Prog&Play : Introduction

Prog&Play est un jeu sérieux basé sur le jeu de Stratégie Temps Réel (STR) multijoueurs « Kernel Panic ». La partie suivante a pour objectif de vous familiariser avec ce jeu.

1 Description générale du jeu

Le jeu Kernel Panic place le joueur au coeur même d'un ordinateur. Au même titre qu'au jeu d'échec le joueur manipule des pions, des tours, des cavaliers, etc., dans Kernel Panic, le joueur manipule des bits, des octets, des pointeurs, des assembleurs, des sockets et un noyau.

Avant d'aller plus en avant sur la description du jeu Kernel Panic, vous noterez que ces derniers éléments ont une origine informatique par exemple un bit est l'abrévation du mot anglais ***bi**nary digi**t***, qui signifie « chiffre binaire ». Il est à la base de toute information numérique. Un byte (ou octet) est un groupement de 8 bits, un socket permet l'enfichage du microprocesseur sur la carte mère, un terminal diffuse des informations à l'utilisateur, l'assembleur est un langage informatique de bas niveau, le pointeur est une variable contenant une adresse mémoire et le noyau est la partie centrale d'un système d'exploitation.

Il est donc important de bien garder à l'esprit que la représentation de ces éléments dans le jeu sous forme d'une boule pour un bit (voir Fig. 2) ou d'une pyramide pour un byte (voir Fig. 3) ainsi que l'arbre de la figure 8 sont purement métaphoriques et ne traduisent en aucun cas la réalité physique de ces éléments.

Fig. 1 – Un Kernel

Fig. 2 – Un Bit

Fig. 3 – Un Byte

Fig. 4 – Un Pointer

Fig. 5 – Un Assembler

Fig. 6 – Un Socket

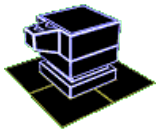

Fig. 7 – Un Terminal

Contrairement au jeu d'échec où l'objectif est d'immobiliser la pièce maîtresse de l'adversaire (à savoir le roi), dans Kernel Panic le joueur doit éliminer toutes les entités contrôlées par son adversaire. Pour ce faire, chaque entité a ses propres caractéristiques dont voici une brève description :

- Le Kernel (ou Noyau, voir Fig. 1) est l'entité maîtresse du jeu. Chaque joueur commence la partie avec cette unique entité. Il est fixe dans l'environnement du jeu et ne peut être déplacé mais permet de générer toutes les entités mobiles du jeu. Il possède une grande résistance avec 40000 points d'énergie.
- Le Bit (voir Fig. 2) est l'entité basique d'attaque. Mobile, rapide à construire, véloce et petit, il peut être généré par le Kernel et le Socket. Sa faible résistance (600 points d'énergie), son attaque basique (80 points de dégâts par attaque) et sa faible portée (256 unités de distance) rendent ses actions solitaires périlleuses.

– Le Byte (ou Octet, voir Fig. 3) peut être généré par le Kernel. Il est mobile, gros, fort et lent mais peut rivaliser avec de nombreux Bits en raison de sa résistance (15000 points d'énergie), de sa force (200 points de dégâts par attaque) et de sa portée (512 unités de distance).
– Le Pointer (ou Pointeur, voir Fig. 4) peut être généré par le Kernel. C'est une « artillerie » en raison de sa longue portée de tir (1400 unités de distance). Efficace contre les entités immobiles (4000 points de dégâts par attaque), il se trouvera rapidement en difficulté face à des entités mobiles en raison de sa lenteur de déplacement et de sa faible résistance (1000 points d'énergie).
– l'Assembler (ou Assembleur, voir Fig. 5) peut être généré par le Kernel. C'est une entité mobile de construction qui a la faculté de pouvoir recharger les entités endommagées. Il ne peut pas attaquer l'adversaire, il est lent et moyennement résistant (2000 points d'énergie).
– Le Socket (voir Fig. 6) est une « usine » qui peut être construit à l'aide de l'Assembler. Il est capable de générer des Bits à une fréquence moindre que celle du Kernel. Il dispose d'une résistance raisonnable avec 20000 points d'énergie.
– Le Terminal (voir Fig. 7) est un silo à missiles qui peut être construit à l'aide de l'Assembleur. Il peut libérer un missile toutes les 90 secondes qui produira 10000 points de dommage sur une large zone.

La figure 8 synthétise la hiérarchie de création des entités, ainsi le Kernel peut générer des Bits, des Bytes, des Pointers et des Assembler.

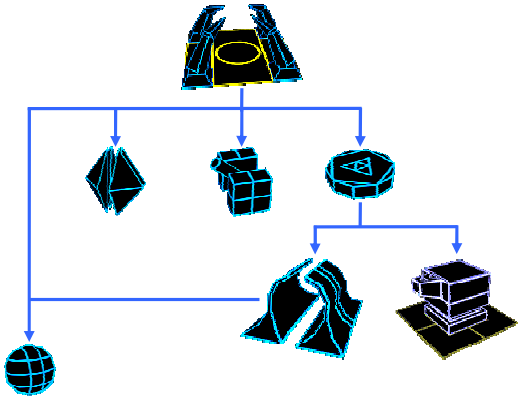

FIG. 8 – Hiérarchie de création des entités

2 Règles principales du jeu

Au même titre que les pièces d'un jeu d'échec se déplacent sur un plateau de jeu, les entités d'un jeu de STR évoluent sur une carte où les positions des entités sont repérées par des coordonnées x et y. Les STR tels que Kernel Panic se basent sur trois règles importantes : le « temps réel », le « brouillard » et le « contrôle indirect ». Le « temps réel » est opposé au « tour par tour » utilisé aux échecs par exemple. Dans ce dernier cas, chaque joueur, l'un après l'autre, modifie l'état du jeu à sa convenance. Un tour est terminé lorsque tous les joueurs ont joué. Dans un jeu en « temps réel », chacun est libre de réaliser une action à n'importe quel instant.

Le « brouillard » limite la vision des joueurs. Les zones visibles dépendent des positions de ses entités sur la carte et de leur champ de vision respectif. Par exemple dans Kernel Panic, un Bit peut voir jusqu'à 512 unités de distance autour de lui alors qu'un Pointer pourra voir jusqu'à 768 unités de distance. La conséquence de cette règle est que les actions de l'adversaire sont cachées tant que ses entités n'entrent pas dans le champ de vision des entités du joueur. Chaque joueur évolue dans un environnement mal connu qu'il devra explorer. La figure 9 illustre l'influence du « brouillard » sur la perception de deux joueurs

d'une même partie. Dans cette situation de jeu, le joueur A contrôle les entités vertes et a ordonné à l'une d'entre elles (numérotée 1) de s'éloigner vers la gauche. En se déplaçant, elle détecte deux entités du joueur B (numérotées 2 et 3). Inversement le joueur B qui contrôle les entités bleues (dont celles numérotées 2 et 3) voit apparaître l'entité du joueur A (numérotée 1) passant à proximité de ses entités.

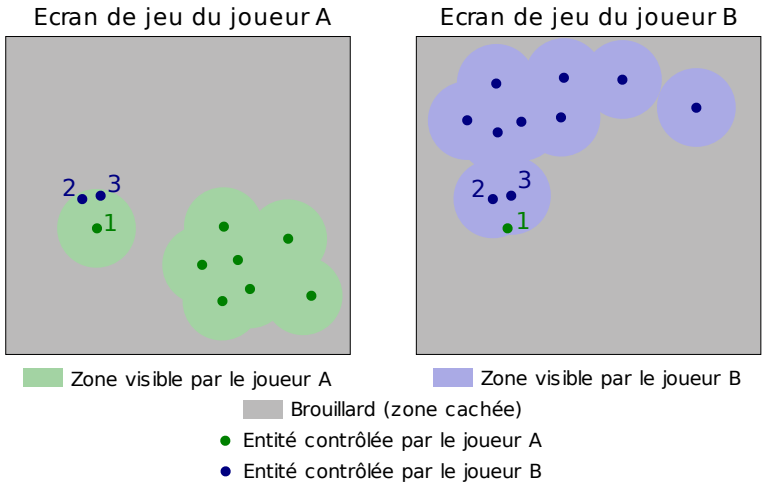

FIG. 9 – Influence du « brouillard » sur la perception de deux joueurs d'une même partie

Le « contrôle indirect » permet au joueur de pouvoir indiquer une action à une ou plusieurs de ses entités. Celles-ci vont alors tenter de la réaliser sans que le joueur ait besoin de les contrôler directement. Par exemple pour déplacer une entité le joueur lui indique la position à atteindre et elle essayera dans la mesure du possible de l'atteindre. Cette règle permet au joueur de pouvoir contrôler facilement plusieurs entités en même temps.

3 But du jeu

Dans un jeu de Stratégie Temps Réel (STR) le but du jeu consiste, en général, à éliminer toutes les entités de l'adversaire. Eliminer une entité revient à amener ses points d'énergie à 0. Pour réduire l'énergie d'une entité adversaire, le joueur doit ordonner à l'une de ses entités d'attaquer l'entité cible. Ces deux entités vont alors engager le combat et produire plusieurs assauts jusqu'à ce que l'une d'entre elles soit éliminée. Chaque assaut réduit l'énergie de l'adversaire du nombre de points de dégâts que peut infliger l'attaquant. Par exemple conformément à la description du Bit faite dans la section 1, à chaque assaut un Bit diminuera l'énergie de son adversaire de 80 points.

En raison du « brouillard », le joueur devra dans un premier temps ordonner à ses entités mobiles d'aller explorer la carte en indiquant pour chacune d'elles une position à atteindre. Lorsque l'adversaire aura été découvert (i.e. lorsque l'une des entités du joueur sera suffisamment proche d'une entité de l'adversaire pour la faire apparaître) le joueur pourra alors ordonner à ses entités d'engager le combat.

Gardez bien à l'esprit l'aspect « temps réel » du jeu. Il est tout à fait possible que votre adversaire vous découvre avant que vous ne le découvriez. Soyez donc prêt à vous défendre à tout moment.