

# Wargame: Red Dragon Internal Mechanics Manual

Resident Mario

November 10, 2016

# Contents

<b>1</b>	<b>Introduction</b>	<b>9</b>
<b>2</b>	<b>Getting Started</b>	<b>10</b>
<b>3</b>	<b>TUniteAuSolDescriptor</b>	<b>12</b>
3.1	DescriptorId . . . . .	12
3.2	_ShortDatabaseName . . . . .	12
3.3	_ClassNameForDebug . . . . .	12
3.4	StickToGround . . . . .	12
3.5	ManageUnitOrientation . . . . .	12
3.6	HitRollSizeModifier . . . . .	12
3.7	DeathExplosionAmmo . . . . .	13
3.8	IconeType . . . . .	13
3.9	PositionInMenu . . . . .	13
3.10	NameInMenuToken . . . . .	13
3.11	AliasName . . . . .	13
3.12	Category . . . . .	13
3.13	AcknowUnitType . . . . .	14
3.14	TypeForAcknow . . . . .	14
3.15	Nationalite . . . . .	14
3.16	MotherCountry . . . . .	14
3.17	ProductionYear . . . . .	15
3.18	MaxPacks . . . . .	15
3.19	Factory . . . . .	15
3.20	ProductionPrice . . . . .	16
3.21	MaxDeployableAmount . . . . .	16
3.22	ShowInMenu . . . . .	16
3.23	ProductionTime . . . . .	16
3.24	CoutEtiole . . . . .	16
3.25	TextureForInterface . . . . .	17
3.26	TextureTransportForInterface . . . . .	17
3.27	TextureMotherCountryForInterface . . . . .	17
3.28	UnitTypeTokens . . . . .	17
3.29	UnitMovingType . . . . .	17
3.30	VitesseCombat . . . . .	18
3.31	UpgradeRequired . . . . .	18
3.32	IsPrototype . . . . .	18
3.33	Key . . . . .	18
3.34	HitRollECMModifier . . . . .	19
3.35	Modules . . . . .	19

<b>4</b>	<b>TTypeUnitModuleDescriptor</b>	<b>19</b>
4.1	ControllerName . . . . .	19
4.2	TypeUnitValue . . . . .	19
4.3	TypeUnitHintToken . . . . .	19
4.4	NameInMenuToken . . . . .	20
4.5	GenerateName . . . . .	20
4.6	Filters . . . . .	20
4.7	MotherCountry . . . . .	21
4.8	UnitInfoJaugeType . . . . .	21
4.9	Training . . . . .	21
4.10	CIWS . . . . .	21
4.11	Sailing . . . . .	21
<b>5</b>	<b>TDebugModuleDescriptor</b>	<b>22</b>
<b>6</b>	<b>TStateEngineModuleDescriptor</b>	<b>22</b>
<b>7</b>	<b>TFlagsModuleDescriptor</b>	<b>22</b>
<b>8</b>	<b>TCriticalEffectModuleDescriptor</b>	<b>22</b>
<b>9</b>	<b>TTargetCoordinatorModuleDescriptor</b>	<b>22</b>
<b>10</b>	<b>TPositionModuleDescriptor</b>	<b>22</b>
10.1	ControllerName . . . . .	22
10.2	InGeoDb . . . . .	22
10.3	ClampInWorld . . . . .	22
10.4	GfxDescriptorPorteur . . . . .	22
10.5	Radius . . . . .	23
10.6	AddToHexagonMap . . . . .	23
10.7	PorteurMustBeVisible . . . . .	23
10.8	RelativeScanningPosition . . . . .	23
10.9	CameraFollower . . . . .	23
10.10	MustAllowZoneIndice . . . . .	23
10.11	LowAltitudeFlyingAltitude . . . . .	23
10.12	NearGroundFlyingAltitude . . . . .	23
10.13	ClampOutMap . . . . .	23
10.14	HasNearlyNullBBox . . . . .	23
10.15	ShortDatabaseName . . . . .	23
<b>11</b>	<b>TInflammableModuleDescriptor</b>	<b>24</b>
<b>12</b>	<b>TLinkTeamModuleDescriptor</b>	<b>24</b>

<b>13 TModernWarfareExperienceManagerDescriptor</b>	<b>24</b>
13.1 ControllerName . . . . .	24
13.2 CanWinExperience . . . . .	24
13.3 ExperienceGainBySecond . . . . .	24
13.4 KillExperienceBonus . . . . .	24
<b>14 TIAStratModule</b>	<b>24</b>
<b>15 TModernWarfareCadavreModuleDescriptor</b>	<b>24</b>
<b>16 TTurretSkeletonModuleDescriptor</b>	<b>25</b>
<b>17 TMissileCarriageModuleDescriptor</b>	<b>25</b>
<b>18 TCommandManagerModuleDescriptor</b>	<b>25</b>
<b>19 TGhostManagerModuleDescriptor</b>	<b>25</b>
<b>20 TScannerModuleDescriptor</b>	<b>25</b>
<b>21 TScannerConfigurationDescriptor</b>	<b>26</b>
21.1 ControllerName . . . . .	26
21.2 UnitType . . . . .	26
21.3 DetectionTBA . . . . .	26
21.4 PorteeVision . . . . .	26
21.5 OpticalStrength . . . . .	26
21.6 OpticalStrengthAltitude . . . . .	26
21.7 SpecializedOpticalStrengths . . . . .	27
21.8 SpecializedDetections . . . . .	27
21.9 PorteVisionTBA . . . . .	27
21.10OpticanStrengthAntiradar . . . . .	27
21.11_ShortDatabaseName . . . . .	27
<b>22 TFuelModuleDescriptor</b>	<b>27</b>
22.1 ControllerName . . . . .	27
22.2 FuelCapacity . . . . .	27
22.3 FuelMoveDuration . . . . .	28
<b>23 TMouvementHandlerLandVehicleDescriptor</b>	<b>28</b>
23.1 ControllerName . . . . .	28
23.2 Maxspeed . . . . .	28
23.3 UnitMovingType . . . . .	28
23.4 SpeedBonusOnRoad . . . . .	28
23.5 TempsDemiTour . . . . .	28
23.6 MaxAcceleration . . . . .	29
23.7 MaxDeceleration . . . . .	29
23.8 VehicleSubType . . . . .	29

23.9 CriticalEffectModule . . . . .	29
23.10TerrainsToIgnoreMask . . . . .	29
<b>24 TMouvementHandlerHelicopterDescriptor</b>	<b>30</b>
24.1 ControllerName . . . . .	30
24.2 Maxspeed . . . . .	30
24.3 MaxAcceleration . . . . .	30
24.4 MaxDeceleration . . . . .	30
24.5 UnitMovingType . . . . .	30
24.6 CyclicManoeuvrability . . . . .	30
24.7 GFactorLimit . . . . .	31
24.8 LateralSpeed . . . . .	31
24.9 Mass . . . . .	31
24.10MaxInclination . . . . .	31
24.11RotorArea . . . . .	31
24.12TorqueManoeuvrability . . . . .	31
24.13UpwardsSpeed . . . . .	31
24.14CriticalEffectsModule . . . . .	32
24.15TempsDemiTour . . . . .	32
<b>25 TMouvementHandlerAirplaneDescriptor</b>	<b>32</b>
25.1 Maxspeed . . . . .	32
25.2 UnitMovingType . . . . .	32
25.3 FlyingAltitude . . . . .	32
25.4 MinimalAltitude . . . . .	32
25.5 PhysicsConfiguration . . . . .	33
25.6 CriticalEffectsModule . . . . .	33
25.7 GunMuzzleSpeed . . . . .	33
25.8 LandingGearOutPhysicalPropertyName . . . . .	33
25.9 LandingGearSubDescriptionName . . . . .	33
25.10FrontLandingGearMeshNodeName . . . . .	33
25.11BackLandingGearMeshNodeName . . . . .	33
<b>26 THaloModuleDescriptor</b>	<b>33</b>
<b>27 TGroupeCombatModuleDescriptor</b>	<b>34</b>
<b>28 TTransportableModuleDescriptor</b>	<b>34</b>
28.1 ControllerName . . . . .	34
28.2 Categories . . . . .	34
28.3 SuppressDamageRatioIfTransporterKilled . . . . .	34
28.4 TransportListAvailableForSpawn . . . . .	34
28.5 TransportedTexture . . . . .	34

<b>29 TModuleModernWarfareSupplyDescriptor</b>	<b>35</b>
29.1 ControllerName . . . . .	35
29.2 SupplyCapacity . . . . .	35
29.3 DeploymentDuration . . . . .	35
29.4 WithdrawalDuration . . . . .	35
29.5 SupplyPriority . . . . .	35
29.6 SupplyDescriptor . . . . .	35
<b>30 TWeaponManagerModuleDescriptor</b>	<b>36</b>
30.1 ControllerName . . . . .	36
30.2 _ShortDatabaseName . . . . .	36
30.3 HasMainSalvo . . . . .	37
30.4 SalvoIsMainSalvo . . . . .	37
30.5 Salves . . . . .	37
30.6 TurretDescriptorList . . . . .	38
<b>31 TurretDescriptorList : TTurret&lt;*&gt;Descriptor</b>	<b>38</b>
31.1 NbFx . . . . .	38
31.2 Tag . . . . .	38
31.3 TagIndex . . . . .	38
31.4 VitesseRotation . . . . .	38
31.5 AngleRotationBase . . . . .	38
31.6 AngleRotationMax . . . . .	38
31.7 AngleRotationMaxPitch . . . . .	39
31.8 AngleRotationMinPitch . . . . .	39
31.9 AngleRotationBasePitch . . . . .	39
31.10UnitIdleManagerDescriptor . . . . .	39
31.11TargetPositionPhysicalPropertyName . . . . .	39
31.12FlyingTimeAndHitPhysicalPropertyName . . . . .	39
31.13OwnerTurnHisChassisVerticallyToAttack . . . . .	39
31.14MountedWeaponDescriptorList . . . . .	39
<b>32 TTurret&lt;*&gt;Descriptor : TMountedWeaponDescriptor</b>	<b>40</b>
32.1 SalvoStockIndex . . . . .	40
32.2 SalvoStock_ForInterface . . . . .	40
32.3 EffectTag . . . . .	40
32.4 TirContinu . . . . .	40
32.5 TirEnMouvement . . . . .	40
32.6 AnimateOnlyOneSoldier . . . . .	41
32.7 Ammunition . . . . .	41
<b>33 Ammunition : TAmmunition</b>	<b>41</b>
33.1 DescriptorId . . . . .	41
33.2 Name . . . . .	41
33.3 TypeName . . . . .	41
33.4 TypeArme . . . . .	41

33.5	Arme . . . . .	42
33.6	RadiusSplashPhysicalDamages . . . . .	42
33.7	PhysicalDamages . . . . .	42
33.8	ProjectileType . . . . .	42
33.9	Puissance . . . . .	42
33.10	TempsEntreDeuxTirs . . . . .	43
33.11	TempsEntreDuexSalves . . . . .	43
33.12	TempsEntreDuexFx . . . . .	43
33.13	NbrProjectilesSimultanes . . . . .	44
33.14	NbTirParSalves . . . . .	44
33.15	AffichageMunitionParSalve . . . . .	44
33.16	PorteeMaximale . . . . .	44
33.17	PorteeMinimale . . . . .	44
33.18	PorteeMinimaleBateaux . . . . .	44
33.19	PorteeMaximaleBateaux . . . . .	44
33.20	PorteeMaximaleTBA . . . . .	44
33.21	PorteeMinimaleTBA . . . . .	44
33.22	PorteeMaximaleHA . . . . .	44
33.23	PorteeMinimaleHA . . . . .	44
33.24	PorteeMaximaleProjectile . . . . .	45
33.25	PorteeMinimaleProjectile . . . . .	45
33.26	AngleDispersion . . . . .	45
33.27	SuppressDamages . . . . .	45
33.28	RadiusSplashSuppressDamages . . . . .	45
33.29	RayonPinned . . . . .	45
33.30	TirIndirect . . . . .	45
33.31	FX_tir_sans_physic . . . . .	45
33.32	FX_vitesse_de_depart . . . . .	45
33.33	FX_frottement . . . . .	45
33.34	FX_tir_trendu . . . . .	46
33.35	Level . . . . .	46
33.36	FireDescriptor . . . . .	46
33.37	FireTriggeringProbability . . . . .	46
33.38	Caliber . . . . .	46
33.39	WeaponCursorType . . . . .	46
33.40	NoiseDisumlationMalus . . . . .	46
33.41	TempsDeVisee . . . . .	46
33.42	InterfaceWeaponTexture . . . . .	46
33.43	AffichageMenu . . . . .	46
33.44	SupplyCost . . . . .	47
33.45	SmokeDescriptor . . . . .	47
33.46	MissileTimeBetweenCorrections . . . . .	47
33.47	Guidance . . . . .	47
33.48	EfficaciteSelonPortee . . . . .	47
33.49	AffecteParNombre . . . . .	47
33.50	NeedModelChange . . . . .	47

33.51	IsFireAndForget	47
33.52	IgnoreInflammabilityConditions	48
33.53	InterdireTirReflexe	48
33.54	TirReflexe	48
33.55	DispersionAtMaxRange	48
33.56	DispersionAtMinRange	48
33.57	CorrectedShotDispersionMultiplier	48
33.58	IsSubAmmunition	48
33.59	RandomDispersion	48
33.60	TempsAnimation	48
33.61	HitRollRule	48
33.62	MissileDescriptor	49
<b>34</b>	<b>TModernWarfareDamageModuleDescriptor</b>	<b>49</b>
34.1	ControllerName	49
34.2	CommonDamageDescriptor	49
34.3	MaxDamages	49
34.4	MaxHPForHUD	49
34.5	Experience	49
34.6	AutoOrientation	49
34.7	Transporter	49
34.8	IsTargetableAsBoat	50
<b>35</b>	<b>TAppearanceModelDescriptor</b>	<b>50</b>
<b>36</b>	<b>CommonDamageDescriptor : TModernWarfareCommonDamageDescriptor</b>	<b>50</b>
36.1	PaliersSuppressDamages	50
36.2	PaliersPhysicalDamages	50
36.3	SuppressDamagesRegenRatio	50
36.4	SuppressDamagesRegenRatioOutOfRange	50
36.5	StunDamagesRegen	50
36.6	StunDamagesToGetStunned	51
36.7	SuppressDamagesEffects	51
36.8	PhysicalDamagesEffects	51
36.9	MaxSuppressionDamages	51
36.10	PhysicalDamagesEffects	51
36.11	TestMoralRollRule	51
36.12	TBlindageProperties	51
<b>37</b>	<b>TVisibilityModuleDescriptor</b>	<b>52</b>
37.1	ControllerName	52
37.2	UnitStealthBonus	52
37.3	_ShortDatabaseName	52
<b>38</b>	<b>TCompanyUnitDescriptor</b>	<b>52</b>



# 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):

Name	Name	Instances	Instance
95	TModuleSelectorFilter	1858	14672
101	TCompanyUnitModuleDescriptor	1837	14673
81	TUnitEuiSolDescriptor	1818	14674
194	TWeaponModuleDescriptor	1815	14675
89	TAmmunition	1756	14676
103	TWeaponManagerModuleDescriptor	1661	14677
137	TTurnetTwoAxisDescriptor	1587	14678
207	TMissileCarriageSubDepictionMis	1568	14679
325	TWargameUnitDeck	936	14680
219	TResourceAnimationMapHolder	915	14681
148	TTurnetUnitDescriptor	668	14682
165	TCompositeHappening	601	14683
145	TTurnetInfanterieDescriptor	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	TMissileCarriageSubDepictionGei	319	14692
209	TMultipleResourceAnimationMap	318	14693
185	TUnit_Odc_AnimationFastAccess	317	14694
57	TSequencingActionHappening	317	14695
229	TGfxDescriptorAnimation	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	TGdcFastAccessKey	254	14703
107	TFuelModuleDescriptor	246	14704
87	TTransportableModuleDescriptor	242	14705
30	TAcknowledgeDescriptor	239	14706
153	TAirplanePhysicsConfiguration	234	14707
121	TMovementHandlerAirplaneDes	229	14708

Name	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	MapPacks	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	CoutToile	Int32	2
462	TextureForInterface	ObjectReference	78 : 16534 (True) - TUJResourceTexture
463	TextureForTransportForInterface	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 that appears, in most cases, to be a copy of NameInMenuToken. This variable does nothing in-game, however, is not editable, and is often (in 485 cases, as of writing) simply "null".

### 3.12 Category

Another categorization variable, filled in with a number between 1 and 6. What unit gets placed in what category does not appear to follow any logic, and this variable is safe to ignore.

### 3.13 AcknowUnitType

This variable categorizes units by type. The scattershot nature of the categorizations seems to imply that at one point the categorizations were of units of specific types, but balancing patches which moved these units around categories seem not to have included updates to this variable, introducing inaccuracies.

Given that fact, it is doubtful that this variable has effect on gameplay. See the "Factory" variable for a cleaner categorization variable.

Value	Type
15	Israeli and Dutch (Paid DLC) Air Superiority Fighters.
16	Israeli and Dutch (Paid DLC) Multirole Planes.
17	Israeli and Dutch (Paid DLC) Ground-Attack Planes.
18	Israeli and Dutch (Paid DLC) SEAD Planes.
22	Tanks, and a few vehicles.
23	Missile platforms, mortars, most vehicles.
24	Most recon units.
25	Supply vehicles.
26	Command units.
27	Most planes and non-recon helicopters.
28	Flamethrower sappers.
29	Transport units, including commands based on them.
30	Missile-based anti-air, including anti-air infantry.
31	Gun-based anti-air.
32	Heavy artillery.
33	Most infantry.
34	Certain special forces infantry.
35	AsHMs, napalm launchers, and cluster artillery.
36	ATGM-carrying infantry and, weirdly, Mistral.
38	Ships, including transport ships.

### 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 that displays on the unit card. Whether or not the unit is actually available in the unit-year categories for deck building (Cat A, Cat B, Cat C) is controlled by an element of the Filter field in the UnitType module.

The record for the oldest unit in the game goes to M-24, a ten-point Norwegian tank that is nothing less and nothing more than the same mainstay tank of World War II provenance. It has a build date of 1943.

Other ancient units are the Czech OT-810 halftrack (1945), Danish M6 Mosegris truck (1947), North Korean Su-76M tank destroyer (1948), and Soviet BTR-152E ZPTU-2 machine-gun truck (1950).

The newest units date to 1996, and include the K9 Thunder, AH-64 Escort, JAS-39 Gripen, AMOS, and Su-27PU.

### 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 placed 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. 248 units in the game are considered prototypes, as of writing, out of 1818 total, a little less than 14 percent of the total.

### 3.33 `Key`

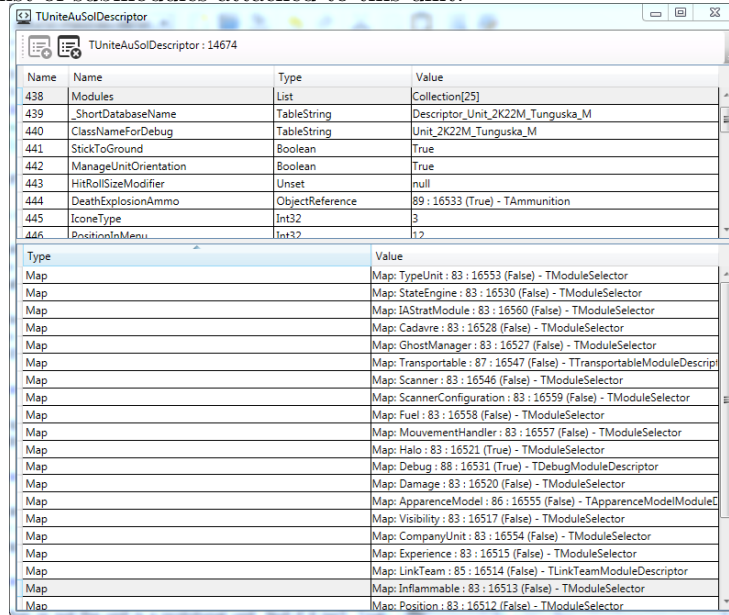
Unknown. Almost always null.

### 3.34 HitRolleECMModifier

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	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 : 16593 (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

If this unit is an infantry unit, this is set to a hash pointing to its training level (otherwise null):

Kind	Hash	Count
Elite	8F37594F19619C07	45
Shock	5593495D19619C07	66
Regular	D6173D5C19619C07	164
Militia	DE644D5719619C07	15

## 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	Count
Exceptional	4F233E0000000000	1
Very Good	4E96452000000000	3
Good	4E96450000000000	4
Medium	D672711906000000	6
Bad	CEC2000000000000	15
None	null	1786

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

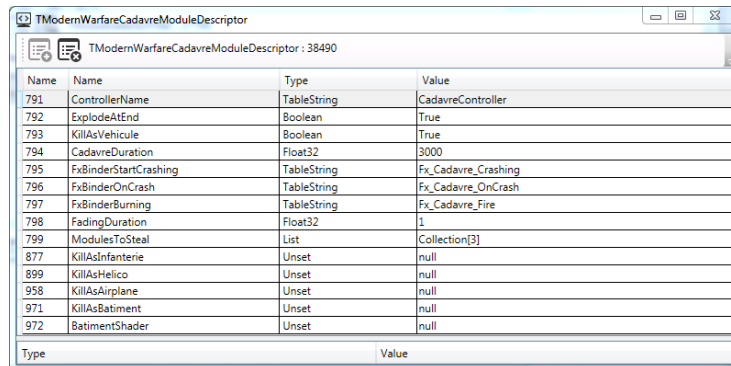
## **14 TIAStratModule**

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 TMod-

ernWarfareVisibilityRollRuleDescriptor 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 carries. While actual autonomy is a function of FuelMoveDuration, this variable controls fuel efficiency with respect to that, and, therefore, how expensive it is to refuel this unit. This variable ranges between 45 (the L3304, a Sewidish recon jeep) and 6500 liters (Su-24, MiG-31, MiG-25, F-15, F-14, and F-111 series planes).

### **22.3 FuelMoveDuration**

Autonomy in seconds. Appears to be the same as the T.O.T statistic displayed in the Armory. Ranges between 60 (MiG-17, F-5, and other cheap planes) and 2000 (CH-46 Phrog, CH-47 Chinook, and MH-53 Pave Low variants).

## **23 TMouvementHandlerLandVehicleDescriptor**

This module is present in the TMouvementDescriptor slot if and only if the unit is a land unit, or, interestingly enough, a ship.

### **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! If this unit is a ship, this value is instead set to null.

### **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 land 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. Ships will execute their turn completely in place (unrealistically but necessarily, given how the game works).

Presumably this value also controls how long it takes to make turns that are less than 180 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.

Value	Units	Count
0.1	Infantry	296
2.0	Jeeps	236
3.0	Light vehicles	404
4.0	Trucks, heavy vehicles, light tanks	167
5.0	Most tanks	366
6.0	Artillery, MLRS, Challenger 1 series tanks	60
7.0	Some MLRS, some tanks (Centurions, Chieftains)	44
10.0	Light ships	8
15.0	Heavy ships	5
20.0	Fleet ships (Sovremenniy, Udaloy II, Kongo)	3

### 23.6 MaxAcceleration

The acceleration the unit applies when it is speeding up.

Value	Units	Count
4030	Infantry	296
2080	Jeeps	100
1950	Light vehicles	131
1560	T-64BM, T-72S Series, T-72M2 Moderna	7
1300	Most tanks and vehicles	443
1040	Centurion series tanks, PTZ-59, ZTZ-59-II	15

### 23.7 MaxDeceleration

How hard to breaks hit. Always 4030 for infantry and 2600 for everything else.

### 23.8 VehicleSubType

1 for lightly armored units, 2 for infantry, 3 for other lightly armored stuff, and 4 for boats. More often then not, however, this variable is set to null, and so may be safely ignored.

### 23.9 CriticalEffectModule

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...

### 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 TMovementHandlerHelicopterDescriptor

This module is present in the TMovementDescriptor slot if and only if the unit is a helicopter.

### 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.

Value	Examples	Count
2080	AH-1 Cobra, Z-9, Gazelle	93
1560	AH-64, Tigre, Mi-28	17
1040	Mi-8, UH-1H Huey, Mi-24, CH-47 Chinook	109

### 24.4 MaxDeceleration

How hard the breaks hit. The values are always the same as those above (e.g. helicopters always decelerate and accelerate at the same rate).

### 24.5 UnitMovingType

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

### 24.6 CyclicManoeuvrability

Controls movement in some way, though in what way is uncertain. Higher is better.

Value	Examples	Count
40	CH-47C Chinook, Mi-6, Mi-26	22
60	Mi-4, Mi-8, Mi-17	31
80	Mi-24, Ka-29TB, UH-1H Huey	46
90	PUMA 330, Z-9	11
100	Mi-2, AH-64, Ka-52	34
110	Z-9, Lynx, Alouette	22
130	Gazelle, AH-1 Cobra, Bo-105	53

## 24.7 GFactorLimit

Controls movement in some way, though in what way is uncertain.

Value	Examples	Count
1.8	CH-47C Chinook, Mi-6, Mi-26	53
2.2	UH-1H Huey, Bell 412	25
2.5	Mi-2, Mi-24, Lynx	70
3.0	AH-64, Ka-52, Tigre	15
3.5	Gazelle, AH-1 Cobra, Bo-105	56

## 24.8 LateralSpeed

Controls movement in some way, though in what way is uncertain. Is always set to 2600.

## 24.9 Mass

Controls movement in some way, though in what way is uncertain.

Values used in-game are 50000.0, 22000.0, 12000.0, 11000.0, 10000.0, 8500.0, 8000.0, 7000.0, 5300.0, 4300.0, 4100.0, 3700.0, 3000.0, 2500.0, or 1800.0.

## 24.10 MaxInclination

Controls movement in some way, though in what way is uncertain.

Value	Examples	Count
89	Mi-2, Mi-24, Lynx AH.7	154
80	PUMA, Z-9	11
75	PUMA 330B Command	1
60	Mi-8, CH-47 Chinook	53

## 24.11 RotorArea

Controls movement in some way, though in what way is uncertain. Ranges between 80 and 800.

## 24.12 TorqueManoeuvrability

Controls movement in some way, though in what way is uncertain. Ranges between 30 and 120.

## 24.13 UpwardsSpeed

Controls movement in some way, though in what way is uncertain. Ranges between 1300 and 3900.

## 24.14 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...

## 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. For helicopters this variable is always set to 2.

# 25 TMouvementHandlerAirplaneDescriptor

This module is present in the TMouvementDescriptor slot if and only if the unit is a plane.

## 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.

## 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.

Value	Examples	Count
78000	F-117 Nighthawk, B-5, MiG-31M, F-14 Tomcat	7
65000	EF-111A Raven, Eurofighter Typhoon	78
52000	A-5, A-4K Kahu, CF-18 Hornet	61
39000	F-111C, A-6A Intruder, Alpha Jet A	26
26000	A-10 Thunderbolt II, F-111G. CF-104	51

## 25.4 MinimalAltitude

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



Value	Examples	Count
15600	F-117 Nighthawk, B-5, MiG-31M, F-14 Tomcat	7
13000	EF-111A Raven, Eurofighter Typhoon	78
10400	A-5, A-4K Kahu, CF-18 Hornet	67
7800	F-111C, A-6A Intruder, Alpha Jet A	26
5200	A-10 Thunderbolt II, F-111G, CF-104	49
2600	J-35D Draken, J-35F Draken	2

## 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 TGroupCombatModuleDescriptor

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

Amongst the various TUnitBehaviorDescriptor 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' TUnitAuSolDescriptor instances.

### 28.4 TransportListAvailableForSpawn

This contains a Collection of references to one or more transporter units' TUnitAuSolDescriptor 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. Ranges between 15000 and 500.

### 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.

Ranges between 10 and 34.

### 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 TWeaponManagerModuleDescriptor

This is one of the two most interesting modules overall, and easily the most complicated one. This module controls everything that there is to know about a unit's weaponry, and to understand its logical structure is to understand how weapons in this game work.

Each `TWeaponManagerModuleDescriptor` contains a `Collection` known as a `TurretDescriptorList` whose items are individual `TTurretTwoAxisDescriptor`, `TTurretUnitDescriptor`, `TTurretInfanterieDescriptor`, or `TTurretBombardierDescriptor` modules.

These names pretty descriptively describe which particular module type corresponds with which weapon domain, and each one describes the weaponry and behavior of a turret of some kind on the unit. Each individual weapon, the things you actually see on the unit card, is in turn one of the `TMountedWeaponDescriptor` modules in the list of them stored in the `MountedWeaponDescriptorList` attribute of `TTurret<*>Descriptor`.

Each of these in turn contains a reference to a single `TAmmunition` object, which describes everything about the damage output of the weapon itself, as opposed to the way it's mounted.

And `TAmmunition` can itself contain references to another object! That object will be a `TUnitDescriptor` instance describing what the missile or round looks like and acts like while it's in flight.

This a very dense structure, and it takes some play and mucking about to make sense of it at all.

### 30.1 ControllerName

Not interesting.

### 30.2 \_ShortDatabaseName

Will be "WeaponDescriptor\_" followed by the unit's debug name. Not very interesting.

### 30.3 HasMainSalvo

You may notice that almost all air unit will automatically "Evac Winchester" when a certain condition or set of condition is met. For air-to-ground units, this will occur whenever the unit has dropped its payload; for air-to-air units, it will occur when it has run out of missiles. There are a few corner cases, like the A-10 Thunderbolt, whose cannon is considered an important enough arnament that it doesn't just evac when it runs out of Mavericks; but it will if it runs out of both missiles and gun rounds (somehow).

This flag indicates whether or not the unit has such a salvo. It is set to true if it does, and set to null if it doesn't. All non-plane units have this set to null.

### 30.4 SalvoIsMainSalvo

This is always a collection of four booleans. Note that it does nothing if HasMainSalvo is null, but is not itself null in that case. We therefore restrict our attention to the case of planes.

The first element controls whether or not the weapon tagged to the first salvo index is considered the main weapon. For planes, the first salvo index will always be the gun, so the boolean in this index will control whether or not the plane's gun is considered primary.

Again in the case of planes, the second element is never tagged to anything, so this is always False.

The third element is whatever is tagged to the second salvo index, and the fourth element is whatever is tagged to the third salvo index.

What do I mean by "tagged to the nth salvo index"? That's convered by the next variable.

### 30.5 Salves

Salves is a variable containing the number of salvos of a weapon available to a unit.

For example, if a unit has a salvo size (set by NbTirParSalve three levels of heirarchy below, in Tammunition) of 24 shots per salvo, and the corresponding entry in Salves gives is a value of 2, then the unit will have two salvos of 24 shots each. For an explanation of what a salvo is, refer to the Tammunition documentation.

Weapons are not listed in Salves in the order that they appear on the unit card. The order that they do appear in somewhat defines explanation. Salves always has 29 values in it, corresponding with a theoretical 29 weapons; in cases in which there is no weapon corresponding with that slot, the value is filled in as a -1.

Which weapon corresponds to which index in this list is controlled by SalvoStockIndex, two levels of heirarchy down in TMountedWeaponDescriptor.

## 30.6 TurretDescriptorList

This is the next level of heirarchy down. Keep reading the section immediately below to study what's found here.

## 31 TurretDescriptorList : TTurret<\*>Descriptor

As previously described, each TWeaponManagerModuleDescriptor contains a Collection known as a TurretDescriptorList whose items are individual TTurretTwoAxisDescriptor, TTurretUnitDescriptor, TTurretInfanterieDescriptor, or TTurretBombardierDescriptor modules. In this section we will describe the components of these modules.

### 31.1 NbFx

Something related to the physics engine.

### 31.2 Tag

The tag will a string of the form "tourelle1", "tourelle2", etcetera. This explicitly states the position of the turret in question in the list of turrets. Turrets are always tagged in the same order that they appear in thee TurretDescriptorList. This property will not appear for TTurretInfanterieDescriptor modules.

### 31.3 TagIndex

An even more literal intepretation of the above: will be one of 1, 2, 3, and so on. This property will not appear for TTurretInfanterieDescriptor modules.

### 31.4 VitesseRotation

Traverse speed of the turret, presumably in radians per second. This only appears if the turret in question is capable of traversal, so it is limited to TTurretTwoAxisDescriptor instances. Ranges between 1300 (I-Hawk family) and 36400 (the Pegasus).

### 31.5 AngleRotationBase

The angle that the turret has with the hull when idle, in radians. This property is unique to TTurretTwoAxisDescriptor instances.

### 31.6 AngleRotationMax

Is the maximum angle that the turret can traverse. If you were to set this value to 6.28, approximately 360 degrees, the turret traversal is 360 all around. This property will not appear for TTurretInfanterieDescriptor modules.

### **31.7 AngleRotationMaxPitch**

Maximum turret elevation. This property will not appear for TTurretIfanterieDescriptor modules.

### **31.8 AngleRotationMinPitch**

Minimum turret depression. This property will not appear for TTurretIfanterieDescriptor modules.

### **31.9 AngleRotationBasePitch**

Resting turret depression. This property is unique to TTurretTwoAxisDescriptor instances.

### **31.10 UnitIdleManagerDescriptor**

If this turret does something when the unit is idle (tank cannons for example will return their resting position), then this action is controlled by a module reference in this field. We will omit exploring that in detail. Only TTurretTwoAxisDescriptor instances have this attribute. This property is unique to TTurretTwoAxisDescriptor instances.

### **31.11 TargetPositionPhysicalPropertyName**

Unknown.

### **31.12 FlyingTimeAndHitPhysicalPropertyName**

Unknown.

### **31.13 OwnerTurnHisChassisVerticallyToAttack**

This variable controls whether or not the unit automatically turns towards its target vertically. Whether or not the unit turns its chassis horizontally is controlled elsewhere, by the AutoRotation parameter in the UnitMouvement module field. This field is null for all land units, which obviously can't turn vertically. It is set to true for fixed-axis weapons, like helicopter or plane unguided rocket pods for example.

This property will not appear for TTurretIfanterieDescriptor modules.

### **31.14 MountedWeaponDescriptorList**

This property contains a list of references to the TMountedWeaponDescriptor elements mounted on this turret. This is the next level of heirarchy down, and is considered in the next section.

## 32 TTurret<\*>Descriptor : TMountedWeaponDescriptor

Each individual weapon, the things you actually see on the unit card, is one of the TMountedWeaponDescriptor modules in the list of them stored in the MountedWeaponDescriptorList attribute of TTurret<\*>Descriptor.

Note that within the game files a unit can have more weapons than appears on its cards. This is true in part because weapons which deal both AP and HE or which can fire smoke have separate weapons for each of these "features", and in part because ships have far more weapons than is listed on their card. The recordholder in this regard is, suprisingly, the Najin, which packs 11 weapons onboard. Way more than the 3 you see on the card!

### 32.1 SalvoStockIndex

Which position in the Salves list two levels of heirarchy up, in the base TWeaponManagerModuleDescriptor, the current weapon takes its salvo count from. A small but important technical rub: if this value is set to null, then it is equivalent to the first value in the Salve list (the zeroth index).

### 32.2 SalvoStock\_ForInterface

This variable controls which position, amongst the weapons displayed on the unit card, the weapon in question occupies. Some weapons are defined twice, once for HE and once for AP, and they may have the same SalvoStock\_ForInterface value, in which case the two weapons will "stack" in the card display (resulting in a weapon with both HE and AP power, as with all tank cannons for example).

### 32.3 EffectTag

Unknown.

### 32.4 TirContinu

Unknown.

### 32.5 TirEnMouvement

Whether or not this weapon can be fired on the move. Null if no (grants the [STAT] tag), True is yes (in which case if additionally MissileDescriptor is populated and IsFireAndForget is set to True, then this weapon is [F&F]; otherwise if MissileDescriptor is populated and IsFireAndForget is False, this weapon is [SA]; if MissileDescriptor is null this weapon receives no additional tag). Note that the actual moving accuracy is set within the HitRollRule inside of Tammunition, two levels of heirarchy deeper.



## 32.6 AnimateOnlyOneSoldier

This variable is set to null if the unit in question is not an infantry unit. If it is, and the weapon in question is a squad weapon limited only to the first man in the squad, then this value is set to True.

## 32.7 Ammunition

The next level of heirarchy. Keep reading...

# 33 Ammunition : TAmmunition

Describes everything about the damage output of the weapon itself, as opposed to the way it's mounted.

## 33.1 DescriptorId

An internal database hash. Don't touch this.

## 33.2 Name

An LocalizationHash which points at a ZZ\_Win.dat file (which one?) to find a name for the cannon as displayed on the unit card. For example, the main cannon of an M1A1 is named M256.

## 33.3 TypeName

Unknown.

## 33.4 TypeArme

This LocalizationHash points at a ZZ\_Win.dat file (which one?) to find the weapon type. Here are the known hash values:

Kind	Hash
MBT Main Cannon	B31E95B36B5E0000
Anti-Ship Missile	57D7010000000000
CQC LMG	E1D2010000000000
More...	Add...

IF, and ONLY IF, TypeArme is set to the Anti-Ship Missile hash, then the AP damage set in Arme, the variable directly below, is multiplied by Physical-Damages to give the unit its true AP damage value. Why? If you pay attention to ATGMs in this game you'll notice that they top out at a universal 30 AP. This is actually an engine limitation, and even getting the limit that high requires a certain trick that will be described later. But all AShMs do way more

than 30 damage in-game (they start at 40). Eugen basically had to hack this in using this clever LocalizationHash abuse.

If this value is set to the hash above, for a close-in LMG, then the weapon will be given the [CQC] tag.

### 33.5 Arme

This variable sets the unit's AP damage value.

If Arme is between 5 and 34 inclusive, the AP damage dealt will be that value less 5, and the weapon will be a kinetic cannon (it will be given the [KE] tag).

If Arme is between 35 and 64 inclusive, the AP damage dealt will be that value minus 34, and the weapon will be a high-energy anti-tank weapon (it will be given the [HEAT] tag).

If Arme is 4, the weapon will do no physical damage (though it may still deal suppression damage).

If Arme is 3, the weapon is given the [AOE] tag and will have an AP of 0 (it will be an HE-only weapon).

The "DeathExplosionArme" top-level variable in TAUUniteSolDescriptor links to one of a few TAmmunition instances, all of which have an Arme value of 4 (for regular units, whose deaths cause suppression damage but not physical damage) or 3 (alongside a certain PhysicalDamages setting, for supply units whose explosions may damage nearby allies).

### 33.6 RadiusSplashPhysicalDamages

This variable controls the area of effect of HE splash damage. It also automatically scales the on-screen impact explosion to match.

FYI, the highest variable value in the game, 33800, is possessed by the B-5.

### 33.7 PhysicalDamages

A float controlling the amount of HE damage this unit in the case of a direct hit.

### 33.8 ProjectileType

Unknown.

### 33.9 Puissance

The direct translation is "noise". This variable is a stealth-negating multiplier that controls how much easier this unit is to spot when it fires this weapon. Values here range from 1 for silenced weapons to the upper two digits.

### 33.10 TempsEntreDeuxTirs

Literal translation is "time between two shots". Each weapon firing in the game is organized in terms of "salvos"; salvo sizes range from 1 for tank cannons to lots and lots for good anti-air pea shooters.

The best way to think of this variable is as the "time between two bursts". If your weapon has a salvo size of 24 bursts, the time between each of those shots in the salvo will be controlled here. But if your weapon has a magazine size of 1 bursts, then TempsEntreDeuxTirs will never kick in, because the magazine is always either full or empty.

For example, take the main weapon of a Biryuska anti-air unit. This weapon fires a hail of bullets in bursts. The time between each bullet in the burst is controlled by TempsEntreDeuxTirs, whilst the time between each burst itself is controlled by TempsEntreDeuxFx.

The critical so-what here is that TempsEntreDeuxTirs is not affected by morale. Once a weapon starts burst-firing, it will continue to burst-fire at its usual rate. It's only the reload time between bursts which is affected by morale fatigue.

Suppose that we have a tank with an autoloader (a T-80U for example). It will have a Salves value (set three levels of hierarchy above, in the WeaponManager) of, say, 24 shots. That means that, technically speaking, a T-80U fires its weapon in a 24-shot burst! It is only in the gap between the 24th and 25th shot that the variable controlling inter-burst time, TempsEntreDeuxFx, kicks in, and so only that shot will be slowed down by morale damage.

An M1A2 without an autoloader, by contrast, will have a Salves value of 1. Each shot is its own burst, so TempsEntreDeuxTirs will never kick in, and the weapon's reload speed will be totally controlled by TempsEntreDeuxFx, the next variable in this list, which is impacted by morale.

The unit here is seconds.

### 33.11 TempsEntreDeuxSalves

This variable controls time between two salvos, bursts, magazines, whatever-you-have-it. Per the terms described in the variable above, it is the only reload time controller for slow-firing weapons without autocannons. For weapons which fire in short bursts, like anti-air guns or machine-guns, this variable controls the size of the gap in between firings.

This variable is impacted by morale.

### 33.12 TempsEntreDeuxFx

This is a visual variable which controls the time in between animations for weapon firings kicking off.

### **33.13 NbrProjectilesSimultanes**

The number of projectiles fired simultaneously. For tank cannons for example this is set to 1, as you only get to fire one shot at a time.

### **33.14 NbTirParSalves**

The number of shots per salvo.

### **33.15 AffichageMunitionParSalve**

The number of displayed ammunition that is depleted after a full salvo is fired.

### **33.16 PorteeMaximale**

Maximal range against ground units.

### **33.17 PorteeeMinimale**

Minimum range against ground units. Set to null if the unit has no minimal range.

### **33.18 PorteeMinimaleBateaux**

Maximal range against ships.

### **33.19 PorteeMaximaleBateaux**

Minimum range against ships.

### **33.20 PorteeMaximaleTBA**

Maximum range against helicopters.

### **33.21 PorteeMinimaleTBA**

Minimum range against helicopters.

### **33.22 PorteeMaximaleHA**

Maximum range against planes.

### **33.23 PorteeMinimaleHA**

Minimum range against planes.

### **33.24 PorteeMaximaleProjectile**

Grants the ability to fire at incoming anti-ship missiles, and sets the maximum range thereof. Grants the [DEF] tag.

### **33.25 PorteeMinimaleProjectile**

Sets the minimum range for firing at incoming anti-ship missiles.

### **33.26 AngleDispersion**

This value controls how much spread a unit has in either of two cases: either when it fails to hit the unit in question directly (a near miss still does HE damage), and how wide the circle is when the unit fires on position.

### **33.27 SuppressDamages**

How much suppression damage this unit does on a direct hit.

Additionally, the "DeathExplosionArme" top-level variable in TAUUniteSolDescriptor links to one of a few TAmmunition instances, examining which shows that most vehicular units deal 180 suppression damage to nearby allies when destroyed.

### **33.28 RadiusSplashSuppressDamages**

The radius of the circle in which the weapon will do suppression damage.

FYI, the highest value in the game, 67600, is possessed by the B-5.

### **33.29 RayonPinned**

The direct translation is "pinned radius". What this variable does is unknown.

### **33.30 TirIndirect**

Null if this weapon is direct-fire, True otherwise.

### **33.31 FX\_tir\_sans\_physic**

A physics variable.

### **33.32 FX\_vitesse\_de\_depart**

A physics variable.

### **33.33 FX\_frottement**

A physics variable.

### **33.34 FX\_tir\_trendu**

A physics variable.

### **33.35 Level**

Unknown.

### **33.36 FireDescriptor**

Unknown.

### **33.37 FireTriggeringProbability**

Unknown.

### **33.38 Caliber**

A hash for the caliber of the weapon, as displayed in the armory screen.

### **33.39 WeaponCursorType**

The type of cursor you get when you hover over a targetable enemy unit. There are three kinds in the game, one for cannons, one for missiles, and one for anti-air. In order these are 1, 2, and 3.

### **33.40 NoiseDissumlationMalus**

### **33.41 TempsDeVisee**

This controls a unit's aim time. Aim time, which is affected by morale, is the amount of time it takes for a unit to go from seeing an enemy to being ready to shoot it.

Note that if the unit is still reloading, or the weapon is on a turret and the other unit is outside of the cone of fire (for example, if you are passing by a helicopter with an airplane, or your tank turret hasn't finished turning around yet), then your unit can finish aiming before it is actually ready to fire.

### **33.42 InterfaceWeaponTexture**

Unknown.

### **33.43 AffichageMenu**

Unknown.

### **33.44 SupplyCost**

The cost, in units of supply (liters), to reload one ammunition instance for this weapon.

### **33.45 SmokeDescriptor**

If this is populated, the unit is granted a [SMK] tag.

### **33.46 MissileTimeBetweenCorrections**

The number of seconds between a missile exiting the tube and the missile receiving a "reroll". When a reroll occurs, a missile that's still in transit has to pass another check to see if it will hit the target or not. For this reason slow long-range missiles are less accurate than they appear to be in the stats alone, because they have to reroll two, possible three times! That has a huge negative effect on actual hit chance, and the result is especially pronounced in the case of infantry ATGMs.

### **33.47 Guidance**

If this value is set to null or 0, this weapon has no guidance. If this is set to 1, this weapon is radar-guided, receiving the [RAD] tag (and the corresponding ability to get its unit killed by SEAD). If this is set to 2, this weapon is anti-radar, receiving the [SEAD] tag.

### **33.48 EfficaciteSelonPortee**

If this is set to True, the weapon's AP scales with distance to its target, as would be expected of a kinetic energy ([KE] tagged) weapon. Null otherwise. Note however that the [KE] tag alone doesn't guarantee this property, or vice versa; properly formatted units must have both set.

### **33.49 AffecteParNombre**

Unknown.

### **33.50 NeedModelChange**

Unknown.

### **33.51 IsFireAndForget**

Rather obviously makes the weapon fire and forget (grants a [F&F] tag).

### **33.52 IgnoreInflammabilityConditions**

Gives this weapon the [NAPALM] tag.

### **33.53 InterdireTirReflexe**

Unknown, but related to the below somehow.

### **33.54 TirReflexe**

Seems to allow CIWS weapons (ones which have PorteeMaximaleProjectile) to target all incoming missiles, including non anti-ship ones.

### **33.55 DispersionAtMaxRange**

Artillery unit radial accuracy at its maximum range.

### **33.56 DispersionAtMinRange**

Artillery unit radial accuracy at its minimum range.

### **33.57 CorrectedShotDispersionMultiplier**

How much better an artillery shot will be if the shot is on a vision-spotted target.

### **33.58 IsSubAmmunition**

Will give the unit a cluster damage ([CLUS]) tag.

### **33.59 RandomDispersion**

Unknown.

### **33.60 TempsAnimation**

Unknown.

### **33.61 HitRollRule**

Contains a reference to a TModernWarfareHitRollRule, which contains four variables: MinimalHitProbability, which sets a floor on how low a shot's accuracy can go (0.05 is usual); minimal crit probability, which sets a floor on how low crit chance can go (0.01 is usual); hit probability, which controls the base chance to hit of the weapon in question; and hit probability while moving, which controls the weapon's chance to hit while the unit is on the move (but only if the weapon is not stationary, which is set elsewhere!).



### **33.62 MissileDescriptor**

This variable contains a reference to a TUnitDescriptor object that describes some kind of missile. Since a missile isn't meant to appear as its own unit, all of the interesting variables thereof are set to null; indeed, this reference only serves to link the art assets of the missile to those of the weapon firing it, as all variables like accuracy and salvo size are set within TAmmunition itself.

## **34 TModernWarfareDamageModuleDescriptor**

This module is the second and less complex of the two modules controlling the most interesting aspect of the game, its combat. This module controls armor and defensive values. Its top-level TModernWarfareDamageModuleDescriptor object references a TModernWarfareCommonDamageDescriptor, which in turn references a TBlindageProperties.

### **34.1 ControllerName**

Not interesting.

### **34.2 CommonDamageDescriptor**

### **34.3 MaxDamages**

The unit's HP.

### **34.4 MaxHPForHUD**

The number of bars of HP to display in-game; this only differs from MaxDamages for ships.

### **34.5 Experience**

Always set to "Experience".

### **34.6 AutoOrientation**

Whether or not the unit automatically rotates itself to face forward against any units that it is targeting. For ground units the unit's best armor is in front, so AutoOrientation is a defensive trait thereof.

### **34.7 Transporter**

Whether or not this unit is a transporter unit. Set to null if it is not, and "Transporter" if it is.

### **34.8 IsTargetableAsBoat**

Whether or not the unit is targetable as a ship (e.g. by AShMs). Set to null if not and True if so.

## **35 TAppearanceModelDescriptor**

Controller for things related to the unit's appearance.

## **36 CommonDamageDescriptor : TModernWarfareCommonDamageDescriptor**

This module is referenced from TModernWarfareDamageModuleDescriptor, and contains further settings related to damage reception.

### **36.1 PaliersSuppressDamages**

Always the list [0, 0.25, 0.5, 0.75]. Controls the percentage of the suppression damage ceiling at which the various morale damage levels kick in.

### **36.2 PaliersPhysicalDamages**

Always the list [0, 0.5, 0.75]. Controls the percentage of the physical health total at which physical damage effects kick in.

### **36.3 SuppressDamagesRegenRatio**

Always a list of float pairs that looks like [1:0.2, 10:0.5, 30:1, 40:2, 50:3]. Controls the multiplier for suppression damage regeneration? But the values don't look quite right...

### **36.4 SuppressDamagesRegenRatioOutOfRange**

How quickly the unit recovers from suppression when not actively receiving suppression damage. Always 20, except in the case of some combination of ships and supply vehicles.

### **36.5 StunDamagesRegen**

Unknown; perhaps this is how much suppression damage the unit recovers while it is stunned? Always 5, except in the case of some combination of ships and supply vehicles.

### **36.6 StunDamagesToGetStunned**

The amount of stun damage the unit has to receive in order to be stunned in the first place. Always 300, except in the case of some combination of ships and supply vehicles; this suppression amount has to be generated within a certain amount of time.

### **36.7 SuppressDamagesEffects**

Contains a table of references to TSuppressDamagesEffects which describe the (global, basically) effects of a unit receiving suppression damage. The values at which these levels kick in are set by PaliersSuppressDamages, described above.

### **36.8 PhysicalDamagesEffects**

Contains a table of references to TPhysicalDamagesEffects which describe the (global, basically) effects of a unit receiving physical damage. The values at which these levels kick in are set by PaliersPhysicalDamages, described above. The effects here are not as extreme as the ones for suppression damage, but still quite noticeable.

### **36.9 MaxSuppressionDamages**

The maximum amount of suppression damage a unit can receive. Always 800, except in the case of some combination of ships and supply vehicles.

### **36.10 PhysicalDamagesEffects**

Contains a table of references to TPhysicalDamagesEffects which describe the (global, basically) effects of a unit receiving physical damage. The values at which these levels kick in are set by PaliersPhysicalDamages, described above. The effects here are not as extreme as the ones for suppression damage, but still quite noticeable.

### **36.11 TestMoralRollRule**

Contains a reference to TModernWarfareTestMoralRollRule. Purpose needs further investigating...

### **36.12 TBlindageProperties**

Contains four armor values, one for each side of the unit: front, sides, rear, top, in that order. Each in turn links to a TArmorDescriptor module with a BaseBlindage integer value. As with weapon damage, BaseBlindage is not a direct translation to armor value, but rather a map. A BaseBlindage of 0 or null means no armor whatsoever. A BaseBlindage of 1 through 4 means an AV of 0 through 3 alongside what has been tested to be a measure of splash damage

resistance. A BaseBlindage of 5 through 34, finally, is AV plus 4 (so for example a BaseBlindage of 21 refers to 17 AV).

## 37 TVisibilityModuleDescriptor

### 37.1 ControllerName

Uninteresting.

### 37.2 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

### 37.3 \_ShortDatabaseName

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

## 38 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.