# Conception d'un dispositif ludopédagogique intégré pour la formation au management des systèmes d'information

Juliette Frédy, Nawel Ghazal, Samy Marchetti

7e Colloque International *Game Evolution*

# Conception d'un dispositif ludopédagogique intégré pour la formation au management des systèmes d'information

**Juliette Frédy**
IAE Paris-Est
juliette.fredy@etu.u-pec.fr

**Nawel Ghazal**
IAE Paris-Est
nawel.ghazal@etu.u-pec.fr

**Samy Marchetti**
IAE Paris-Est
samy.marchetti@etu.u-pec.fr

**Résumé :**

Dans le cadre du projet tuteuré *Global Minetest Technical and Educational Expertise Center* de l'IAE Paris-Est, notre objectif est de concevoir un environnement intégré pour la formation au management des systèmes d'information en nous appuyant sur trois logiciels : le jeu vidéo Minetest, l'*Enterprise Resource Planning* (ERP) Dolibarr et le Système d'Information Géographique (SIG) QGIS. Après une présentation de ces logiciels, nous détaillons l'état d'avancement du projet ainsi que les travaux qui devront être entrepris afin de réaliser un premier prototype entièrement gratuit et *open source*.

**Mots-clefs :**

Management des systèmes d'information, Ludopédagogie, Minetest, Intégration, *open source*

# 1 GLOBAL MINETEST TECHNICAL AND EDUCATIONAL EXPERTISE CENTER

## 1.1 CONTEXTE ET GENESE DU PROJET

Le projet *Global Minetest Technical and Educational Expertise Center* est un projet universitaire porté par des enseignants et étudiants de l'IAE Paris-Est (Université Paris-Est Créteil, UPEC). Ce projet tuteuré s'inscrit dans le cadre de la montée en compétence des étudiants en informatique du Parcours Informatique et Management[1]. Au cours de cette année universitaire, l'objectif principal de notre travail a été de réaliser un module afin de récupérer certaines données du jeu *open source* Minetest. Après récupération de ces données, nous devions les migrer dans un outil externe pour pouvoir les exploiter dans un dispositif pédagogique innovant de formation au management des systèmes d'information. Plus précisément, nous devions exporter les coordonnées géographiques des avatars du jeu, exporter les matériaux portés par l'avatar (inventaire) et exporter les informations liées à l'orientation du corps de l'avatar ainsi que ses mouvements. Un travail de recherche et de développement de modules en langage de script Lua[2] a d'abord été nécessaire pour mener à bien ce travail. Puis, les recherches se sont orientées dans les deux progiciels destinés à recevoir les données du jeu : l'*Enterprise Resource Planning* (ERP) Dolibarr[3] et le Systèmes d'Information Géographique (SIG) QGIS[4]. Dolibarr a été choisi pour sa simplicité et son intuitivité. Par ailleurs, l'un de nos critères était la gratuité du logiciel, nous souhaitions développer notre projet à partir d'un logiciel libre. Il s'agissait là également d'un moyen de s'assurer de la pérennité et de la liberté de notre projet. Enfin, Dolibarr est l'ERP utilisé par le parcours Informatique & Management de l'IAE Paris-Est. Les retours étaient très positifs et le caractère modulable de Dolibarr a fait la différence par rapport à d'autres ERP plus monolithiques. QGIS a également été choisi pour sa gratuité et sa disponibilité en *open source*. L'équipe ne connaissant pas l'outil, cela a également été l'occasion de le tester. Cet article présente l'état d'avancement de la connexion entre le jeu Minetest et l'ERP Dolibarr ainsi que celui concernant Minetest et QGIS afin d'effectuer un suivi en temps réel des avatars[5].

## 1.2 MINETEST

[1] Présentation du Parcours Informatique & Management : https://bit.ly/3E9nMU3.
[2] Présentation du langage de programmation Lua : https://www.lua.org/about.html.
[3] Présentation de l'ERP Dolibarr : https://www.dolibarr.org/#features.
[4] Présentation du SIG QGIS : https://www.qgis.org/fr/site/about/index.html.
[5] Cette idée a déjà fait l'objet d'un projet tuteuré, SICRAFT. Nos camarades disposaient alors de MinecraftEdu qu'ils ont réussi à connecter à Dolibarr (Lépinard, 2016 ; Albessard et *al.*, 2016 ; Sarda & Lépinard, 2018). SICRAFT a remporté le 2e Prix du concours innovation du groupe ADIS en 2017.

Minetest est un jeu vidéo multiplateforme, gratuit et disponible en *open source*[6]. Le jeu s'inspire principalement de Minecraft mais il dispose d'une API de *modding* (une interface de programmation). Conçu pour Windows, Linux, Android, macOS et FreeBSD, il s'agit d'un jeu de construction où les joueurs n'ont pas d'objectifs précis, aucun scénario n'est préétabli. Deux modes de jeu s'offrent aux joueurs : un premier mode où le joueur rassemble dans son inventaire un ensemble de matières premières. Il s'agit du mode survie. Le joueur ne peut construire au-delà de ce qu'il a récolté. Le second mode permet quant à lui de disposer d'un nombre illimité de matières premières, il s'agit du mode créatif. Les possibilités de ce jeu sont immenses, les joueurs peuvent combiner les blocs récoltés pour construire des structures. Le monde virtuel de Minetest se compose d'environ 62 000 blocs de côté (-30 912, 30 927) aux textures variées (du bois, des pierres, de l'herbe, des fleurs, etc.). Les possibilités dans le *gameplay* sont nombreuses, le joueur peut construire, marcher, voler, se téléporter, passer au travers des murs. Il peut casser les blocs de son choix, les repositionner ailleurs (*mining*). Il peut aussi se fabriquer et améliorer des outils pour récolter de nouvelles matières premières (*crafting*). Le joueur dispose d'un inventaire limité à 99 objets et peut fabriquer des coffres pour entreposer ses outils. En résumé, Minetest est un jeu vidéo de type *sandbox* (c'est-à-dire qu'il n'y a pas d'objectifs prédéterminés), offrant une grande liberté créative aux joueurs. Sa grande flexibilité et sa disponibilité en open source en font une alternative populaire à d'autres jeux comme Minecraft.

# 2 CONNEXION ENTRE MINETEST ET DOLIBARR

## 2.1 PRESENTATION DE L'ERP DOLIBARR

Dolibarr est un progiciel de gestion intégré et de gestion de la relation client *open source* pour les entreprises de toutes tailles. S'agissant d'un logiciel modulable, il s'adapte parfaitement à notre projet universitaire. Dans ce contexte d'importation de données Minetest, Dolibarr propose un ensemble de fonctionnalités intéressantes. En effet, ce logiciel repose sur plusieurs modules permettant aux utilisateurs de gérer différentes catégories de tâches indépendamment. Il existe d'ores et déjà un ensemble de modules Dolibarr pré-intégrés au logiciel permettant la gestion intégrée d'une entreprise tels que « Stocks », « Achats et approvisionnements », « Facture et paiement », etc. Dolibarr étant initialement conçu pour les entreprises, il convient de l'adapter aux besoins et spécificités du projet Minetest par le développement d'un module spécifique à l'importation de données du jeu afin de gérer ces

[6] Site Internet du projet Minetest : https://www.minetest.net/.

données provenant et surtout de les exploiter : ces données seront centralisées et accessibles à toute personne ayant les droits. Dolibarr est déjà utilisé au sein du Parcours Informatique et Management de l'UPEC. La subtilité du projet Minetest tient donc dans la création d'un module dédié et adapté au projet. Il existe dans Dolibarr un service « Intégration, Développement » qui comporte quelques fonctionnalités importantes dans le cadre de ce travail de recherche :

- La fonctionnalité d'import / export : Il s'agit d'un assistant d'importation ou d'exportation qui permet d'aider à importer ou extraire des données dans ou à partir d'une application. L'assistant d'importation permet d'importer un grand nombre de données à partir de n'importe quelle application externe via un fichier *comma-separated values* (CSV) ou Excel. Il est possible d'effectuer les importations en mode insertion ou mise à jour et le mode exportation permet d'exporter toutes les données de l'application dans un fichier CSV ou Excel. Dans notre cas, nous sommes plus amenés à importer des données qu'à en exporter. Nous verrons donc plus en détails les possibilités offertes par l'importation de données dans la suite de l'article.
- Le module *Builder* pour développeurs : Il s'agit d'un studio *low code* pour aider les développeurs à créer une application complète en quelques minutes. Nous verrons dans la partie suivante comment nous exploitons cette fonctionnalité pour créer un module spécifique à Minetest puis comment nous pouvons l'améliorer.
- Fonctionnalité Connectivité et Interfaces : Cette fonctionnalité peut être intéressante car elle permet de connecter Dolibarr à d'autres services externes comme PostgreSQL. Il faut noter que PostgreSQL n'est pas intégré dans les logiciels partenaires de base mais des membres de la communauté ont pu rendre la connexion entre ces deux plateformes possibles en créant des modules.

### 2.2 CREATION DU MODULE SPECIFIQUE

Comme mentionné précédemment, le Module *Builder* intégré à Dolibarr est un studio *low code* permettant la création d'un module ou d'une application de manière très simplifiée. Nous avons donc fait usage de ce service afin de générer un module pour Minetest. La première étape de ce travail était donc de déterminer le type de données que nous souhaitions importer dans Dolibarr.

Dans un premier temps, nous avons choisi d'importer les données liées à l'inventaire de l'avatar. Lorsque ces données seront bien implémentées dans l'ERP, nous pourrons ajouter les contenus des coffres et d'autres données. Pour l'import des données de l'inventaire des

joueurs, il est d'abord nécessaire de récupérer ces dernières afin de pouvoir les exporter du jeu. Pour ce faire, nous avons alors programmé un *addon* en Lua permettant de choisir un joueur et de récupérer l'ensemble de son inventaire. Ces données s'affichent alors dans la console Minetest à l'appel de la commande *listinventory* avec, en paramètre, le pseudonyme du joueur en question, et sont transférées simultanément dans un fichier CSV généré par le code.

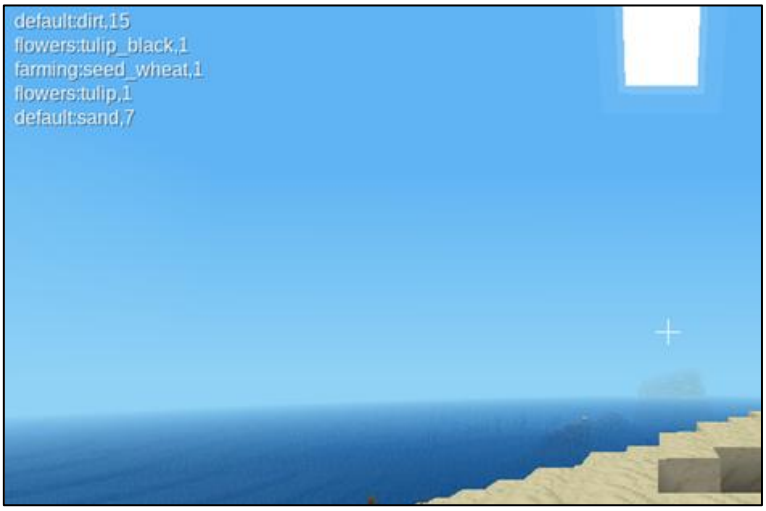



*Figure 1. Résultat en sortie de la commande listinventory sous Minetest.*

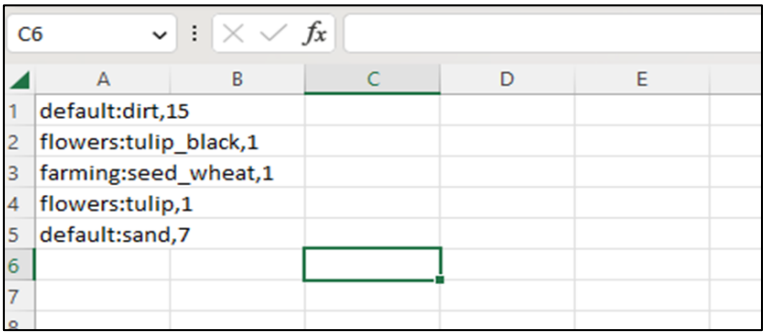



*Figure 2. Fichier CSV généré par la commande "listinventory".*

Ce modèle de fichier CSV nous permet ensuite d'exporter ces données vers différents logiciels tels que Dolibarr. En effet, comme nous pouvons le voir dans la documentation officielle, l'assistant d'import-export mis à disposition par le logiciel nous permet d'importer des données à partir de n'importe quel fichier CSV ou Excel. Une fois le module Minetest créé, il s'agira de faire un mappage entre les champs du fichier source (donc celui généré par notre code Lua dans notre cas) et les champs Dolibarr (ceux du module). Nous avons donc résolu le problème de récupération de données et avons un script CSV. Nous avons maintenant besoin d'un modèle de tables pour stocker les données en *Structured Query Language* ou SQL (tables : JOUEUR et ITEM) :

- Nom de la table : JOUEUR
- Champs :  ID_JOUEUR(PK), PSEUDO_JOUEUR
- Nom de la table : ITEM
- Champs : ID_ITEM(PK), NOM_ITEM, QUANTITE_ITEM, #ID_JOUEUR(FK)

Un joueur peut posséder plusieurs items mais un item ne peut être possédé que par un joueur : ID_JOUEUR est une clé étrangère (FK) de la table ITEM. Nous pouvons noter que le *Module Builder* de Dolibarr permet une incrémentation automatique des valeurs des clés primaires lors de la création de tables SQL. Nous utilisons cette auto-incrémentation pour les champs ID_JOUEUR et ID_ITEM. Si cela n'était pas faisable de manière automatique, nous aurions pu ajouter une variable globale dans le module faisant office de compteur pour chaque item et chaque joueur afin d'implémenter les clés primaires. Maintenant que tout est en place, nous pouvons passer à sa création concrète. Nous pourrions légitimement être amenés à nous demander pourquoi créer un module complet pour Minetest sous Dolibarr tandis que nous pourrions tout simplement importer les données dans un module déjà existant. Il faut savoir qu'il existe sous Dolibarr des modules Apache permettant d'ajouter un Système de Gestion de Bases de Données (SGBD), c'est ce que nous voulons faire.

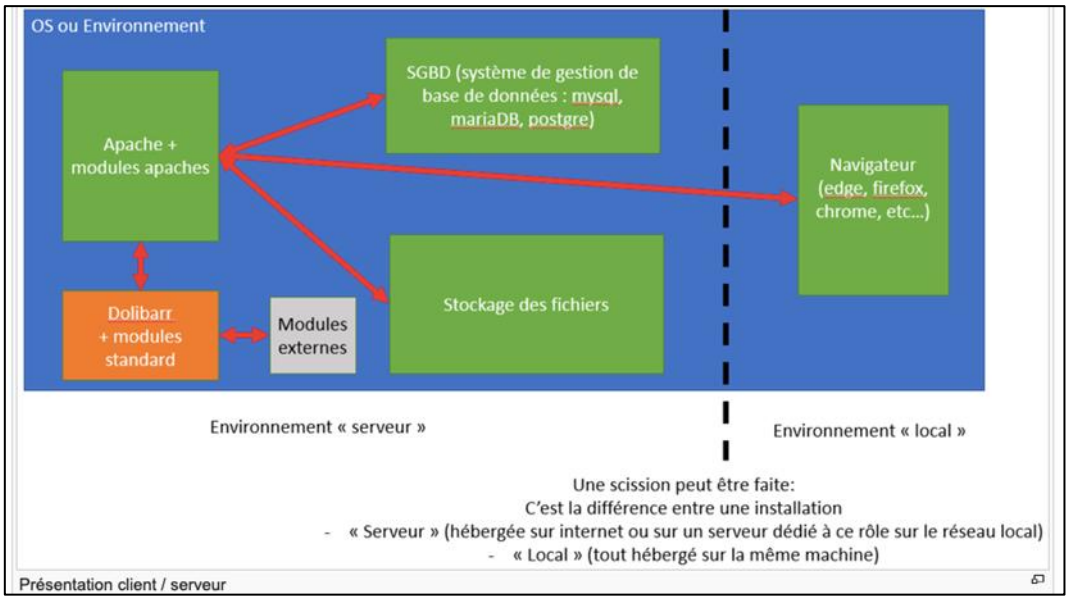



*Figure 3. Architecture de Dolibarr selon wiki.dolibarr.org.*

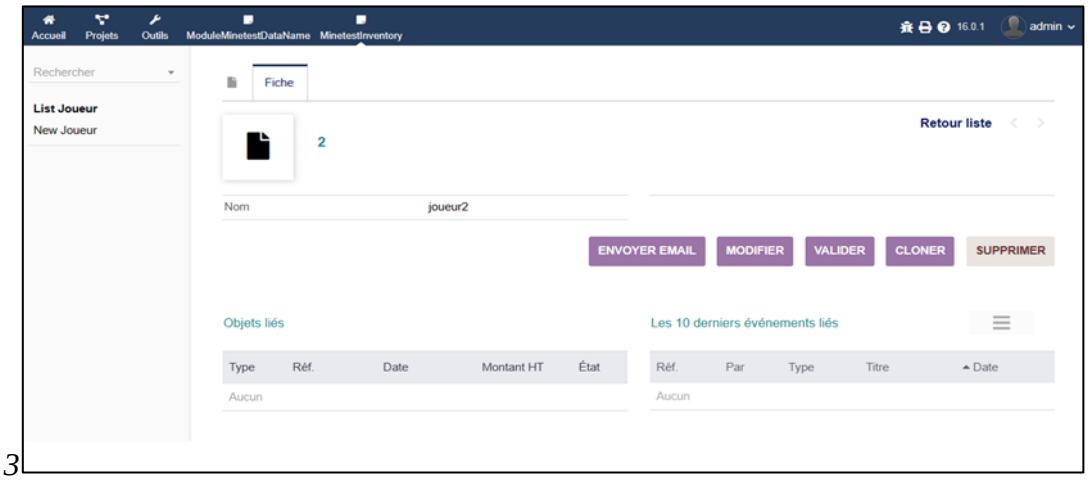
La création du module permet entre autres d'ajouter un menu avec des entrées, d'afficher la liste des joueurs, ou encore d'exécuter du code de manière automatique après une action sous Dolibarr. Cette dernière fonctionnalité peut s'avérer très utile si elle peut permettre, par exemple, de rafraîchir l'importation de données. On peut imaginer un bouton « mettre à jour »

qui permettrait d'ajouter de nouvelles données à la base de données si le script généré par le *mod* Lua a subi des modifications.

### 2.3 MISE EN PLACE CONCRETE DU MODULE

Nous avons installé Dolibarr sur notre machine afin d'avoir les droits d'administrateur. Seuls les administrateurs ont la possibilité de créer des modules et de les mettre à disposition des autres utilisateurs. Nous avons alors créé un nouveau module appelé *MinetestInventory* afin d'essayer d'implémenter notre solution. Nous avons créé deux tables dans ce module. Sous Dolibarr, les tables SQL sont appelées « objets ». Créer un nouvel objet revient donc à créer une table. Nous avons créé pour chaque objet les champs associés. Nous avons actuellement deux objets à disposition : JOUEUR et ITEM. Comme vu plus haut, l'ID_JOUEUR est une clé étrangère de la table ITEM, néanmoins, nous avons rencontré un problème au sujet de la clé étrangère : nous n'arrivons pas encore à enregistrer ce champ dans la table ITEM. De plus, contrairement aux données des joueurs qui apparaissent dans le module, nous n'avons pas encore accès aux données des items à partir de l'interface. Le but est alors de réussir à lier les deux objets via la notion de clé étrangère, et finalement, de faire en sorte de pouvoir accéder aux données des items également via l'interface du module. Lorsque ces fonctionnalités seront ajoutées au module, nous pourrons le personnaliser et gérer l'esthétique de l'interface afin de la rendre plus facilement utilisable et plus intuitive. Voici quelques pistes d'amélioration en ce qui concerne l'ergonomie. Nous pouvons en premier lieu nous concentrer sur l'affichage de la liste des joueurs : la liste s'affiche selon les clés primaires des joueurs et non selon leurs noms. Avoir les deux seraient une meilleure solution. On peut ensuite envisager d'ajouter le logo de Minetest en tant que logo du module. Pour ce faire, on peut le remplacer en indiquant le chemin vers le logo en question dans le code de description du module. Pour le moment, le module permet de voir la liste des joueurs mais ces derniers s'affichent selon leurs clés primaires et non selon leurs noms, ce qui s'avère peu pratique mais reste très logique étant donné que les clés primaires sont des identifiants uniques pour chaque objet. Cette manière de les afficher permet donc d'être sûr de ne pas avoir de doublons. En effet, si deux joueurs avaient le même nom, il serait difficile de les distinguer, ou de savoir s'il s'agit d'une erreur.


*Figure 4. Module MinetestInventory sous Dolibarr : liste des joueurs affichée via leur clé primaire.*




3

*Figure 5. Module MinetestInventory sous Dolibarr : affichage des informations.*

On peut voir qu'il reste un certain nombre d'informations inutiles dans le cadre du projet Minetest. Il y avait plus d'informations que cela initialement mais nous en avons supprimé la plupart. Celles-ci sont un peu plus compliquées à supprimer car elles sont intégrées à la fiche du joueur, mais nous allons les supprimer en modifiant le code via le module *Builder* :

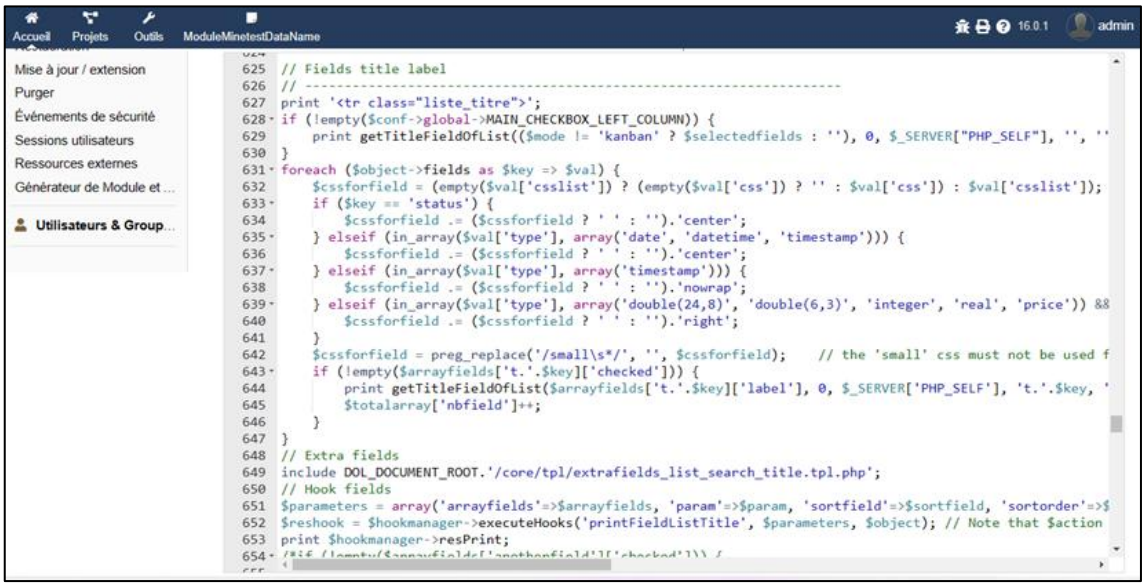


*Figure 6. Module MinetestInventory sous Dolibarr : code généré par le module builder.*

Le travail mené permet de mettre en évidence les axes d'améliorations :

- Ajouter le logo Minetest au module (via le code de description du module).
- Lier les tables JOUEUR et ITEM via une clé étrangère.
- Modifier les différentes informations qui s'affichent sur la fiche des joueurs.
- Ajouter les noms des joueurs (en laissant les clés primaires affichées afin d'éviter les problèmes de doublons) dans la liste des joueurs.

Une fois que ces tâches seront réalisées, notre module sera prêt à être testé.

# 3 CONNEXION ENTRE MINETEST ET QGIS

## 3.1 PRESENTATION DU SIG QGIS

QGIS est un système d'information géographique professionnel et *open source*, distribué sous licence publique générale GNU. Il permet de créer et visualiser des données géospatiales. Cet outil a été choisi pour visualiser en temps réel les joueurs présents dans Minetest comme actuellement (mais de manière relativement succincte) avec le module *Mapserver*. QGIS propose de multiples affichages : des cartes géologiques, des cartes de rues, des photographies satellites ou encore des cartes de visualisation de fonds marins. L'intégration des données de Minetest doit se faire en trois temps :

1. L'exportation des coordonnées des joueurs depuis Minetest
2. La transformation en une base de données PostgreSQL.
3. L'importation de cette base de données dans QGIS.

## 3.2 EXPORTATION DES COORDONNEES ET IMPLEMENTATION DANS QGIS

Pour exporter les coordonnées géographiques des joueurs de Minetest, il est nécessaire de créer un module en Lua. Ce module présente deux fonctionnalités : récupérer les coordonnées d'un joueur spécifique ou simplement récupérer les coordonnées de tous les joueurs présents dans le serveur de jeu. Ensuite, ces coordonnées seront automatiquement insérées dans un fichier SQL.

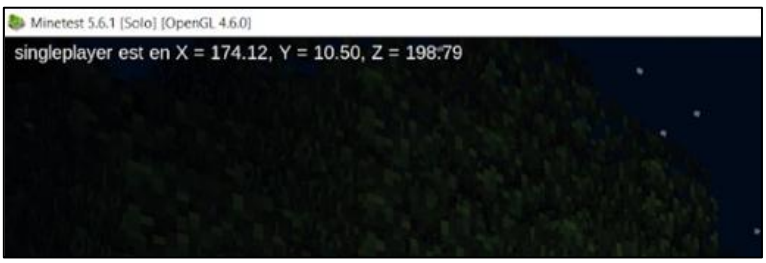
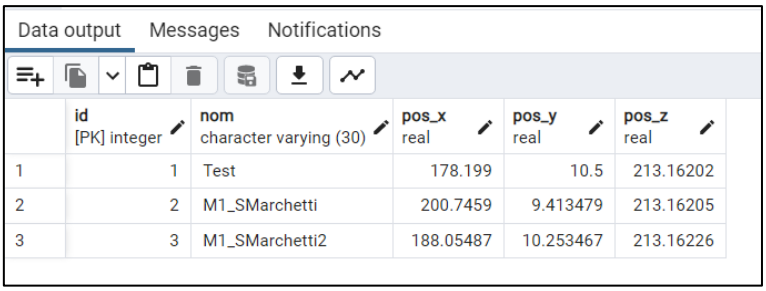



*Figure 7. Résultat de la commande « returnpos ».*




| | id [PK] integer | nom character varying (30) | pos_x real | pos_y real | pos_z real |
|---|---|---|---|---|---|
| 1 | 1 | Test | 178.199 | 10.5 | 213.16202 |
| 2 | 2 | M1_SMarchetti | 200.7459 | 9.413479 | 213.16205 |
| 3 | 3 | M1_SMarchetti2 | 188.05487 | 10.253467 | 213.16226 |

*Figure 8. Exemple de table dans PostgreSQL.*

Une fois l'exportation des coordonnées achevée, nous pouvons passer à l'implémentation dans QGIS. QGIS supporte de manière native l'import de base de données. Il est possible d'afficher les données contenues dans celle-ci sans avoir à les extraire. Pour cela, il faut connecter la base de données PostgreSQL à QGIS depuis la Gestion des Sources. Il faut renseigner l'adresse IP sur laquelle la base de données se situe (en local pour notre cas), le port par défaut, ainsi que le nom de la base et enfin s'authentifier avec le mot de passe PostgreSQL. C'est à partir de cette étape que nous avons rencontré les principales difficultés.


*Figure 9. Gestionnaire de source PostgreSQL.*

La base de données précédemment créée est bien connectée (QGIS peut vérifier la connexion avec la base de données), mais aucune donnée n'apparaît sur la fenêtre de Gestion des sources. En effet, les données renvoyées (des réels) ne sont pas reconnues par QGIS comme des données géographiques. Deux types de données sont acceptées par QGIS : des données vecteurs (exemple : *shapefile*) et des données raster (exemple : *geotiff*). De plus, il n'y a pas d'affichage : aucune des données fournies par la base de données ne constitue de couche, terme employé par QGIS pour désigner un fond de carte. Nos pistes de travail sont actuellement les suivantes : il s'agirait en premier lieu de récupérer la carte affichée par le module *Mapserver* et l'afficher en tant que fond de carte. Enfin, il s'agira de convertir les

coordonnées de position obtenues depuis Minetest en coordonnées géographiques reconnues par QGIS. Grâce aux spécificités de l'*open source*, QGIS bénéficie d'une large communauté sur laquelle il est possible de compter pour solutionner les problématiques techniques rencontrées.

## 4 CONCLUSION

L'objectif de cet article était de présenter le travail réalisé cette année pour connecter le jeu *open source* Minetest aux progiciels Dolibarr et QGIS. Nous avons montré toutes les étapes nécessaires à la création d'un module dans Dolibarr pour exporter les données de l'inventaire. Nous avons aussi exploré l'utilisation de QGIS pour récupérer les positions géographiques des joueurs lorsqu'ils sont connectés. L'objectif pédagogique de cette exportation de données est le suivant : concevoir un environnement intégré pour la formation au management des systèmes d'information. Les données collectées pourront ainsi être exploitées dans différents scénarios pédagogiques. Des pistes d'améliorations pour Dolibarr notamment ont pu être développées comme le fait de lier les tables JOUEUR et ITEM via une clé étrangère et de modifier le code généré par le *Module Builder* pour enlever les informations inutiles. Pour QGIS, nous devons poursuivre le travail pour générer les positions des joueurs et les traiter dans QGIS. Ce travail très enrichissant nous a permis de développer nos compétences en informatique et montre les nombreuses possibilités du projet. Minetest n'est pas seulement un jeu vidéo, cela va bien au-delà. Les possibilités sont immenses. Le projet *Global Minetest Technical and Educational Expertise Center* va pouvoir continuer son développement. Nous prévoyons de poursuivre le travail en créant des scripts Lua qui exporteront les axes et mouvements des personnages. Pour conclure cet article, nous pouvons souligner le potentiel du projet et notre engouement à poursuivre son développement.

## 5 REMERCIEMENTS

Nous remercions les autres membres de l'équipe du projet *Global Minetest Technical and Educational Expertise Center* : Lukas Dreyer (L1), Léni Beaulaton (BUT 1), Lahna Ould-Mohand (L1), Carine Ho (M1) et Julien Menier (M2).

## 6 REFERENCES

Albessard, C., Conrard, J., Va Duong, N. T., Hajji, Z., Hang, J. N., Lépinard, P., Lohez, G., Messoussi, M., Sarda, B. (2016). Minecraft and I.S.: A winning combinaison for education, *Virtual reality international conference*, Laval.

Lépinard, P. (2016). SICRAFT, une plateforme de serious gaming pour les enseignements aux systèmes d'information. *34e Congrès INFORSID*, Grenoble.

Sarda, B., Lépinard, P. (2018). Apprentissage des SI via le jeu vidéo. *3Festival Pas Sage en Seine*, Choisy-le-Roi.