**Warhammer 40K : Army Factory**

De la conception à la réalisation d’une application de gestion de créations d’armées dans l’univers de Warhammer 40000

Alexis CORNET, Nicolas APERCE

Le 14 décembre 2016

Warhammer 40K : Army Factory est un projet débuté dans le cadre du cours « 8INF957 Programmation objet avancée » dispensé au cours du semestre d’automne 2016 à l’Université du Québec à Chicoutimi (UQAC).

Ce projet a ainsi été pensé, conçu et réalisé par Alexis CORNET et Nicolas APERCE, tous deux étudiants à l’UQAC.

CORA24129403, [alexis.cornet1@uqac.ca](mailto:alexis.cornet1@uqac.ca)

APEN05049401, [Nicolas.aperce1@uqac.ca](mailto:Nicolas.aperce1@uqac.ca)

# Abstract (100 à 150 mots)

# Mots Clef

Projet informatique, Études, Warhammer 40K, Programmation Orientée Objet, C++, Conception, UML

Table des matières

[Abstract (100 à 150 mots) 2](#_Toc469338285)

[Mots Clef 2](#_Toc469338286)

[1. Introduction 2](#_Toc469338287)

[2. Games Workshop et Warhammer 4](#_Toc469338288)

[3. La préparation aujourd’hui 5](#_Toc469338289)

[4. Matériel et méthode 7](#_Toc469338290)

[4.1. Conception 7](#_Toc469338291)

[4.2. Réalisation 8](#_Toc469338292)

[5. L’application 9](#_Toc469338293)

[5.1. Préambule 9](#_Toc469338294)

[5.2. Objectif 9](#_Toc469338295)

[5.3. Atteint 9](#_Toc469338296)

[5.4. Fonctionnement 10](#_Toc469338297)

[Table des légendes 13](#_Toc469338298)

[Références 13](#_Toc469338299)

# Introduction

Warhammer 40000 est, à la base, un jeu de figurines existant depuis 1987 réalisé et produit par la société Games Workshop.

Univers et jeu de plus en plus connus des joueurs et collectionneurs de jeux de figurines et/ou de plateaux, il met en opposition un minimum de deux joueurs autour d’un terrain, disposant chacun d’une armée de figurines, qu’il aura préalablement assemblées et peintes.

Bien que cet univers soit en constante expansion, aucune application n’existe actuellement dans le but d’aider ou d’assister le joueur à la conception et à la création d’une armée avant de réaliser une partie.

Par conséquent, l’application « Warhammer 40K : Army Factory », constituant le cœur de cet article, est conçue dans ce but. Le joueur peut désormais penser, créer, maintenir et optimiser son armée via cette application.

Le document qui suit explique les phases de conception et de réalisation d’un projet visant à concevoir une telle application, réalisée par deux étudiants à l’Université du Québec à Chicoutimi. Bien que cette application ne soit, ni complète, ni achevée, la phase de conception ayant permis de procéder à sa réalisation restera, par la suite, similaire.

# Games Workshop et Warhammer

Premièrement, il est nécessaire de savoir ce qu’est l’univers de Warhammer 40000.

... Blablabla … (Ou voir pour compléter l’intro au lieu de faire cette partie)

# La préparation aujourd’hui

Aujourd’hui, lorsqu’un joueur est désireux de créer une nouvelle armée afin de pouvoir participer à des parties de Warhammer 40K, ce dernier doit procéder comme suit :

* Effectuer des recherches dans des livres de règles
* Effectuer des recherches dans un livre appelé « Codex » (propre à chaque race et corps d’armée)
* Recenser les différentes figurines et unités jouées dans l’armée, sur feuille ou autre support personnel, mais aucune application ayant cet objectif n’est mis à disposition des joueurs
* Calculer les points de la création de chacune de ces unités en fonctions des figurines et équipements la composant, et ce, dépendant des différentes règles respectées

Lorsqu’il souhaite optimiser son armée, le joueur doit calculer l’intégralité de ses points avant d’effectuer un changement et gérer ses figurines, procéder à des schémas d’armées et à la réalisation de calculs statistiques.

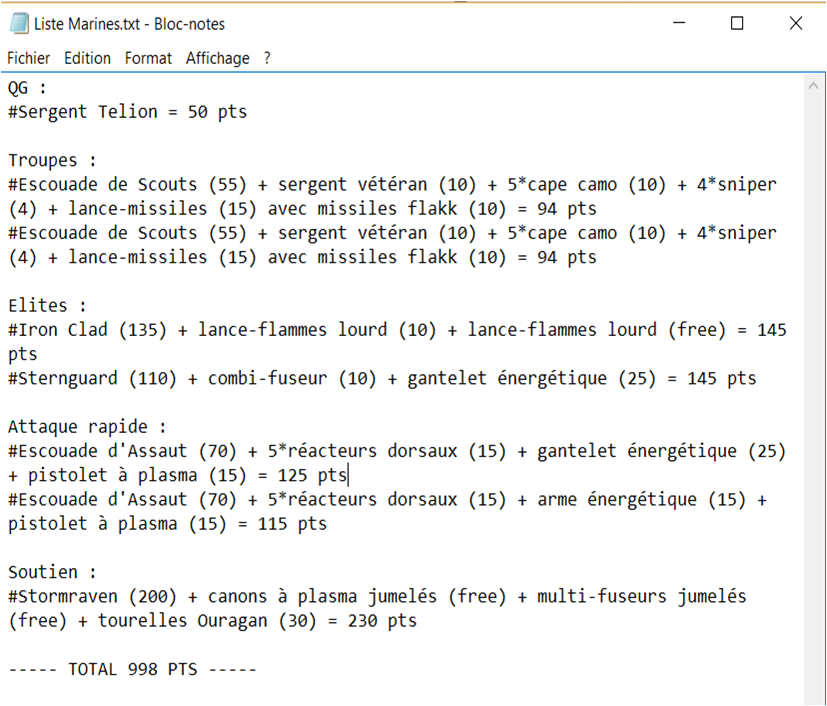
Tout ceci étant fastidieux, et les calculs pouvant s’avérer complexes en fonction des corps d’armées et des règles choisies, il est aisé pour le joueur de se tromper dans ses calculs, et, par conséquent, dans la création même de son armée.

Figure 1 : Exemple création armée

La figure 1 ci-dessus consiste en un exemple de création d’une armée.

On y retrouve des rôles tels que « QG », « Troupes », « Elites », « Attaque rapide » et « Soutien » ; et on remarque que cette armée contient 998 points.

L’armée dispose des unités « Sergent Telion », « Escouade de Scouts », « Iron Clad », « Sternguard », « Escouade d’Assaut », « Stormraven ».

On aperçoit de plus les figurines composant ces différentes unités, ainsi que les équipements leur étant attachés.

On retrouve bien comme indiqué, le calcul menant à la somme de 998 points

Les parties s’effectuent en disposant d’armées respectant un certain nombre de points préalablement défini entre les joueurs. L’armée valant ici 998 points, elle est faite pour participer à des parties où chacune des armées vaut un maximum de 1000 points.

De la sorte, si le joueur souhaite désormais améliorer et ainsi optimiser son armée pour la rendre plus compétitive, il se doit de trouver par quel moyen (changement de figurine, d’équipement, etc.), tout en effectuant un nouveau calcul de points pour chaque changement envisagé.

# Matériel et méthode

L’application appelée « Warhammer 40K : Army Factory » a été conçue et réalisée à l’aide de logiciels entièrement gratuits. (Il sera possible de trouver le terme WH40KAF dans la suite du document afin de faire référence à ce nom.)

## Conception

« Warhammer 40K : Army Factory » a d’abord été pensée sur feuille de papier afin de regrouper l’ensemble des idées et informations que l’application doit nécessairement contenir.

L’application a été découpée en une architecture 3-tiers dont les 3 couches sont les suivantes :

* DAL pour « Data Access Layer »
* BLL  pour « Business Logic Layer »
* IHM pour « Interface Homme Machine »

La conception du projet fut définie en employant le langage UML via le logiciel open source UMLet. Deux types de diagrammes UML furent réalisés.

Des diagrammes de classes ont été dessinés dans le but de montrer les interactions des dites classes au sein d’une même couche.

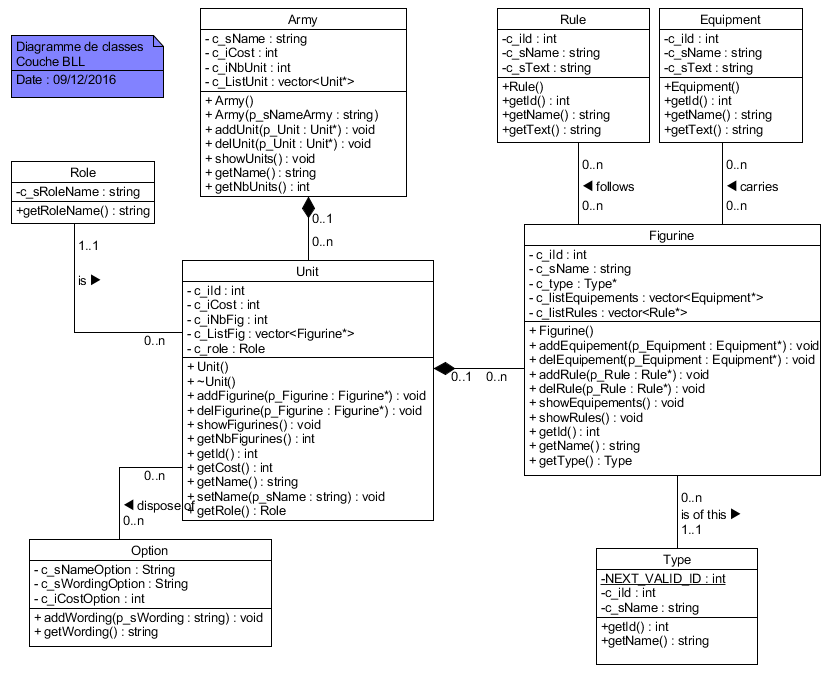


Figure 2 : Diagramme de classes de la couche BLL

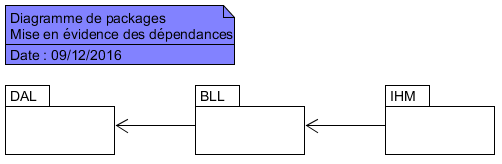
Un diagramme de package est également conçu, mettant en évidence les dépendances architecturales des 3 couches.

Figure 3 : Diagramme de packages et dépendances

On remarque sur la figure 3 ci-dessus, que la couche IHM dépend de la couche BLL, elle-même dépendante de la couche DAL.

La couche DAL sert à la connexion avec la base de données.

La couche BLL permet la réalisation d’opérations métiers sur les différents éléments mis en base.

La couche IHM s’assure de l’affichage de l’application.

## Réalisation prévue

Afin de réaliser l’application WH40KAF, les principes du paradigme de la programmation orientée objet furent respectés. On retrouve ainsi les principes classiques comme le polymorphisme (au travers de l’héritage), la réutilisabilité ou la séparation des responsabilités (une classe appelée « Figurine » ne traitera que des dites figurines).

L’application est une application dite Windows Form réalisée avec l’IDE « Microsoft Visual Studio Community ». Il s’agit donc d’un projet de type « CLR », et employant le langage Visual C++ basé sur le FrameWork .NET4.5 développé par Microsoft.

L’application est connectée à une base de données MySQL réalisée via PHPMyAdmin et gérée par un serveur Apache.

La couche DAL telle réalisée définit les méthodes de connexion et de déconnexion à la base de données.

La couche BLL permet la récupération et la modification des éléments mis en base, et sert de « base des opérations » pour toute interaction entre ce que voit l’utilisateur et la base de données.

La couche IHM dispose quant à elle seulement de fichiers de type fenêtres ou vues, appelées respectivement « Windows Form » et « CLR User Control » sur l’IDE. Ces fichiers correspondent ainsi à la partie visible par l’utilisateur.

## Réalisation effectuée

La conception UML de l’application montrait que cette dernière devait suivre une architecture 3-Tiers, cependant, et par soucis de programmation, il ne put être fait comme prévu.

En effet, il a été pensé de réaliser une « solution » dans l’IDE « Microsoft Visual Studio », qui contiendrait 3 projets « DAL », « BLL », « IHM », et où les références DLL à chacun de ces projets permettrait le respect de la dite architecture. Néanmoins, un problème au niveau de l’IDE (certainement au niveau des propriétés de l’IDE ou des différents projets) a conduit à revoir ce schéma, et un seul projet fut finalement réalisé. Ce projet est ainsi séparé en 3 « namespaces » différents, représentant chacun une partie de l’architecture ; on retrouve ainsi un namespace « DAL », un namespace « BLL » et un namespace « IHM ».

# L’application

## Préambule

Reprenant la partie « La préparation aujourd’hui » telle définie précédemment, notre application a pour objectif de simplifier cette préparation.

L’application se doit par conséquent d’éviter à l’utilisateur d’effectuer ses recherches et calculs de façon continue. Il continuera d’effectuer ses recherches de figurines afin de savoir laquelle ajouter à son armée, mais n’aura nullement nécessité à vérifier l’intégralité des interactions possibles entre la dite figurine et les équipements ou règles auxquels la lier, ni même besoin de calculer le nouveau coût de son armée, le tout devant s’effectuer automatiquement sur l’application.

## Objectif

Ayant pour objectif d’aider et d’assister l’utilisateur dans la création et l’optimisation d’armées jouables, l’application se doit de permettre à l’utilisateur de réaliser l’ensemble des tâches nécessaires à la création de son armée.

Autrement dit, l’utilisateur cherche dans la base de données de l’application, la figurine qu’il souhaite ajouter à son unité. De là, l’application lui fait connaître, en le lui affichant, le coût de cette figurine, puis le coût de l’unité contenant cette figurine.

Lorsque l’utilisateur décide de modifier son unité, par le biais d’un ajout ou suppression, modification d’une figurine, l’application affiche le nouveau coût de l’unité.

Les équipements disposant de coûts respectifs dépendamment des armées jouées, l’application doit également afficher à l’utilisateur le nouveau coût de l’unité et de l’armée lorsque celui-ci décide de la modifier par le biais d’un ajout, suppression, ou modification d’un équipement.

L’application propose à l’utilisateur d’enregistrer les unités correspondant à ses choix, ainsi que l’enregistrement et la validation de son armée après y avoir inclus les différentes unités la composant.

L’ensemble de ces informations, c’est-à-dire, informations auxquelles accède l’utilisateur, informations mises à jours par l’utilisateur lorsqu’il crée ou modifie une armée ou une unité ; sont présentes en une base de données MySQL hébergée sur un serveur Apache.

## Atteint

Aujourd’hui, l’intégralité des points cités en 5.2. ne sont pas fonctionnels.

En effet, l’application est liée à une base de données hébergée sur un serveur Apache.

## Fonctionnement

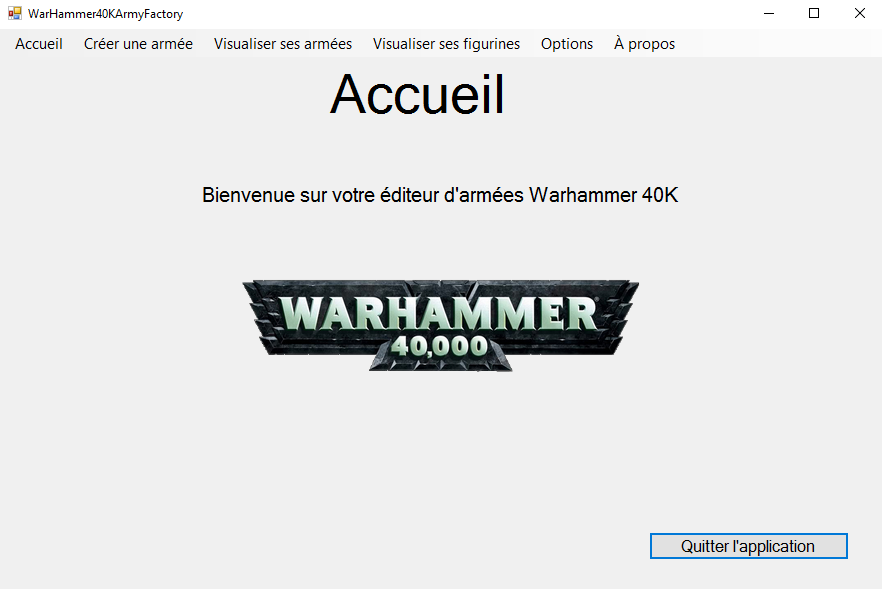
Lorsque l’utilisateur démarre l’application « Warhammer 40K : Army Factory », celle-ci se connecte à sa base de données « test\_army\_factory.sql » et affiche l’écran d’accueil. L’utilisateur n’a nullement connaissance de l’établissement de connexion, et ne voit que la fenêtre d’accueil telle présentée sur la figure 4 présente ci-dessous.

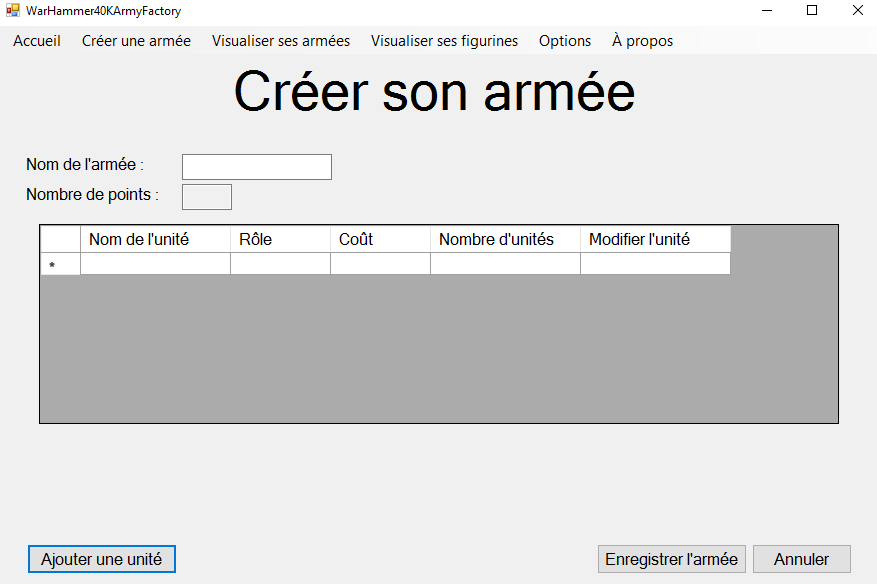
Figure 4 : Écran d'accueil de l'application

L’utilisateur a dès lors la liberté de :

* Créer une armée ;
* Voir (et modifier) une armée ;
* Voir les figurines ;
* Vérifier les options propres à l’application ;
* Revenir à cet écran lorsqu’il n’est plus dessus ;
* Quitter l’application

On retrouve sur les figures 5 et 6 suivantes, l’écran de création d’une armée ainsi que la possibilité d’y ajouter des unités

Figure 5 : Écran de création d'une armée

L’écran représenté par la figure 5 ci-dessus propose à l’utilisateur de choisir un nom pour son armée. Ce même écran lui propose également d’ajouter une unité (cf. figure 6 ci-dessous), ainsi que de valider, ou bien au contraire, d’annuler la création de la dite armée.

L’application affiche à l’utilisateur, l’ensemble des unités qu’il aura préalablement définies et qu’il souhaite incorporer dans son armée, ainsi que la totalité des points relative à ces unités.

Lorsque l’utilisateur appuie sur « Enregistrer l’armée », cette dernière est automatiquement enregistrée dans la base de données ; tandis que lorsqu’il clique sur « Annuler », cette fenêtre se vide.

La fenêtre présente sur la figure 6 ci-dessous propose à l’utilisateur de sélectionner, via une liste déroulante, les modèles de base possibles pour une unité.

L’utilisateur saisi de plus un nom pour son unité, et sélectionne son rôle dans l’armée. La liste déroulante d’équipements affiche à l’utilisateur les équipements indiqués par celui-ci.

Afin de remplir cette liste, l’utilisateur appuie sur « Ajouter un équipement » qui ouvre une nouvelle fenêtre indiquant les informations à renseigner.

Il procède de même pour ajouter une nouvelle figurine en appuyant sur « Ajouter une figurine ».

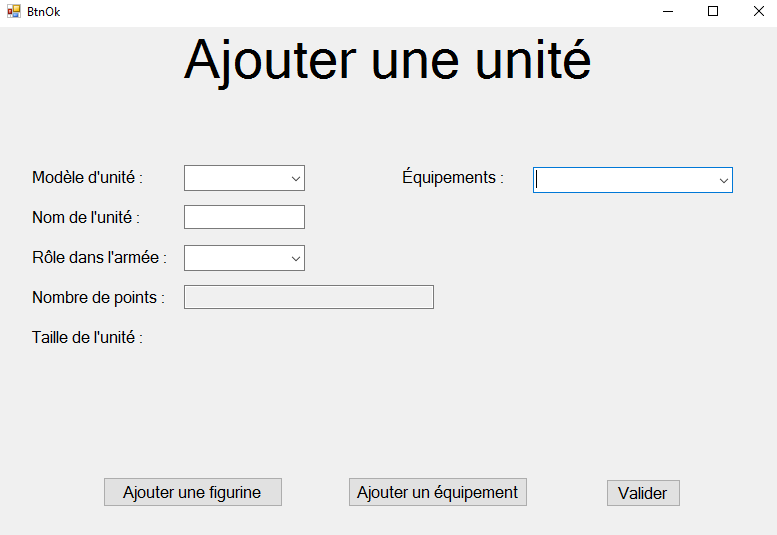


Figure 6 : Écran d'ajout d'une unité

# Table des légendes

[Figure 1 : Exemple création armée 5](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338279)

[Figure 2 : Diagramme de classes de la couche BLL 7](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338280)

[Figure 3 : Diagramme de packages et dépendances 8](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338281)

[Figure 4 : Écran d'accueil de l'application 10](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338282)

[Figure 5 : Écran de création d'une armée 11](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338283)

[Figure 6 : Écran d'ajout d'une unité 12](file:///D:\Stockage\Documents\Cours\UQAC\Programmation%20Orientée%20Objet\Projet%20final\Rapports\Article%20Projet%20WH40K.docx#_Toc469338284)

# Références