NameInMenuToken

This localization hash controls the display name of the unit in the menu.

As previously mentioned, Wargame database files are backed up, with the structure and contents of each previous version of the game backed up in its own folder on disk. To get what this token corresponds to, we actually have to exit NDF_Win.dat and load ZZ_Win.dat instead (if this file is not present in your folder, check the immediately prior patch versions, one of them should have it). This file contains all of the text localization strings for the game. The `unites.dic` in particular contains all of the English-language menu texts, and if you for example look up our Dana's 8E14D59D2C0F8600 hash in this table you will find that it is, indeed, called the DANA in-game.

Thus you can change this yourself by editing this hash or by creating new ones and referring to them using the Modding Suite.

It needs to be noted that this piece of text corresponds to the name in the armory screen only. A second hash with the same name stored in the UnitType submodule, which we will get to later, controls the unit's name in-game.

3.11 AliasName

A better-formed name for the unit. Does nothing in-game. Not editable. Some are null.

3.12 Category

Unknown.

3.13 AcknowUnitType

Unknown.

3.14 TypeForAcknow

Unknown.

3.15 Nationalite

NATO units are null, PACT units are 1.

3.16 MotherCountry

An abbreviation for the country of origin of this unit.

Value	Nation
US	United States
UK	United Kingdom
FR	France
RFA	West Germany
CAN	Canada
SWE	Sweden
NOR	Norway
DAN	Denmark
ANZ	ANZAC
JAP	Japan
ROK	South Korea
ISR	Israel
HOL	The Netherlands
URSS	Soviet Union
RDA	East Germany
TCH	Czechoslovakia
POL	Poland
CHI	China
NK	North Korea

These are French acronyms, hence why URSS is "backwards".

3.17 ProductionYear

The year that the unit was produced. Note that this variable only controls the text; whether or not the unit is actually available in the unit-year categories for deck building (Cat A, Cat B, Cat C) is controlled elsewhere.

3.18 MaxPacks

The number of cards of this unit which are available for deck-building. Ranges from 1 (542 instances) or 2 (995 instances, the most common) up to 9 (16 instances, for certain transports).

3.19 Factory

Armory tab. Values are:

Value	Tab	Count
3	Logistics	177
6	Infantry	233
7	Planes	214
8	Vehicles	335
9	Tanks	196
10	Recon	197
11	Helicopters	142
12	Ships	65
13	Support	259

3.20 ProductionPrice

The in-game production price. This value isn't provided straight, but is instead embedded into a Collection of five integer elements. This may be because production price was at one point experimentally dependent on the veterancy of the unit, but this mechanic is not used in-game. Instead, the first value in this list is the actual price, and the remainder are all placeholder values of 15.

3.21 MaxDeployableAmount

A Collection of integers. The amount of copies of this unit deployable at each veterancy level, per card. 0 for veterancies this unit is not available at. Ordered Rookie to Elite.

3.22 ShowInMenu

A Collection of booleans which is always set to True for true units and always set to False for missiles.

3.23 ProductionTime

How long it takes, between an air unit being clicked on or a squad of land units being places on the map, for the unit to appear where it's expected. Airplanes have a ProductionTime of null (instantaneous), most units have a ProductionTime of 10, and helicopters have a ProductionTime of 5. This value is in seconds.

3.24 CoutEtiole

Means "Price Star" in French. A quickly-dropped mechanic in the first iteration of the series, Wargame: European Escalation, was that unit cards for your deck, besides the basic ones, would cost an in-game currency known as "stars" to unlock. This variable used to control this cost, and continues to exist today as a holdover. Seems to always be set to 1, 2, or 3, and doesn't do anything.

3.25 TextureForInterface

A reference to the texture resource for this object's deck display.

3.26 TextureTransportForInterface

Transport units can appear in the armory screen as separate units, but most of the time they are viewed during deck building "behind" the units they are carrying, in which case this texture reference is called up and placed. Is set to null for non-transport units.

3.27 TextureMotherCountryForInterface

Controls which flag gets displayed in the corner of the card display.

3.28 UnitTypeTokens

A list of LocalizationHash instances which control which deck type the unit falls into. Valid inputs are:

Kind	Hash
Mechanized	8BD43C9757360E00
Armored	5C76718B57360E00
Marine	23B8605ED9380000
Airborne	0BB7685ED9380000
Motorized	5E767965E3000000
Support	DAD77965E3000000

3.29 UnitMovingType

Flags the unit movement type. Values are:

Value	Kind	Count
1	Foot	296
2	Wheeled, Supply Truck, Land-Only	36
3	Wheeled, Non-Supply Truck, Land-Only	215
5	Tracked, Land-Only	445
6	Planes, Helicopters	448
7	Wheeled, Amphibious	121
8	Tracked, Amphibious	225
9	Ship	32

A unit's track style controls how it moves, and this field contains some important information on this subject.

Units moving on foot (1) move at the same speed on land everywhere, can move on steep slopes, and cannot enter water.

Wheeled units in general (2, 3, 7) move at 150 kph on roads, regardless of off-road speed (specified in `MouvementManager`), and move at 33% of their off-road speed in forests. Amphibious wheeled units (7) can additionally enter water, moving at a globally-set 50% of their off-road speed when doing so. I do not know what the difference between (2) and (3) is.

Tracked units in general (5, 8) move at 110 kph on roads, regardless of off-road speed (specified in `MouvementManager`), and move at 50% of their off-road speed in forests. Amphibious tracked units (8) can additionally enter water, moving at a globally-set 50% of their off-road speed when doing so.

It should be noted that while 150 kph and 110 kph are global values, they are not controlled globally. There is instead a multiplier in another submodule, `MouvementDescriptor`, called `RoadSpeedBonus`, which is always set to exactly what it has to be set to to make this fact true. Amphibiousness, by contrast, is universal: if the unit can tread water, it will do so at half of its off-road speed, and as no other variables control this behavior this happens without exceptions.

3.30 VitesseCombat

According to the name, the speed of the unit when it is in combat (e.g. whenever the unit can see enemy units). It is surprising that this is a top-level variable and unknown whether or not this variable has any effect. The input is an unsigned int; for information on what the unit of measurement is, refer elsewhere in this guide.

3.31 UpgradeRequired

Contains a reference to another `TUnitAuSolDescriptor`. When this unit is pulled up in the armory, assuming you did all of the other categorization placements right, this unit will display after the linked unit in the horizontal list. If this is set to null, the unit will appear either on its own line or as the first unit in the horizontal list, if another unit references it itself.

Five is the maximum value, as no more are allowed to stack in the armory; any number higher will move the unit to its own line. It's also notable that when transport units are linked via UpgradeRequired, if one unit is made available as a transport for a unit all of its children are, too (for example: enabling Mi-8T for Gornostrelki also enables Mi-8TVs).

3.32 IsPrototype

Whether or not the unit is a prototypal unit. Null if it isn't, True if it is. Affects deck-building, as prototypal units can't be used in global decks.

3.33 Key

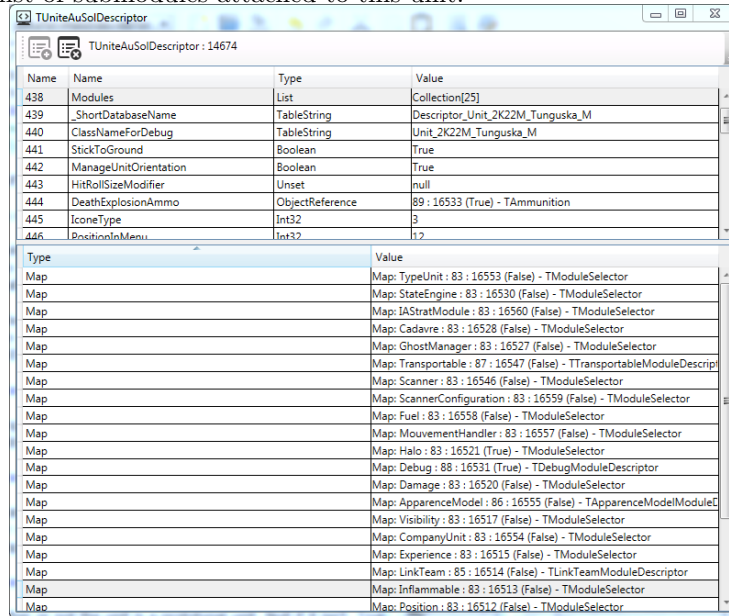
Unknown. Almost always null.

3.34 HitRolleCMModifier

The unit's ECM level. This only applies to planes and ships with higher than 0% ECM; all other units (including planes without ECM) have this value set to null.

3.35 Modules

A list of submodules attached to this unit:



Name	Type	Value
438 Modules	List	Collection[25]
439 _ShortDatabaseName	TableString	Descriptor_Unit_2K22M_Tunguska_M
440 ClassNameForDebug	TableString	Unit_2K22M_Tunguska_M
441 StickToGround	Boolean	True
442 ManageUnitOrientation	Boolean	True
443 HitRollSizeModifier	Unset	null
444 DeathExplosionAmmo	ObjectReference	89 : 16533 (True) - Tammunition
445 IconType	Int32	3
446 PositionInMenu	Int32	12

Type	Value
Map	Map: TypeUnit : 83 : 16553 (False) - TModuleSelector
Map	Map: StateEngine : 83 : 16530 (False) - TModuleSelector
Map	Map: IAStratModule : 83 : 16560 (False) - TModuleSelector
Map	Map: Cadavre : 83 : 16528 (False) - TModuleSelector
Map	Map: GhostManager : 83 : 16527 (False) - TModuleSelector
Map	Map: Transportable : 87 : 16547 (False) - TTransportableModuleDescriptor
Map	Map: Scanner : 83 : 16546 (False) - TModuleSelector
Map	Map: ScannerConfiguration : 83 : 16559 (False) - TModuleSelector
Map	Map: Fuel : 83 : 16558 (False) - TModuleSelector
Map	Map: MouvementHandler : 83 : 16557 (False) - TModuleSelector
Map	Map: Halo : 83 : 16521 (True) - TModuleSelector
Map	Map: Debug : 88 : 16531 (True) - TDebugModuleDescriptor
Map	Map: Damage : 83 : 16520 (False) - TModuleSelector
Map	Map: ApparenceModel : 86 : 16555 (False) - TApparenceModelModuleDescriptor
Map	Map: Visibility : 83 : 16517 (False) - TModuleSelector
Map	Map: CompanyUnit : 83 : 16554 (False) - TModuleSelector
Map	Map: Experience : 83 : 16515 (False) - TModuleSelector
Map	Map: LinkTeam : 85 : 16514 (False) - TLinkTeamModuleDescriptor
Map	Map: Inflammable : 83 : 16513 (False) - TModuleSelector
Map	Map: Position : 83 : 16512 (False) - TModuleSelector

Each of these passes through a TModuleSelector to another table of some kind. We will now go through each of these in order of appearance.

4 TTypeUnitModuleDescriptor

4.1 ControllerName

Always TypeUnitController. Immutable.

4.2 TypeUnitValue

Unknown.

4.3 TypeUnitHintToken

Unknown.

4.4 NameInMenuToken

Just like its top-level NameInMenuToken, except that this copy of the hash controls what the unit's name is in-game, whereas the top-level one controls what it is in-menu. For more details see the UniteAuSol NameInMenuToken field.

4.5 GenerateName

Unknown. Always True.

4.6 Filters

This complex field is a Map containing four sets of things.

The first element is an in-game hover name which displays when a unit is hovered over by an enemy. An example value is "Anti-Air Vehicle".

The second element controls which of the year-based deck types the unit falls into (Cat A, Cat B, or Cat C). A dictionary for the values is:

Value	Kind
41E22D4DD9380000	1980 and less
46E22D4DD9380000	1981 to 1985
81E22D4DD9380000	1986 and later

Note that it is this field, not the YearProduced top-level variable (which only controls an element of the card display), which controls which year-limited deck categories the unit is available in.

The third element is a set of the deck types that the unit appears in. But wait, you may ask, doesn't the top-level UnitTypeTokens already do that? It turns out that to make things work you have to properly set both that value and this one. Again, here's a list of valid hashes:

Kind	Hash
Mechanized	8BD43C9757360E00
Armored	5C76718B57360E00
Marine	23B8605ED9380000
Airborne	0BB7685ED9380000
Motorized	5E767965E3000000
Support	DAD77965E3000000

The fourth element is a set of hashes for miscellaneous filters that the unit falls under in the army. So for example if you want a unit to fall the "Cavalry Tank" filter there, you would need to set the appropriate hash here. The values for these hashes in particular are located in the interface.ingame.dic file inside ZZ_Win.dat.

4.7 MotherCountry

A copy of the top-level attribute.

4.8 UnitInfoJaugeType

Unknown.

4.9 Training

Unknown. Usually, maybe always null.

4.10 CIWS

The CIWS statistic, as displayed in the armory, carried by a naval unit. Set to null for non-naval units and for naval units without CIWS. This is a localization hash; to set it to a particular value, use one of the following:

Kind	Hash
Exceptional	4F233E0000000000
Very Good	4E96452000000000
Good	4E96450000000000
Medium	D672711906000000
Bad	CEC2000000000000
None	null

Changing this value does NOT change a ship's CIWS, it only changes the quality of CIWS reported on the card in-game. In other words, this only controls a text display element.

4.11 Sailing

The ship's sailing type. This is similarly a LocalizationHash controlling a text display, not the real value (which is in MouvementControl). Values are:

Kind	Hash
Deep Sea	CBD32D65B4780000
Coastal	CBD33165B4780000
Riverine	CBD33565B4780000
None	null

5 TDebugModuleDescriptor

There is nothing interesting here.

6 TStateEngineModuleDescriptor

There is nothing interesting here.

7 TFlagsModuleDescriptor

Unknown.

8 TCriticalEffectModuleDescriptor

A reference to one of a small number of tables which control the critical effects a unit may be subjected to that occur due to fire from enemy units (critical effects affecting movement are in a different module, Mouvement). We will omit further details, as the resultant tables are various, singular, and pretty easy to parse.

9 TTargetCoordinatorModuleDescriptor

There is nothing interesting here.

10 TPositionModuleDescriptor

10.1 ControllerName

Not interesting.

10.2 InGeoDb

Unknown.

10.3 ClampInWorld

Unknown.

10.4 GfxDescriptorPorteur

Links to a module containing information of interest to the physics engine, which shouldn't be messed with.

10.5 Radius

Unknown.

10.6 AddToHexagonMap

Unknown.

10.7 PorteurMustBeVisible

Unknown.

10.8 RelativeScanningPosition

Unknown.

10.9 CameraFollower

Links to a TGfxDescriptorCameraFollower object that sets variables for the camera, meant for focus-watching a plane, land unit, ship, or helicopter.

10.10 MustAllowZoneIndice

Unknown.

10.11 LowAltitudeFlyingAltitude

If the unit is a helo, the altitude that the helo flies at by default.

10.12 NearGroundFlyingAltitude

If the unit is a helo, the altitude the helo goes to when told to descend (which helps with stealth, but is barely ever used anymore).

10.13 ClampOutMap

Unknown, always null.

10.14 HasNearlyNullBBox

Unknown, always null.

10.15 _ShortDatabaseName

Useless, always null.

11 TInfammableModuleDescriptor

There is nothing interesting here.

12 TLinkTeamModuleDescriptor

There is nothing interesting here.

13 TModernWarfareExperienceManagerDescriptor

13.1 ControllerName

Disinteresting.

13.2 CanWinExperience

There are only two possible table references: one with CanWinExperience set to null, which is applied to supply units and units without weapons, and one with CanWinExperience set to True, for everything else.

13.3 ExperienceGainBySecond

Always set to 0.1.

13.4 KillExperienceBonus

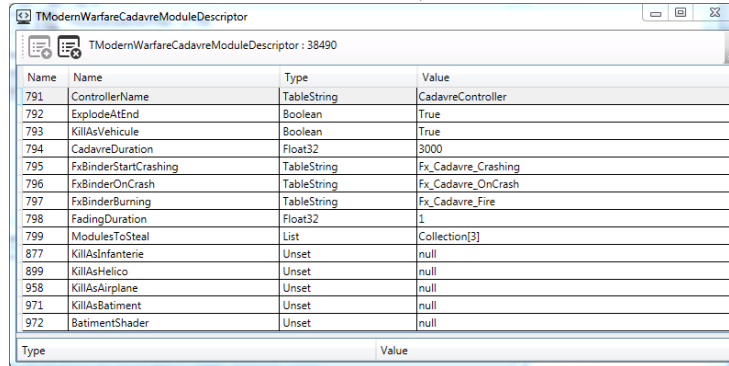
Always set to 2.5

14 TIAStratModule

This module contains a few hard-coded numerical references which are used for internal AI orchestration.

15 TModernWarfareCadavreModuleDescriptor

This module is buried in the logic section of the CadavreController top-level module object. It contains a number of interesting variables, all controlling what happens when a unit dies: how long it sticks around afterwards, how quickly it fades, whether or not it exploded on death (if it does, it uses the DeathExplosionAmmo top-level object), and a bunch of flags:



The screenshot shows a window titled "TModernWarfareCadavreModuleDescriptor" with a subtitle "TModernWarfareCadavreModuleDescriptor : 38490". It contains a table with the following data:

Name	Name	Type	Value
791	ControllerName	TableString	CadavreController
792	ExplodeAtEnd	Boolean	True
793	KillAsVehicule	Boolean	True
794	CadavreDuration	Float32	3000
795	FxBinderStartCrashing	TableString	Fx_Cadavre_Crashing
796	FxBinderOnCrash	TableString	Fx_Cadavre_OnCrash
797	FxBinderBurning	TableString	Fx_Cadavre_Fire
798	FadingDuration	Float32	1
799	ModulesToSteal	List	Collection[3]
877	KillAsInfanterie	Unset	null
899	KillAsHelico	Unset	null
958	KillAsAirplane	Unset	null
971	KillAsBatiment	Unset	null
972	BatimentShader	Unset	null

We will omit examining it in detail.

16 TTurretSkeletonModuleDescriptor

Contains a pointer to the unit's primary mesh.

17 TMissileCarriageModuleDescriptor

Contains pointers to art resources constituting missile carriages, which control all of the art involved in units having and then not having certain visible weapon elements on-body. So for example, a F-15D Eagle with its bombs would look different from an F-15D Eagle without its bombs, obviously. That's controlled here.

We will omit further details, except to say that the difficulties of working with this module make modding planes and helicopters (some of which have this module, some of which haven't) challenging.

18 TCommandManagerModuleDescriptor

When this module is set up properly, the unit is treated as a command unit in-game.

19 TGhostManagerModuleDescriptor

This module controls things related to what happens when a unit is spotted, but not identified: what you see when something is shaded in black.

20 TScannerModuleDescriptor

This module is only present if the unit has special vision privileges. It contains a TModernWarfareVisibilityRollRule as a subtable, which in turn contains TModernWarfareVisibilityRollRuleDescriptor and TModernWarfareDistanceMultiplierRollRuleDescriptor dependencies.

A relatively small number of tables of the latter two types ultimately control large swatches of unit vision ranges, so you might find that e.g. tweaking one Very Good recon unit's table's vision range up will turn up vision for ALL Very Good recon units, and other similar shenanigans.

Types of vision controlled by this module are Air, Sea, and Land.

This module only contains multipliers. Base vision values are set by the module below.

21 TScannerConfigurationDescriptor

This module, if present, controls the base vision values for units granted above-normal vision. All of the base values come from here, but interact with values in TScannerModuleDescriptor above. If one of these two modules is present, both are, as they are dependent on one another.

21.1 ControllerName

Always ScannerConfigurationController. Immutable.

21.2 UnitType

A reference to the domain of the unit in question itself. UnitType 1 refers to ground, UnitType 4 refers to planes, and UnitType 6 refers to ships.

21.3 DetectionTBA

Maximum range at which you can see an unidentified helicopter.

21.4 PorteeVision

Maximum range at which you can see an unidentified ground unit.

21.5 OpticalStrength

Optical strength against ground units, used to determine whether a unit can see enemy units in cover.

21.6 OpticalStrengthAltitude

Optical strength against aircraft (including helicopters).

21.7 SpecializedOpticalStrengths

This is a Collection of float pair mapping; each of the maps binds a UnitType and then the range at which they can be detected. UnitType 4 refers to planes, while UnitType 6 refers to ships, and UnitType 1 refers to Land. The value Optican strengths against

21.8 SpecializedDetections

Maximum range at which you can see an unidentified unit. This is a Collection of float pair mapping; each of the maps binds a UnitType and then the range at which they can be detected. UnitType 4 refers to planes, while UnitType 6 refers to ships, and UnitType 1 refers to Land.

21.9 PorteVisionTBA

Maximum spotting range for helicopters. This attribute is only populated for helicopters, in which case it controls helicopter-to-helicopter sighting. It is otherwise null.

21.10 OpticanStrengthAntiradar

Maximum anti-radar spotting range. Only populated for anti-radar units; null otherwise.

21.11 _ShortDatabaseName

Same as the top-level parameter, but this seems to always be null.

22 TFuelModuleDescriptor

22.1 ControllerName

Uninteresting.

22.2 FuelCapacity

The number of units of fuel that this unit can stockpile.

22.3 FuelMoveDuration

Autonomy in seconds. Appears to be the same as the T.O.T statistic displayed in the Armory.

23 TMovementHandlerLandVehicleDescriptor

This module will only be present if the vehicle in question is, indeed, a land unit.

23.1 ControllerName

Uninteresting.

23.2 Maxspeed

The unit's max speed, in engine distance units (see the next major section for unit explanations).

23.3 UnitMovingType

The same as that set in TUnitMouvementDescriptor.

23.4 SpeedBonusOnRoad

A multiplier on the unit's base speed. Always set to whatever it needs to be set to to give the unit 110kph tracked speed or 150kph wheeled speed, meaning that these are constants but must be set on the local level!

23.5 TempsDemiTour

The amount of time, in seconds, it takes for a unit to make a half-turn (the translation is literal). If you tell a unit to move in a direction directly opposite to its current orientation, it will do a reverse-accelerate V-turn to the side, and this variable controls how long this takes.

Presumably this value also controls how long it takes to make turns that are less than 360 degrees, but still significant. The cutoff value at which a unit starts to make a reverse turn, instead of just turning in place, is unknown.

23.6 MaxAcceleration

The acceleration the unit applies when it is speeding up.

23.7 MaxDeceleration

How hard to breaks hit. Tends to be twice MaxAcceleration for tracked and wheeled land units.

23.8 VehicleSubType

Unknown, often null.

23.9 CriticalEffectModule

A module pointer whose contents is a TCriticalEffectModuleDescriptor describing what happens when this unit is hit with a critical. Unfortunately it ultimately points to hashes, not to any descriptions of the effects of the criticals themselves...

23.10 TerrainsToIgnoreMask

This is set to certain magic values for ship units which are allowed and not allowed to enter different water depths: 16 if it can enter all waters, 24 if it's a coaster, and 26 if it's a bluewater ship.

Otherwise null.

24 TMouvementHandlerHelicopterDescriptor

24.1 ControllerName

Uninteresting.

24.2 Maxspeed

The unit's max speed, in engine distance units (the meaning of which is covered elsewhere).

24.3 MaxAcceleration

The acceleration the unit applies when it is speeding up.

24.4 MaxDeceleration

How hard the breaks hit.

24.5 UnitMovingType

The same as that set in TUnitMouvementDescriptor. Here this will always be 6, for air.

24.6 CyclicManoeuvrability

Controls movement.

24.7 GFactorLimit

Controls movement.

24.8 LateralSpeed

Controls movement.

24.9 Mass

Controls movement.

24.10 MaxInclination

Controls movement.

24.11 RotorArea

Controls movement.

24.12 TorqueManoeuvrability

Controls movement.

24.13 UpwardsSpeed

Controls movement.

24.14 CriticalEffectsModule

A module pointer whose contents is a TCriticalEffectModuleDescriptor describing what happens when this unit is hit with a critical. Unfortunately it ultimately points to hashes, not to any descriptions of the effects of the criticals themselves...

24.15 TempsDemiTour

The amount of time, in seconds, it takes for a unit to make a half-turn (the translation is literal). For helicopters this requires a dive to the left. It's uncertain whether or not this variable is still used for anything.

25 TMouvementHandlerAirplaneDescriptor

25.1 Maxspeed

Maximum speed this unit can go at. Since airplanes fly at a constant speed, this is also the only speed.

25.2 UnitMovingType

Always 6, for air (other values are 4 for sea and 1 for land).

25.3 FlyingAltitude

The height the plane prefers to fly at. Notably, all planes spawn at the same height then ascend or descend to their preferred height.

25.4 MinimalAltitude

Below which the plane will not dip. Will cause it to break off from certain attacks at certain heights.

25.5 PhysicsConfiguration

A reference to a `TAirplanePhysicsConfiguration` module, which controls the plane's various movement parameters.

25.6 CriticalEffectsModule

A module pointer whose contents is a `TCriticalEffectModuleDescriptor` describing what happens when this units is hit with a critical. Unfortunately it ultimately points to hashes, not to any descriptions of the effects of the criticals themselves...

25.7 GunMuzzleSpeed

Unknown. Always 300000.

25.8 LandingGearOutPhysicalPropertyName

Unknown. Always "InShowRoom".

25.9 LandingGearSubDescriptionName

Unknown. Always "Landing Gear".

25.10 FrontLandingGearMeshNodeName

Unknown. Always "Train_Avant".

25.11 BackLandingGearMeshNodeName

Unknown. Always "Train_Arriere".

26 THaloModuleDescriptor

Does a bunch of things which control the circle that gets drawn when a unit is selected.

27 TGroupeCombatModuleDescriptor

This module, which is only attached to infantry units. This table has a sub-module, TUniteBehaviorDescriptor, which has a large number of variables controlling the unique aspects of infantry combat stats in the game.

Amongst the various TUniteBehaviorDescriptor tables, all of the variables contained therein are the same (making them globals, essentially) except for two: NbSoldatInGroupeCombat, which controls the number of men to the squad, and AnimationFastAccessor, whose purpose is unknown.

28 TTransportableModuleDescriptor

This module, which is only attached to infantry units, controls what transports are available to an infantry unit.

28.1 ControllerName

Not interesting.

28.2 Categories

Always a list with two elements: "infanterie" and "barge".

28.3 SuppressDamageRatioIfTransporterKilled

This float controls what percentage of the unit's total suppression ceiling (which is 800 for all units) the unit will take in suppression damage if it is on a transport, the transport is destroyed, and the unit survives.

As such it contains direct references to other transporter units' TUniteAuSolDescriptor instances.

28.4 TransportListAvailableForSpawn

This contains a Collection of references to one or more transporter units' TUniteAuSolDescriptor instances. Those units become available to this unit as an on-card transport.

Note that, due to the previously described mechanics of the UpgradeRequired top-level variable, a reference in this table to a transporter unit with an upgrade path will cause all of those other units to also become available. So the exactly

list of transporters for this unit may be longer than it appears to be from this list alone!

28.5 TransportedTexture

Unknown. Transported units have no texture, no?

29 TModuleModernWarfareSupplyDescriptor

29.1 ControllerName

Uninteresting.

29.2 SupplyCapacity

Units of supply (liters) this unit carries.

29.3 DeploymentDuration

How long after a unit has stopped before it begins to provide supply, in seconds. Always 0.2 seconds currently, except for FOBs, which have this value set to null.

29.4 WithdrawalDuration

How long after a unit is ordered to start moving before it stops providing supplies and starts moving. Currently always 0.2 seconds currently, except for FOBs, which have this value set to null.

29.5 SupplyPriority

It is possible for supply units to supply other supply units; for example a common tactic is to shorten the supply chain trip length by buying a Mi-26, dropping it halfway between the FOB and the battlefield, and running ammo trucks between the helo and the front line. The lower this number the lower in this stack this unit is, and the greater the number of other supply units this supply unit could itself draw from. Generally 1-to-1 with total supply onboard: the bigger the capacity, the lower the SupplyPriority.

29.6 SupplyDescriptor

Links to another module, an instance of TModernWarfareSupplyDescriptor, which contains the variables used for calculating supply cost and supply per second. There are only two instances of TModernWarfareSupplyDescriptor, one for FOBs and one for all other supply units. The latter includes the following essentially-globals:

45	FuelSupplyCostBySecond	Float32	5
46	FuelSupplyBySecond	Float32	30
47	HealthSupplyCostBySecond	Float32	5
48	HealthSupplyBySecond	Float32	0.1
49	AmmunitionSupplyBySecond	UInt32	25
50	AmmunitionSupplyCostBySecond	UInt32	25
51	SupplySupplyBySecond	Float32	50
52	SupplySupplyCostBySecond	Float32	50
53	DefaultSupplyRange	Float32	52000
54	SupplyPointCostInComma	Unset	null

The former:

45	FuelSupplyCostBySecond	Float32	0.00001
46	FuelSupplyBySecond	Float32	80
47	HealthSupplyCostBySecond	Float32	5
48	HealthSupplyBySecond	Float32	0.03
49	AmmunitionSupplyBySecond	UInt32	35
50	AmmunitionSupplyCostBySecond	UInt32	35
51	SupplySupplyBySecond	Unset	null
52	SupplySupplyCostBySecond	Unset	null
53	DefaultSupplyRange	Unset	null
54	SupplyPointCostInComma	Float32	0.01

30 TWeaponManagerDescriptor

WRITE THIS UP.

31 TModernWarfareDamageModuleDescriptor

WRITE THIS UP.

32 TAppearanceModelDescriptor

Controller for things related to the unit's appearance.

33 TVisibilityModuleDescriptor

33.1 ControllerName

Uninteresting.

33.2 UnitStealthBonus

A multiplier applied to the unit's visibility, and the only interesting contents of this module. The multipliers are:

Value	Stealth	Examples	Counts
0.1	Bad	Hatsuyiki	1
0.3	Bad	Luda	1
0.4	Bad	Oliver Hazard Perry	1
0.5	Bad	Jianghu III	3
0.6	Bad	Nanushka III	4
0.7	Bad	Donghae	5
0.8	Bad	Chamsuri	4
0.9	Bad	Shmel	2
1	Poor	AMX-30B	1349
1.2	Medium	Komar	1
1.25	Poor (!)	MiG-29S	1
1.5	Medium	VLB Minstral Recon	139
1.75	Good	PAH-2 Tiger	4
2	Good	US Marines	200
2.5	Very Good	Spetsnaz GRU	91
3	Exceptional	Spetsnaz VMF, Nighthawk	7

33.3 `_ShortDatabaseName`

The same as that set in the top-level object, but seemingly always null.

34 `TCompanyUnitDescriptor`

Companies are what the game terms sets of units ranging in size from 1 (lone) to 4 (max size; helicopters can only be grouped in 2s and planes and boats in 1s). This module contains a bunch of elements which define the logic of giving orders to a company of units, as well as the number limiting how many of the unit stack.