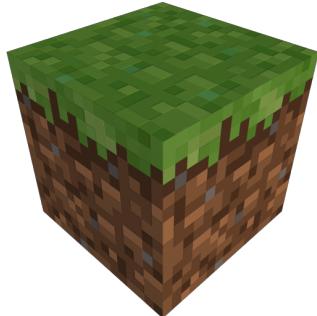


Emile Bex
Bastien Angénieur
Pierre-Louis Duthoit
Louis Derollepot
Marie Page

Projet Tuteuré S2

La programmation Minecraft



N° du sujet	227
Titre	Programmation Minecraft
Tuteur	Amélie Cordier
Membres du groupe	Marie Page
	Pierre-Louis Duthoit
	Louis Derollepot
	Bastien Angénieur
	Emile Bex
Résumé du projet en quelques lignes	Nous souhaitons faire un inventaire de tous les différents composants et portes logiques reproductibles dans Minecraft, qu'on implémentera ensuite dans un mod, afin de facilement les reproduire.
Technologies envisagées pour la réalisation du projet	Les technologies nécessaires sont le jeu Minecraft, sur lequel nous testerons les composants qui seront implantable, et essayerons de les assemblés pour réaliser d'autres composants (additionneur 1 bit, bistable...), qui reenrichiront le catalogue de composant du mod. Afin de créer le mod, nous utiliserons Forge, une API de Minecraft sous Java permettant de directement coder notre mod.
Méthode de gestion de projet envisagée	Nous utiliserons un planning (GanttProject), sur lequel nous nous assignerons des tâches et une date limite pour le faire, avec une estimation du temps nécessaire. Pour l'assignation des tâches, nous utiliserons Trello. Nous utiliserons également des outils de collaboration et de gestion de version comme GitHub.
Quelle est, selon vous, la probabilité de poursuivre ce projet au prochain semestre ?	Même si le sujet pourrait être poursuivi l'année prochaine, les difficultés qu'on a eu notamment de gestion de projet nous donnent envie de changer de sujet pour l'année prochaine. L'intérêt pour ce projet reste assez limité à notre sens, car il ne répond pas à un besoin premier et ne pourrait pas être concrétisé dans le cadre d'une entreprise ou dans la vue d'une monétisation. Cependant il pourrait être très intéressant de travailler sur un tel projet, car il permettrait de largement développer nos compétences en analyse de programme et en Java.
Résumé des difficultés rencontrées pour la réalisation de ce projet (quelques lignes)	Difficultés à reproduire un transistor, donc obligé d'utiliser un composant de plus haute abstraction. Impossibilité d'utiliser l'équivalent d'une horloge. Difficultés à appliquer les méthodes de gestion de projet à un sujet comme celui-ci.

I. Introduction

Minecraft est un jeu développé par Mojang et de type "sandbox" (bac à sable), dans lequel le joueur a une liberté totale et peut construire tout ce qu'il imagine grâce à des cubes de matières qu'il peut poser ou enlever. Le monde est générée de façon procedural, c'est à dire que le niveau est généré à la volée, au fur et à mesure que le joueur explore, en suivant un certain algorithme. Minecraft dispose également d'un mode "survival", dans lequel les ressources sont limitées et le joueur doit survivre dans un monde hostile.

Un des matériaux de Minecraft est la redstone qui a la particularité d'agir comme un fil électrique, ce qui apporte une nouvelle facette au jeu. En utilisant certains objets déjà présent dans celui-ci (torche de redstone, plaque de pression, levier, bouton, piston...), il est très facile de construire des circuits électroniques, que ce soit dans un but récréatif ou éducatif.

Mais sachant que les constructions se font bloc par bloc, Minecraft ne permet pas de facilement reproduire des constructions déjà créées au préalable. Si l'on veut créer plusieurs fois une même construction, comme une porte logique par exemple, il faut à chaque fois la construire soi-même.

Comment pourrait-on simplifier cette tâche afin de proposer aux joueurs l'utilisation de portes logiques facilement ?

L'idée est de créer un mod. Un mod est une modification du jeu original qui vient s'ajouter à celui-ci en ajoutant des fonctionnalités ou en modifiant des éléments du jeu. Ce mod mettrait à disposition du joueur une interface graphique, un menu transparent qui s'ajoutera au jeu minecraft et qui permettrait de construire automatiquement des composants logiques et de les faire disparaître afin de créer facilement des circuits logiques, des jeux ou des structures dans Minecraft.

La création d'une porte logique dans Minecraft requiert du temps et de l'expérience, surtout sur les portes les plus compliquées. Avec notre outil, l'utilisateur pourra se simplifier la tâche.

Le menu sera dirigé par 5 touches : 2 directionnelles (haut & bas) une touche de sélection, une touche de retour et une touche pour faire apparaître le menu.

Il sera disponible à tout moment dans le jeu et n'interrompra pas le fonctionnement du jeu, afin de faciliter la création. (*cf Figure 1*)

L'utilisateur aura également la possibilité d'enregistrer des nouveaux composants qu'il aura au préalable construit et de supprimer ceux dont il ne se sert plus.

Ce menu pourrait aussi permettre par exemple d'accéder à de la documentation sur les portes logiques et leur utilisation dans Minecraft.

Pour créer ce mod nous aurons besoin de nous familiariser avec les outils de développement Minecraft, tel que le logiciel Mcreator ou l'API Minecraft pour Java ou JavaScript, ainsi que comprendre le fonctionnement des différentes portes logiques afin de les implémenter. Minecraft est entièrement programmé en Java, ce qui nous facilitera la tâche et nous permettra d'exploiter nos connaissances dans ce langage.



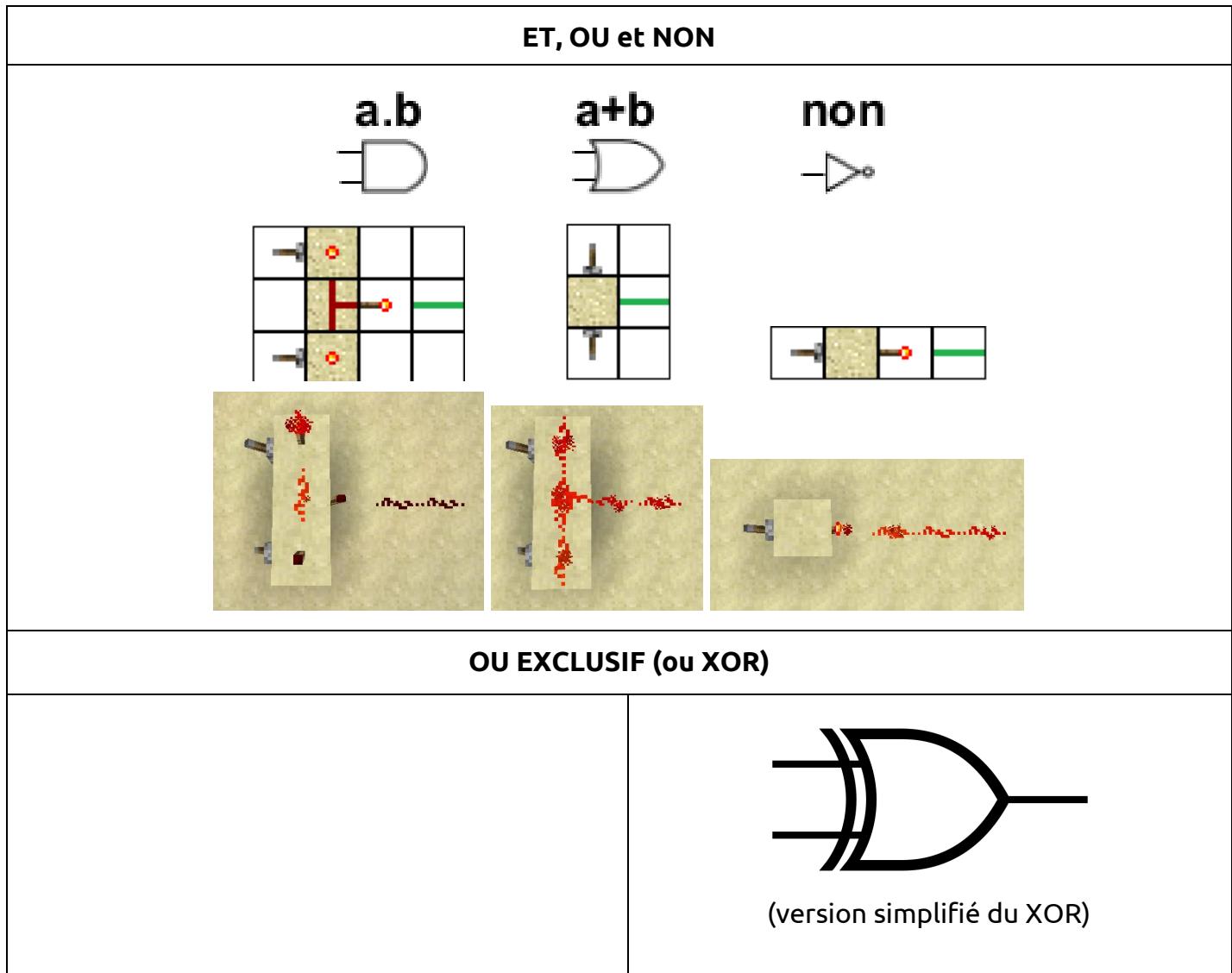
Figure 1 : Image d'illustration du mod

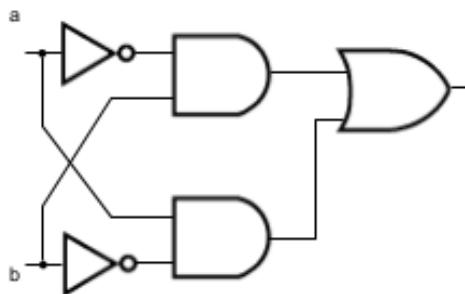
II. Présentation et explication des différentes portes logiques

Pour la reproduction des portes logiques sur Minecraft, on a besoin de 3 composants essentiels: le ET, le OU et le NON. Ceux-ci peuvent ensuite être utilisés pour créer des composants de plus haute abstraction, tels que le OU EXCLUSIF (XOR), ou d'autres, comme les additionneurs, les comparateurs, les bistables...

Cette liste serait donc la liste des premiers composants à planter dans le mod.

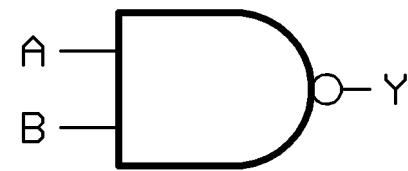
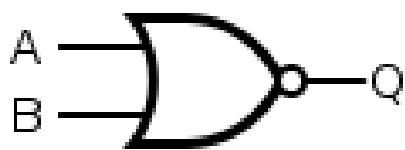
Dans les schémas suivants (*cf Figure 2*), les leviers peuvent être remplacés par de la redstone afin de relier plusieurs composants entre eux.





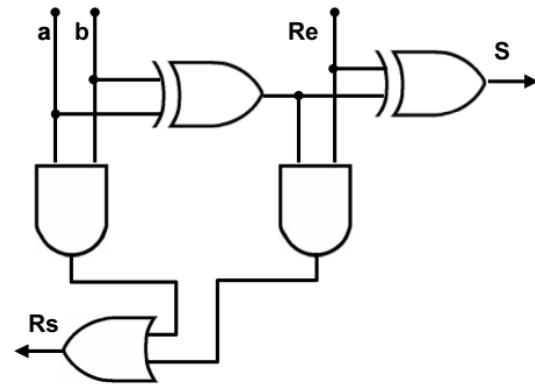
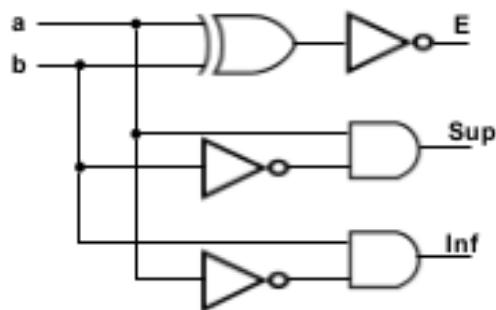
NOR (OU suivi d'un NON)

NAND (ET suivi d'un NON)



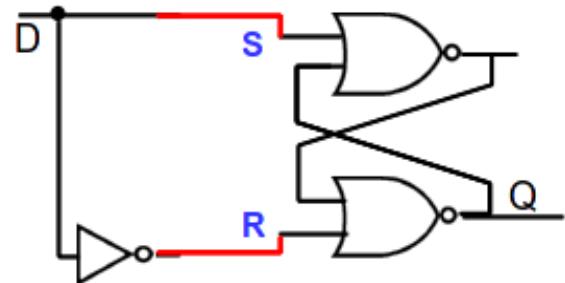
COMPARATEUR 1 BIT

ADDITIONNEUR 1 BIT



BISTABLE RS

BISTABLE D



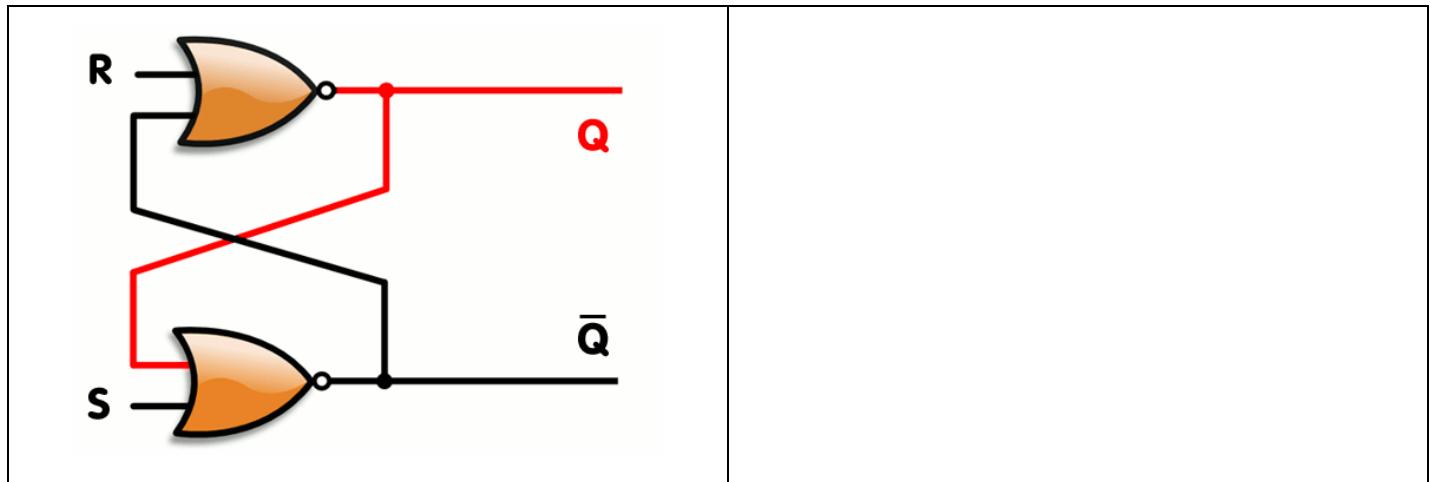


Figure 2 : Schéma de différentes portes logiques

III) Techniques d'implémentation et matériel utilisé

Pour créer notre mod, nous avons besoin d'outils : d'une version de Minecraft compatible avec Minecraft Forge (cité plus tard), de Minecraft Coder Pack (MCP), qui est un ensemble de scripts et d'outils permettant de décompiler, modifier et de recompiler les classes de Minecraft, et enfin, d'un IDE pour créer le mod. En l'occurrence nous avons choisi d'utiliser Eclipse puisque l'ensemble des tutoriels de création de mods que l'on peut trouver sur internet utilisent pour la large majorité Eclipse. Nous aurons donc besoin de Minecraft Forge; celui-ci va vous permettre d'installer de nombreux mods. Il s'agit tout simplement d'une API qui permet aux développeurs de concevoir des mods sur Minecraft et qui est un outil indispensable à la conception d'un mod.

Pour concevoir un mod, il faut tout d'abord installer Minecraft Forge qui va nous permettre de créer notre mod sous Eclipse puis en faisant un simple "Run Client", Forge va recompiler Minecraft en utilisant les scripts de Minecraft Coder Pack (*cf Figure 3*). Un client Minecraft va alors s'ouvrir et l'on pourra tester notre mod.

Pour construire et rendre le mod utilisable par d'autres personnes, il faudra utiliser un script disponible dans le dossier de Forge qui permet de créer le mod sous la forme d'un fichier .jar que l'utilisateur, pour utiliser le mod, pourra mettre dans son dossier ".\AppData\Roaming\.minecraft\mods".

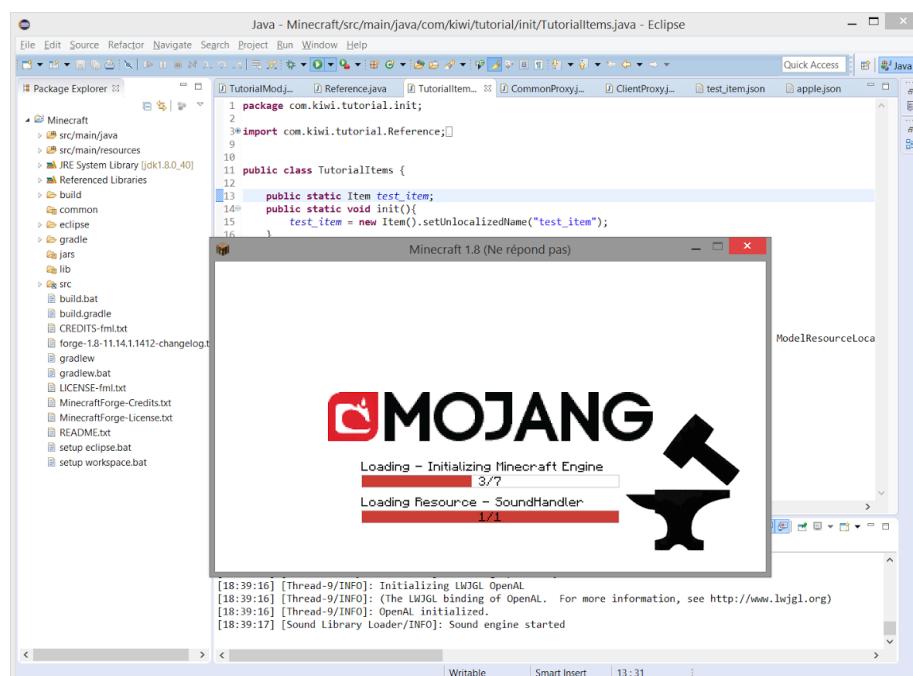


Figure 3 : Minecraft Forge en train de recompiler Minecraft à l'aide de MCP

IV) Analyse de la charge de travail et des problèmes posés.

Nous avons d'abord répartit le projet en en différents blocs, à la manière d'un projet informatique : apprentissage des technologies, codage fonctionnel, codage graphique, implémentation fonctionnelle, implémentation graphique et les tests. Ne connaissant pas le temps réel que cela nous prendrait, nous utiliserons des durées relative en fonction de x , x étant la durée du bloc le plus court.

La première partie consiste à l'apprentissage des technologies. C'est-à-dire apprendre à créer un mod, en suivant des tutoriels, en faisant des tests, en s'appropriant les bases de l'API Minecraft, etc. Nous avons estimé cette étape fondamentale comme durant $2x$. Vient ensuite le codage fonctionnel, c'est-à-dire la programmation de toutes les fonctionnalités du mod : faire apparaître les constructions, ajouter des constructions à l'inventaire, faire disparaître les constructions, etc. Cette étape étant la plus technique, nous avons estimé qu'elle prendrait $5x$ et serait le bloc le plus long. Celui-ci est suivi du codage de l'interface graphique (le menu et la selection des composants à faire apparaître) que nous avons estimé à $4x$. Ces deux blocs sont suivis de tests que nous avons estimé à x . Restent les implémentations des deux codes (l'interface graphique devant être implémentée obligatoirement après la partie fonctionnelle), que nous avons estimé chacun à $3x$ et pour finir, les tests finaux, que nous avons estimé à $2x$.

Le temps total serait donc de $13x$. Si l'on prend le temps de travail personnel suggéré par le Programme Pédagogique National, nous sommes censé passé environ 160 heures sur le projet. En majorant le temps de travail personnel à 169 heures, on peut donc estimer que $x = 13$ heures.

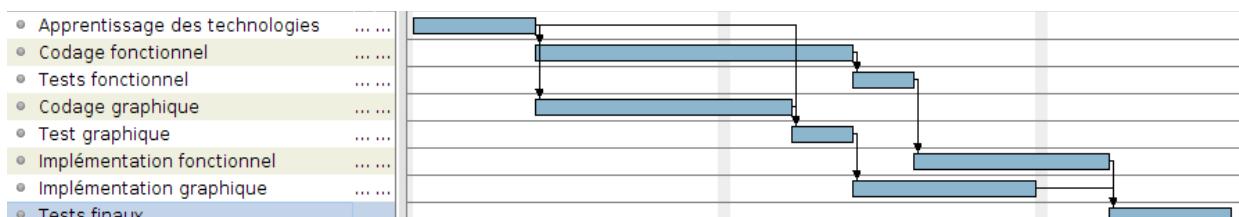


Figure 4 : Diagramme de Gantt simplifié (durées relatives)

Mais créer un mod n'est pas si simple que ça et pose parfois des difficultés qui nécessitent de prendre certaines mesures.

La difficulté majeur, est de s'approprier le code de Minecraft et de comprendre comment il est organisé, d'autant plus que depuis la version 1.8 la création de mod est devenu beaucoup compliquée.

Une autre difficulté concerne l'IDE. En effet, la majorité des tutoriels sont expliqués avec Eclipse car Minecraft Forge utilise des outils compatibles avec celui-ci. Cela nécessite donc de nous adapter à cet IDE.

Un autre problème est celui des versions de Minecraft. Il ne faut jamais tenter d'installer un mod tant qu'il n'a pas été mis à jour pour correspondre au numéro de version de notre jeu. Même un changement mineur de version peut être suffisant pour causer des problèmes (plantages, corruption des sauvegardes...). Ceci pose donc problème au niveau de la mise en place du mod. De plus, l'outil que nous disposons pour décompiler Minecraft est compatible pour la version 1.8, cependant la dernière version de Minecraft disponible est 1.8.4. Notre mod sera donc accessible uniquement pour la version 1.8.

Les outils et les versions de Minecraft évoluent régulièrement, c'est pourquoi si l'on veut mettre à disposition un mod fonctionnel sur une longue période, nous devons mettre à jour notre mod régulièrement, ce qui demande du travail sur le long terme.

Sources:

<http://liris.cnrs.fr/mmrissa/>

<http://commons.wikimedia.org/wiki/File:Logic-gate-nand-us.png>

http://commons.wikimedia.org/wiki/File:XOR_ANSI.svg

<https://www.youtube.com/watch?v=VhOSL7rGb10>

<http://files.minecraftforge.net/>