



# MITALWAR

## *Conception BDD-Protocole MitalWar*



## ***Historique des modifications***

Date	Version	Nom	Modifications	Paragraphes concernés
11/06/07	0-1	Inimz	Création du document initial.	Tous
24/06/07	0-2	Olivier	Modification de paragraphes + Modèle UML	1 et 2



## *Table des matières*

Table des matières .....	3
1.1 Méthode de conception de la base de données .....	4
i. Règles méthodologiques suivies .....	4
ii. Modélisation de la base de données .....	4
1.2 Identification des données à stocker .....	4
2 Modèle conceptuel .....	5
i. Tables et associations relatives au serveur .....	5
ii. Tables et associations relatives au jeu .....	5
iii. Tables et associations relatives à la configuration courante (joueur) .....	5
iv. Résumé des associations .....	5
v. Résumé des tables et attributs .....	5
vi. Modélisation UML du modèle conceptuel .....	7
3 Modèle physique .....	7



## 1) Conception BDD MitalWar

### 1.1 Méthode de conception de la base de données

#### i. Règles méthodologiques suivies

Notre objectif premier était d'aboutir sur un modèle conceptuel de données normalisé jusqu'en 3<sup>ième</sup> forme normal. Pour cela, nous avons veillé à ce que le modèle mis en place respecte la méthodologie Merise qui sert à garantir la cohérence des données ainsi que des traitements, on peut par conséquent être sûre du résultat qui sera produit par l'application développée. Pour cela cette méthodologie doit suivre respectivement les pratiques suivantes :

- Définition des entités ;
- Définitions des attributs propres à chaque entité ;
- Définition d'un identifiant unique pour chaque entité ;
- Définition des relations entre les entités et éventuellement de leurs attributs spécifiques, qui ne peuvent ou ne doivent pas être affectés aux entités appartenant à la relation ;
- Définition des cardinalités ;
- Suppression des éventuelles transitivités ;
- Vérification que le schéma est connexe, c'est-à-dire que l'ensemble de ces composants interagissent entre eux ;
- Vérifier qu'il n'y a pas de redondances de données ;
- Vérifier qu'il n'y a pas de données calculées ;
- Vérifier que toutes les fonctionnalités pré requises sont incluses.

Néanmoins, nos contraintes de temps nous obligeant à coder rapidement pour respecter le planning visant à produire une démo du jeu pour la fin 2007, nous devons impérativement connaître les champs manipulés en minimisant les allers-retours à la documentation. Ainsi, nous avons choisi des noms sémantiquement parlant au détriment de la normalisation. En d'autres termes, nos clés étrangères ne portent pas obligatoirement le nom de la clé primaire à laquelle elles se réfèrent. Ce choix est d'autant plus important que pour la gestion des différentes langues, une même table pourra contenir une clé étrangère vers une clé primaire issue de la même table. La distinction est donc inévitable pour éviter tout problème de collisions.

En bref, nous avons respecté les standards en matière de développement afin d'assurer une clarté maximum au travail produit, ceci est indispensable pour ce type de projet dont l'avancement est basé sur le mode 'Usine de développement'.

#### ii. Modélisation de la base de données

Pour une meilleure compréhension, le modèle conceptuel est représenté par un diagramme UML et le modèle physique par un schéma fait avec Visual Studio 2005 C# Express<sup>1</sup>.

Nous avons décidé de réaliser deux modélisations différentes pour couvrir tous les aspects utiles de la modélisation. Le diagramme UML est en effet plus illustratif pour définir les besoins de la base de données et il est donc plus clair et plus accessible pour le raisonnement. Nous pouvons aussi pour réaliser un suivi quotidien retranscrire ce diagramme sous Visual Studio 2005 Express (en version plus simplifiée). Mais c'est bien le diagramme UML évoqué au début qui fait autorité sur la structure de la base, les requêtes à réaliser, etc. Il apporte également une génération automatique du code source qui réduit un risque de désynchronisation entre le modèle imaginé et le code écrit.

### 1.2 Identification des données à stocker

Les différentes données pourront être déterminées en se basant sur les documents :

1 Ce logiciel est utilisé dans le projet MitalWar pour la programmation.



- 060724\_MITALWAR\_GameDesign\_v1-0\_Marc.doc
- 060822\_MITALWAR\_Unites\_v0-6\_Marc.xls
- 070503\_MITALWAR\_Evenements\_v0-1\_Marc.doc

Il sera important de suivre les différentes modifications de ces trois derniers documents car ils pourront influencer sur la suite du présent document.

## 2 *Modèle conceptuel*

### *i. Tables et associations relatives au serveur*

### *ii. Tables et associations relatives au jeu*

Un Joueur(PlayerInstance) sera identifié par une table. Afin de garder une certaine logique dans le modèle de données, on différencie le titulaire d'un compte(Account) avec le joueur d'une partie. Mais ces 2 entités sont tout de même en relation. Un type de joueur(PlayerType) permet de définir l'instance du joueur. Cela permet d'attribuer au joueur son avatar. Chaque type de joueur a un pouvoir(PowerStrike) propre à lui même en multijoueur. Par contre en singleplayer, le joueur gagne en grade lors de ses victoires, lui permettant d'obtenir de nouveaux pouvoirs. La table (PowerStrike) est donc en association avec la table (Ranking) qui représente la liste des grades que peut avoir le joueur.

Lors d'une partie, le joueur(PlayerInstance) gère plusieurs unités(UnitInstance). Chaque instance d'unité est défini selon un type d'unité(UnitType). La table (UnitState) liste tous les états que peut avoir une unité. Elle est donc associée à l'entité (UnitInstance) afin de définir l'état courant de l'unité. On peut imaginer les états « camouflage », « défense », etc...

Une unité(UnitInstance) peut agir sur différents plans (eau,terre,air,sous l'eau). Elle est également positionnée sur le plateau de jeu(Level) qui est composé de plusieurs cases (Square). En associant l'unité à une case du jeu cela permet de déterminer sa position actuelle.

Une case (Square) est définie selon son type(SquareType). En fonction du type de terrain(SquareType) et du plan(Plan) sur lequel une unité agit, cela modifie certains attributs de l'unité. Il en va de même avec l'entité (Weather) qui représente les différentes météo du jeu. Celle ci est aussi associé avec la table Plan qui modifie les attributs des unités.

L'instance d'un niveau est relié à cette table (Weather) afin de connaître la météo actuelle du jeu.

Les entités ayant un libellé permettant d'informer le joueur sur le nom de certains objets(unités, pouvoirs,etc) sont associées à la table (Text) afin de déterminer la traduction des libellés selon le langage utilisé par le client.

### *iii. Tables et associations relatives à la configuration courante (joueur)*

### *iv. Résumé des associations*

### *v. Résumé des tables et attributs*

Dans le tableau suivant, nous allons définir les principales tables et donner leurs attributs. Par convention, la clé primaire de chaque table sera soulignée<sup>2</sup>. Notons que les clés primaires sur des éléments critiques destinés à être stockés plusieurs fois et/ou à être accéder fréquemment utiliseront une clé de type Int64.

---

2 Cette convention sera gardée dans tout le document. Toute clé primaire a pour but d'identifier de manière unique chacun des éléments stockés dans une table donnée.



Conception BDD-Protocole MitalWar

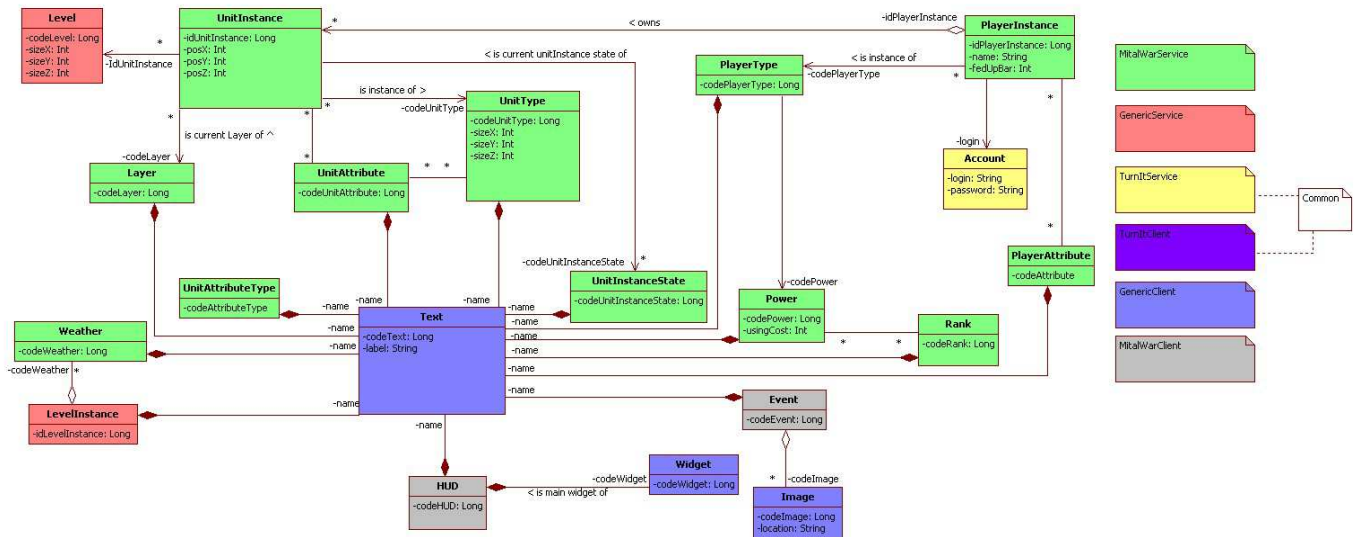
Tables	Identifiant français	Attributs	Type	Commentaire
PlayerInstance	Contexte Joueur	<u>Name</u> FedUpBar	String Int64	Un contexte de joueur est représenté par son nom de général. Il contient aussi la barre de « ras le bol » du joueur. Cette entité permet de différencier le compte du joueur avec le personnage qu'il incarne pendant le jeu.
PlayerType	Type de joueur	<u>idGeneral</u>	Int64	Cette table liste tous les personnages que le joueur peut incarner durant une partie
PowerStrike	Pouvoir	<u>idPower</u> cost	Int64 Int	Un pouvoir est un bonus que peut utiliser un joueur, celui ci a un prix d'utilisation(cost).
Ranking	Classement	<u>IdRank</u>	Int64	Cette table liste tous les rangs que peuvent avoir les joueurs. Cela va du simple soldat jusqu'au maréchal.
UnitType	Type d'unité	<u>IdTypeUnit</u> Faction CombustibleAmount AmmoAmount ShieldAmount Mobility Perception PowerAtk RangeAtk AmmoSupply CombustibleSupply ShieldSupply	Int64 Int Int Int Int Int Int Int Int Int Int Int	Cette table va lister tous les modèles d'unité présent dans MitalWar. Chaque type est constitué de plusieurs attributs comme sa quantité de munition(AmmoAmount), le nombre de déplacement qu'elle peut effectuer de base(Mobility), sa puissance de feu(PowerAtk) ou bien sa portée d'attaque(RangeAtk),la faction à laquelle elle appartient,etc.
Plan	Plan	<u>idPlan</u>	Int64	Il s'agit de la liste des plans dans le jeu: subaquatique, aquatique, terrestre, aérien.
UnitInstance	Contexte d'unité	<u>idUnitInstance</u> CurrentCombustible CurrentAmmo CurrentShield CurrentMobility AmmoSupply CombustibleSupply ShieldSupply	Int64 Int Int Int Int Int Int Int	Liste de toutes les unités créées dans la partie. Elles sont définies par un type d'unité.
UnitState	Etat de l'unité	<u>idUnitState</u>	Int64	Liste des différents états dans laquelle peut se trouver une unité(camouflage,defense,plan actuel)
Weather	Météo	<u>idWeather</u>	Int64	Table contenant les différentes météo utilisée dans une partie. Le type de météo peut influencer sur 4 caractéristiques d'unité: la mobilité, le combustible,l'attaque et la défense
SquareInstance	Contexte de case	<u>PosX,posY,posZ</u>	Int64	Table contenant toutes les cases du damier d'un niveau
SquareType	Type de case	<u>idSquareType</u>	Int64 Int Int	Liste des types de case définissant une case d'un damier. En fonction de son type, la case peut influencer sur la



## Conception BDD-Protocole MitalWar

Tables	Identifiant français	Attributs	Type	Commentaire
			Int	mobilité des unités.
Event	Événement	<u>IdEvent</u>	Int64	

### vi. Modélisation UML du modèle conceptuel



### 3 Modèle physique

Nous faisons apparaître les nouvelles tables créées de jointure et les différentes clés étrangères (en italique souligné pointillé) dans le tableau ci-dessous :

Tables	Identifiant français	Attributs	Type	Commentaire
PlayerInstance	Contexte Joueur	<u>Name</u> <u>FedUpBar</u> <u>idGeneral</u>	String Int64 Int64	Personnage choisi
PlayerType	Type de joueur	<u>IdGeneral</u> <u>idPower</u> <u>name</u>	Int64 Int64 String	Pouvoir du personnage son nom
PowerStrike	Pouvoir	<u>idPower</u> <u>cost</u> <u>name</u>	Int64 Int String	Nom du pouvoir
Ranking	Classement	<u>IdRank</u> <u>name</u>	Int64 String	Nom du rang
A_PowerStrike_Ranking	Association pouvoir et classement	<u>IdPower</u> <u>idRank</u>	Int64 Int64	Identifiant du pouvoir Identifiant du rang



Conception BDD-Protocole MitalWar

Tables	Identifiant français	Attributs	Type	Commentaire
UnitType	Type d'unité	<u>IdTypeUnit</u> Faction CombustibleAmount AmmoAmount ShieldAmount Mobility Perception PowerAtk RangeAtk AmmoSupply CombustibleSupply ShieldSupply	Int64 Int Int Int Int Int Int Int Int Int Int Int	
Plan	Plan	<u>idPlan</u> <u>name</u>	Int64 String	Nom du plan
UnitInstance	Contexte d'unité	<u>idUnitInstance</u> CurrentCombustible CurrentAmmo CurrentShield CurrentMobility AmmoSupply CombustibleSupply ShieldSupply <u>idUnitType</u> <u>idPlan</u> <u>idUnitState</u>	Int64 Int Int Int Int Int Int Int Int Int64 Int64 Int64	Type d'unité Plan d'action actuel de l'unité Etat actuel de l'unité
UnitState	Etat de l'unité	<u>idUnitState</u> <u>name</u>	Int64 String	Nom de l'état
Weather	Météo	<u>idWeather</u> <u>name</u>	Int64 String	Nom du temps
A_Weather_Plan	Association plan et météo	<u>idWeather</u> <u>idPlan</u> MobilityInfluence AtkInfluence DefInfluence EnergyInfluence	Int64 Int64 Int Int Int Int	Identifiant de la météo Identifiant du plan influence sur la mobilité influence sur l'attaque influence sur la défense influence sur l'énergie
SquareInstance	Contexte de case	<u>PosX,posY,posZ</u> <u>idLevel</u> <u>idSquareType</u>	Int64 Int64 Int64	Identifiant du niveau Identifiant du type de case
SquareType	Type de case	<u>idSquareType</u> <u>name</u>	Int64 String	Nom du type de case
A_Plan_SquareType	Association Plan et SquareType	<u>idPlan</u> <u>idSquareType</u> mobilityInfluence	Int64 Int64 Int	Identifiant du plan Identifiant du type de case influence sur la mobilité
Event	Evénement	<u>IdEvent</u> <u>name</u>	Int64	Libellé de l'événement

Ce tableau est notre modèle physique.

#### 4) Protocole d'échange MitalWar



Document sous licence [Creative Commons Paternité-NonCommercial-NoDerivs 2.5 License](https://creativecommons.org/licenses/by-nc-nd/2.5/)

Droits réservés à l'association Futurn et à son auteur : D. Lesage