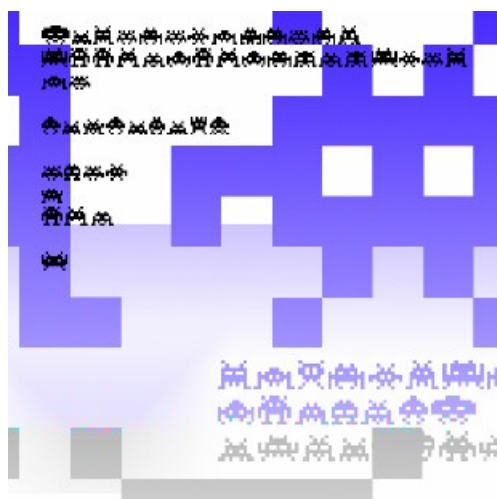


Stage d'Ingénieur **ENST** - rapport technique

Projet RNRT **SAFARI** : Réseaux ad-hoc et applications

Jeu vidéo tirant profit des caractéristiques des réseaux ad-hoc

SNCF Direction de la Recherche
et de la Technologie



- Version du 15 décembre 2003 -

Remerciements

Je tiens tout d'abord à remercier *David Sanz* et *Jean-Noël Temem*, respectivement chercheur et responsable à l'unité *TIC* (nouvelles Technologies de l'Information et de la Communication) de la Direction de la Recherche et de la Technologie de la *SNCF*, puisque ce sont eux qui ont retenu ma candidature et ont encadré ce stage.

Je tiens à remercier également toute l'équipe de la *DRT SNCF* ainsi que mon maître de stage pour l'écoute et l'aide qu'ils ont su m'apporter au jour le jour, ainsi que pour la confiance qu'ils m'ont accordée.

Enfin, je tiens à remercier particulièrement *Antoine Kah*, *Sébastien Kuntz*, *Samuel Hocevar*, et *Sherine Khoury* pour avoir participé à l'établissement d'une ambiance de travail aussi chaleureuse qu'agréable.

Introduction

Mon stage s'est déroulé de juillet à décembre 2003 au sein de l'unité de recherche nouvelles technologies de la direction de la recherche et de la technologie de la *SNCF*. Il s'agissait pour moi de ma troisième expérience professionnelle dans le cadre d'une l'entreprise du secteur informatique, ayant effectué un cursus complet à l'*IUP GMI* de l'Université Paris-IX avant *Télécom Paris*.

Le présent rapport consiste, après une présentation détaillée de l'entreprise et du contexte de recherche dans lequel il s'inscrit, en une explication et une justification des solutions informatiques mises en place pour répondre aux exigences de mon sujet de stage.

Bonne lecture.

Sommaire

1. MON STAGE A LA SNCF	8
1.1. L'ENTREPRISE SNCF	8
1.1.1. HISTORIQUE	8
1.1.2. LA SNCF AUJOURD'HUI	9
1.1.3. STATUS ECONOMIQUE	9
1.1.4. STRATEGIE	9
1.2. LA DIRECTION DE LA RECHERCHE ET DE LA TECHNOLOGIE	10
1.2.1. PRESENTATION GENERALE	10
1.2.2. L'UNITE <i>NTIC</i>	10
1.3. LA TECHNOLOGIE AD-HOC ET LE PROJET RNRT SAFARI	11
1.3.1. QU'EST-CE QU'UN RESEAU AD-HOC ?	11
1.3.2. OBJECTIFS DU PROJET <i>SAFARI</i>	11
1.3.3. LES PARTENAIRES DU PROJET	12
1.3.4. CONTEXTE DE MISE EN ŒUVRE	13
1.3.5. ORGANISATION DU PROJET	14
1.3.6. RETOMBEES DU PROJET	14
1.4. UN STAGE A LA DRT, DANS LE CADRE DU PROJET SAFARI	15
1.4.1. FICHE DE STAGE	15
1.4.2. DEROULEMENT PREVU EN TROIS PARTIES	16
2. ETAT DE L'ART : LES JEUX VIDEO SUR <i>POCKETPC</i>	17
2.1. INTRODUCTION	17
2.2. JEUX OFF-LINE	18
2.2.1. "PUZZLES GAMES"	18
2.2.2. EMULATEURS	20
2.2.3. JEUX D'AVENTURE, D'ACTION OU DE STRATEGIE	22
2.3. JEUX EN RESEAUX	24
2.3.1. GENERALITES SUR LES JEUX EN RESEAU	24
2.3.2. JEUX <i>POCKETPC</i> EN RESEAU NON AD-HOC	28
Par connexions Infrarouge / Bluetooth	28
En réseau sans fil WiFi	28
2.3.3. EXPERIMENTATION DE JEUX <i>POCKETPC</i> EN RESEAU AD-HOC	30
2.4. ASPECTS TECHNIQUES	31
2.4.1. LIBRAIRIES MULTIMEDIA POUR <i>POCKETPC</i>	31
Côté Linux	31
Côté WinCE	31
Librairies multi-plateformes	32
2.4.2. LA TROISIEME DIMENSION SUR <i>POCKETPC</i>	34
2.4.3. INTERFACES HOMME-MACHINE	36
2.4.4. STUDIOS DE DEVELOPPEMENT SPECIALISES	39
2.5. CONCLUSION	40
2.6. REFERENCES	41
2.6.1. REFERENCES INDEXEES	41
2.6.2. REFERENCES ADDITIONNELLES	42
3. RECHERCHE D'UN SCENARIO DE JEU	43

3.1. INTERROGATIONS PRELIMINAIRES	43
3.1.1. SERVEUR DE STOCKAGE	43
3.1.2. IDENTIFIANT BILLET DE TRAIN	43
3.1.3. INFORMATIONS SNCF	44
3.1.4. PROXIMITE SPATIALE EN RESEAU AD-HOC	44
3.1.5. CONTROLE DE LA PROXIMITE SPATIALE	44
3.2. CONTRAINTES PORTANT SUR LE SCENARIO DE JEU	45
3.3. SCENARIO : LE PEUPLE DE L'ARCHE (MONDE PERSISTANT)	46
3.3.1. PRINCIPE DU JEU	46
3.3.2. LE MONDE DE L'ARCHE	47
3.3.3. INTERACTIONS JOUEUR-JOUEUR ET JOUEUR-MONDE	47
Interaction entre les Joueurs	47
Interaction avec le monde	48
3.3.4. LA QUETE DE L'ARCHE ET LA QUETE DES POINTS	49
3.3.5. ASPECTS AD-HOC DU JEU	49
3.3.6. TECHNIQUE	50
3.3.7. AMELIORATIONS ET EXTENSIONS POSSIBLES	52
3.3.8. AVANTAGES ET INCONVENIENTS DU SCENARIO	53
Avantages	53
Inconvénients	53
3.3.9. REFERENCES	54
3.4. SCENARIO : PONG PARTY (PONG MULTIJOUEUR AD-HOC)	55
3.4.1. PRINCIPE DE BASE	55
3.4.2. CARACTERE AD-HOC DU JEU	56
3.4.3. SCORE ET PROFILING	56
3.4.4. LISIBILITE ET FIN DE JEU	56
3.4.5. EXTENSIONS ET AMELIORATIONS	56
3.4.6. AVANTAGES / INCONVENIENTS DU SCENARIO	57
3.4.7. REFERENCES	58
3.5. SCENARIO : COLISEUM PARTY (BEAT THEM ALL AD-HOC)	59
3.5.1. PRINCIPE DU JEU	59
3.5.2. CARACTERE AD-HOC DU JEU	60
3.5.3. EXTENSIONS ET AMELIORATIONS	60
3.5.4. AVANTAGES / INCONVENIENTS DU SCENARIO	60
3.5.5. REFERENCES	61
3.6. SCENARIO : COOPERATIVE STAR-TANKERS (SHOOT THEM ALL AD-HOC)	62
3.6.1. PRINCIPE DU JEU	62
3.6.2. CARACTERE AD-HOC DU JEU	63
3.6.3. EXTENSIONS ET AMELIORATIONS	63
3.6.4. AVANTAGES / INCONVENIENTS DU SCENARIO	63
3.6.5. REFERENCES	64
3.7. BILAN SUR LES SCENARIOS PRESENTES	65
 4. SCÉNARIO FINAL : TREENOR, RAILWAY PIRACIES	 66
4.1. INTRODUCTION	66
4.2. LES PHASES DE JEU	67
4.2.1. PHASES TREENORIENNES	67
4.2.2. PHASES SPATIALES	68
4.3. CHASSEUR DE PRIMES OU PIRATE ?	68
4.3.1. SUR TREENOR	69
4.3.2. DANS L'ESPACE	69
4.4. UN CONCEPT DE JEU ORIGINAL	70

4.5. ASPECT AD-HOC ET LIEN AVEC LA GARE	70
4.6. ASPECT TECHNIQUES	72
4.7. EVOLUTIONS POSSIBLES	74
4.8. CONCLUSION	75
4.9. REFERENCES	75
4.9.1. REFERENCES INDEXEES	75
4.9.2. REFERENCES ADDITIONNELLES	75
 5. DEVELOPPEMENTS PRELIMINAIRES ET PARALLELES	 76
5.1. CHOIX DU LANGAGE DE PROGRAMMATION	76
5.2. DEVELOPPEMENTS PRELIMINAIRES : TRPONG	76
5.2.1. PRESENTATION GENERALE	76
5.2.2. REALISATION TECHNIQUE	77
Des éléments de bases	77
Information distribuée et <i>Dead-Reconning</i>	77
5.2.3. RESULTAT EN IMAGES	78
5.3. DEVELOPPEMENTS PARALLELE : LE TRENGINE	80
5.3.1. PRESENTATION GENERALE	80
5.3.2. PRINCIPALES FONCTIONNALITES	81
La couche graphique	81
La couche réseau	82
La couche dédiée au son	82
Des outils spécifiques	82
Une interface de programmation	83
 6. DEVELOPPEMENT : TREENOR RAILWAY PIRACIES	 84
6.1. PRESENTATION GENERALE	84
6.2. PROBLEMATIQUE D'ORDRE GENERAL ET SOLUTIONS RETENUES	85
6.2.1. CONSOMMATION DE LA MEMOIRE	85
6.2.2. INDEPENDANCE VIS-A-VIS DE LA FREQUENCE DE RAFRAICHISSEMENT	85
6.2.3. CARACTERISTIQUE « MULTIPLATEFORME »	86
6.2.4. AVATAR ET PERSONNALISATION	87
6.2.5. ESTHETISME	89
6.3. PROBLEMATIQUES EN PHASE PLANETAIRE ET SOLUTIONS RETENUES	89
6.3.1. REPRESENTATION D'UN MONDE	89
6.3.2. INTERFACE HOMME-MACHINE ET JOUABILITE	91
Menu principal	91
Déplacement de l'avatar	91
Maniement et Interactions en phase planétaire	92
6.3.3. LA GESTION DES COLLISIONS	93
6.3.4. DES ROBOTS « INTELLIGENTS »	94
6.3.5. LES TRAINS SPATIAUX	95
6.3.6. IMMERSION SONORE ET AUTRES DETAILS	95
6.4. PROBLEMATIQUES EN PHASE SPATIALE ET SOLUTIONS RETENUES	96
6.4.1. REPRESENTATION DE L'ESPACE INTERSTELLAIRE	96
6.4.2. INTERFACE HOMME-MACHINE	96
6.4.3. COMBATTRE DANS L'ESPACE	96
6.4.4. ROBOTS ET TRAINS DE L'ESPACE	97
6.5. ARCHITECTURE RESEAU	97
6.5.1. LES DIFFERENTS MODES DE JEU	97

6.5.2.	EN PHASE PLANETAIRE	98
6.5.3.	EN PHASE SPATIALE	99
6.6.	PROTOCOLE DE COMMUNICATION	100
6.6.1.	EN PHASE PLANETAIRE	100
6.6.2.	EN PHASE SPATIALE	100
6.7.	RÉFÉRENCES	101
6.7.1.	RÉFÉRENCES INDEXÉES	101
6.7.2.	REFERENCES BIBLIOGRAPHIQUES	102
<u>7.</u>	<u>RESULTAT FINAL</u>	<u>103</u>
7.1.	TREENOR : INTRODUCTION, MENU ET CREATION D'AVATAR	103
7.2.	TREENOR : LES PHASES PLANETAIRES	104
7.3.	TREENOR : LES PHASES SPACTIALES	107
7.4.	TREENOR : LES STATIONS ORBITALES	110
7.5.	THE TREENOR PROJECT : SITE INTERNET (NON PUBLIE)	110
<u>8.</u>	<u>CONCLUSION</u>	<u>112</u>

1. Mon stage à la SNCF

1.1. L'entreprise SNCF

1.1.1. Historique

En 1914, la longueur du réseau national ferroviaire atteint 39400 km. Il est exploité par des compagnies privées. La prolifération des lignes dont beaucoup ne sont pas rentables, met de nombreuses compagnies dans une situation financière difficile, nécessitant une intervention de l'Etat dans la gestion des chemins de fer.

Les réseaux étant pour la plupart déficitaires, les chemins de fer sont alors nationalisés en 1937 et donnent naissance à la Société Nationale des Chemins de fer Français.

La convention du 31 août 1937 permet entre autres, de fusionner les divers réseaux de chemins de fer en un réseau unique, de placer celui-ci sous la responsabilité de l'Etat et de gérer le réseau de façon industrielle et non administrative afin d'équilibrer le budget économique. La réunion des capitaux privés et publics fait de la SNCF une société d'économie mixte.

Mais l'entreprise doit affronter dès 1937, et plus encore à partir de 1946, la concurrence croissante des transports routiers, de l'aviation et de l'oléoduc. Afin de la mettre sur un pied d'égalité avec les autres transporteurs, l'Etat apporte de profondes modifications au régime financier de l'entreprise. La SNCF devient pleinement responsable de son équilibre budgétaire.

Libre de sa gestion, la SNCF se lance dans une politique commerciale active. La qualité du service se renforce par la succession de performances techniques qui placent l'entreprise à l'avant-garde du progrès ferroviaire dans le monde.

En 1955, deux locomotives électriques battent le record du monde de vitesse sur rail en roulant à la vitesse de 331 km/h.

En 1974, la construction de la ligne nouvelle Paris-Lyon à grande vitesse est approuvée par le gouvernement. 1981, le TGV Sud-Est établit le nouveau record du monde à 380 km/h. Puis c'est au tour du TGV Atlantique qui atteint la vitesse de 515.3 km/h en mai 1990.

Depuis mai 1995, le TGV *Eurostar* reliant Paris, Bruxelles et Londres, est accessible aux voyageurs. Ce service est offert par un Groupement Européen d'intérêt Economique regroupant la SNCF, la *SNCB* (Société Nationale des Chemins de fer Belges) et la *BR* (chemins de fer britanniques). Dès juin 1995, d'autres partenariats se sont joints à la SNCF et à la SNCB dans un nouveau groupement: l'Allemagne et la Hollande, pour mettre en service le *Thalys* (Paris Bruxelles Koln Amsterdam) .

A l'échelle des moyennes distances, le rail concurrence désormais sérieusement l'avion et l'automobile.

1.1.2. La SNCF Aujourd'hui

La SNCF est chargée du transport ferroviaire de voyageurs et de fret sur le réseau national ainsi que de l'exploitation et de la gestion de l'infrastructure (voies, signaux). Elle répond aux besoins des clients en proposant des produits et services de qualité à des prix compétitifs. Elle assure sa mission dans les meilleures conditions de confort et de ponctualité et réalise des performances techniques remarquables.

En 1996, le gouvernement a procédé à une réforme du système ferroviaire, d'où la création d'un établissement public "Réseau Ferré de France" (*RFF*), chargé d'assurer, pour le compte de l'Etat, la responsabilité de propriétaire du réseau des infrastructures ferroviaires. La SNCF est devenue une société exploitante utilisant un réseau appartenant à RFF.

Afin d'assurer son développement et la reconquête de ses clients, la SNCF s'est engagée dans un projet industriel, qui repose sur trois thèmes: le client, l'entreprise, les hommes. Il prévoit un certain nombre d'actions à mener et propose à chaque cheminot de participer à leur mise en œuvre. Elles garantissent le développement de l'activité ferroviaire et un redressement financier durable de la SNCF. "Pour être, en 2003, l'entreprise de service public de référence en France et en Europe".

1.1.3. Status économique

Le 1er janvier 1993, conformément à la loi d'Orientation des Transports Intérieurs (LOTI), la SNCF est devenue un Etablissement Public Industriel et Commercial (EPIC). Son cahier des charges lui confie la mission d'exploiter, d'aménager et de développer, selon les principes du service public, le réseau ferré national en même temps qu'il définit certaines obligations :

- entreprise publique devant tenir compte de l'intérêt général
- entreprise en situation de concurrence pour toutes ses activités
- service public et compétitivité - réalisation de nouveaux projets, adaptation de son organisation.
- équilibre financier - l'équilibre des comptes est une exigence fondamentale.
- concours à l'exportation - présence sur les marchés d'ingénierie et d'études à l'étranger grâce à sa filiale SYSTRA.

1.1.4. Stratégie

La stratégie SNCF s'ordonne à moyen terme autour de trois objectifs essentiels :

- développer les liaisons rapides de voyageurs
- améliorer les transports de la vie quotidienne
- consolider la situation du fret (transport de marchandises)

Leur mise en œuvre passe par l'écoute de la clientèle car le développement du trafic suppose une adéquation permanente aux besoins de transport. La SNCF doit également diffuser une image forte par l'offre d'une grande variété de produits et posséder une gestion dynamique de son domaine d'activité.

1.2. La Direction de la Recherche et de la Technologie

1.2.1. Présentation générale

Cette direction est apparue avec le projet C03 qui a donné naissance au TGV, il y a tout juste 30 ans. Elle a aujourd'hui pour mission de piloter la Recherche et le Développement de l'entreprise et de conduire les programmes de recherche dans le cadre d'un fonctionnement en réseau.

Elle est organisée en trois domaines de responsabilités définis à partir des trois principaux axes d'activité de la Direction :

- l'animation des pôles de compétences du réseau de la Recherche
- la production de la recherche
- l'élaboration du Programme de recherche, prospective et développement.
en passant par l'aide à la gestion et à l'optimisation du trafic.

1.2.2. L'unité NTIC

La Direction de la Recherche et de la Technologie de la SNCF est composée de plusieurs unités de recherche dont l'unité N.T.I.C - Nouvelles Technologies de l'Information et de la Communication - conduite par Jean-Noël TEMEM.

Les activités de l'unité de recherche Nouvelles Technologies de l'Information et de la Communication (NTIC) sont par nature pluridisciplinaires. Les techniques utilisées et mises en œuvre relèvent à la fois de l'informatique, des télécommunications, du traitement du signal, de l'intelligence artificielle de la communication interactive et des facteurs humains.

Outre les aspects veille technologique et fonctionnelle, qui permettent de surveiller les nouvelles technologies et d'imaginer de nouveaux concepts, les travaux de recherche et de développement s'appuient sur la réalisation d'études et de projets qui prennent en compte les besoins des utilisateurs et des clients. Ces travaux sont une première étape pour assurer la maturation qu'impose l'innovation, et permettre le passage vers l'opérationnel.

Les développements en cours concernent principalement les domaines suivants:

- réseaux
- télécommunications
- commerce électronique
- traitement automatique de la parole
- systèmes de dialogue
- réalité virtuelle
- multimédia
- et depuis juillet 2003, applications/services pour mobiles et PDA.

1.3. La technologie ad-hoc et le projet RNRT SAFARI

1.3.1. Qu'est-ce qu'un réseau ad-hoc ?

Un réseau ad-hoc est un réseau sans infrastructure dédiée au routage, généré « spontanément » par l'ensemble des terminaux le constituant. En effet, les terminaux sont également routeurs et constituent autant d'étapes permettant d'acheminer une information d'un point du réseau à un autre.

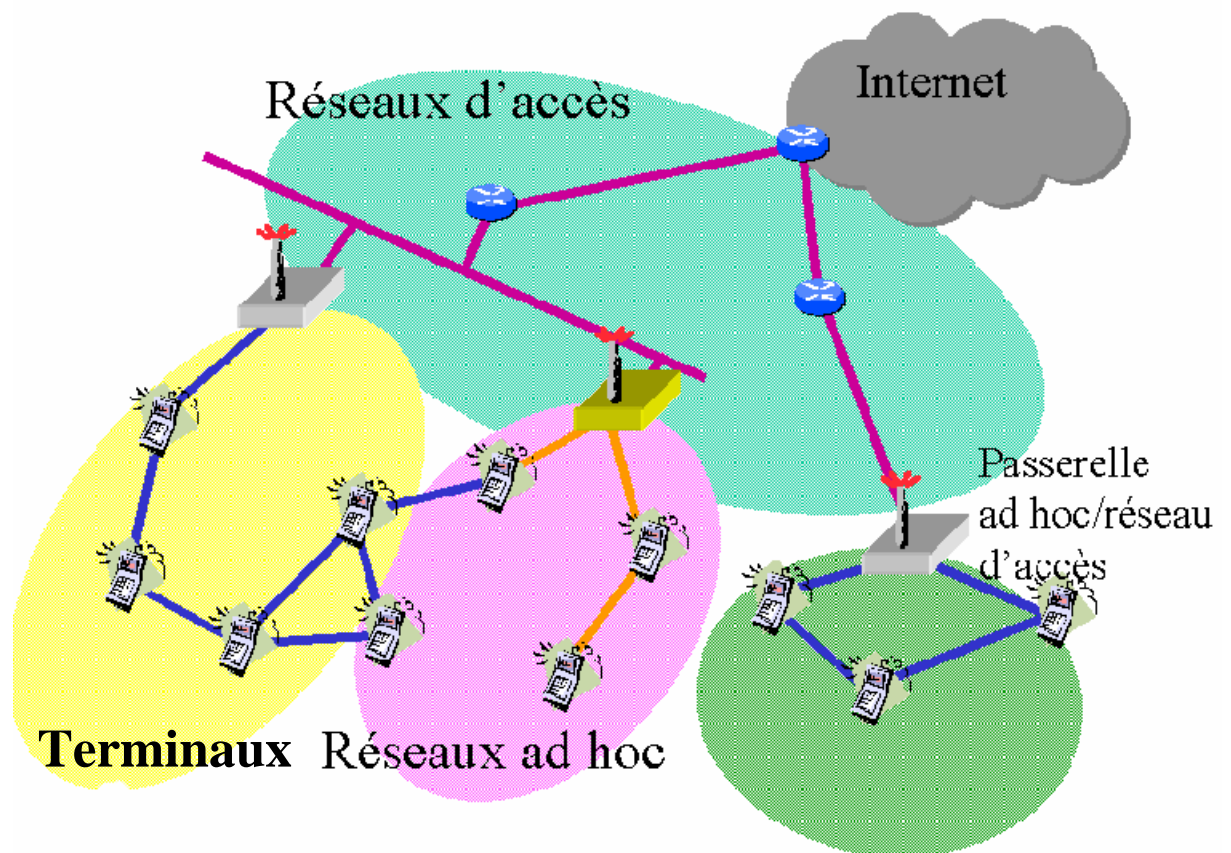
Ainsi, en mode **ad hoc** les machines sans fil clientes se connectent les unes aux autres afin de constituer un réseau point à point (*peer to peer* en anglais), c'est-à-dire un réseau dans lequel chaque machine joue en même temps de rôle de terminal et le rôle de routeur.

1.3.2. Objectifs du projet SAFARI

SAFARI a pour objectif de concevoir, combiner et réaliser une infrastructure protocolaire et logicielle nécessaire à l'accès transparent, la configuration automatique, l'intégration et l'adaptation des services sur un réseau IPv6 en **mode ad hoc** comportant des accès filaires. Son apport est la conception de solutions protocolaires et logicielles novatrices basées sur des standards existants et éprouvés (IPv6, multicast, proxy applicatifs, réseaux programmables, ...) capables de répondre aux contraintes de dynamique à la fois de l'infrastructure réseau et de l'offre et de la continuité des services.

Celle-ci comporte les composants suivants :

- terminaux : ces entités épousent à la fois le rôle de terminal usager et celui de composant de routage, rôle inhérent au mode ad hoc du réseau sans fil.
- passerelles : ces entités sont caractérisées par la disponibilité en leur sein d'une interface en mode ad hoc, ainsi que d'une interface vers l'Internet filaire. Une passerelle peut être un équipement dédié ou un terminal qui, à un moment, est branché directement sur un réseau d'entreprise (par exemple, dans une gare, l'un des PC est connecté au réseau de la gare et à une carte 802.11b/a en mode ad hoc).



La contribution majeure du projet SAFARI est de proposer la continuité des services dans le monde ad hoc au travers d'une approche globale des besoins (d'IPv6 aux services applicatifs extensibles en passant par la QoS et la supervision). Cette contribution est le résultat des contributions dans chacune des briques qui constituent la plate-forme de services visée, à savoir : les services d'interconnexion de base (IPv6, multicast sur la totalité du réseau), les services avancés (QoS dans la partie ad hoc, sécurité ad hoc et multicast), les services d'adaptation de contenus multimédia et de découverte dynamique des ressources et services. La composition optimale de ces contributions permettra d'aboutir à l'architecture visée.

1.3.3. Les partenaires du projet

Le projet *SAFARI* rassemble neuf partenaires, dont trois industriels et six « académiques » :

1. France Télécom R&D
2. ALCATEL
3. SNCF
4. INRIA - Institut National de Recherche en Informatique et en Automatique
5. LIP6 – Le Laboratoire d'Informatique de l'Université Paris 6
6. LRI - Laboratoire de Recherche en Informatique
7. LSIIT - Laboratoire des Sc. de l'image, de l'informatique et de la télédétection
8. LSR-IMAG - Institut National Polytechnique de Grenoble

9. ENST - Ecole Nationale Supérieure des Télécommunications

1.3.4. Contexte de mise en œuvre

La convergence des services et le support multi-technologies ne couvrent pas ou très peu les réseaux ad hoc, voire même les WLAN. Néanmoins un certain nombre de briques logicielles en vue d'une convergence à l'initiative du réseau sont disponibles mais souffrent de quelques limitations :

- les environnements d'exécution logiciels ouverts dans les terminaux mobiles en cours de standardisation (J2ME, MIDP) se limitent principalement à des MIDlets téléchargeables dans un navigateur. Il est attendu la possibilité de stocker des MIDlets sur le terminal, de fiabiliser leur exécution, et de fiabiliser et sécuriser leur téléchargement en définissant leur équivalent dans le réseau (dans notre cas, la passerelle fixe/mobile) ;
- les algorithmes de routage ad hoc sont en plein essor, comme on peut le constater dans le groupe MANET à l'IETF. Néanmoins, étant donné la variété des caractéristiques des différents domaines d'application des réseaux ad hoc (lieux publics comme les gares ou aéroports, complexes de salles de réunion, campus universitaires ou industriels, canaux ferroviaires ou routiers, festivals culturels, ...) ;
- les protocoles spécifiques adaptés aux contraintes de chacun de ces domaines fleurissent. Les besoins d'adaptabilité à ces différentes caractéristiques ne sont pas pris en compte alors qu'ils le devraient dès l'origine. Etant donné le fait que ce sont les terminaux qui font offices de routeurs, la problématique se rapproche de celle de la radio logicielle, qui, elle, a pris en compte ce besoin d'adaptabilité dès la conception ;
- les passerelles fixe/mobile sont essentiellement l'apanage des opérateurs/équipementiers télécoms actuellement, et traitent essentiellement de conversions protocolaires. D'un autre côté, l'hétérogénéité des terminaux (PC, assistants personnels, smartphones) est souvent gérée par les fournisseurs de services eux-mêmes, qui souffrent des problèmes de montée en charge et de manque de dynamique pour prendre en compte les nouveaux terminaux. À l'instar des passerelles WAP, il est attendu la mutualisation de fonctions d'adaptation de contenu multimédia par les passerelles fixe/mobile pour décharger les serveurs applicatifs de certaines conversions de contenus élémentaires, pour offrir aussi ces fonctions aux applications « Peer to Peer », pour prendre en charge la personnalisation par l'utilisateur (le filtrage de contenus douteux, l'introduction/retrait de bandeaux publicitaires), et pour prendre en compte l'hétérogénéité des terminaux et également les contraintes de réseaux d'accès mobiles, de surcroît lorsque ceux-ci présentent de fortes contraintes de variabilité (mobilité, bande passante, gigue, ...). Ce besoin est clairement identifié par les études sur le "mobile computing depuis une dizaine d'années. Les études sur les réseaux actifs ou services actifs viennent renforcer cette idée.

1.3.5. Organisation du projet

Le projet comporte 5 sous-projets qui abordent de façon complémentaire les différents défis relevés dans le projet et qui permettront d'aboutir au démonstrateur intégrant l'ensemble des composants nécessaires à la délivrance de services sécurisé avec qualité de service dans une infrastructure hybride fixe/mobile - ad hoc :

SP 1 : Définition de l'architecture fonctionnelle: interconnexion ad hoc / filaire

- Protocoles OLSR, ANANAS, IPv6
- Considérations d'adressage et de nommage.

SP 2 : Qualité de Service ad hoc

- routage QoS, Bruit, Contrôle d'admission / noeud

SP 3 : Environnement de service

- Plate-forme d'adaptation (gateway de service filaire fixe)
- Plate-forme de service
- Environnements terminaux
- Chargement de code dans les terminaux
- Découverte et activation de services.

SP 4 : Administration et contrôle de l'infrastructure

- COPS Outsourcing
- Monitoring
- Accounting
- Sécurité ad hoc

SP 5 : Etudes d'usage et démonstrateurs

1.3.6. Retombées du projet

Le projet SAFARI aura des retombées dans tous les domaines qui entrent dans la réalisation de l'architecture visée. La retombée majeure du projet sera de démontrer la faisabilité d'une architecture de services sur une infrastructure ad hoc avec QoS avec une couche de convergence IPv6. Dans chaque domaine (passerelle applicative, supervision, QoS), les propositions auront des impacts sur les groupes de normalisation, sur les développements futurs dans ces domaines via la distribution de composants logiciels issus du projet.

Le démonstrateur permettra de plus d'offrir une plate-forme de référence (qui plus est extensible) à la communauté permettant à de nouveaux composants d'être expérimentés.

1.4. Un stage à la DRT, dans le cadre du projet SAFARI

1.4.1. Fiche de stage

Type de stage : Stage DEA

Sujet du stage : Développement de maquettes de services pour terminaux nomades en gare

Domaines du stage : Informatique - Réseaux

Objectif du stage :

Le projet de recherche SAFARI (Services Ad-hoc/Filaires : développement d'une Architecture de Réseau Intégré) a été labellisé par le Réseau National de Recherche en Télécommunications. Les objectifs du projet sont l'étude, la réalisation et l'expérimentation d'une architecture de réseau intégrée pour la conception, le déploiement et l'exploitation optimale de services dynamiques sur un réseau IPv6 hybride ad-hoc/filaire. Le projet comprend deux démonstrateurs, dont un dans une grande gare parisienne, qui visent à montrer le fonctionnement de l'architecture développée.

Principales tâches de la mission :

- 1ère partie : Dans un premier temps le stagiaire devra prendre connaissance du contexte scientifique et technique du projet SAFARI et se familiariser avec un certain nombre de sujets en rapport avec la SNCF et ses clients (les gares, ses acteurs, clients, exploitants, et leurs besoins), le projet SAFARI (les réseaux ad-hoc, les terminaux nomades communicants, ...) et suivre les premiers résultats du groupe de design en cours à Telecom Paris. Une fois le groupe de design terminé (fin juin), ses résultats devront être analysés afin de choisir les applications innovantes à développer (maquetter).

- 2ème partie : Développement des applications retenues. Les développements devront prendre en compte l'ensemble de contraintes SAFARI connues au début de cette phase : terminaux Linux, débits disponibles, environnement logiciel de la plateforme de services développée dans le projet SAFARI, ...

Un jeu vidéo en réseau ad-hoc :

Parmi les pistes qui ont été avancées par les groupes de design en cours à l'ENST, la SNCF souhaite approfondir celle d'un jeu en réseau qui prendrait en compte à la fois les caractéristiques des réseaux ad-hoc : mobilité des noeuds (avec apparition/disparition de noeuds), débits et délais très variables,... et des éléments contextuels liés à l'utilisateur (profil, temps disponible, ...) et à l'endroit où il se trouve (une gare, avec des grands espaces, des couloirs, des commerces, des salles d'attente, ...).

Après une prise de connaissance du contexte dans lequel se situe le stage, le stagiaire devra définir et proposer plusieurs scénarios de "jeu ad-hoc en gare" qui seront soumis aux équipes SAFARI de l'ENST et de la SNCF. Un des jeux proposés sera alors sélectionné et le stagiaire devra développer (de préférence en Java) une maquette pour PDA communicant (le terminal cible est un iPAQ sous Linux et doté d'une carte WiFi). Comme les modules ad-hoc SAFARI pour les terminaux risquent de ne pas être disponibles avant la fin du stage, il est probable que dans la maquette, le mode ad-hoc doive être remplacé par le mode point à point de WiFi avec les limitations évidentes que cela implique (un noeud ne voit pas tous les noeuds du réseau, mais uniquement ceux qui se trouvent dans sa zone de couverture, les noeuds ne routent pas les paquets des autres noeuds,...).

Environnement de travail :

Stage à la Direction de la Recherche et de la Technologie de la SNCF, au sein de l'unité TIC (nouvelles Technologies de l'Information et de la Communication). Tous les outils nécessaires à la réalisation du stage sont fournis.

Profil, Connaissances prérequis :

Techniques : Linux, Java, C++, VisualBasic, HTML, Macromedia Flash

Modalités :

Date de début : 01/07/2003

Durée : 6 mois

Indemnisation : oui

Lieu : SNCF - Direction de la Recherche et de la Technologie
45, rue de Londres
75379 Paris cedex 08

Maître de stage : David SANZ
Chercheur
Tel: 01 53 42 02 90
Email: david.sanz@sncf.fr

1.4.2. Déroulement prévu en trois parties

Partie 1 : Contexte

- Entrée dans le sujet
- Lecture des documents disponibles sur les sujets suivants : SNCF, SAFARI, réseau ad-hoc
- Installation de *Linux* sur quelques *PocketPC*

Partie 2 : Etat de l'art et scénarios

- Etat de l'art : les jeux vidéo sur *PocketPC*
- Etablissement de trois scénarii de jeux tirant le mieux partie du contexte et répondant le mieux aux contraintes
- Election d'un scénario

Partie 3 : Développement du jeu

2. Etat de l'art : les jeux vidéo sur *PocketPC*

2.1. Introduction

Depuis le premier lancement commercial du *PocketPC* le 19 avril 2000 [Réf 0], le panel de logiciels et de jeux disponibles sur ce support a considérablement augmenté et les systèmes d'exploitation associés sont aujourd'hui multiples.

Nous nous intéresserons plus précisément, dans cette étude, à l'état actuel du jeu vidéo pour « PC de poche ». Ce domaine applicatif s'est très vite développé pour devenir une véritable industrie avec ses studios de développement spécialisés (dont les plus célèbres sont américains : *ZIOSoft* et *Hexacto*) et sa nouvelle communauté de joueurs passionnés (notez l'apparition du néologisme « *PocketGamers* »).

L'E³ - *Electronic Entertainment Expo* – est le plus grand salon de jeux vidéos du monde qui se tient chaque année à Los Angeles et réunit tout les grands acteurs de cette industrie. Il accueille dorénavant un ensemble de stands dédiés au *PocketPC* et le reconnaît, de ce fait et aux yeux du monde, comme une nouvelle plateforme de jeu à part entière.



Entrée de l'E³ au Los Angeles Convention Center

On recenserait à l'heure actuelle 1895 jeux vidéo commercialisés pour *WinCE* dont les prix de vente varieraient de quelques dollars à plus de 35\$, c'est-à-dire un prix similaire à un jeu PC classique, pour le dernier jeu de golf de *ZIOSoft* (source [Réf 1]).

Du point de vue des jeux vidéos portés sur les distributions de Linux pour *PocketPC*, la gamme des produits proposés est moins large avec un peu moins de 300 titres (source [Réf 2]). Rappelons que l'ensemble du parc des *PocketPC* est aujourd'hui livré avec *WinCE* à l'exception du *Zaurus* de *Sharp*, ce qui peut expliquer en partie ce déséquilibre.

Cet état de l'art s'attachera tout d'abord à présenter les jeux disponibles off-line puis en réseau sans fil, infrarouge ou bluetooth, et enfin les expériences récentes de jeux portables en réseau ad-hoc. La seconde partie consistera, quant à elle, en un tour d'horizon des technologies multimédia mises en œuvre dans ces nouveaux logiciels.

2.2. Jeux off-line

Les jeux off-line représentent une écrasante majorité des jeux vidéo sur *PocketPC* même si les jeux multi-joueurs en réseau se développent, comme nous le verrons dans la partie suivante. Ces jeux sont présentés ici à travers trois catégories distinctes qui dépendent du degré de complexité et d'innovation qu'ils apportent.

2.2.1. "puzzles games"

Cette catégorie inclut les jeux « les plus simples » qui ont tous eu leurs équivalents sur d'autres supports non portables dans la longue histoire du jeu vidéo. Certains ont même eu leur équivalent en vrai jouet en carton ou en bois.

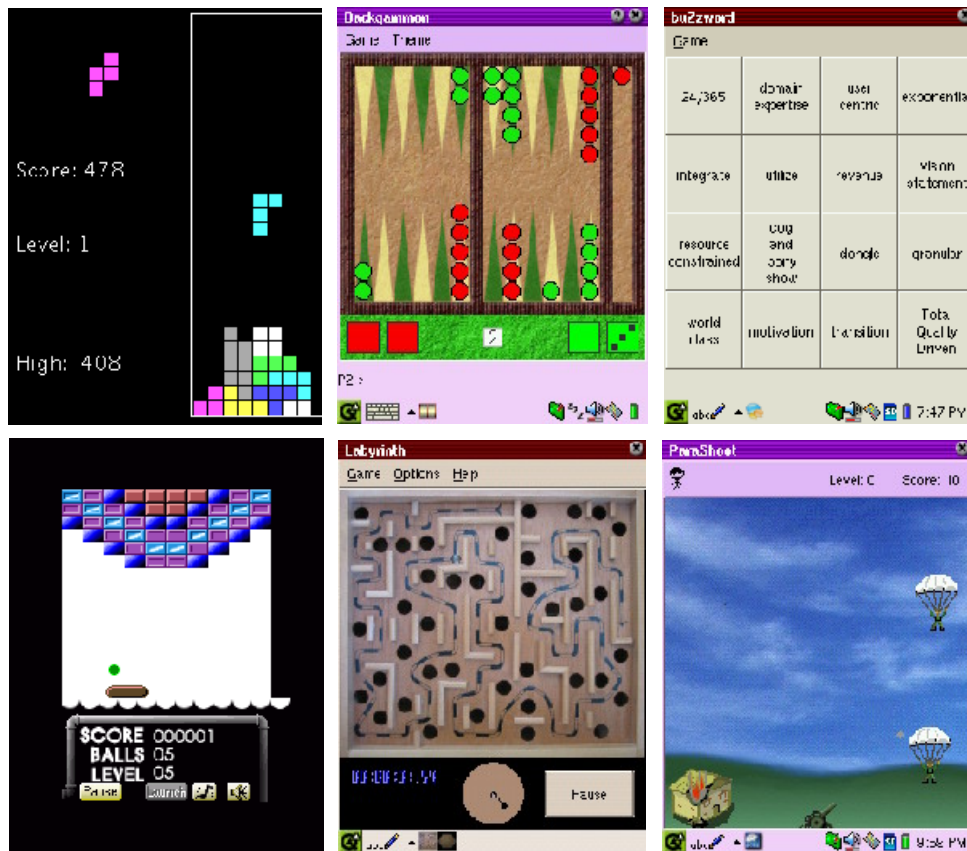


Il s'agit plus sérieusement des jeux de plateaux, les jeux de cartes, les casses briques et assimilés, les puzzles, mais aussi les grands classiques comme *Pac-Man*, *Pong*, *Tetris*, *Snake* ... Les « Puzzles Games » sont très nombreux sur *PocketPC*. Pour donner un ordre de grandeur, sur les 1895 jeux annoncés plus haut pour *WinCE*, environ 800 sont de ce type. Pour linux, la proportion atteint près des deux tiers des développements.

On peut expliquer cette réussite en remarquant que ces jeux réunissent les avantages suivants qui satisfont à la fois les développeurs et les joueurs :

- rapidité de développement, donc coût de fabrication réduit et donc prix attractifs.
- principes de jeux qui ont déjà fait leurs preuves et très largement connus de tous.
- Parties généralement assez courtes qui correspondent bien aux usages actuels en matière de *PocketPC* (Une petite partie entre deux arrêts de métro ?).
- Enfin, logiciel léger en espace mémoire pour des plateformes qui ne disposent parfois de pas plus de 16Mo d'espace dédié au stockage (iPAQ H3600).

Enfin, voici un tour d'horizon des « puzzles games » en images, tout d'abord pour linux (et plus précisément *Familiar* qui n'est autre que la version *PocketPC* de *Debian*), puis pour *WinCE* (Windows Compact Edition de *Microsoft*) :



“Puzzle Game” pour Linux – Tetris, Backgammon, Bingo, Brickout, Labyrinth, ParaShoot ...



“Puzzle Game” pour WinCE – Freecell, Pocket Snake, PocketFarm™ ...

2.2.2. Emulateurs

Quel passionné de jeux vidéo (nous les désignerons par la suite par le terme consacré de *hardcore-gamers*) n'a jamais rêvé de retrouver dans sa poche la console de jeu préférée de son enfance accompagnée des hits sur lesquels il a déjà passé des dizaines d'heures ?

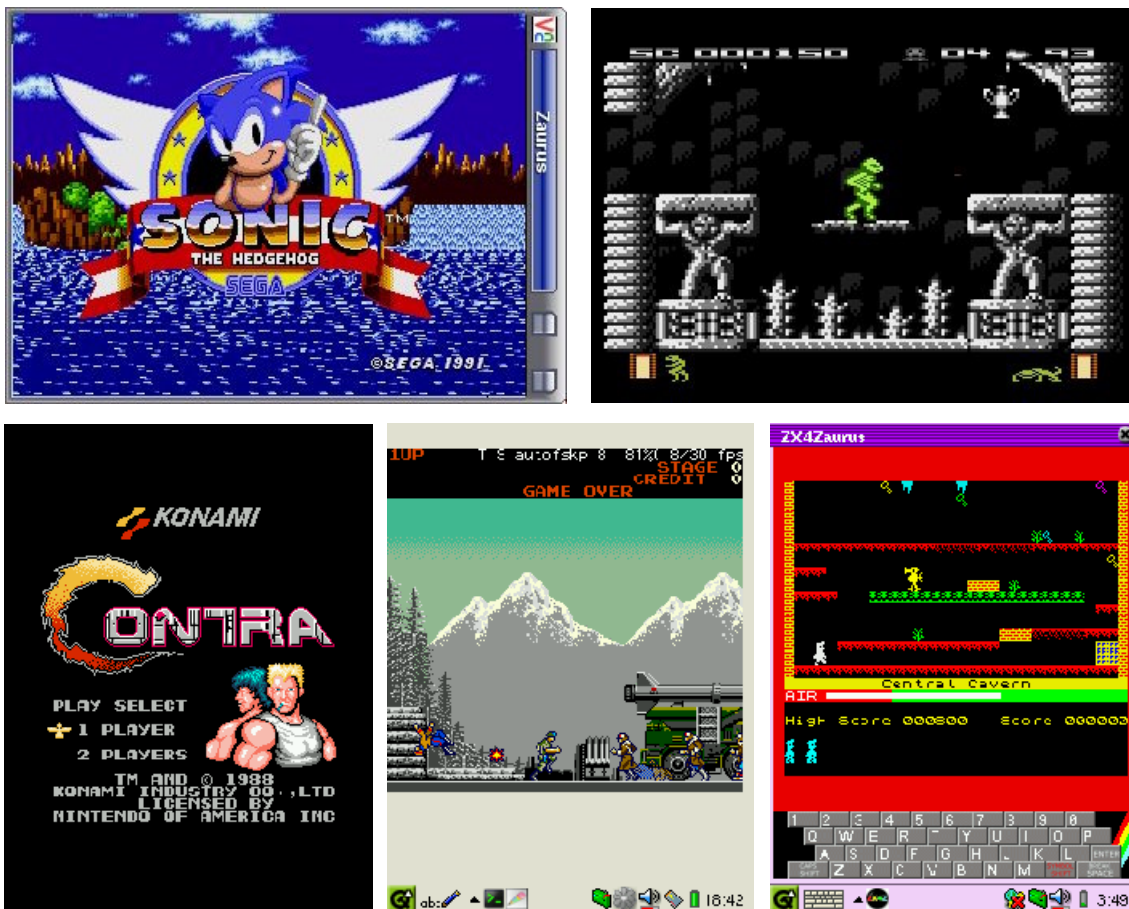
C'est exactement ce que les Emulateurs proposent. Ce ne sont donc pas des jeux à part entière mais bien des programmes qui permettent de porter l'ensemble de la ludothèque d'une plateforme de jeu donné pour l'architecture sur laquelle ils sont exécutés. Les différents jeux sont alors chargés sous forme d'images – on parle de ROMS – copies conformes des données incluses dans le support original (Cartouche de jeu, CD-ROM, DVD-ROM ou autre format propriétaire) dans un fichier.

Malgré le grand intérêt que constituent les Emulateurs pour le développement de l'offre ludique sur *PocketPC* (la ludothèque de la Super Nintendo constitue à elle seule plus de 500 titres ! [Réf 3]) les émulateurs ne sont pas des produits destinés à la vente puisqu'ils sont le fruit du développement acharné d'amateurs passionnés en violation avec le droit d'auteur et le copyright. De rares constructeurs cautionnent le développement des Emulateurs, comme ce fut le cas de *Sony* qui n'a pas caché son enthousiasme en 1999 en voyant émuler sa console *PlayStation* sur la console concurrente de l'époque, la *Dreamcast* de *Sega* (Note : cette console avait la même architecture qu'un PC et fonctionnait avec l'OS *winCE* de *Microsoft*, ce qui a très largement facilité l'émulation).

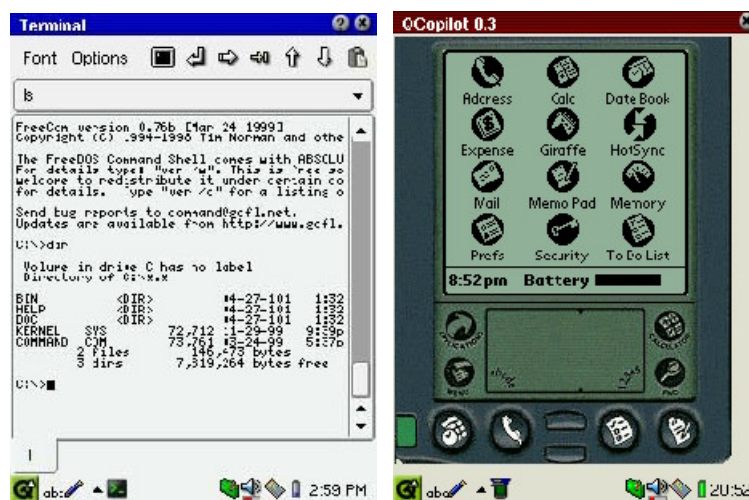
La possession de ROMS est illégale si l'on ne possède pas le jeu original. Ils sont donc officiellement des copies de sauvegarde. En pratique, leur possession est tolérée pour des jeux qui ne sont plus commercialisés.

Le *PocketPC* propose une quarantaine d'Emulateurs divers et variés pour *Linux* et un peu plus d'une cinquantaine pour *WinCE*. Citons parmi les plateformes émulées les plus éminentes : la NES et la Super NES de *Nintendo*, la Megadrive de *Sega*, l'Atari 800 et 2600, la ZX-Spectrum, mais aussi un ensemble de bornes d'arcade et la toute récente Game Boy Advance de *Nintendo* (sortie début 2002).

Remarquons enfin que l'Emulation ne se cantonne pas au seul domaine des jeux vidéo. Par exemple, *zTi85emu* permet par exemple de transformer un *PocketPC* en calculatrice Ti85 de *Texas Instruments* tandis que *Bochs* met à votre disposition un PC x86 sous DOS. Enfin, *QCopilot* permet tout simplement d'Emuler un *PalmPilot* !



Emulateur pour PockeTPC : Megadrive, Atari 800, Super Nintendo, Bone d'arcade, ZX-Spectrum...



Emulateur pour PockeTPC : PC x86 sous FreeDOS (gauche) et PalmPilot (droite)

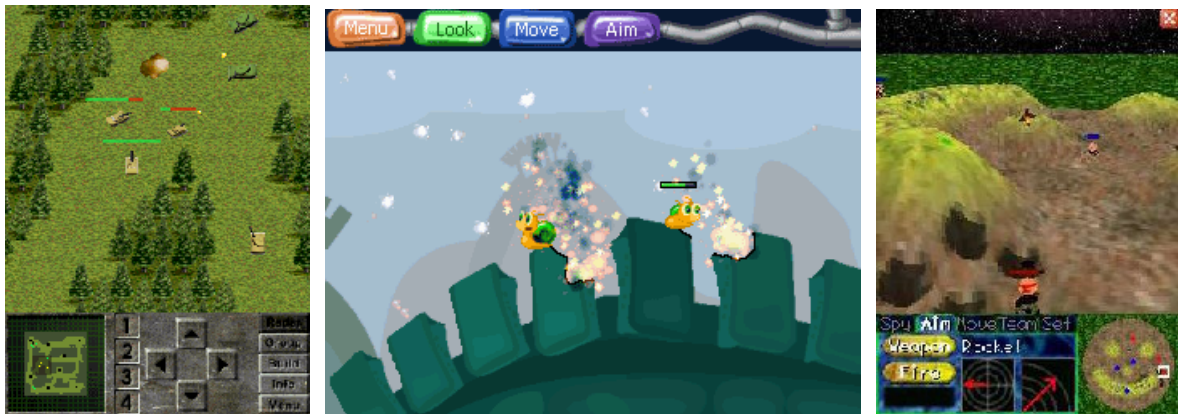
2.2.3. Jeux d'aventure, d'action ou de stratégie

Cette dernière catégorie de jeux off-line inclut l'ensemble des autres titres disponibles, allant du jeu d'action au jeu de stratégie en passant par le jeu d'aventure. Contrairement à ce qu'on aurait pu attendre, la proposition de cette catégorie (plus complexe à développer mais présentant aussi plus d'intérêt) est très restreinte sous *Linux* (moins d'une cinquantaine de titres !) et à peine majoritaire sous *WinCE* (moins d'un millier de titres sur les 1895).

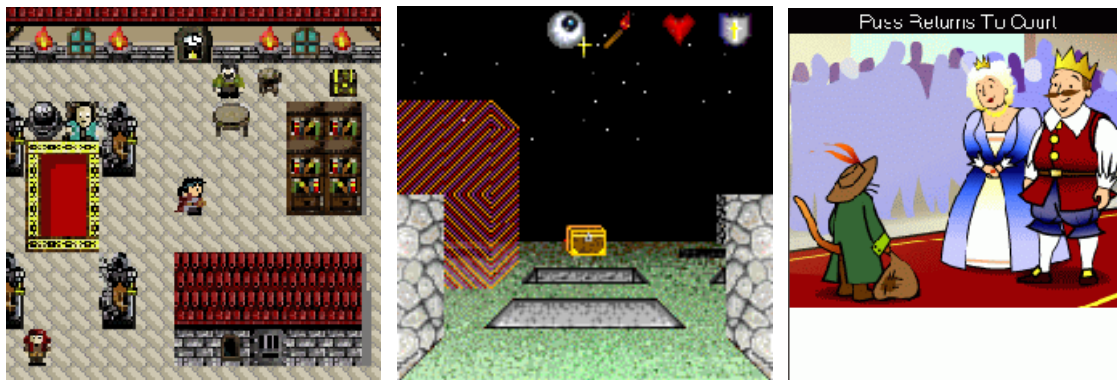
Si tous les genres présents sur PC ou consoles sont représentés (Arcade, Action, stratégie, aventure, simulation ...), il est important de distinguer les jeux originaux développés spécifiquement pour le *PocketPC* des adaptations plus ou moins fidèles de jeux préexistants.

Parmi les adaptations à succès, il faut mentionner *Snails* qui transpose le concept de *Worms* (Jeu PC édité en 1994 par *Team17*) en remplaçant les vers de terre par des escargots. Le succès de ce jeu au tour par tour est toujours au rendez-vous. Pour les jeux originaux d'action, le haut de l'affiche est actuellement tenu par *VRally*, une simulation de course automobile assez bien réalisée (test complet [Réf 4]).

Encore une fois, faisons un tour d'horizon en images des jeux actuels :



Jeux de stratégie - *Strategic Assault* (War game, Linux), *Snails* ("worms like", Linux), *SupeTtrooper* (WinCE)



Jeux d'aventure - Kyle's Quest 2 (Linux), Dragon Bane II (WinCE) et Puss In Boots (WinCE)



Jeux d'action - Zraycast ("doom like", Linux), le célèbre VRally (WinCE) et Siberian Strike (Shoot, WinCE)

2.3. Jeux en réseaux

2.3.1. Généralités sur les jeux en réseau

Les jeux en réseau se divisent fondamentalement en deux grandes familles :

- Les Mondes persistants, ou *massively multiplayer game* (MMPG)
- Les matchs à mort, ou *head-to-head death-match*

Mondes persistants

Quelque soit le rendu final, en mode texte (cf : fig. Mes premiers pas sur *FAZULL*) ou en trois dimensions (cf : fig. *There*, un MMPG en full-3D), ce type de jeu plonge le joueur dans un monde virtuel, réaliste (le deuxième monde de *Canal+*), imaginaire ou fantastique (cf : fig. *Ultima Online* de *Origin*).



Ultima Online – Monde persistant médiéval fantastique en vue isométrique, *Origin* 1997 (source [Réf 5])

Quatre vingt huit mondes persistants seraient actuellement disponibles ou en développement à travers le monde (source [Ref 2]), et ceux-ci suscitent un engouement de plus en plus important comme le démontre en France le succès de *ManKind* (cf : [Réf 6] *Mankind*, le succès malgré les imperfections).



Mankind - monde persistant français, entre combat interstellaire et relation commerciale, *Vibes* (2000)

Les interactions que peut avoir le joueur avec le monde et les êtres qui le peuplent varient sensiblement d'un produit à l'autre. Malgré ces variantes, on retrouve les interactions classiques des jeux off-line : combat, découverte d'objet, résolution d'énigmes ou gestion de ressources, auxquels s'ajoutent une nouvelle forme de « tourisme » et une tendance au dialogue exacerbé par la présence, dans le même monde, d'autre joueurs humains.

En effet, comme le souligne la définition ci-dessous, ce qui fait la force des mondes persistants est le fait qu'ils immergent un très grand nombre de joueurs en même temps dans un même jeu. C'est ce que signifie l'adjectif Massively Multiplayer.

Massively Multiplayer and Persistent World

Thousands of players can be *in the same game at the same time* in a world that is available 24 hours, 7 days a week.

Définition des MMPG par Atriarch (source [Réf 7])

On parle de mondes persistants car on évolue dans des univers continus dans le temps. Qu'on soit connecté ou pas, le monde évolue par lui-même. D'autres personnes peuvent jouer et il est possible que le monde change en votre absence. Les MMPG vous offrent en quelque sorte un monde parallèle, non pas centré sur vous - le joueur - comme dans les jeux traditionnels, mais avec sa vie propre.

Notons que les MMPG encouragent en un sens les interactions sociales. Tout d'abord, ils jouent un rôle secondaire mais non négligeable de « chat graphique » géant. De plus, en fonction des univers, la coopération entre joueurs est plus ou moins possible pouvant encourager au fil du temps la création de véritables communautés de joueurs se réunissant derrière une même entité, un emblème, une race, une alliance ... Le terme approprié, entré dans le vocabulaire des hardcore-gamers, est *Guildes*.

Citons par exemple, la Guilde « l'empire du scorpion noir » qui regroupe des joueurs de trois mondes persistants différents (Mankind, Eve et le dernier StarWars Galaxies) autour d'une même « idéologie de jeu » (cf : <http://www.scorpion-noir.org>). La guilde est même organisée en trois castes de joueurs placées sous l'égide du conseil Gris du Scorpion noir ...



Emblème du scorpion

Pour finir, d'un point de vue technique il est important de signaler que les mondes persistants sont synonyme de serveurs. Les joueurs disposent sur leur plateforme d'un client (contenant le moteur graphique, les données images et son, et l'ensemble des données « non-dynamiques » du jeu) qui pour pouvoir ouvrir une session de jeu doit se connecter à un serveur dédié (en pratique les serveurs sont disposés chez les fournisseurs du jeu).

Les serveurs conservent les données persistantes et dynamiques (c'est à dire inconnues lors du développement du jeu) du monde et se chargent de les fournir aux clients. Il s'agit par exemple des données des autres joueurs, leurs positions, ou les changements qui ont eu lieu sur une planète par exemple.

Pour le lecteur désireux d'en savoir plus sur les mondes persistants et la révolution à venir qu'ils représentent, on recommande vivement la lecture des très bons articles de *Crosbie Fitch* intitulés *Cyberspace in the 21st Century* disponibles à l'adresse suivante [Réf 8].

Ci-dessous on trouve un exemple de premières minutes de jeu dans le monde de *Fazuul*. Il s'agit d'un monde persistant en mode texte, appelé plus communément *MUD* (Multi-User Dungeon), se jouant directement à partir d'un client *Telnet*. Ce type de *MMPG* n'a pas encore sa place dans les musées puisqu'il en existe de nombreux en activité encore aujourd'hui.

```
FAZUULjorMUD
by Tim Stryker 35 104 DWELLE
Copyright (C) 1984, 1986, 1991, 1992 Galacticomm, Inc.
33 Kreg          MajorMUD
-----

>move north
...You can't go that way.
>move south
...You can't go that way.
>move east
...You can't go that way.
>move west
...You're standing on a broad outdoor walkway running east and
west, with tall crystalline towers rising up on either side. To
the east is a gate leading to the spaceport. Further on down the
walkway to the west is a giant glowing sphere on a pedestal, and
to the south is an open portal leading into the base of one of
the crystalline towers.
>move east
...You're at the spaceport.
>west
...You're on the east-gate walkway.
>move south
...You're in the Hall of Umflungoo, an auditorium-sized room with
Umflungoo banners hanging from the rafters. There is a large
ircular hole in the middle of the floor; portals lead out of the
hall to the north and south. A small panel attached to the wall
near the north portal appears to have some writing on it.
```

Les MUDS (mondes persistants en mode texte) – Mes premiers pas sur FAZUUL...

Enfin, une illustration de *There*, le monde persistant en trois dimensions cité ci avant. En texte ou en image, la « drague » reste un loisir pour les joueurs de *MMPG* ...



There, un MMPG en full-3D : crobin et shartley02 en pleine discussion ...

Death match

La seconde grande famille de jeu vidéo en réseau est constituée des jeux de type matchs à mort, ou *head-to-head death-match*. Si les titres à succès sont les éminents *Quake*, *Starcraft* ou *Tetris*, cette catégorie de jeux en réseau est la plus représentée qu'il s'agisse de jeu de tir à la première personne (*FPS* : first-person shooters), de simulation de guerre (War Games) ou d'autres concepts d'affrontement plus ou moins originaux.

Ce succès peut s'expliquer par deux phénomènes : tout d'abord, l'architecture matérielle qu'ils nécessitent est moins lourde que les mondes persistants puisqu'ils ne nécessitent pas (à priori) de serveur centralisé et donc pas d'abonnement à un service comme condition nécessaire au jeu. Chaque joueur est un client et l'un d'entre eux fait office de serveur pour échanger les données évolutives (qui sont beaucoup plus limitées que dans les mondes persistants : typiquement la position ou l'état des autres joueurs).

Une deuxième raison du succès est peut-être le fait que ce type de jeux, consistant en l'affrontement pur entre les joueurs, répond plus directement à ce qu'on pourrait appeler une « pulsion ludique ». L'action est plus directe que dans les autres types de jeu en réseau.

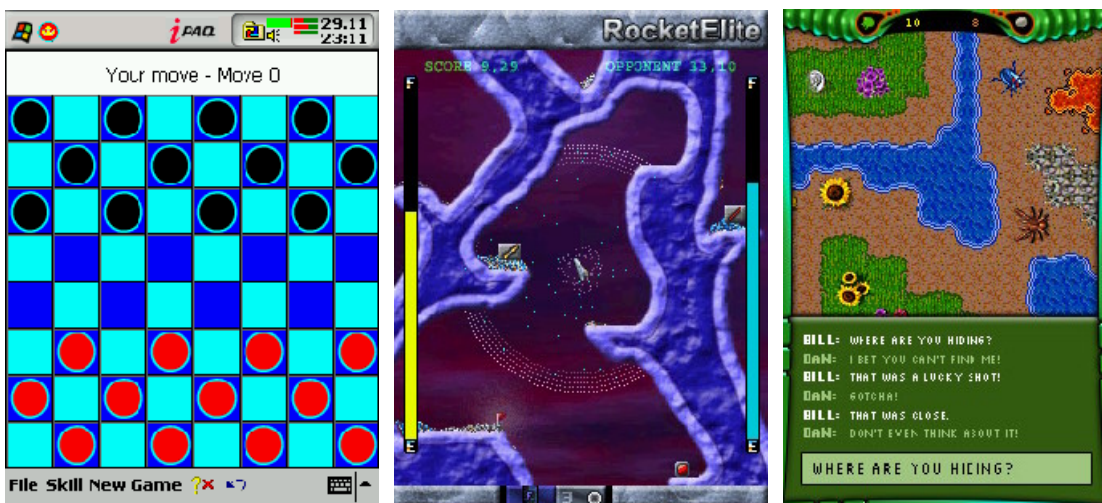
Pour les Death match, il existe également des serveurs spécifiques mais ceux-ci n'ont plus un rôle de stockage. Ils permettent en revanche de faciliter la mise en relation des joueurs afin qu'ils constituent une session de jeu. Des milliers de serveurs de ce type se sont développés à travers le monde (source : About Online Multiplayer Games, Sugih Jamin University of Michigan). Citons à titre d'exemple www.Gamespy.com.

2.3.2. Jeux *PocketPC* en réseau non ad-hoc

Par connexions Infrarouge / Bluetooth

Pour jouer à plusieurs sur *PocketPC* il y a tout d'abord des solutions plus simples et moins lourdes à mettre en place que des réseaux sans fils. Il s'agit de relier deux PC de poche en utilisant directement une connexion *Infrarouge* ou *Bluetooth*.

Cette alternative à été implémentée dans un certain nombre de jeux *PocketPC* afin d'offrir un mode deux joueurs facile à mettre en place (il suffit de placer les PDA côte à côte !). Remarquons que la proportion de jeux *Bluetooth* est aujourd'hui minime face à ceux qui utilisent une connexion *Infrarouge*.



Jeux infrarouge - Checkers 1.0 (jeu de dame, Ask4Soft) et RocketElite (jeu de tire, Digital Concept)

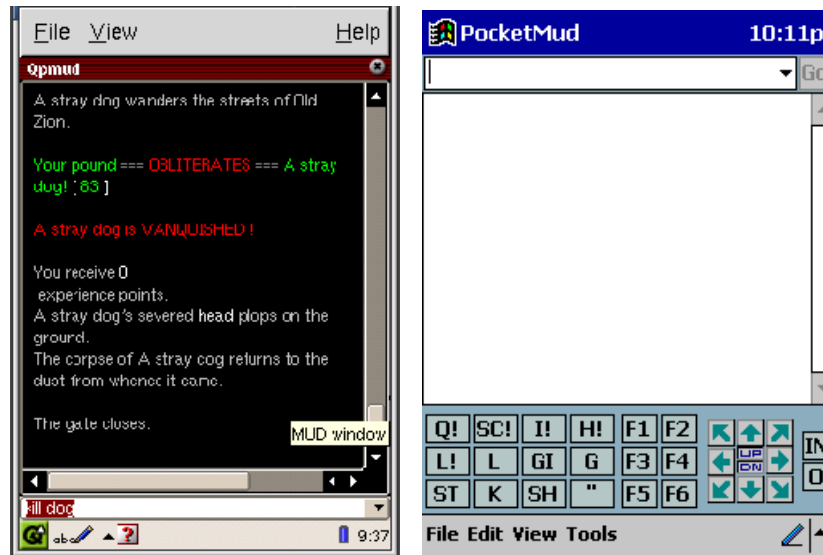
Jeu Bluetooth – Buglord (jeu de stratégie sur le thème des insectes, BigBrainGames)

Le retard du *PocketPC* dans l'usage du *Bluetooth* dans les jeux devrait être renforcé par l'usage que va bientôt en faire la console portable *GameBoy Advance SP* de *Nintendo* qui devrait très largement démocratiser le procédé. Le lecteur intrigué par ce projet ce rapportera à l'article [Réf 9]. Nous verrons plus tard que le *PocketPC* est en revanche en avance sur la *GBA* dans le domaine de la 3D.

En réseau sans fil WiFi

Grâce à la technologie *WiFi* (pour Wireless Fidelity ou IEEE 802.11b wireless networking) il est possible pour un *PocketPC* de se connecter à un réseau local et même à Internet. Cette mise en réseau des terminaux mobiles permet d'imaginer la mise en place de tous les types de jeux réseaux présentés dans le paragraphe 2.3.1 *Généralités sur les jeux en réseaux*. C'est exactement ce qui a été fait par les studios de production spécialisés qui seront présentés plus loin.

On retrouve ainsi les *MUD* (Multi-User Dungeon, MMPG en mode texte) sur *PocketPC* qui se connectent à un serveur distant afin d'obtenir les textes répondant aux interactions et sollicitations du joueur. Le logiciel sur le terminal mobile n'est autre qu'un équivalent de client Telnet.



OpMud MUD Client pour Opie (Linux) et InterString PocketMud (WinCE, InterString Software)

Parmi les 88 mondes persistants actuellement disponibles ou en développement dont il était question plus haut, aucun n'intègre la possibilité de recevoir des clients *PocketPC* et encore moins sont spécialement « conçus » pour. Jusqu'ici, le domaine des MMPG sur PC de poche est un sous domaine des jeux vidéo restant inexploré !

En revanche, les jeux de type « Death Match » (cf : paragraphe précédent) sans fil font d'ores et déjà partie de la ludothèque des *PDA*. A signaler par exemple *3D Mini-Jetfight*, de *OmniGSoft*, qui offre la possibilité d'affronter ses amis en réseau dans une simulation d'avion de chasse très réaliste (nous reviendrons sur les outils graphiques réalisés par *OmniGSoft* dans le paragraphe 2.4.1). La particularité unique de ce titre et de toute la série des *3D Mini* du même éditeur est la possibilité que les terminaux du réseau sans fil soient indifféremment des PC de poche, de bureau, des *SmartPhones* ou des *Palm Pilot* ! C'est en quelque sorte un pas de plus vers les réseaux hétérogènes...



3D Mini-JetFight, Play multiplayer game over wireless LAN on PC, Pocket PC or Smartphone

2.3.3. Expérimentation de jeux *PocketPC* en réseau ad-hoc

Les jeux *PocketPC* commerciaux en réseau ad-hoc sont encore inexistants. En effet, aucune infrastructure commerciale de réseau ad-hoc n'a encore été mise en place. En revanche, des projets de recherche ont été réalisés sur le sujet par le laboratoire de recherches suédois *SITI AB* (Swedish Research Institute for Information Technology).

Les travaux du *SITI AB* consistent à étudier de quelle manière les jeux vidéo sur réseau ad-hoc pourraient faciliter et encourager dans un lieu public les interactions entre utilisateurs en face à face. Afin de réaliser leur étude, ils ont développé un jeu *PocketPC* intitulé *Cafétrek* qui regroupe plusieurs modules différents permettant aux joueurs du réseau de s'affronter l'un contre l'autre. Le jeu a ensuite été confié à trente étudiants d'une université suédoise afin qu'ils le testent durant quatre semaines puis rapportent le fruit de leurs expériences respectives.

Le principal résultat de cette étude a été de confirmer le résultat attendu. En effet, l'affrontement distant mais peer-to-peer que permet le réseau ad-hoc encourage de manière significative les interactions réelles ultérieures entre les joueurs.

Pour en savoir plus sur cette expérience dont le maître mot est « Adhoc gaming is the social networking of the future », consulter [Réf 10].

2.4. Aspects techniques

2.4.1. Librairies Multimédia pour *PocketPC*

Afin de réaliser l'ensemble des jeux ci-dessus, les développeurs ont dû concevoir un certain nombre d'outils génériques facilitant l'exploitation des capacités Multimédia des *PocketPC* (graphisme, calcul géométrique, mais aussi son...). Dans ce paragraphe, on dresse une liste non exhaustive de ces outils que sont les librairies graphiques et environnements de développement Multimédia (*SDK*, Software Development Kit).

Faisant le lien entre « le bas niveau » et la couche applicative, ces librairies sont fortement liées au système d'exploitation pour lequel elles sont conçues. Nous nous intéresserons alors aux développements pour *Linux* et *WinCE*.

Côté Linux

Dans le monde des *Desktop PC* (c'est-à-dire des PC classiques) fonctionnant sous Linux, l'environnement de développement multimédia le plus largement employé se nomme *SDL* pour *Simple DirectMedia Layer* [Réf 11]. L'aspect graphique 3D de *SDL* utilise la technologie multi plateforme *OpenGL*. *SDL* remplit en fait les mêmes fonctionnalités que *DirectX* de *Microsoft* pour PC sous *Windows*.

En ce qui concerne le *PocketPC*, *SDL* a été adapté afin de fonctionner sur ce support. Très vite adopté par les développeurs, *SDL* est le SDK multimédia le plus utilisé sur *PocketPC* même si d'autres SDK co-existent comme *X-Forge* (présenté côté WinCE mais également compatible Linux).

Pour les jeux les plus simples nécessitant de tirer beaucoup moins partie des capacités multimédia des PDA (type « puzzle game » par exemple), ce sont directement les technologies graphiques *GTK* (GIMP toolkit) ou *QT* qui sont utilisées. Ce sont en effet ces boîtes à outils graphiques qui sont exploitées pour proposer les GUI – Graphic User Interface - *Opie* (QT) et *Gpe* (GTK) compatibles avec la distribution *Familiar* de Linux pour *PocketPC*.

Côté WinCE

Les premières recherches ont consistées à rechercher une adaptation pour *WinCE* de l'hégémonique *DirectX* de *Microsoft*. En effet, l'entreprise fondée par Bill Gates a tout de suite pensé à doter la version portable de son système d'exploitation d'un équivalent de *DirectX*. Les premiers travaux réalisés en ce sens ont d'ailleurs permis à la firme de concevoir, en partenariat avec *Sega*, la couche graphique de la défunte console de jeu *Dreamcast* 128 bits, concurrente annoncée de la *Playstation 2* de *Sony* en 1998.

Comme l'explique ci-dessous un responsable marketing de chez *Microsoft*, il s'est avéré qu'en raison des contraintes matérielles fortes qui s'exercent sur les systèmes portables, il a été jugé plus pertinent de réaliser un système dédié au *PocketPC* plutôt

que de chercher à adapter le complet mais lourd *DirectX*. Cette solution customisée pour PDA a été baptisée *GAPI*.

Where is DirectX?

So if the Pocket PC is great for games, does it include DirectX support? The answer is no. The Pocket PC product team looked at porting DirectX over (it's in use on the Dreamcast now), but when they saw the sheer size and lack of any dedicated graphics acceleration hardware on the devices, it quickly became apparent that they'd need something smaller and more tuned to the Pocket PC platform. Thus was born GAPI!

Pour en savoir plus sur l'environnement de développement *GAPI* consulter [Réf 12] d'où provient l'extrait précédent. Ce qu'il faut retenir côté vidéo est que l'accent est principalement mis sur la vitesse d'accès au buffer de l'écran qui se doit d'être accéléré le plus possible. *GAPI* propose également aux applications multimédia une standardisation pour la gestion du mode plein écran des PDA (horizontale ou verticale) et pour le « mapping » automatique des boutons.

Même si *GAPI*, sous sa forme actuelle, est assez récent il semble déjà s'être imposé comme leader en matière de *SDK* Multimédia pour *Windows CE* et n'a pas de concurrents solides. En effet, *Microsoft* joue ici sur son propre terrain.

Librairies multi-plateformes

Il existe malgré cela des librairies graphiques multi-plateformes compatibles avec *WinCE* qui méritent d'être présentées puisqu'elles sont utilisées dans un certain nombre de jeux actuellement sur le marché. Les plus abouties sont *X-Forge* et le *Diesel Engine* de Inmar Software Ltd. Ces librairies s'attachent principalement à optimiser et à accélérer les rendus en trois dimensions. Elles sont compatibles avec *WinCE* mais également *Symbian OS* (système d'exploitation pour les téléphones portables dit intelligents : Nokia N-Gage, Sony Ericsson P800 et Orange SPV) et *Palm OS* (système d'exploitation des *PalmPilot*).

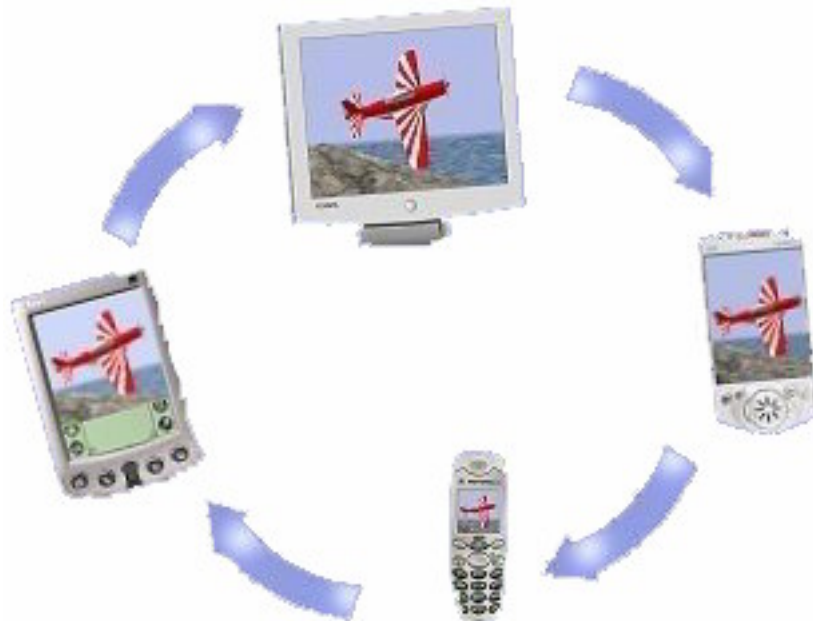


Exemple en image : *Diesel Minigolf* (*Diesel Engine*) et *Geopod* (*X-Forge*)

A signaler également les travaux de *Sundial Soft* qui cherche à adapter au *PocketPC* la librairie graphique *OpenGL* issue du monde du *PC* en créant *PocketGL*.

OpenGL est, comme nous l'avons signalé précédemment, un ensemble de librairies graphiques très largement déployé pour les applications multimédia PC sous *Linux* (SDL en dépend). Il constitue une des seules alternatives sérieuses aux librairies graphiques de *DirectX* (en dehors des librairies propriétaires) pour développer des applications PC graphiquement riches sous *Windows*. Pour en savoir plus sur *PocketGL*, consulter [Réf 13].

Enfin toujours en matière de librairies graphiques, il semble primordial de souligner les importants travaux de la firme américaine *OmniGSoft* qui s'attache à développer un moteur graphique *OmniGraphics* « Cross-platform ». L'objectif, d'ores et déjà atteint, est que le rendu final soit de même qualité qu'il soit projeté sur l'écran d'un ordinateur, d'un Palm Pilot, d'un téléphone portable intelligent ou d'un PC de poche.



OmniGraphics, un moteur graphique « inter plateforme » développé par OmniGSoft [Réf 14]

2.4.2. La Troisième dimension sur *PocketPC*

Si aujourd'hui dans le monde des jeux vidéo pour stations fixes (PC, consoles, bornes d'arcades ...) la 3D c'est banalisée au point de devenir quasi impérative pour les réalisations commerciales, elle n'en est pas moins naissante sur les stations de jeux portables.

Pour donner un exemple concret, la toute récente console portable de *Nintendo* (la *GameBoy Advance SP* sortie en France en 2003) n'intègre toujours aucune routine de calcul de 3D en temps réel. Malgré ses 32 bits les seuls jeux en 3D sont en réalité en « pseudo-3D » dont le meilleur exemple est *Wolfenstein 3D* de ID Software (cf : figure suivante).



la pseudo-3D sur *GameBoy Advance SP* avec *Wolfenstein 3D* de ID Software

On aurait donc pu au premier abord s'attendre à une gestion de la 3D relativement peu complexe sur *PocketPC*. Pourtant, dans ce domaine, le PC de poche nous rappelle son lien de parenté avec son grand frère de bureau. En effet, les librairies graphiques présentées dans le paragraphe précédent offrent des rendus très satisfaisants, laissant loin derrière les consoles portables actuelles (Sans compter bien sûr sur la prochaine future console portable de *Sony* ...).

On retrouve ici le travail de *OmniGSoft* - la librairie *OmniGraphics* - qui permet de générer ce que l'on pourrait établir comme le plus beau rendu en matière d'images en trois dimensions calculées en temps réel sur *PocketPC*.

A vous de vous faire votre propre opinion avec les images qui suivent :





3D Mini-Sportsbike, 3D Mini-TransCanada, 3D Mini-Dogfight et 3D Mini-Jetfight (OmniGSoft Inc.)



Radium, un clone de Tomb Raider (Karama Studios) [Réf 15]



Métalion 1.1 (gauche) et ZIOGolf 2 (droite) (ZIO Interactive)

2.4.3. Interfaces Homme-Machine

Le PocketPC est une station résolument portable et les choix des constructeurs en matière d'interfaces de communication sont allés bien évidemment dans ce sens. Les PDA en général (*PocketPC*, *Palm*, ...) sont munis des deux interfaces distinctes, complémentaires ou proposant les mêmes fonctionnalités en fonction des applications : un pavé directionnel assorti d'une série de boutons et un écran tactile.

Du point de vue des jeux vidéo, le pavé directionnel et les boutons – aux nombres de quatre sur les *PocketPC* – offrent les mêmes possibilités que celles des consoles de jeux vidéo de salon en vogue de 1983 à 1990 et que celles des consoles portables actuelles, même si la prise en main est moins évidente.



De deux à huit boutons, l'évolution des joypads des consoles de jeux de salon – NES à 2 boutons (1983), la Mégadrive à 3 (1988), la SNES à 4+2 latéraux (1990), et enfin la Playstation à 4+4 latéraux (1995).

L'IHM du *PocketPC* est donc raisonnablement adapté à l'ensemble des jeux qui ont eu leur équivalent sur consoles (et ils sont plus que nombreux!).

L'écran tactile des PDA et le stylet qui l'accompagne constituent une alternative convenable pour les jeux dont la jouabilité s'effectue traditionnellement à la souris (issu essentiellement du monde du PC). On pense ici d'avantage aux jeux d'aventures/réflexion où la souris permet de « pointer » un objet qu'aux jeux « à la première personne » (*FPS – First Person Shooter*) où la souris permet d'orienter une vue en trois dimensions.

A noter le portage sur *PocketPC* des grands succès du jeu d'aventure PC qui ont fait les bonnes heures de *Lucasfilm Games* - rebaptisé *Lucas Arts* (cf : fig. suivante). L'écran tactile s'adapte donc aisément à ces jeux dont la jouabilité a été exclusivement prévue pour la souris.



Ecran tactile et stylet - Day Of The Tentacles (à gauche) et Monkey Island II (à droite) sur PocketPC

Pour améliorer la prise en main de l'ensemble pavé directionnel / boutons, des accessoires sont apparus petit à petit sur le marché. Ils donnent au *PocketPC* des allures de console de jeux portable en se fixant directement à sa base. Citons les produits phares qui sont le *QPad* [Réf 16] et le *DPad* [Réf 17] de *Tigerex* et *Hanao*.



DPad de Tigerex (à gauche) et le QPad de Hanaho (à gauche)

Enfin, pour une jouabilité optimale, il existe des manettes spécialement adaptées aux PDA. Nous présentons ici le modèle Zeta de *Be Interactive* dans sa version *PocketPC*.



La manette PocketPC de BE interactive – Le Zeta Game Pad [Réf 18]

Le clavier est un point faible de l'IHM des *PocketPC*. Il existe heureusement des alternatives aux claviers tactiles miniatures proposés dans les interfaces graphiques qui rendent la saisie moins fastidieuse.

Avant de faire un tour d'horizon des solutions proposées, signalons que le clavier n'est quasiment jamais utilisé dans les jeux vidéo PDA sauf si, bien entendu, il est nécessaire de saisir un texte. En effet, si le clavier est souvent utilisé dans les jeux pour ordinateur traditionnel c'est bien grâce à l'accès immédiat et simultané aux touches.

De simples adaptateurs permettent tout d'abord de brancher un clavier de PC standard sur un *PDA*. Il existe également des claviers plus allégés qui accueillent les PC de poche comme le font les stations de rechargement (cf : fig. suivante).



Foldable Keyboard for the Compaq iPAQ

Plus transportable, *Targus* et *Compaq* proposent des extensions claviers miniatures qui se fixent directement sur l'appareil à la manière des *QPad* et *DPad* présentés précédents.



Targus Click 'n' Type Keyboard (à gauche), SnapNType Portable Mini-Keyboard (à droite).

Enfin, la dernière solution qui allie portabilité et confort d'utilisation nous vient de la société Israélienne *VKB*. Elle projete un clavier lumineux sur toute surface plane puis détecte automatiquement la position des doigts.



Le Virtual Keyboard de VKB dans sa version PocketPC [Réf 19]

2.4.4. Studios de développement spécialisés

Comme il a été annoncé dans l'introduction de ce document, le jeu vidéo sur *PocketPC* représente aujourd'hui une industrie à part entière et la clientèle est suffisamment large pour permettre l'émergence d'entreprises spécialisées dans la conception de jeux pour ce support.

Les studios de développement spécialisés pour *PocketPC* sont même très nombreux. Il est intéressant de noter que la plupart d'entre eux développent également leurs jeux dans des versions pour *Palm Pilot*, mais parfois aussi *SmartPhone* (téléphone dit intelligent) ou *Tablet PC* (PC portable prenant la forme d'une Tuile).

A titre indicatif, citons les deux acteurs majeurs du domaine que sont *ZIO Interactive* et *Hexacto*. Les autres studios remarquables par la presse spécialisée seraient *OmniGSoft*, *Emodiv Inc.*, *Resco*, *Liquid Impact Inc.*, *Pocky Games*, *Jimmy Software Co.*, *Carrot4Free*, *Karma-studios*, *CyberPlanet Interactive* ...



Parmi les meilleures ventes PocketPC: ZioGolf, StarWars, XBall, Metalion ...

2.5. Conclusion

Dans cet état de l'art, nous avons présenté l'industrie du jeu vidéo sur *PocketPC* à travers ses aspects techniques et commerciaux. Nous avons dressé un panel des différents types de jeux déjà développés depuis la naissance de la plateforme en avril 2000, qu'ils soient à un joueur ou à plusieurs. Nous avons ensuite découvert les outils de développement multimédia pour environnements *Windows* et *Linux* versions de poche pour enfin aborder les problèmes d'interface homme machine.

La communauté de « joueurs *Pocket* » naissante ainsi que la profusion des studios de développement dédiés font du *PocketPC* une véritable plateforme de jeu nouvelle qui a très largement sa place auprès des récentes consoles de jeu portables.



*La future plateforme de jeu portable
par excellence ?*

2.6. Références

2.6.1. Références indexées

Réf 0 : Happy Birthday PocketPC !

<http://www.pocketpcthoughts.com/forums/viewtopic.php?t=11516&highlight=history+pocketpc>

Réf 1 : PocketPC software and freeware for all devices

www.pocketpcsoft.net

Réf 2 : Games and Linux for PocketPC

www.killefiz.de

Réf 3 : La super Nintendo à la loupe

http://www.nintendo-europe.com/NOE/fr/FR/system/snes_topic1.jsp

Réf 4 : Test complet de VRally

<http://www.ppcfrance.com/default.asp?r=vrally>

Réf 5 : Ultima Online Official Website

<http://www.uo.com/>

Réf 6 : Mankind - Le succès malgré les imperfections

<http://echo.levillage.org/dossier/jeu/mankind.cbb>

Réf 7 : Massively Multiplayer and Persistent

<http://www.vervetech.com/atriarch/features/massive.htm>

Réf 8 : Cyberspace in the 21st Century, Crosbie Fitch's articles

http://www.gamasutra.com/php-bin/article_display.php?category=14

Réf 9 : La famille GameBoy bientôt au régime sans-fil version Bluetooth

<http://www.zdnet.fr/actualites/technologie/0,39020809,2101743,00.htm>

Réf 10 : Cafétrek - Adhoc gaming is the social networking of the future

<http://www.cafetrek.com>

Réf 11 : Simple DirectMedia Layer

<http://www.libsdl.org>

Réf 12 : Here Comes GAPI!

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnppc2k/html/ppc_gapi.asp

Réf 13 : Le « diesel Engine »

<http://www.inmarsoftware.com/diesel.htm>

Réf 14 : *OmniGraphics* de *OmniGSoft*
<http://www.omnigsoft.com/products/OmniGraphics/OmniGraphics.html>

Réf 15 : *Radium*, « Tomb Raider like » en 3D pour *PocketPC*
<http://www.radiumgame.com/>

Réf 16 : le *DPad* de *Tigerex* pour *PocketPC*
<http://www.tigerex.com.tw>

Réf 17 : le *QPad* de *Hanaho* pour *PocketPC*
<http://www.hanaho.com>

Réf 18 : *Zeta Game Pad* de *Be Interactive*
<http://www.pocketpcminds.com/reviews/zetajoypad.php>

Réf 19 : *VKB - Virtual Keyboard*
<http://www.vkb.co.il/>

2.6.2. Références additionnelles

Réf : L'histoire des consoles de jeux de salon
<http://jeux.video.tripod.com/histoire.html>

Réf : Tous savoir sur *PocketGL*, adaptation d'*OpenGL* pour *PocketGL*
<http://www.sundialsoft.freemove.co.uk/pgl.htm>

Réf : Mondes persistants - le portail francophone des MMGP
<http://www.mondespersistants.com>

Réf : Persistent online Worlds list
<http://hem.passagen.se/ulkis/onlineworlds/>

Réf : About Online Multiplayer Game
www.cs.umn.edu/Research/networking/itrworkshop/slides/sugih_jamin.pdf

3. Recherche d'un scénario de jeu

3.1. Interrogations préliminaires

3.1.1. Serveur de stockage

Question : Malgré le caractère ad-hoc du réseau sur lequel le jeu doit fonctionner, y sera-t-il intégré un serveur ou du moins un espace de stockage indépendant des terminaux utilisateurs ?

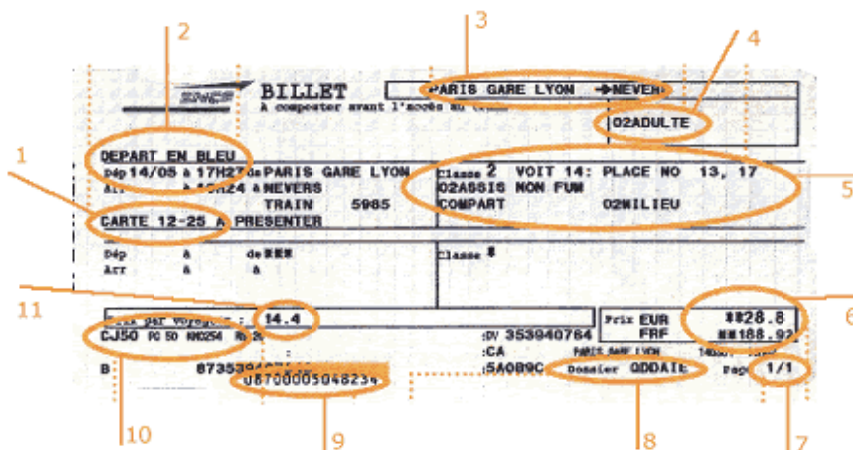
En effet, il est important de savoir si les données dynamiques du jeu (c'est-à-dire celles qui ne sont pas connues à priori au développement) doivent être stockées uniquement sur chacun des terminaux ou en partie centralisées. Par exemple, il paraît judicieux de stocker les données spécifiques aux utilisateurs dans leurs terminaux (profiles) et les données évolutives spécifiques au jeu lui-même (et donc indépendantes des joueurs) dans un serveur.

Par exemple, pour implémenter un monde réellement persistant qui évolue au fil du temps (les joueurs peuvent effectuer des modifications durables : construire ou détruire des bâtiments, modifier la géographie des lieux, placer des pièges à lapin dans une forêt ou labourer un champ ...) il faut nécessairement un serveur pour stocker les données évolutives du monde. Sans celui-ci, le monde est « pseudo » persistant (la carte est par exemple fixe) et les évolutions du monde au cours du temps sont préprogrammées (alternance jour/nuit, déplacement des monstres ...).

Réponse : Oui, les passerelles en gare qui se chargeront du lien entre le réseau ad-hoc et le réseau filaire peuvent servir de serveur et donc d'espace de stockage pour les données dynamiques du jeu.

3.1.2. Identifiant billet de train

Question : Y a-t-il un identifiant unique sur les billets de train permettant de les dissocier et un algorithme qui permette d'en vérifier le plus facilement possible la validité ?



Transformer un billet SNCF en billet de LOTO ?

L'idée est ici d'associer et de créer un lien entre le futur voyage du joueur et le jeu. On peut par exemple imaginer une loterie où certains identifiants de billet seraient gagnants. Egalement, on peut imaginer d'obtenir certaines caractéristiques pour son personnage (pouvoirs, puissance, objets ...) à l'aide de son billet. La seule contrainte est qu'il doit exister un algorithme ou un moyen simple de vérifier si une suite de caractères est un identifiant valide ou non, afin d'éviter des tests multiples « au petit bonheur la chance ».

Réponse : Non. Il existe bel et bien un identifiant appelé « numéro de dossier (suite de six lettres – indiquée 8 sur la figure ci-dessus) mais il n'y a pas de règles de vérification. Le seul moyen est d'effectuer une requête au serveur SNCF central de Lille ce qui paraît un peu lourd dans le cadre d'un jeu même si cela est techniquement possible.

3.1.3. Informations SNCF

Question : Est-il techniquement possible et raisonnable (du point de vue de la complexité) d'accéder aux « informations train » de la SNCF (départs, arrivées, horaire, ...) dans le jeu ?

Si oui, il serait intéressant de tirer partie de ses informations pour lier plus étroitement le jeu avec le contexte de la SNCF en transformant les départs et les arrivées des trains en des instants particuliers dans le jeu. Par exemple, dans un monde persistant, des événements particuliers peuvent advenir en synchronisation avec les trains.

Réponse : Oui mais à un prix discutable. Comme pour l'identifiant billet de la question précédente, il faut accéder au serveur d'informations SNCF de Lille. N'est ce pas trop lourd ?

3.1.4. Proximité spatiale en réseau ad-hoc

Question : A-t-on connaissance des joueurs distant d'un bond (au sens du réseau ad-hoc) de nous ?

Si oui, il serait très intéressant de tirer partie de la proximité spatiale des joueurs dans le jeu. La coopération entre les gens proches (même équipe, même unité, même vaisseau, ...) ou l'affrontement (même partie, même arène de jeu, ...) sont des concepts qui permettraient d'apporter une forte dimension ad-hoc au jeu.

Réponse : Oui, cela sera possible avec *OLSR*, la librairie ad-hoc développée par l'*INRIA* dans le cadre du projet *SAFARI*.

3.1.5. Contrôle de la proximité spatiale

Question : un bond (au sens du réseau ad-hoc) a-t-il à une longueur maximale fixe, variable, ou contrôlable ? Si fixe, quel est son ordre de grandeur ?

Réponse : La portée dépend de la carte réseau utilisée, mais également de la gestion éventuelle de sa puissance qui peut-être faite. Par exemple, lorsque aucun nœud n'est découvert à une certaine puissance (donc distance), la puissance d'une carte peut-être augmentée pour atteindre une plus grande portée.

Cette gestion dynamique de la puissance est implémentée dans OLSR (pour carte réseaux avec driver *orinoco*). Il devrait être possible de limiter la puissance d'émission de la carte radio afin de fixer la distance maximale d'un bond (par exemple moins de 10 mètres) et qu'ainsi le jeu tire partie de la proximité spatiale.

3.2. Contraintes portant sur le scénario de jeu

Le jeu vidéo qui sera développé au cours de ce stage devra satisfaire un ensemble de contraintes liés d'une part au projet *SAFARI* et aux exigences de la *SNCF*. Voici, dans un ordre d'importance décroissante, un ensemble de contraintes que devront satisfaire au mieux les scénarios proposés :

- 1) Le principe du jeu devra tirer au maximum parti du caractère ad-hoc du réseau pour lequel il est destiné. C'est la contrainte la plus forte que soumet la *SNCF* et plus généralement le projet *SAFARI*. En effet, la *SNCF* se positionne dans le projet comme un client final permettant de tester la technologie ad-hoc dans ses gares. Le jeu qui fait l'objet de ce stage est en fait un aspect de la démonstration, en gare, du réseau ad-hoc.

De plus, la *SNCF* s'intéresse depuis longtemps aux services ludiques pour ses clients. La portée de ce stage va donc plus loin que le strict cadre du projet *RNRT SAFARI*.

- 2) Le jeu, étant en réseau et dans un contexte où le nombre de joueurs est potentiellement élevé, doit être massivement multijoueur (cf : chapitre 1, état de l'art).
- 3) Le scénario peut être innovant mais le principe du jeu doit être suffisamment simple et clair pour être assimilé rapidement par le plus grand nombre. Idéalement, un joueur occasionnel doit y trouver ses marques et se divertir rapidement tandis qu'un hardcore-gamer, en rentrant plus en profondeur dans le jeu, trouvera la complexité et la richesse ludique qu'il recherche.
- 4) Le jeu doit proposer un environnement graphique aussi riche que possible puisque cet aspect du développement est particulièrement motivant. Il s'agit de proposer une interface agréable et ergonomique à l'utilisateur, une certaine fluidité visuelle ...
- 5) Sa complexité doit correspondre au temps imparti pour le développer, à savoir cinq mois. L'idéal est un jeu dont le concept est suffisamment large pour autoriser le plus grand nombre d'améliorations et d'extensions possibles afin que le temps de développement soit modulable à souhait.

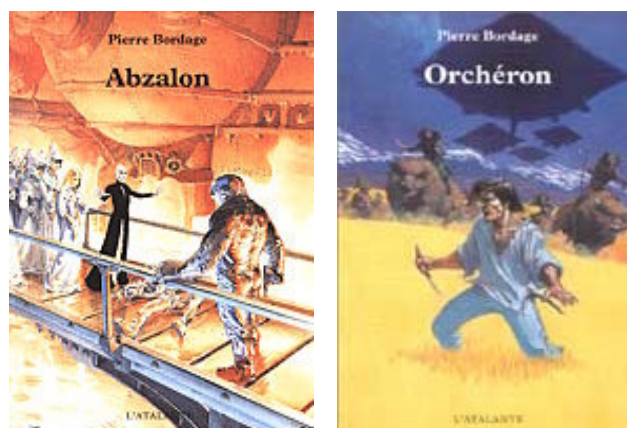
3.3. Scénario : le peuple de l'arche (monde persistant)



3.3.1. Principe du jeu

Après toutes les réflexions et remarques qui ont pu être faites tout au long de ce document sur les mondes persistants, le lecteur ne sera pas surpris de trouver un monde persistant comme premier scénario de jeu proposé. Le paragraphe « avantages » de la dernière partie de ce chapitre s'attachera à présenter à quel point ce scénario semble répondre aux contraintes décrites précédemment.

Le Peuple De l'Arche (*PDA*) est un monde persistant en deux dimensions dans un univers proche de l'univers médiéval-fantastique, avec quelques « mystères technologiques » qui s'y introduisent tel que l'arche lui-même (cf : la légende de l'arche). L'histoire de ce monde est inspirée des romans *Abzalon* et *Orchéron* de *Pierre Bordage* et se placerait au moment où le peuple décrit dans ces romans oublie ses origines.



Abzalon et Orchéron de Pierre Bordage, aux éditions de l'Atalante (1998 et 2000) [Réf 1]

Pour l'esthétisme général et l'univers graphique, la principale source d'inspiration serait *The legend of Zelda: a link to the past* (Jeu Nintendo, sortie en 1992 sur *Super NES* et réédité en 2003 sur la console portable *GameBoy Advance SP*).

Le principe du jeu est donc de permettre aux joueurs de faire évoluer un avatar dans un monde virtuel (constitué de villages, de prairies, de champs, de lacs, de forêts, ...) dans lequel il peut à tout moment rencontrer l'avatar d'autres joueurs jouant au même jeu au même moment (et faisant donc partie du même réseau ad-hoc).

Outre le tourisme et les interactions multiples qui peuvent s'exercer entre les joueurs et entre les joueurs et le monde (cf : paragraphe 3.3.3), la dimension ludique du jeu réside dans l'affrontement d'adversaires non humains et plus globalement dans la poursuite de la « quête de l'arche ». Cette quête consistera à résoudre le mystère qui plane sur les origines du peuple de l'arche ...

3.3.2. Le monde de l'arche

Sans entrer dans les détails de ce qui constituera un travail d'imagination relativement conséquent si le projet est retenu, on présente ici le monde de l'arche sous ses traits les plus caractéristiques.

En analogie avec l'espèce humaine, disons que le peuple de l'arche a atteint un degré de civilisation moyenâgeux. Les habitants sont pour la majorité agriculteurs, chasseurs, pêcheurs ou artisans.

Les rivalités, conflits d'intérêts et luttes intestines sont relativement peu nombreuses et le peuple vit dans une paix relative depuis une époque qui échappe maintenant à la mémoire collective. Seule trace indiscutable d'un passé lointain, les vestiges d'une structure étrange et incongrue semblant provenir de nulle part, trônant du haut de ses 50m de haut en plein centre de la ville de *Uspun* et que l'on désigne sous le terme « Arche ».

En ce qui concerne les décors du jeu, la zone dans laquelle les joueurs pourront évoluer (il ne s'agit pas en effet de modéliser une planète toute entière !) est une réunion de plusieurs régions géographiques du globe terrestre. Mélangez les steppes russes avec les forêts bavaroises, les côtes bretonnes avec les Alpes italiennes et vous obtiendrez un aperçu du monde de l'arche.

3.3.3. Interactions joueur-joueur et joueur-monde

Interaction entre les Joueurs

- Les utilisateurs peuvent dialoguer entre eux directement dans le monde. Lorsqu'un texte est saisi, une bulle s'affiche au dessus du personnage à la manière des bandes dessinées et affiche le texte. Toutes les personnes dans l'entourage proche du joueur « entendent », c'est-à-dire peuvent lire, la phrase prononcée.

Un système de proposition de phrases toutes faites (« bat toi ! », « où es-tu dans la gare ? », ...) vient compléter ce système de discussions qui n'a rien d'innovant. (Exemple : *Ultima Online*, *There*, *Meridian59*, *Graal-Online*, ...).

- Les combats se déroulent comme dans la plupart des jeux d'actions en deux dimensions à la troisième personne. Exemple : *Zelda*, *Graal Online*, ... C'est-à-dire traditionnellement à l'épée, avec un système de cœurs pour représenter le niveau de vie du personnage.
Le monde est parfois hostile et regorge en certains endroits de monstres et bêtes féroces qu'il faudra être en mesure de maîtriser. La quête de l'arche amènera également les joueurs à affronter des monstres plus imposants ...
- Dans le jeu, les comportements des joueurs peuvent être multiples. En effet, on est libre de parler et de sympathiser avec les autres joueurs, ou au contraire, de les affronter et pourquoi pas chercher à les massacrer tous ! Ange ou démon, c'est souvent dans les jeux que se libère votre vraie personnalité. C'est en tout cas la richesse comportementale de l'homme qui fera la richesse du jeu *PDA*.
- Un certain nombre de « mini-jeux » collectifs seront disponibles dans le jeu. Par exemple, on peut imaginer qu'il soit possible de « jouer à chat » avec un ensemble d'autres joueurs dans une zone du village assimilable à un jardin d'enfant : le chat est désigné puis un compte à rebours s'enclenche laissant aux autres le temps de s'éparpiller. La partie s'arrête lorsque tous les joueurs sont retrouvés par le chat ou que celui-ci abandonne la partie.
- Une arène de jeu à la romaine, au plein centre du village principale, peut également permettre d'intégrer d'autres type de « mini-jeux ». L'idée est ici d'intégrer complètement le scénario *Coliseum Party* dans *PDA* ! (cf : 3.5).

Interaction avec le monde

- Le monde de l'arche regorge d'objets aux propriétés différentes qu'il est possible de collecter et conserver dans son inventaire. Il peut s'agir d'armes, d'équipements de protections ou d'objets magiques en tout genre. L'inventaire fait partie des données stockées dans le profil du joueur avec l'apparence de son avatar, son nom, son score actuel, ...
- Un système de magie peut venir enrichir les combats. Chacun dispose d'un potentiel magique limité dont la capacité augmente au fur et à mesure des combats. Tout un arsenal de magie se débloquent alors.
- Le monde de l'arche a aussi sa monnaie locale. L'argent peut être gagné de différentes manières (victoires dans un affrontement, loterie, ...) et dépensé dans des magasins locaux qui sont un bon moyen d'enrichir son inventaire.
- S'il est possible de mettre en place la loterie utilisant les identifiants billet de train dont il était question dans le chapitre précédent, on peut mettre en place la loterie

nationale de l'arche, permettant d'obtenir différents gains comme de l'argent, des objets ou de nouveaux pouvoirs magiques.

- L'heure locale au monde de l'Arche est en synchronisation avec l'heure réelle en gare. L'alternance jour/nuit s'effectue en 24h.
- Certaines parties du monde sont sans ennemis ou monstres pilotés par l'ordinateur (la ville centrale par exemple) tandis qu'il existe des zones à risque (forêt, steppe, donjon, ...) regorgeant de monstres, d'affrontements ... et d'aventure ! A noter que dans les zones sans risque, personne n'est à l'abri d'un joueur « fou » qui prendrait un malin plaisir à attaquer tout le monde.
- Chaque combat, résolution d'énigme ou collecte d'objet permet d'obtenir des points. Les scores sont stockés dans le profil des joueurs et correspondent à l'ensemble de leurs sessions de jeu. Les high-scores sont disponible en temps réel aux yeux de tous.
- Si un serveur de stockage est disponible, le monde de l'arche sera réellement évolutif comme il a été expliqué dans le paragraphe 3.1.1. Par exemple, on peu imaginer que chaque joueur dispose de sa propre battisse dans le monde.

3.3.4. La quête de l'arche et la quête des points

La quête des points, plutôt à destination des hardcore-gamers, consiste alors à obtenir le meilleur score de toute la gare exactement comme dans le scénario *Pong Party* du paragraphe 3.4.

Si les cinq mois de développement permettent de l'implémenter, la quête de l'arche est une suite d'énigme, d'action à accomplir et de monstres à affronter qui mèneront le joueur sur les traces de l'origine du monde de l'Arche. Cette quête consistera notamment à réunir un ensemble de fragments de l'arche dissimulées dans des emplacements périlleux du monde ...

3.3.5. Aspects ad-hoc du jeu

La dimension ad-hoc de l'aventure de l'arche se révèle dans l'aspect collaboratif du jeu. Exactement comme dans *Cooperative Star-Tanker* (cf : scénario présenté au paragraphe 3.6), la collaboration entre joueurs s'établit comme suit :

Lorsqu'un joueur s'approche suffisamment d'un autre pour être distant d'un bond au sens du réseau ad-hoc, une proposition de coopération est alors soumise aux deux joueurs. Si la coopération est acceptée par chacun d'entre eux, leurs forces et puissances magiques s'additionnent rendant le consortium plus puissant. En revanche, les points gagnés et objets récoltés seront également partagés.

La coopération sera nécessaire pour résoudre certaines énigmes de la quête de l'arche et combattre certains monstres plus gros que les autres (on parle de « boss »). Le

hardcore-gamer sera alors partagé entre la quête des points qui le pousse à rester seul, et la quête de l'arche qui le force à coopérer ... à moins de passer des centaines d'heures sur le jeu et de faire parti des meilleurs joueurs.

3.3.6. Technique

- L'écran est orienté horizontalement afin d'offrir une meilleur visibilité latérale.
- Les déplacements de l'avatar sont gérés avec l'ensemble pavé centrale et boutons du *PocketPC*. Une alternative consiste à utiliser le crayon pour pointer la direction vers laquelle on désire que le personnage se dirige, ce qu'il fera en ligne droite. On mène ainsi l'avatar avec le bout du crayon comme on mène un âne avec une carotte. Cette technique a l'avantage d'être beaucoup moins complexe à implémenter que lorsque le personnage se déplace de façon « intelligente » d'un point à un autre.
- Les ennemis pilotés par l'ordinateur ont des techniques d'attaque assez simples. En effet, ils se dirigent tête baissée et en ligne droite sur le héros puis l'attaquent lorsqu'il est à portée d'arme. Là encore, il s'agit d'une hypothèse simplificatrice. En dehors des combats, on peut imaginer un système de chemin de ronde pouvant être affecté aux ennemis afin de proposer une alternative à l'immobilité.
- Comme il a été signalé précédemment, un usage du *profiling* client est fait afin de stocker entre deux sessions de jeu les données des joueurs : Nom, apparence, pouvoirs, objets ...
- Le nom (=login) des joueurs que l'on rencontre s'affiche pendant une seconde au dessus de leur tête lorsqu'ils apparaissent à l'écran.
- Du point de vu graphique et de la modélisation du monde à l'écran, un procédé simple est envisagé : la technique du *Tilling*.

Il s'agit en fait de considérer l'ensemble du monde comme un immense pavage de carrés de taille fixe. A chaque pavé d'apparence différente (herbe, pavé, sable, eau, coin droit d'une maison, ...) correspond un identifiant. La carte et le monde entier n'est alors plus qu'un tableau d'ID et tout le graphisme des décors se limite à une grosse image contenant tout les pavés possibles (cf : le fichier de décor de Graal Online). Notez que les « pavés » peuvent être animés.

Le gain de place et la simplification du travail graphique qu'apporte cette technique est très sensible. Il y a également un second tableau similaire pour compléter la description du monde. Celui-ci indique si le pavé est un terrain : praticable, impraticable (mur, arbre, ...), ou semi-praticable (eau, boue, sables mouvants, ...).

La technique du *Tilling* est une vieille technique, mais toujours mise en oeuvre dans de nombreux jeux. Citons par exemple le dernier *Pokemon adventure* de Nintendo éditée en juillet 2003.



La technique du Tilling - Zelda I: The Wand of Gamelon (Game Boy Color) et Zelda II: The Wand of Gamelon (Super NES)



Graal Online : v1.0 avec la technique du Tilling (droite), v2.0 en vue isométrique (gauche)



Technique du Tilling - Le fichier des décors utilisé dans Graal Online

3.3.7. Améliorations et Extensions possibles

Elles sont extrêmement nombreuses et c'est ce qui fait la richesse de ce scénario.

On peut imaginer que chaque personnage dispose d'un semblant de téléphone portable avec lequel il peut communiquer avec toute personne connectée et donc présente au même moment dans le monde (Système de numéro de téléphone locale ou pseudo). On réinventerait ainsi un chat « en privé ».

Mais le concept de ce scénario est tellement large qu'il permet d'y intégrer en partie les deux scénarios qui seront présentés ultérieurement : Comme il a été dit précédemment, l'arène du village, peut permettre d'intégrer *Coliseum Party* dans PDA. (cf : chap 3.5).

Mais il me semble surtout intéressant d'intégrer le monde de *Cooperative Star-Tanker* (cf : 3.6) dans le peuple de l'Arche. Les compagnies marchandes de ce jeu s'avèreraient exploiter les habitants du monde de l'Arche en retirant une large partie de leur récolte.

L'idée la plus novatrice serait sans doute alors que les *Star-Tanker* atterrissent sur le monde à heure fixe ... en synchronisation avec l'arrivée et le départ des trains dans la gare ! (Il faut bien sûr avoir accès aux « info trains », ce qui ne pourra se faire en temps réel)

3.3.8. Avantages et inconvénients du scénario

Avantages

- Le temps nécessaire au développement est modulable et surtout peut être étendu à l'infini (Avec d'autres objectifs, ce scénario pourrait facilement donner du travail à une équipe de vingt personnes pendant deux ans).
Si permettre aux joueurs de se déplacer et de discuter dans une prairie verte peut constituer un premier objectif, les améliorations, mini-quêtes, petits-jeux et interactions en tout genre pouvant être intégrées permettront de constituer autant d'objectifs secondaires que nécessaire.
- Les mondes persistants ont l'avantage de permettre aux usagers de se rencontrer et de discuter librement comme dans un chat traditionnel, recette qui a déjà fait ses preuves (pensez au succès de *caramail*).
- Un autre grand avantage du point de vue de la *SNCF* est la création, à terme, de communautés de joueurs, ce qui tisse un nouveau type de lien entre les usagers et la société des chemins de fer français. C'est une forme de fidélisation de clientèle cachée.
- Ce scénario tire réellement partie de la technologie ad-hoc avec la coopération inter-joueurs. Le peuple de l'arche constituerait ainsi le premier monde persistant ad-hoc au monde !!
- Il répond à la fois à la demande d'un joueur néophyte et/ou occasionnel, qui peut se contenter de se balader dans le monde et chatter avec les autres joueurs, qu'à celle d'un hardcore-gamer qui trouvera son compte dans la quête du meilleur score (tuer le plus d'ennemis possible) ou « la quête de l'arche ».
- Enfin, ce scénario a l'avantage d'autoriser une grande richesse graphique; l'imagination y prend également une place prépondérante.

Inconvénients

- Le principe même du jeu est clairement plus complexe que celui des scénarios qui seront présentés ultérieurement. Or un concept simple est souvent un bon ingrédient pour réussir un jeu ...

- Le peuple de l'Arche répond moins directement à l'attente d'un joueur qui souhaiterait « se défouler » un petit moment. Il demande un peu plus de temps ... avant de devenir complètement accro et déménager en face du kiosque à journaux de la gare Montparnasse pour jouer à PDA !!

3.3.9. Références

Ref 1 : Abzalon et Orchéron, œuvre de Pierre Bordage (SF, éditions de l'Atalante)

http://www.actusf.com/SF/articles/Bordage_orcheron.htm

<http://www.noosphere.com/heberg/bordage/livres/abzalon.htm>

Réf 2 : The Legend of Zelda: A Link to the Past

<http://snes.mobygames.com/game/shots/p./gameId,6608/gameShotId,27759/>

<http://www.zeldatherevolution.fr.st/>

<http://www.zelda-passion.com/alttp/accueil.htm>

3.4. Scenario : Pong Party (Pong multijoueur ad-hoc)

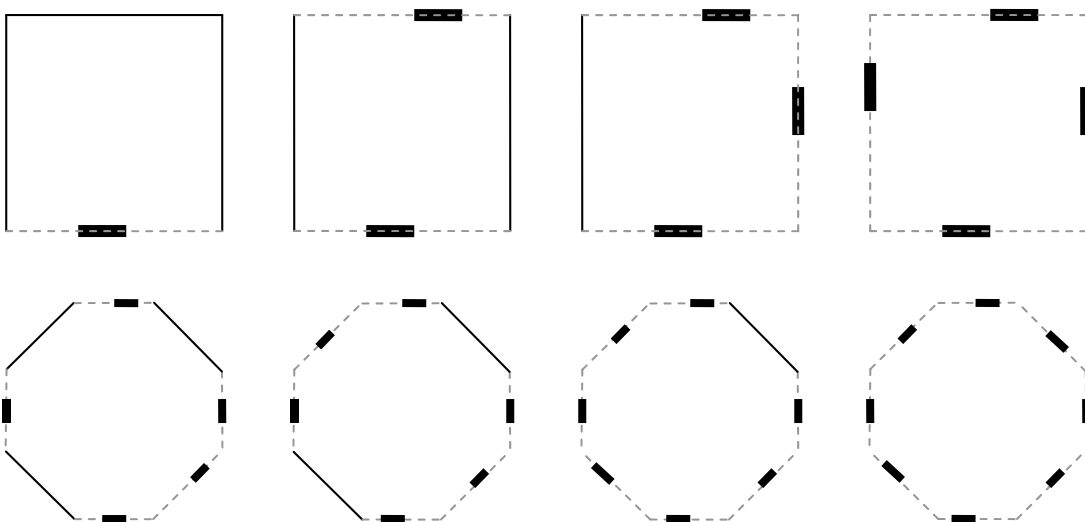


Le jeu Pong original : Shields & Dash jouant au MIT

3.4.1. Principe de base

Pong Party se joue seul ou contre un nombre illimité de joueurs. Les jeux solo et deux contre deux sont similaires au bien célèbre mais néanmoins préhistorique jeu « Ping-Pong ».

Le concept de *Pong Party* est que lorsqu'un joueur entre dans l'arène de jeu, celle-ci change de forme afin de proposer une nouvelle face pour accueillir le joueur. Ainsi, l'arène passe d'une forme carrée à hexagonale, puis 16-xtagone, 32 ... (cf : figure ci-dessous).



Pong Party : Une chambre de jeu de un a huit joueurs

3.4.2. Caractère ad-hoc du jeu

L'exploitation du caractère ad hoc des réseaux pour lesquels *Pong Party* est dédié se fait comme suit : le joueur rejoint automatiquement la partie de *Pong Party* ayant cours entre les possesseurs des PocketPC du réseaux se trouvant distants d'un bond (au sens ad-hoc) dans le réseau.

En d'autres termes, si on choisit de jouer seul au début (dans une arène carrée à trois murs), dès qu'un joueur s'approche suffisamment de vous, celui-ci se voit « parachuté » dans votre arène dont un pan de mur est supprimé. Un affrontement à deux joueurs peut alors commencer.

Si d'autres personnes s'approchent également suffisamment d'un des joueurs de l'arène, ils constitueront autant de nouveaux opposants dans l'arène. Enfin, si on se déplace dans la gare, on peut à tout moment se voir disparaître de l'arène et apparaître dans l'arène d'une autre partie de *Pong Party* ayant lieu à quelques distances de là (entre le tabac journaux et le buffet de la gare par exemple ..). Si plusieurs arènes sont « visibles », un choix aléatoire est réalisé.

3.4.3. Score et profiling

Différentes parties ont donc lieu dans l'enceinte de la gare, avec un nombre de joueurs extrêmement variable. En revanche, un système de score s'accumulant sur l'ensemble des parties (avec stockage dans le profil) et comparant l'ensemble des joueurs en gare est là pour récompenser les bons joueurs.

3.4.4. Lisibilité et fin de jeu

A partir de 9 joueurs, donc lorsque l'arène atteint 16 faces, celle-ci n'apparaît plus en entier sur l'écran afin qu'elle reste lisible. Le joueur ne peut alors surveiller qu'une zone réduite autour de sa raquette. La balle peut alors apparaître à tout moment, ce qui augmente la difficulté du jeu et oblige le joueur à rester concentré !

Notez que dans le mode principal du jeu (cf : Extensions et améliorations) il n'y a pas réellement de fin de partie. Les joueurs entrent et sortent en fonction de la configuration immédiate du réseau ad-hoc et de leur envie de jouer (on peut quitter l'arène à tout moment). Ainsi, le vainqueur est celui qui à un instant donné a le score le plus élevé de toute la gare. Mais attention, il peut être détrôné à n'importe quel moment par des nouveaux joueurs.

3.4.5. Extensions et Améliorations

On peut penser à améliorer le concept de base de *Pong Party* en mettant en place un système de bonus du même type que dans le célèbre *Arkanoïd* ou tous les casses briques en général qui s'en sont plus ou moins inspiré.

Des bonus, sous forme de lettres, tombent à la verticale du plan de la raquette et doivent être touchés par celle-ci pour être activés. Parmi les bonus (et les malus !) classiques, citons l'allongement (ou le rétrécissement) de la raquette, le retour du mur de l'arène derrière la raquette (pour une période limitée !) ...

On peut également étendre le concept avec une autre variante du jeu : un grand nombre de balles sont placées dans l'arène et la fin de la partie (qui dans ce cas existe vraiment) à lieu lorsque toutes les balles sont sorties de l'arène. Les points sont comptés en fonction du nombre de rebonds entre les balles et la raquette du joueur.

3.4.6. Avantages / Inconvénients du scénario

Avantages :

- jeu simple, donc les règles sont facilement assimilables par tous (faire rebondir la balle coûte que coûte) même si le contexte ad-hoc ajoute une nouvelle dimension.
- Le ad-hoc est bien mis à profit et pousse les rencontres : on va chercher des opposants dans la gare pour faire des points !
- *Pong Party* est basé sur l'affrontement direct ce qui peut répondre facilement et rapidement aux éventuelles « pulsions ludiques » de l'utilisateur en gare, contrairement à des jeux plus complexes (ex : Monde persistant) où les sensations ludiques sont peut-être moins directes de prime abord.

Inconvénients :

- jeu simple et peut-être trop pour occuper un stage de six mois. La simplicité est donc un semi avantage pour les futurs joueurs et un véritable inconvénient du point de vue du stage.
- Ping-Pong est-il un concept de jeu qui peut réellement passionner les joueurs et défouler des foules de voyageurs ??



Le concept du jeu Ping-Pong faisant fureur dans les chaumières ... des années 70

3.4.7. Références

Réf 1 : *iPong* game

www.prism.gatech.edu/~gte410r/iPong_proposal.htm

Réf 2 : Site of the 1st video game

www.pong-story.com

3.5. Scenario : Coliseum Party (*Beat them all ad-hoc*)

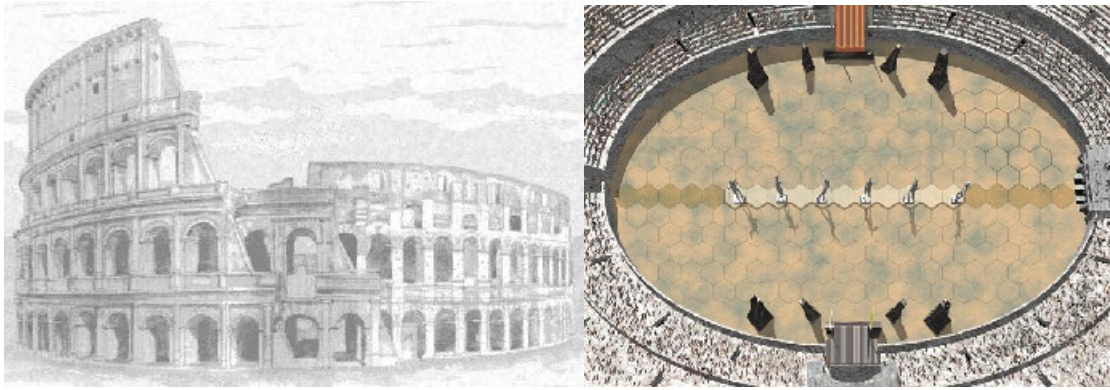


3.5.1. Principe du jeu

Fondamentalement, le principe de *Coliseum Party* se rapproche considérablement de celui de *Pong Party*. Les affrontements entre les joueurs se déroulent également dans des arènes de jeu qui peuvent accueillir depuis un seul joueur jusqu'à un nombre illimité de joueurs qui s'affrontent dans l'optique de se divertir tout d'abord, mais aussi d'obtenir le score le plus élevé de tous les joueurs de la gare.

La différence est qu'ici le jeu ne consiste pas à faire rebondir une balle avec une raquette, mais de « massacrer » des ennemis en tout genre à savoir tout ce qui peut s'aventurer sur l'arène de jeu (C'est pourquoi on appelle ces types de jeu des *Beat them all* c'est à dire des « tape les tous ! »). Les opposants à supprimer seront donc les autres joueurs présents dans l'arène, mais également un certain nombre d'autres monstres divers et variés pilotés par l'ordinateur.

Comme vous l'aurez compris, le terme « arène de jeu » est ici prit au pied de la lettre puisque les combats se déroulent dans des équivalents de l'arène romaine : le Colisée (modélisé en deux dimensions et en vue aérienne). Les joueurs sont donc tels des gladiateurs qui doivent lutter pour leur propre survie. Les armes sont donc les mains, toutes les armes inventées ou utilisées par nos ancêtres les romains (glaives, épées, dagues, fourches, lances, masses d'arme ...) mais aussi d'autres armes innovantes (n'ayons pas peur des anachronismes, l'analogie avec la Rome antique s'arrête à l'arène elle-même).



Le Colisée de Rome – arène de jeu déjà utilisée au temps des Romains, face (gauche) et dessus (droite)

3.5.2. Caractère ad-hoc du jeu

L'exploitation du caractère ad-hoc du réseau pour lequel le jeu est dédié est rigoureusement identique à *Pong Party*. Nous invitons donc le lecteur à se rapporter au paragraphe 3.4.2.

La seule innovation qu'apporte *Coliseum Party* est la possibilité pour le joueur de se placer en spectateur d'une arène. Il assiste alors aux combats, juste pour le spectacle, comme pouvait le faire les 50 000 spectateurs du Colisée romain (source : [Réf 3]) à la différence près qu'il peut modifier son point de vue à volonté en déplaçant sa caméra. On pourra donc se déplacer tout au long de la gare pour assister à divers affrontements et pourquoi pas voter pour les gladiateurs offrant le meilleur spectacle ?

3.5.3. Extensions et Améliorations

La richesse des opposants non-humains en matière d'apparence (fauves, autres gladiateurs, monstres ...), d'intelligence artificielle, d'arme et de technique de combat sont des axes qui permettent de développer considérablement l'intérêt et la complexité du jeu.

Egalement, il est possible de développer énormément la palette d'armes à la disposition des joueurs (blanche, à poing, à feu, ...) et le système de bonus permettant de les acquérir. On peut enfin introduire la magie comme autre type d'arme avec un potentiel magique à recharger et des coups spéciaux qui s'apprennent après un certain nombre de combats victorieux.

3.5.4. Avantages / Inconvénients du scénario

Les avantages et les inconvénients de *Coliseum Party* sont encore à rapprocher avec ceux de *Pong Party*. En revanche, l'inconvénient qui portait sur l'extrême simplicité et surtout sur l'intérêt discutable du principe de jeu est ici beaucoup moins applicable. Même simple, les *Beat them all* et les *Death Match* ont fait leurs preuves et continuent de plus belle à servir de défouloir aux joueurs du monde entier.

La question est plutôt : Un jeu violent répond t'il à la demande de la SNCF ?

3.5.5. Références

Réf 1 : Le Colisée en image

http://www.gmtgames.com/rome/sneakpeek_map.html

Réf 2 : Pour en savoir plus sur la culture Romaine

<http://home.freeuk.com/elloughton13/romans.htm>

Réf 3 : Pour en savoir plus sur le Colisée

<http://www.hmm-162.com/History/news/News2001/2001News006.htm>

3.6. Scenario : Cooperative star-tankers (Shoot them all ad-hoc)

3.6.1. Principe du jeu

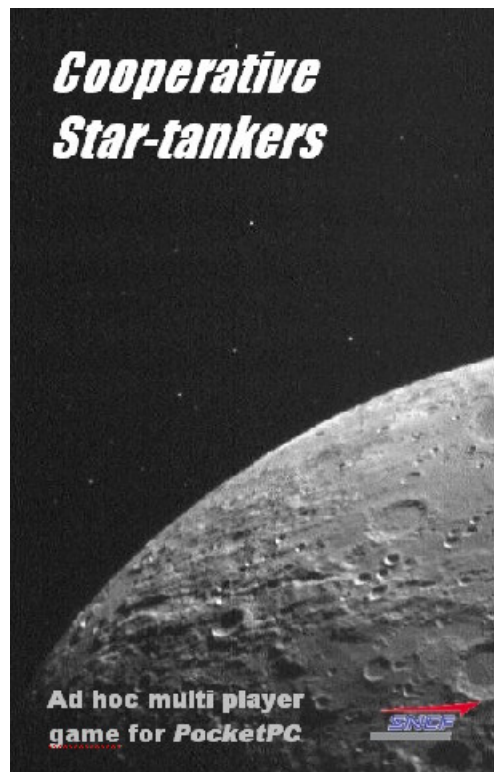
Cooperative Star-tankers est ce que l'on appelle communément un *Shoot them all* (ou *Shoot them up*) en deux dimensions en vue aérienne et à scrolling multidirectionnel avec la particularité peu commune pour le genre d'être massivement multijoueur et, ce qui est une exclusivité, en réseau ad-hoc.

Contrairement à la série des « *Party* » des scénarios précédents, *CST* fait jouer l'ensemble des joueurs dans un seule et même espace de jeu beaucoup plus vaste. L'espace de jeu figure ici le vide inter stellaire quelque part entre la galaxie du *Vieux soleil* et le trou noir de *Ners*.

Aux commandes d'un *Star-tanker*, vaisseau spatial initialement conçu exclusivement pour le transport de marchandises mais équipé pour le combat suite aux périodes de troubles que traversent actuellement l'*UPT* (l'Union des Planètes Terra formées), vous devez impérativement vous défendre afin de survivre dans cet univers pour le moins hostile.

Afin que les marchandises transportées par les *Star-tankers* aient le plus de chances d'arriver à bon port, la société des marchands de l'*UPT* a équipé ses vaisseaux de ses pilotes les plus chevronnés (à savoir les usagers de la SNCF patientant à la gare Montparnasse). Mais elle s'est surtout remémoré un vieux proverbe d'un âge révolu qui disait simplement : « l'union fait la force ». Ainsi, les voyages commerciaux prendront la forme de convois de plusieurs *Star-tankers* qui se protégeront mutuellement contre la faune interstellaire. C'est là que le terme *Cooperative Star-tankers* est apparu.

On dit même que d'autres unions planétaires ou marchandes auraient constitué leurs propres flottes de *Cooperative Star-tankers*. Certain peuples étant pour le moins sauvages, on peut s'attendre à ce qu'ils soient prêts à tout pour conserver leurs marchandises et surtout pour piller et s'approprier celles des autres !



3.6.2. Caractère ad-hoc du jeu

CST est un jeu fondamentalement ad-hoc puisqu'il met en place une coopération entre le joueur et les autres joueurs du réseau proche (au sens ad-hoc, c'est-à-dire situé à un nombre de bonds limité) qui s'opposent aux autres joueurs du réseau global. En analogie avec la série des « *Party* » (Pong et Coliseum), ici l'ensemble des joueurs d'une arène de jeu constitue une équipe de *Star-tankers* qui évolue dans le même espace de jeu que toutes les autres équipes de la gare contre lesquelles elle doit se défendre. Elle doit également répondre aux attaques d'autres vaisseaux ou autre objets volants non identifiés pilotés par l'ordinateur.

Pratiquement, lorsqu'un joueur se connecte à *CST* il a le choix entre faire partie du convoi de *Cooperative Star-tankers* du joueur placé le plus proche de lui dans le réseau ad-hoc, ou former son propre convoi (à *Star-tanker* unique).

Comme pour tous les jeux présentés dans ce document, le profil du joueur permet de stocker son score au cours de ses parties, mais aussi l'argent qu'il peut collecter (à chaque résolution de mission marchande) afin d'upgrader son vaisseau, acheter de nouvelles armes, réparer son vaisseau ...

3.6.3. Extensions et Améliorations

L'univers de *CST* est suffisamment riche pour être étendu et permettre au concept de base de s'étoffer. En effet, au lieu de se « contenter » du vide interstellaire, pourquoi ne pas imaginer la possibilité de survoler des planètes, des stations spatiales, des fermes galactiques ou des élevages de raies stellaires ? On pourrait même penser à des phases de jeu au sol.

Comme pour *Coliseum Party*, le développement des armes ou la customisation du vaisseau permet d'apporter des étapes supplémentaires possibles au développement. (Transformer après beaucoup d'heures de jeu son vieux *Star-tanker* commercial en un *Amiral Star-Cruiser* !).

3.6.4. Avantages / Inconvénients du scénario

Avantages

- ce scénario tire excessivement bien partie du caractère ad-hoc du réseau pour lequel il est destiné et propose le concept innovant de coopération (également présent dans le projet de monde persistant).
- En se basant sur l'affrontement direct, il est avant tout un jeu d'action qui peut répondre à l'attente d'un joueur occasionnel qui cherche à se défouler quelques instants tout comme à celle d'un *hardcore-gamer* qui plantera encore une fois sa tente dans sa gare SNCF préférée pour jouer à *CST*.

- L'univers est suffisamment riche et le principe suffisamment souple pour étendre la période de développement à l'infini et donc satisfaire largement les cinq mois de stage restant.
- Le travail graphique nécessaire est tout à fait satisfaisant (Contrairement à *Pong Party* par exemple).

Inconvénients

- Le principal problème de ce scénario est qu'il nécessite une certaine masse critique d'utilisateurs. En effet, si l'ensemble de l'espace interstellaire n'est constitué que de deux ou trois convois chacun contenant une poignée de *ST*, le jeu se résumera vite à un *shoot them up* classique et même bas de gamme en raison de l'absence de niveau. Le seul « but » sera de détruire les quelques vaisseaux extraterrestres pilotés par l'ordinateur.
- le jeu reste moins riche qu'un monde persistant mais il souffre surtout de la comparaison car il n'inclue pas l'aspect « relation humaine » (avec le chat) et « tourisme » (avec le monde) des mondes persistant.

3.6.5. Références

Réf 1 : Archives et musée du shoot them up (= shoot them all)

<http://www.shmup.com/index.php?page=pourquoi>

Réf 2 : Un site à hacker ...

<http://www.nasa.gov>

3.7. Bilan sur les scénarios présentés

Le 7 août 2003, s'est tenu une réunion bilan au département *TIC* - nouvelles technologies de l'information et de la communication - de la *DRT* - direction de la recherche et de la technologie de la *SNCF*. Les travaux des chapitres précédents y ont été présentés. Présent à cette réunion, *Jean-Noël Temem* Directeur du département, *David Sanz* chercheur à la *DRT* et responsable de ce stage, *Sylvain Gitton* chercheur, *Shérine Khoury* stagiaire de *Télécom Paris*, et moi-même.

Globalement c'est le principe de monde persistant qui a été retenu, contrairement aux jeux basés sur l'affrontement - *Death Match* - tel que *Coliseum Party* (cf 3.5). En revanche, la principale remarque qui a été émise sur les scénarios est leur lien assez faible avec l'univers de la gare et du transport ferroviaire. D'une façon générale, le jeu doit tenir plus en compte le contexte de l'utilisateur : il est en gare, mobile et dispose souvent de peu de temps. Cette contrainte a effectivement été omise dans les contraintes exposées au début de ce chapitre. Le scénario final y répondra de manière beaucoup plus directe.

Il a été demandé en outre que le jeu tire plus partie de la spécificité des nœuds faiblement mobiles du réseau ad-hoc. En effet, le réseau ad-hoc du projet *SAFARI* est constitué d'un ensemble de nœuds mobiles - les terminaux utilisateurs -, mais également de nœuds dit « faiblement mobiles » qui sont en fait des nœuds fixés dans différents emplacements de la gare (face à l'accueil, devant le tabac-journaux, face au quai 7 et 8 ...). Grâce à ces nœuds particuliers, il est possible d'effectuer une localisation spatiale approximative des terminaux du réseau.

En effet, lorsqu'un nœud faiblement mobile détecte un nœud mobile dans son entourage proche (c'est-à-dire à un bond au sens du réseau ad-hoc), il peut lui indiquer la zone dans laquelle celui-ci se positionne. Le jeu peut donc tirer parti de cette information spatiale et ainsi avoir un lien plus proche avec l'univers de la gare. On peut alors imaginer un jeu de pistes dans la gare poussant les joueurs à se déplacer d'une zone à une autre de la gare, des points de rechargement ou des trous noirs décimés à différents endroits ...



Hall de la gare *S^t Lazare* à Paris, à quelques mètres des locaux de la *DRT*.

4. Scénario final : Treenor, railway piracies



4.1. Introduction

Par sa position stratégique, au milieu des deux superamas colonisés par l'espèce humaine, la planète *Treenor* s'est imposée comme un centre économique majeur de l'Union (UPT - Union des Planètes Terraformées). Sa taille imposante a permis de transformer la planète en une gigantesque station de transit, plaque tournante inévitable pour toutes les marchandises acheminées par convois ferrés de l'espace aux quatre coins des galaxies de l'Union.

Treenor est donc une pièce angulaire du système économique et commercial de la civilisation humaine. Mais cette position centrale est également synonyme de fragilité. En effet, toutes les ressources échangées sur la planète attirent les convoitises et les convois de l'espace eux-mêmes sont régulièrement attaqués, victimes de pirateries et de pillages à répétition.

Auteurs de ces piratages intergalactiques, des rebelles, qu'il est commun d'appeler les « non-unifiés » contestent l'hégémonie de l'union. En réponse aux attaques de ces pirates, le gouvernement de l'*UPT* offre de très fortes récompenses à qui en débarrassera la planète et l'espace. Des primes sont également allouées aux pilotes qui escorteront les convois jusqu'à leur destination afin de les protéger des attaques.

Treenor : railway piracies propose aux joueurs d'incarner, au choix, un chasseur de primes citoyen de *Treenor* travaillant pour le compte de l'union ou un « non-unifié » pillard.

4.2. Les phases de jeu

Deux phases de jeu sont à dissocier dans *TRP* : celles qui se déroulent sur la planète *Treenor* elle-même et celles qui se déroulent dans l'espace.

4.2.1. Phases treenoriennes

Pendant les phases treenoriennes, le joueur dispose d'un avatar qu'il déplace dans la cité marchande de *Treen*, capitale de *Treenor*. Pour donner une idée générale de l'apparence de cette ville, on dira quelle s'apparente à une version moins désertique de la ville de *Bestine*, capitale de la planète *Tatooine* du film *Star Wars* [Réf 1].

L'action se déroule principalement sur la place centrale de la ville, gigantesque marché et gare de transit pour les marchandises extra-treenoriennes. Dans ce lieu, on trouve les éléments suivants :

- Des plateformes d'embarquement/débarquement pour les convois ferrés de l'espace. Telles des pistes de décollage, ces plateformes sont prolongées par de longues portions de rails qui s'élèvent vers le ciel à leurs extrémités. Les convois spatiaux s'élancent sur ces voies avant de s'envoler dans le vide interstellaire vers leurs destinations respectives.

On retrouve ce concept de transport ferroviaire de l'espace dans des œuvres de science fiction telles que celle de *Matsumoto Leiji*, auteur du film d'animation *Galaxy Express 999* (mais aussi de la série animée *Albator*) sorti en 1978 au Japon [Réf 2-3]. La figure suivante illustre les plateformes de lancement imaginées par *Leiji*.

- Des bars, espaces de rencontre, lieux de prédilection pour que les joueurs dialoguent entre eux grâce à un système de bulle s'affichant au dessus de leurs avatars respectifs. Certains disposent même de pièces plus exigües dans lesquelles les conversations peuvent rester plus discrètes ...
- Des magasins d'armement, où il est possible de se procurer de nouvelles armes treenoriennes, mais également de réparer ou d'améliorer la puissance offensive et les performances de son vaisseau (cf : phase spatiale).



Plateforme de lancement du train Galaxy Express 999 dans le manga du même nom (M. Leiji - 1978)

- Des commerces à ciel ouvert, où les voyageurs de tout l'univers viennent se procurer les marchandises acheminées par convois. Les commerçants treenoriens - pilotés par l'ordinateur - ne font pas la fine bouche sur la provenance des marchandises qu'ils achètent. Ils rachèteront avec plaisir les marchandises que les chasseurs de primes auront récupérées des bras des pillards, mais seront également tout à fait intéressés par les marchandises volées par les pirates ...

4.2.2. Phases spatiales

Pendant ces phases, les joueurs sont aux commandes de leurs vaisseaux spatiaux armés et suivent un convoi en particulier, depuis l'embarquement des marchandises, au cours de son voyage interstellaire et jusqu'à son entrée en hyperspace. La fin d'une phase spatiale se traduit soit par le passage en hyperspace du convoi, soit par son anéantissement.

Pour entrer dans ces phases, les avatars doivent se présenter devant les plateformes d'embarquement des convois. Le but de chacun des joueurs dépend alors du type de personnage qu'il a choisi.

4.3. Chasseur de primes ou pirate ?

Le choix entre les deux types de personnage s'effectue à la première partie, lors de la création de l'avatar où l'on choisit également un nom, un sexe, une apparence, ... Les choix effectués sont alors stockés dans le profil pour les sessions ultérieures. On dispose alors d'un crédit de trois vies. Si celui-ci est épuisé, la partie est terminée mais

le personnage créé ne disparaît pas. Il perd tous ses crédits, bonus, armements ou upgrades et repart avec un capital de trois vies. C'est au début de chaque partie qu'il est alors possible de changer de camp.

Intéressons nous à présent aux buts respectifs et aux différentes interactions qui peuvent avoir lieu entre ces deux familles de joueurs, et ceci, au cours des deux phases de jeu possibles.

4.3.1. Sur Treenor

Sur la planète, rien ne permet de distinguer à priori un pirate d'un chasseur de primes. Ce n'est qu'avec le dialogue, en faisant confiance à son interlocuteur ou en menant soit même sa mini enquête que l'on peu deviner le camp auquel appartient tel ou tel autre joueur.

Les chasseurs de primes et les pirates peuvent faire usage de leurs armes sur *Treenor*. Les chasseurs obtiennent en effet la prime de l'*UPT* pour tout pirate abattu. Les pirates, quand à eux, dépouillent leurs victimes du camp adverse et s'emparent de leur argent. Mais attention, le meurtre est sévèrement puni dans la ville de *Treen* et si un chasseur de primes en supprime un autre par erreur, il encourt une sévère amende. De la même manière, les meurtres entre pirates sont mal perçus par la confrérie et les traîtres sont dépouillés et exclus. Ainsi, les rues de *Treen* seront d'avantage un espace de dialogue qu'un no man's land où l'on risque sa vie à chaque recoin.

4.3.2. Dans l'espace

Dans l'espace, la motivation principale des chasseurs de primes est de toucher la grosse récompense que l'Union offre aux pilotes protégeant les convois spatiaux. Ils ont donc tous intérêt à se lier pour protéger le convoi ferré qui n'est pas invulnérable. La prime est distribuée équitablement à chaque chasseur à la condition que le train de l'espace effectue sans encombres son saut dans l'hyperespace.

Les pirates en revanche, cherchent à anéantir complètement le convoi afin de se partager l'imposant butin qu'il représente. Ils ont donc également tous intérêt à agir de concert mais cette fois, pour empêcher le saut hyperespace et détruire le convoi.

Parallèlement à ces gains importants acquis de manière collective et équitablement distribués à chacun des membres d'un clan, de l'argent peut être gagné indirectement dans les phases spatiales et de manière individuelle.

En effet, lorsqu'un wagon est partiellement détruit, les containers de marchandises qu'il contient sont alors apparents. Les pirates peuvent alors en prendre possession et les conserver dans leur vaisseau. S'ils ne se font pas détruire tous au long de la phase spatiale, ils pourront revendre leurs marchandises aux commerçants de *Treen* à leur retour sur *Treenor* et en retirer une certaine somme.

Les chasseurs de primes quant à eux, n'ont aucun intérêt à voler les marchandises des convois qu'ils protègent. En revanche, lorsqu'ils détruisent un vaisseau pirate, la marchandise volée que celui-ci pouvait transporter reste en suspend dans le vide

interstellaire et peut être récupérée par un chasseur. Celui-ci pourra, à son retour sur *Treenor*, rapporter la marchandise au marchand qui le récompensera alors.

4.4. Un concept de jeu original

Treenor : railway piracies est un projet qui se positionne entre le scénario du *Peuple De l'Arche* (cf : 3.3) et celui de *Cooperative Star-Tankers* (cf : 3.6). Le jeu est donc un mélange entre monde persistant - phases treenorriennes - et shoot them all coopératif - phases spatiales -.

Le lecteur remarquera que si le concept ludique de monde persistant est un lieu commun (cf : 1.3.1), celui de shoot them all coopératif apporte déjà un caractère inédit au jeu. Mais sa véritable force, ce qui le rend unique, est l'exploitation qu'il fera du caractère ad-hoc du réseau pour lequel il est dédié et surtout le lien fort qui en découle entre l'univers imaginaire de *Treenor* et la gare *SNCF* dans laquelle le réseau est déployé.

4.5. Aspect ad-hoc et lien avec la gare

Le jeu est en fait principalement basé sur l'information spatiale qui peut être tirée de la mise en œuvre de nœuds faiblement mobiles dans le réseau ad-hoc du projet *SAFARI*.

Revenons brièvement sur cette notion : le réseau ad-hoc du projet est constitué d'un ensemble de nœuds mobiles - les terminaux utilisateurs -, mais également de nœuds dit « faiblement mobiles » qui sont en fait des nœuds fixés dans différents emplacements de la gare (face à l'accueil, devant le tabac-journaux, face au quai 7 et 8 ...). Lorsqu'un nœud faiblement mobile détecte un nœud mobile dans son entourage proche (c'est-à-dire à un bond au sens du réseau ad-hoc), il peut lui indiquer sa présence. Le nœud mobile peut alors en déduire sa position approximative si la position des nœuds faiblement mobile dans la gare est connue.

Ce procédé a déjà été retenu à la suite de réflexions menée conjointement avec *Alcatel*. Des nœuds particuliers du réseau (que nous appellerons maîtres) permettent donc de délimiter des zones fixes dans la gare (différents quais, tabac journaux, ...) dans lesquelles les terminaux utilisateurs pourront avoir conscience d'être présent. On utilise également dans le jeu les informations horaires des trains arrivant dans la gare *SNCF* où le réseau est mis en œuvre.

L'univers virtuel de *Treenor* est ainsi mis fortement en parallèle avec l'environnement réel de la gare *SNCF*. En effet, les départs et les arrivées sur *Treenor* des convois ferrés de l'espace sont synchronisés avec ceux des trains *SNCF* dans la gare. Ensuite, pour suivre un convoi dans l'espace – phase spatiale –, il faut que l'avatar treenorien du joueur se présente devant les plateformes d'embarquement ET que le joueur se place devant le quai de la gare qui lui correspond, c'est-à-dire dans la zone de proximité (à un bond au sens ad-hoc) du nœud faiblement mobile du quai.

De même, le hangar et magasin d'armement n'est accessible que pour les joueurs se positionnant à proximité du tabac-journaux de la gare. La place principale de *Treen*,

marché et gare de transit pour les marchandises de l'Union, est donc agencée de manière similaire à la gare *SNCF* pour en constituer un véritable monde parallèle. Le joueur en gare, nœud mobile du réseau ad-hoc, est donc par défaut sur la planète *Treenor*. Lorsqu'un train *SNCF* arrive à un quai, un convoi spatial débarque également sur *Treen*. Le joueur et l'avatar peuvent alors rejoindre de concert leur version respective, virtuelle ou non, du train.

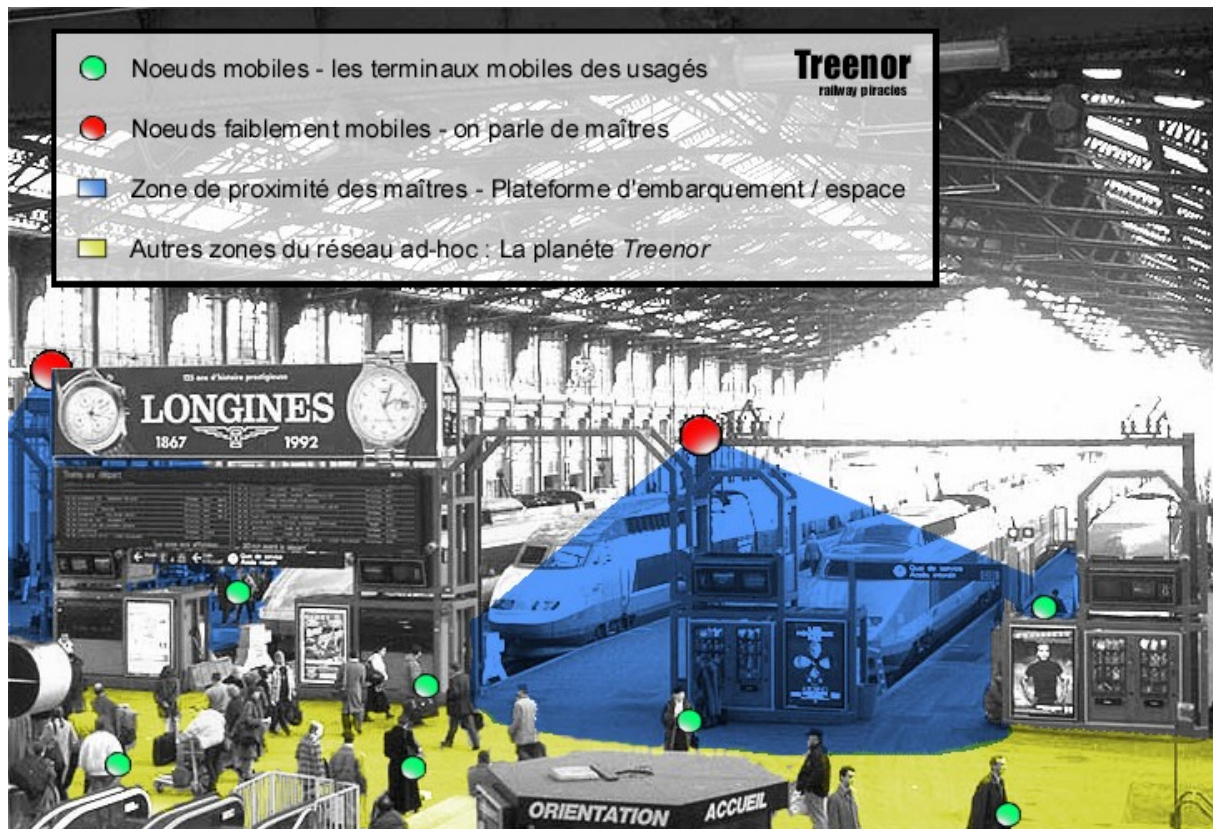
Une fois le joueur à quai et l'avatar devant la plateforme d'embarquement, le jeu passe en phase « shoot » et c'est aux commandes de son vaisseau que le chasseur de primes protégera le convoi spatial pendant la phase d'embarquement ou que le pirate cherchera à le piller. Il y a donc affrontement entre les joueurs chasseurs de primes et les joueurs pirates se trouvant au même quai. Si les effectifs des clans sont mal équilibrés, ou pire, qu'un seul d'entre eux est représenté, des pirates ou chasseurs pilotés par l'ordinateur (et commandés par le nœud maître) sont ajoutés au champ de bataille.

Lorsque le train *SNCF* démarre effectivement, le convoi part également pour son voyage interstellaire. S'il n'est pas détruit par les pirates, il effectuera son saut dans l'hyperespace quatre minutes après son départ. La destruction ou le saut signifie la fin de la partie pour tous les joueurs positionnés sur le quai. Ils retrouvent alors leurs avatars respectifs qui se posent de nouveau sur *Treenor*.

Chacun des trains arrivant en gare sont donc autant de niveaux du jeu où les joueurs s'affrontent en équipe - *Multiplayer Death match*. Tandis que la gare *SNCF* fait office de monde persistant - *MMPG*. C'est le concept de *Treenor*.

A noter que l'apparence des convois de l'espace et leur destination sont fonction de la destination des trains *SNCF* qui leur correspondent (Nantes, Rennes, Toulouse, ...)

La figure suivante illustre l'exploitation de l'architecture du réseau ad-hoc décrite précédemment permettant de mettre en parallèle l'univers de la gare et celui du jeu :



Exploitation de l'architecture ad-hoc : Parallèle spatial entre l'univers de la gare SNCF et celui de Treenor

4.6. Aspect techniques

- L'écran est orienté horizontalement afin d'offrir une meilleure visibilité latérale.
- Les déplacements de l'avatar sur *Treenor* et les mouvements du vaisseau dans l'espace sont gérés avec l'ensemble pavé centrale et boutons du *PocketPC*. Une alternative pour l'avatar consiste à utiliser le crayon pour pointer la direction vers laquelle on désire qu'il se dirige. La technique utilisée pour que celui-ci suive une trajectoire intelligente (évitement des obstacles, plus court chemin, ...), appelé *path-finding*, sera conforme aux travaux de Fabrice Rossi - Chercheur au Lamsade, Université Paris IX Dauphine - et basé sur un algorithme A* allégé.
- Les comportements des ennemis pilotés par l'ordinateur seront décrits sous forme d'un script, comme c'est le cas pour une grande majorité de jeux, et ne feront donc nullement appel à des techniques d'intelligence artificielle ou collective. En effet, ces techniques peuvent être lourdes et complexes. Elles ne seront donc employées dans ce projet que pour résoudre le problème du *path-finding*.
- Comme il a été signalé précédemment, un profil du joueur est généré et stocké sur son terminal afin de conserver entre deux sessions de jeu les données des joueurs : Nom, apparence de l'avatar, capacité du vaisseau, armement, argent, score ...
- Le nom (=login) des joueurs que l'on rencontre s'affiche pendant une seconde au-dessus de leur tête lorsqu'ils apparaissent à l'écran.

Du point de vue graphique et pour la modélisation de la planète *Treenor* à l'écran, on envisage le même procédé que dans le Peuple de l'Arche : la technique du *Tilling* (cf : 3.3.6).

Pour l'arrivée et le départ des convois de l'espace, ce sont les nœuds maîtres qui disposent des informations horaires des trains. Dans le cadre de ce stage, les nœuds maîtres pourraient ne disposer que d'informations de trafic prévisionnelles. Pour la démonstration *SAFARI* prévue à la Gare Montparnasse de Paris en 2005, il faudrait prévoir une réactualisation automatique (par exemple toutes les vingt minutes) de ces informations à partir d'un serveur se trouvant dans le réseau filaire de la gare par le biais de la passerelle faisant le lien entre le monde ad-hoc et le monde filaire.

Enfin, puisque le réseau ad-hoc du projet *SAFARI* n'est pas encore une réalité commerciale et reste un sujet de recherche, nous ne sommes pas certains du débit effectif qu'il sera possible d'obtenir dans l'architecture réseau nécessaire au jeu. On pense surtout à deux avatars se déplaçant à proximité sur *Treenor* (chacun voit l'autre se déplacer en temps réel sur son écran) alors qu'ils sont chacun à l'opposé de la gare et donc à deux extrémités du réseau ad-hoc (les temps de transfert pouvant être très variables ...).

Pour les phases spatiales, le fait d'être certain que les nœuds mobiles peuvent tous communiquer entre eux par deux bonds en passant par le maître, permet d'espérer un temps de réponse plus rapide et uniforme pour tous les joueurs.

Le jeu est réalisable comme il a été décrit précédemment si les débits entre tous les terminaux du réseau, quelque soient leurs positions dans la gare, sont assimilables à un débit de réseau *WIFI* (Wireless Fidelity) ou *LAN* (Local Area Network) classique. En effet, le jeu de shoot et la partie planétaire exigent que chaque terminal envoie la position de l'avatar ou du vaisseau en temps réel aux autres nœuds.

Dans le cas où il ne s'avérerait pas réalisable d'obtenir ces débits en dehors des zones de proximité des nœuds maîtres, les phases treenoriennes seraient modifiées comme suit : les avatars se déplacent toujours dans *Treen* mais n'y apparaissent plus les avatars des autres joueurs. On peut toujours se balader et se diriger vers les magasins, point d'embarquement ou bars de la ville, mais *Treenor* n'est plus qu'une interface entre les autres points forts du jeu et non un monde persistant. L'interaction entre les joueurs pour cette phase se jouerait alors dans les bars qui seraient en fait des chats traditionnels en mode texte (Pas besoin d'un débit très élevé !).

Si les débits sont également trop faibles pour les phases spatiales, il faudra remplacer les combats *shoot them all* en combats au tour par tour. Ce type de combat a été directement emprunté aux jeux de rôle traditionnels où les joueurs s'affrontent autour d'une table à l'aide de dés et de règles particulières. Les jeux vidéo utilisant ce principe, appelés *Turn based game*, sont nombreux et toujours au goût du jour. Citons par exemple la série bien connue des *Final Fantasy* qui a conservé ce système de combat depuis la première version parue en 1987 sur console *Nintendo NES* jusqu'à la toute dernière version sur *Playstation 2* [Réf 4].



Turn based games : les plus célèbres, Final Fantasy opus II, III, IX et X de Square Soft

L'intérêt de ce type de combat, dont le lecteur intéressé trouvera une définition précise en [Réf 4], est en effet que le débit qu'il nécessite est assez faible et comparable à celui d'un chat : on envoie tour à tour sa commande offensive.

4.7. Evolutions possibles

Enfin, le concept de *Treenor*, *railway piracies* a été imaginé de façon à pouvoir évoluer, que ce soit au niveau graphique, des fonctionnalités ou même des lieux de déploiement. En effet, dans le cas où les réseaux ad-hoc viendraient à se généraliser, on peut bien imaginer un déploiement sur toutes les grandes gares SNCF par exemple et surtout faire en sorte qu'il reste opérationnel sur les lignes de train elles-mêmes.

Ainsi, il n'y aurait plus de sauts dans l'hyperespace des convois spatiaux et les joueurs continueraient à défendre ou attaquer le train tout le long de leur voyage ! A la gare d'arrivée, ils entreraient sur une autre planète marchande (carte et données à télécharger à l'arrivée) où ils rencontreraient de nouveaux joueurs, fréquenteraient d'autres bars, pourraient échanger leurs marchandises volées à d'autres cours ...

4.8. Conclusion

En conclusion, on dira que ce projet est le fruit des travaux et réflexions qui ont été menés pour réaliser les quatre scénarios précédents et effectuer un mélange des genres intéressant pour proposer un concept de jeu original.

En effet, rappelons tout d'abord que le concept ludique de *shoot them all* multi-joueur coopératif est déjà assez peu commun. Ce concept est de plus couplé avec celui des mondes persistants, ce qui rend le principe de jeu unique. *Treenor* est enfin et surtout une exclusivité puisqu'il s'agit d'un des premiers jeux vidéo tirant partie des caractéristiques des réseaux ad-hoc. Il est également unique puisqu'il en profite pour tisser un lien entre un univers imaginaire et la réalité, liant l'emplacement physique du joueur dans la gare avec la vie propre de l'avatar dans le jeu.

Treenor : railway piracies, profitant des caractéristiques des réseaux ad-hoc, met l'environnement de la gare et le monde du transport ferroviaire en son centre, ce qui répond aux attentes de l'entreprise et en fait une exclusivité SNCF.

4.9. Références

4.9.1. Références indexées

Réf 1 : Star Wars Databank : Tatooine

<http://www.starwars.com/databank/location/tatooine/eu.html>

Réf 2 : Critique du film *Sayonara Galaxy Express 999* de Matsumoto Leiji

<http://www.cinemasie.com/fr/fiche/oeuvre/sayonaragalaxyexpress/noscritiques.html>

Réf 3 : Critique du film *Galaxy Express 999* de Matsumoto Leiji (1978)

<http://www.cinemasie.com/fiche/oeuvre/galaxyexpress999/noscritiques.html>

Réf 4 : Turn-based game definition

http://www.wikipedia.org/wiki/Turn-based_game

Réf 5 : The World of Final Fantasy

http://www.mobygames.com/featured_article/feature,18/section,114/

4.9.2. Références additionnelles

Réf : The Extrasolar Planets Encyclopaedia

<http://www.obspm.fr/encycl/encycl.html>

Réf : Fiche technique de la planète Epsilon Eridani, alias HD 22049

<http://cfa-www.harvard.edu/planets/eps-Eri.html>

Réf : Le jeu Playstation : Galaxy Express 999, Story of Galaxy Express

http://www.asylum-anime.com/games/psx_ge999

5. Développements préliminaires et parallèles

5.1. Choix du langage de programmation

D'après la fiche de stage rédigée début 2003 (cf : 1.4.1), les développements informatiques étaient initialement prévus en *Java*.

Java est un langage de programmation orienté objet conçu par *Sun Microsystems*. Sa caractéristique principale est d'être un langage semi-interprété. En effet, contrairement aux langages classiques, il est pré- compilé dans un pseudo-langage bien connu - le bytecode - puis interprété à l'exécution par une machine virtuelle. Ce mécanisme rend les codes *Java* indépendants de la plate-forme sur lesquels ils sont exécutés (Windows, Linux, téléphone portable, Pocket PC, calculatrice, puces électroniques ...) au prix d'un substantiel surplus de calcul qui rend les programmes *Java* en général plus lents que les programmes issus d'autres langages tels que le C ou le C++.

Dans le projet *SAFARI*, la *SNCF* fait partie du sous-projet SP5 (cf : 1.3.5) et prévoit de s'appuyer sur la plate-forme de services basée sur *Java* développée par *ALCATEL* pour concevoir ses démonstrateurs. *Treenor : railway piracies*, faisant partie de ces démonstrateurs, devait donc initialement être écrit en *Java*.

Au regard de la relative lenteur de *Java* - les jeux vidéo sont des applications très gourmandes en puissance de calcul, concevoir *Treenor* en C++ s'est avéré être un choix plus pertinent. Enfin, à l'issue de l'état de l'art dressé dans le chapitre 2, c'est la librairie multimédia *SDL* (cf : 2.4.1) qui fut retenue.

Enfin, remarquons que tous les développements furent soumis par un logiciel de suivi de version de type CVS – *Subversion*.

5.2. Développements préliminaires : TRPong

5.2.1. Présentation générale

Le *TRPong* ou *Treenor Pong* est le premier jeu qu'il m'a été donné de réaliser dans le cadre de ce stage. L'objectif de ce jeu simple était de mettre en œuvre les mécanismes de communication sans fil entre deux *PocketPC*. Il reprend le principe du scénario *Pong Party* présenté en 3.4, et décline dans une version spatiale l'ancestral jeu « Pong ».

La version finale de *Treenor Pong* permet à un joueur d'affronter l'ordinateur ou à deux joueurs de s'affronter en réseau. L'apparence des raquettes est paramétrable et les décors spatiaux sont fournis.



Le Treenor Pong sur un PocketPC iPAQ H3600

5.2.2. Réalisation technique

Comme tous les projets de ce stage, le *Treenor Pong* a été réalisé en C++ et à l'aide de la bibliothèque multimédia *SDL*. En tant que premier projet, il fut surtout une occasion de jeter les bases du développement, qui plus tard, formalisées et conceptualisées, donneront naissance au *TREngine* (cf : chap 6).

Des éléments de bases

Parmi les processus que ce projet a permis d'appréhender puis de mettre en œuvre, citons : la compilation de programmes C++ sur PC pour architecture PocketPC – on parle de cross-compilation, mais aussi l'initialisation d'un environnement graphique, l'accès à la mémoire vidéo, les mécanismes de recopie accélérée de données graphiques à l'écran (« blit de surfaces »), la gestion des couleurs dans différents modes graphiques (24 bits, 16 bit ou 256 couleurs), la gestion de la fréquence de rafraîchissement, la gestion des événements système, le tracé de figures simples (lignes, rectangles, cercles), la gestion de l'interface *PocketPC* ... Un système de « Font » - affichage graphique et dynamique de textes - a également été développé.

Ce projet a été également l'occasion de mettre en œuvre les routines et mécanismes de communications réseaux qui seront réutilisés par la suite dans la partie spatiale du jeu final. Les développements s'appuient sur la librairie réseaux *SDL_Net*, et permet une communication *UDP IPv4* ou *v6* entre deux terminaux joueurs.

Information distribuée et *Dead-Reconning*

Enfin, *Treenor Pong* a permis de soulever des problèmes non triviaux et inhérents aux jeux en réseaux. En effet, à une fréquence fixe et en mode deux joueurs, chaque terminal transmet à son opposant la position propre de sa raquette. Comme chacun dispose de la position des deux raquettes à chaque instant, on peut imaginer que la trajectoire de la balle soit calculée par chacun des terminaux indépendamment – on parlera de calculs « en local ».

Malheureusement, on ne peut obtenir un seul et même jeu cohérent sur les deux terminaux avec cette logique. En effet, un décalage peu s'induit entre les deux jeux pour de multiples raisons : une latence excessive du réseau - temps de transmission

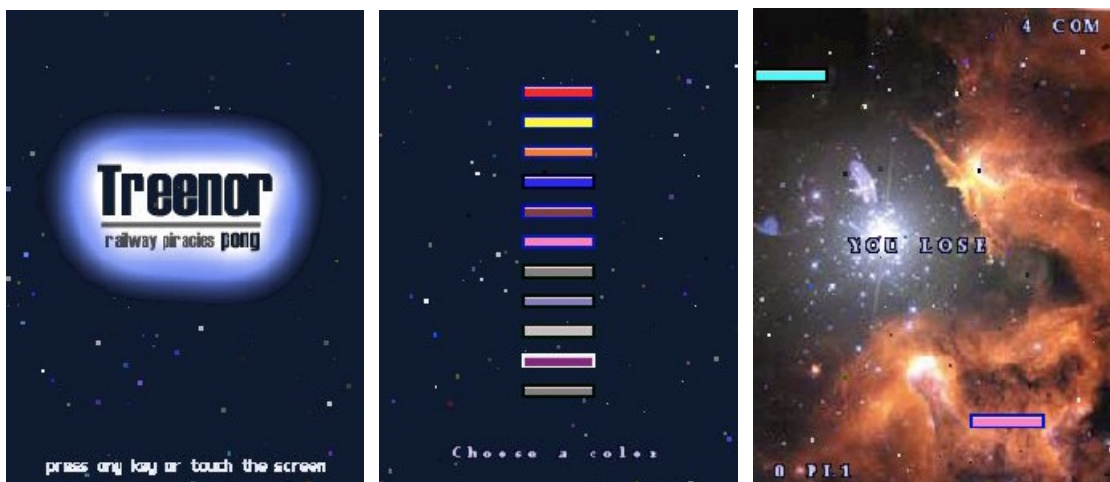
d'un paquet d'informations, une perte importante de paquets sur le réseau - UDP est un protocole de communication avec perte, un ralentissement ponctuel ou prolongé d'un des systèmes ...

La position de la balle est typiquement une donnée dont chaque joueur du réseau doit disposer, au même moment et avec la même valeur, mais que personne ne peut « calculer soi-même ». On parle alors d'*information distribuée*. Dans les réseaux classiques, on retient pour ce type « d'information communautaire » une architecture de type client/serveur ou un serveur est chargé de calculer puis diffuser ces informations à ses clients. Dans le cadre d'un réseau ad-hoc, et d'avantage dans celui d'un jeu *Peer-to-Peer* comme le *TRPong*, on ne peut pas compter sur la présence d'un serveur.

Pour résoudre ce problème, la solution retenue fut la suivante : la position de la balle est calculée en local par un terminal élu maître et transmis au terminal opposant. Par convention, le maître est toujours celui venant d'effectuer le service et change donc à chaque remise en jeu. Ainsi par exemple, si le maître décide d'un rebond de la balle sur la raquette adverse, celle-ci rebondira également au même moment sur le terminal de l'opposant même si sur son écran un énorme décalage rendait le rebond non justifié.

Enfin, le *Treenor Pong* met en œuvre de façon simple les techniques de *Dead reckoning*. L'objectif de ses techniques propres aux NVE - Networked Virtual Environments - est de compenser l'impact du décalage induit par le temps de transfert des informations sur le réseau entre toutes les instances d'un environnement partagé. En pratique dans le jeu, le maître envoie à l'instant t les coordonnées de la balle à l'instant $t+L$. Si L est le temps de transfert d'un paquet d'information sur le réseau - latence du réseau, le terminal de l'adversaire recevra à l'instant $t+L$ les coordonnées de la balle en $t+L$ et affichera ainsi des données cohérentes.

5.2.3. Résultat en images





Treenor Pong - copies d'écran : introduction, options, phases de jeu, « game over »

5.3. Développements parallèle : Le *TREngine*



TREngine - logo

5.3.1. Présentation générale

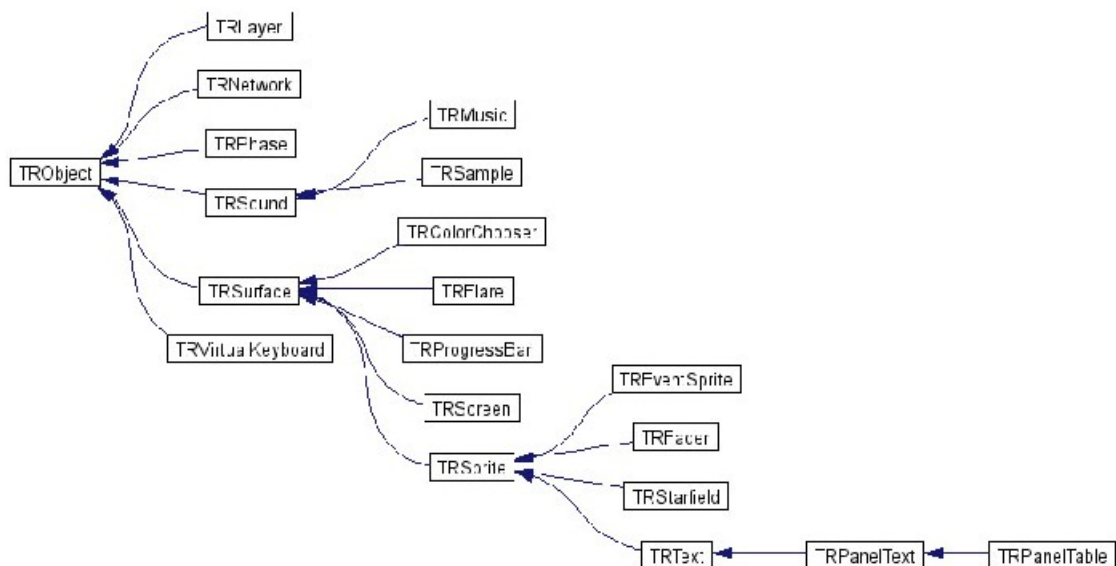
Pendant le développement du *Treenor Pong*, j'ai dissocié les mécanismes généraux communs à tous les jeux (Initialisation de l'écran, routines d'affichage, gestion de la souris, mécanismes de communications, effets spéciaux ...) de ceux propres au projet lui-même dans des classes bien distinctes afin qu'ils soient plus facilement réutilisables pour les développements futurs d'applications ludiques *SNCF*.

De ce travail est né, avec l'accord de mon maître de stage, le *TREngine*.

Le *TREngine* est donc un ensemble de classes *C++* - on parle de *framework* - ayant pour objectif de faciliter le travail des programmeurs réalisant un jeu vidéo en deux dimension sur PC, PocketPC, Mac, Solaris, Amiga, console GBA, Playsation 2, Xbox, et toute les autres plateformes compatibles avec la librairies multimédia *SDL* [Réf 7].

Si ce type de *framework* existe déjà dans une dizaine de versions sous licence *GPL* (citons *GameBlade* et *Adgali*, cf : chap 1), la singularité du *TREngine* - et ce qui m'a poussé à le réaliser - est son implémentation fortement orientée objet. En effet, la plupart des frameworks sont, soit écrits en *C* - comme *SDL* - et donc non orientés objet, soit de conception relativement « monolithique » et/ou relativement « bas niveau ».

Le *TREngine* se propose de fournir les outils les plus génériques et complets possibles pour la réalisation de projets ludiques fortement conceptualisés et hiérarchisés. Son développement a commencé avec le projet *TRPong*, mais s'est prolongé en parallèle de tous les autres travaux de développement composant ce stage. Enfin, le *TREngine* n'est pas statique. Il devrait être amené à évoluer bien au delà de ces six mois de stage, et ceci, particulièrement s'il devient un projet *GPL* à part entière.



TREngine v1.5, framework de développement de jeu vidéo en 2D orienté objet – diagramme de classes

5.3.2. Principales fonctionnalités

Sans constituer une documentation technique détaillée, ce chapitre se propose de faire un tour d'horizon des principales fonctionnalités du *TREngine* et des avantages dont qu'il présente face aux autres *frameworks* du même type.

La couche graphique

En deux dimensions, les besoins d'aide à la programmation de jeux se font principalement sentir lorsqu'il s'agit d'orchestrer toutes les surfaces graphiques constituant un écran de jeu à un instant donné. Par exemple, pour faire marcher un personnage, il faut afficher successivement à l'écran et à une fréquence donnée chacune des images constituant l'animation de la marche. Si ensuite on décide de modifier légèrement la couleur de son personnage - en plus de devoir coder l'algorithme de changement de couleur - il faut penser à l'appliquer sur toutes les images constituant l'animation de la marche. Également, si le personnage change de direction, c'est une autre séquence d'images qu'il faudra penser à afficher ...

Le *TREngine* formalise et simplifie toutes ces opérations. Il introduit la notion de *TRSurface* correspondant à une surface graphique affichable. Cette classe contient un grand nombre de méthodes permettant de modifier la surface graphique que le programmeur n'a plus qu'à invoquer directement : modification du taux de transparence, de la colorimétrie, du Hue, de la luminance, du contraste, tracés de primitives (points, lignes et rectangles), système de masquage à l'écran ...

Il introduit également la notion de *TRSprite* correspondant à une surface animée - comme celle du personnage marchant vers la droite - séquence de *TRSurface*. Les sprites du système incluent des routines de détection de collision inter-sprites permettant, par exemple, de détecter automatiquement si un missile rentre en collision

avec un vaisseau spatial - Les algorithmes actuellement implémentés sont « Bounding-Box » et « détection parfaite » présentés plustard.

Le *TREngine* contient également une notion assez unique de « sprite événementiel », le *TREventSprite*. Ce type de sprite modifie automatiquement sa séquence animée sur apparition d'un événement préalablement défini par le programmeur. Par exemple, on peut imaginer qu'en définissant l'événement « VAS A GAUCHE », notre personnage modifie tout seul son apparence lorsque la flèche gauche est pressée.

Les *TRLayer* permettent de regrouper les surfaces et sprites par couches ou plans dont il sera par la suite aisé d'intervertir l'ordre d'affichage à l'écran.

Enfin, le système simplifie la gestion de l'écran dans son ensemble à travers l'objet *TRScreen* qui centralise les opérations d'affichage des sprites et surfaces animées. *TRScreen* hérite également de toutes les méthodes et algorithmes de traitement d'images de la classe *TRSurface*. Modifier tous les pixels de l'écran en renforçant la composante bleu pour donner un effet « nuit », c'est une ligne de code avec le *TREngine* !

La couche réseau

La classe *TRNetworks*, basé sur *SDL_Net*, centralise les fonctionnalités réseaux nécessaires aux jeux multi-joueurs. Dans la version la plus récente, *TRNetworks* supporte les communications UDP en IP v4 et v6, et propose quelques fonctionnalités spécifiques aux réseaux ad-hoc.

La couche dédiée au son

Cette couche permet de gérer facilement les musiques et effets sonores et s'appuie sur *SDL_Mixer*, la couche son de *SDL*. Il est possible de jouer une musique de fond unique au format *MP3*, *WAV* et *OGG*, et de jouer simultanément différents échantillons sonores au format *WAV*. Le mixage de toutes les voies est automatisé et la structure en objet du *TREngine* rend l'orchestration plus intuitive avec les objets *TRSample* et *TRMusic* et leurs méthodes de contrôle de la lecture, du volume et du padding.

Des outils spécifiques

Le *TREngine*, c'est aussi les outils génériques suivants :

- *TRText* : un moteur de « Font » pour afficher des textes en mode graphique
- *TRPanelTable* : permettant de créer des tableaux 2D textuels
- *TRFader* : permettant de générer des effets de fondu - fade – montant, descendant ou enchaîné.
- *TRVirtualKeyboard* : assurant la gestion d'un clavier virtuel rétractable - essentiel pour les applications *PocketPC* ! (utilise *xkbd*)
- *TRProgressBar* : une barre de progression graphique paramétrable.
- *TRColorChooser* : un « sélecteur de couleurs » parmi une palette ou une gamme donnée.
- *TRStarfield* : un effet « fond étoilé » dynamique

- *TRFlare* : un effet « lens flare 2D »
- ...

Une interface de programmation

Enfin, avec la classe de base *TROject* dont héritent tous les composants du moteur et un ensemble de macros C spécifiques, le *TREngine* propose quelques outils de programmation complémentaires bien utiles. Citons les commandes arithmétiques *TRRandom*, *TRCountain* et les commandes de log *TRCOUT*, *TRCERR* et *TRCTRACE* paramétrables en fonction du mode debug.

6. Développement : Treenor railway piracies



6.1. Présentation générale

Le projet *Treenor* consiste à implémenter le scénario spécifié dans le chapitre 4 en C++/SDL. Il s'agit donc de réaliser un jeu vidéo dans son ensemble, contenant deux phases de jeu multi-joueur – un monde persistant et un shoot'them'up coopératif en arène – reposant sur la technologie ad-hoc *SAFARI*.

La réalisation d'un jeu vidéo est une vaste entreprise qui incombe de nos jours d'avantage à des équipes de développeurs, graphistes, designers et musiciens qu'à une personne isolée comme cela fut encore le cas en 1991 lorsque *Eric Chahi* créa seul *Another World*. [Réf 1].

« Aujourd'hui, développer un jeu vidéo, c'est faire partie d'une équipe dans laquelle se mélangent allègrement infographistes, programmeurs, musiciens, acteurs, chefs de produits, chacun ayant un domaine et des compétences propres. Lorsque l'on constate que l'élaboration de certains produits occupe toute une équipe de ce type plusieurs années durant, on comprend aisément que personne ne peut cumuler le temps et les compétences requises pour mener le projet à bien de bout en bout ... »

[Réf 2] Extrait article sur la mort des "artisans du jeu vidéo" - PC Team N°47, Juin 1999

Les trois mois de stage restant, consacrés au projet *Treenor* furent très denses en travail de conception, création et codage et on ne compte plus, ni les cafés, ni les nuits passées sans fermer l'œil. Ce chapitre, loin d'être exhaustif, se propose de faire un tour d'horizon des problématiques majeures que j'ai rencontrées et des solutions qui y furent apportées.

6.2. Problématique d'ordre général et solutions retenues

6.2.1. Consommation de la mémoire

Comme tout programme, une application ludique consomme de la mémoire pour fonctionner. Quelque soit la plate-forme d'exécution, la mémoire est une ressource limitée partagée par l'ensemble des processus actifs dans le système et affectée parcimonieusement à chacun d'eux par le système d'exploitation.

Lorsque l'on programme, on définit des processus qui seront plus ou moins gourmands en mémoire en fonction de ce qu'il font mais aussi de comment ils le font. Pour les mêmes fonctionnalités, un programme mal conçu et/ou non optimisé peut utiliser des quantités démesurées de mémoire et bloquer partiellement le système ou tout simplement ne pas pouvoir être exécutée. Il incombe donc au programmeur de gérer efficacement la mémoire.

Dans le cadre du projet *Treenor*, la gestion de la mémoire fut une quête de tous les instants, les contraintes étant d'autant plus fortes que la plate-forme cible – le *Pocket PC* – dispose de beaucoup moins de mémoire vive que les PC fixes.

En C++, il est très facile de consommer énormément de mémoire. En effet, contrairement à d'autres langages tels que *Java* qui inclut un processus automatique de « *garbage collecting* », c'est au programmeur de désallouer - rendre disponible pour le système d'exploitation - les zones mémoire qu'il a réservées pour stocker ses objets. Même avec la plus grande rigueur, un oubli est vite arrivé ...

Quelque soit le langage de programmation, l'optimisation des algorithmes et le choix pertinent des structures de données permettent d'améliorer la taille mémoire requise à l'exécution. En cela, une application est toujours perfectible et *Treenor* n'échappe pas à la règle.

6.2.2. Indépendance vis-à-vis de la fréquence de rafraîchissement

Un jeu dispose de ce que l'on appelle communément une boucle de jeu. Il s'agit de la suite des opérations réalisées par l'ordinateur, en boucle, pendant le jeu. Formalisons la de façon simple :

Tant que « le jeu n'est pas fini »

1. *Recevoir et traiter les éventuelles commandes utilisateurs*
2. *Modifier « l'état du jeu »*
3. *Afficher à l'écran « l'état du jeu »*

Ainsi, l'état du jeu est modifié et l'écran actualisé à chaque boucle. La fréquence de rafraîchissement – ou *frame rate*, temps entre deux images – correspond donc au temps dont l'ordinateur a besoin pour exécuter une fois la boucle. Comme au cinéma ou en télévision, il apparaît une sensation d'animation à partir de 25 images par secondes, mais ce n'est que lorsqu'on atteint la fréquence de rafraîchissement de l'écran PC (typiquement entre 60 et 80 images par seconde) que l'animation paraît réellement fluide.

Sur une machine lente ou momentanément ralentie – à cause d'un grand nombre de logiciels déjà lancés par exemple, le temps pour exécuter une boucle de jeu est plus long et donc le *frame rate* plus bas, que sur une machine puissante qui sera en mesure de fournir un *frame rate* convenable. Pour donner un ordre d'idée, le *frame rate* en phase planétaire sur un *PocketPC H3900* atteint péniblement 30 images par secondes alors qu'il peut dépasser 250 images par secondes sur un *PC Celeron 2GHz*.

Prenons maintenant le cas simple d'un programme faisant marcher un petit personnage. Si l'étape 2 (modifier « l'état du jeu ») correspond au déplacement vers la droite de 5 pixels du personnage, celui-ci marchera beaucoup plus vite sur une plateforme puissante que sur une plus lente. Cette situation est intolérable, particulièrement si les deux personnages se déplacent dans le même monde partagé en réseau. Il y a donc nécessité d'uniformiser la fréquence de rafraîchissement pour qu'elle soit commune sur toutes les architectures et constante au cours du temps.

Pour les machines rapides, il est facile de fixer le *frame rate* à la valeur uniformisée (posons 60 images/s). En effet, il suffit, à la fin de chaque boucle de jeu, de faire « patienter » le processus du jeu le temps restant pour que le temps d'exécution de la boucle atteigne les $1 / 60 = 0,0166..$ secondes réglementaires.

Pour les machines plus lentes ou momentanément ralenties, le problème est plus délicat. Il est en effet impossible de les « faire calculer plus vite » ! La clef du problème consiste à programmer le jeu uniquement en raisonnant dans le temps. Par exemple, pour le cas de notre programme simple, notre personnage devra se déplacer de 5 pixels tout les $1 / 60 = 0,0166..$ secondes. L'étape 2 (Modifier « l'état du jeu ») consisterait donc alors à déplacer le personnage d'un nombre de pixels proportionnel au temps qui a été nécessaire pour effectuer le dernier tour de boucle : si par exemple 0,03 secondes ont été nécessaires, on se déplacera de $5 * 0,03 / 0,0166.. = 9$ pixels.

Avec ce mécanisme, même si la vitesse de rafraîchissement des images à l'écran varie, le jeu se déroule de la même façon sur tous les types de plates-formes et quelque soit leur capacité de calcul du moment.

6.2.3. Caractéristique « Multiplateforme »

La plateforme cible de *Treenor* est le *Linux PocketPC* puisqu'il s'inscrit dans le projet *SAFARI*. Malgré cela, les développements ont été faits de façon à tirer partie de la compatibilité ascendante de *SDL* qui existe dans des versions pour : PC sous Windows, Linux, BeOs, FreeBSD, OpenBSD, BSD/OS, Solaris, IRIX, QNX, mais aussi sur Amiga, Dreamcast, Atari, ...

Actuellement, le projet *Treenor* existe en version *Linux PocketPC* mais également pour PC classique sous *Linux* (Debian, red-hat, ...) et *Windows* (NT, 95 à XP). Du point de vue du développement, cette caractéristique multiplateforme implique des efforts supplémentaires. Par exemple, la taille de l'écran géré est différente sur *PocketPC* (240x320) et sur *PC* (mode plein écran en 640x480, 800x600, 1024x768, ...), il faut donc gérer tous les cas. Le nombre de bits sur lequel sont codées les

couleurs peut également être différent ce qui nécessite une réécriture complète de certains algorithmes – par exemple, l’algorithme de modification du *Hue* d’une surface graphique.

Enfin, *Treenor* tire partie de la puissance qui est plus importante sur *PC* que sur *PocketPC* – de fréquence de l’ordre de 400MHz – en y rendant disponible certaines options supplémentaires. Par exemple, tous les animaux sauvages d’une même espèce sont de la même couleur sur *PocketPC* alors qu’elle varie pour chaque individu sur *PC*.

6.2.4. Avatar et personnalisation

Treenor est un jeu multi-joueur où chacun dispose d’un avatar virtuel qu’il fait vivre dans un univers commun. Lorsque plusieurs personnes se rencontrent virtuellement via leurs avatars, il est extrêmement important que chacun puisse se reconnaître et se distinguer parmi la masse.

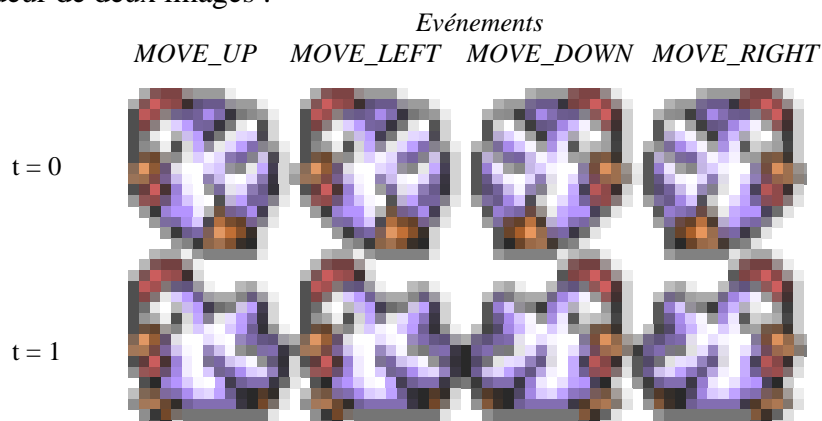
Pour cela, le jeu doit permettre de « customiser » son avatar de façon suffisante. En dehors du camps du personnage – pirate ou chasseur de primes – *Treenor* propose les choix suivants :

- Le nom de l’avatar : on peut taper le nom de son choix ou conserver la proposition aléatoire générée par le générateur automatique de nom. Si des générateurs complexes existent, celui réalisé dans le projet se contente de mettre bout à bout un préfixe, quelques syllabes et un suffixe aléatoirement choisis. Exemple de nom généré : Phylizu, Gizaple, Pheerizur, Folec, Laramoe, ...
- l’aspect général de l’avatar : « Enfant, Homme 1, Homme 2, pingouin, ... ». Le *TREngine* est alors mis à contribution puisque l’avatar n’est autre qu’un *TREventSprite* (cf : chapitre 6). Chacune des apparences des sprites composant le sprite événementiel sont stockées dans un fichier image.



Treenor – le fichier image décrivant le sprite événementiel du personnage « Homme 1 »

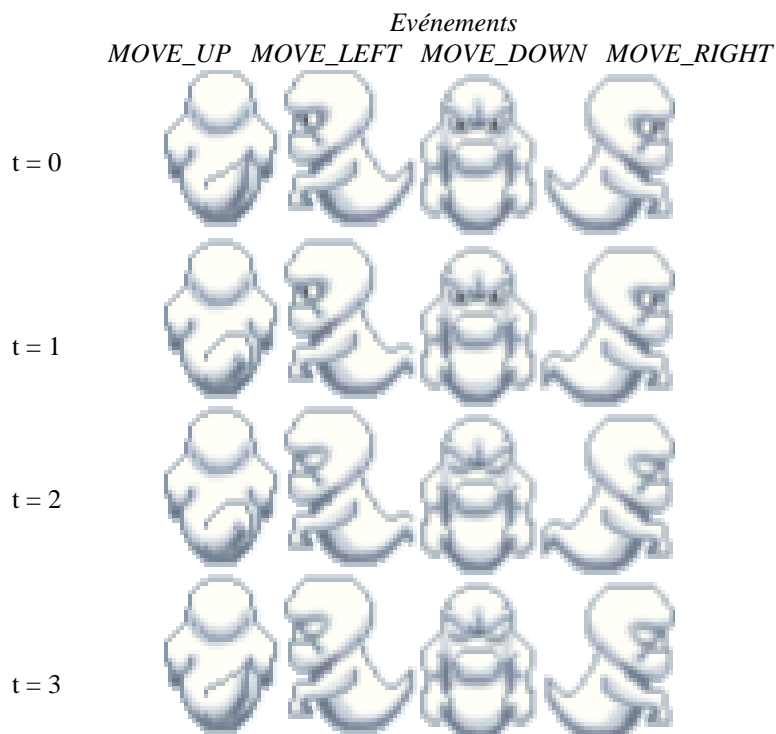
Est également défini un jeu de sprites correspondant à la position au repos - événements STAND_UP, STAND_DOWN, STAND_LEFT et STAND_RIGHT. A noter que la « profondeur » des sprites - dans l'exemple de la figure, la profondeur est de trois - varie en fonction des personnages. La poule, quant à elle, a par exemple une profondeur de deux images :



Treenor – le fichier image décrivant le sprite événementiel des poulets

Enfin, il faut remarquer que dans la grande majorité des 16 personnages définis actuellement dans *Treenor*, le fichier défini pour les positions au repos n'est autre qu'une référence au fichier des événements de déplacement mais interprété avec une profondeur de un – au repos, les personnages ne sont pour la plupart pas animés.

Le contre exemple est le fantôme. N'ayant pas de pieds, il n'y a pas de distinction apparente entre ses états en déplacement et au repos. Les deux références pointent sur le même fichiers et les deux profondeurs valent quatre :



- le joueur choisit également le vaisseau dans lequel son avatar affrontera ses adversaires et collaborera avec ses coéquipiers en phase spatiale. Sans entrer dans les détails, le système de chargement des vaisseaux développé est similaire à celui des personnages. Aux vaisseaux est également associé un jeu de munitions ayant une apparence propre.
- La touche finale de la personnalisation est le choix du coloris de l'avatar qui se répercute également sur le vaisseaux et ses armes. La colorisation est basé sur un algorithme de modification de *Hue*, codé dans le *TREngine*, et offrant une palette de 360 coloris.

6.2.5. Esthétisme

La référence pour le projet fut *Zelda : The link to the Past*, un jeu *Nintendo* sur console *SNES* qui a marqué des générations de joueurs et dont il a déjà été question dans les chapitres précédents. L'intérêt de ce type d'univers graphique est qu'il est particulièrement bien adapté à la technique du *Tiling* (cf : 3.6.6). De plus, l'aspect un peu « enfantin » est plus adapté à un projet 2D qui peut difficilement viser le réalisme. Enfin, il correspond à un goût personnel.

L'aspect brumeux et mystérieux de l'introduction puis les couleurs froides tirant vers le bleu métallisé que l'on retrouve tout au long du jeu donnent une cohérence esthétique à l'ensemble du monde imaginaire et du vide interstellaire qui l'entoure.

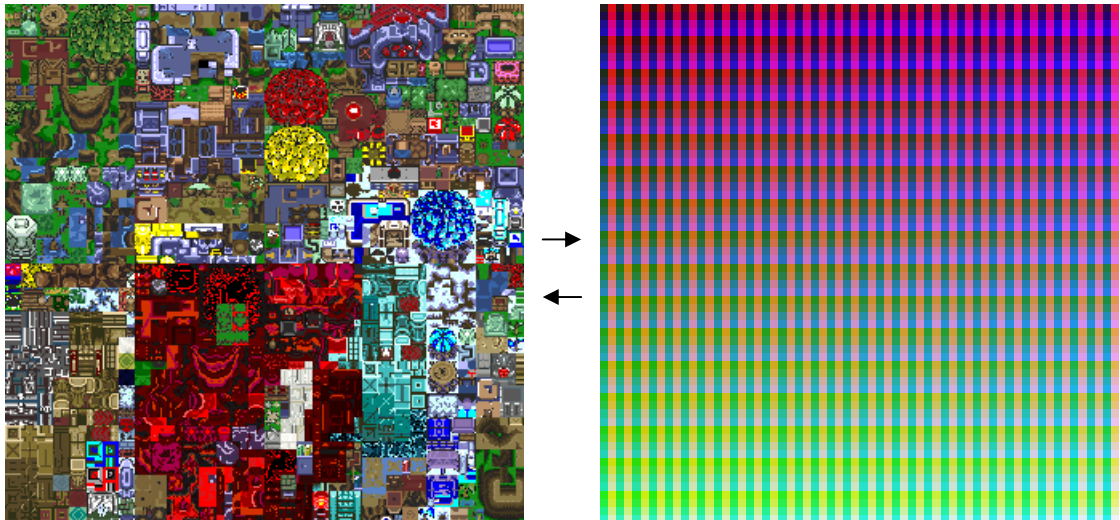
6.3. **Problématiques en phase planétaire et solutions retenues**

6.3.1. Représentation d'un monde

Pour représenter la planète *Treenor* à l'écran, la technique utilisée est donc celle du *Tiling* (cf : 3.6.6) qui permet de modéliser un monde en deux dimensions en vue aérienne avec une légère perspective.

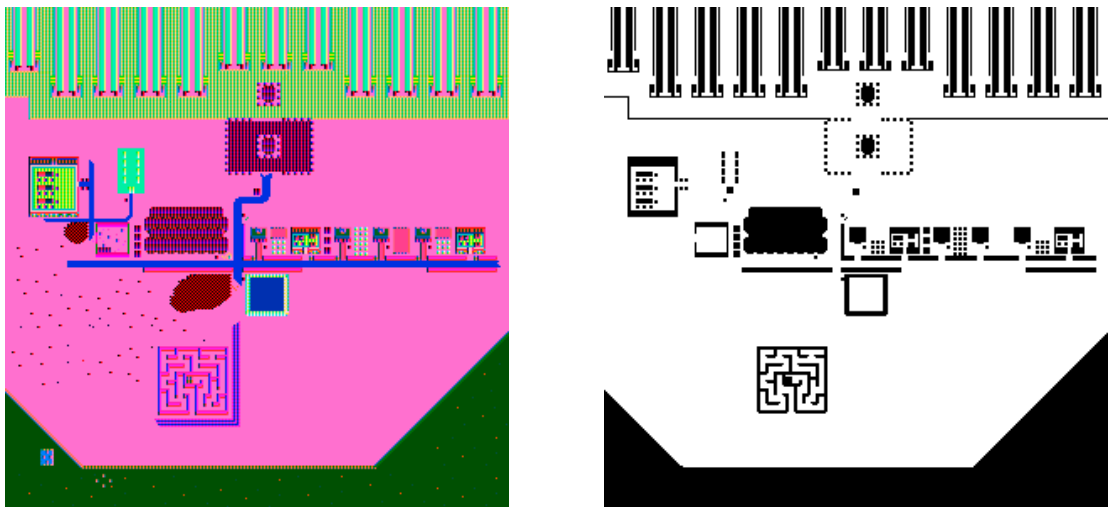
La solution technique retenue pour implémenter le *Tiling* est la suivante : la taille unitaire des « pavés » est de 16x16 pixels. Un tableau de correspondances a été dressé faisant correspondre à chacun des 4096 carrés composant l'univers une couleur unique – combinaison donnée des trois composant RGB en 32 bits.

Pour fixer les idées, la figure suivante présente les pavés à gauche et les couleurs qui leurs sont affectées à droite.



Treenor – Implémentation du Tiling, tableau de correspondance « pavés » / index coloré

« Le monde » est ensuite codé au 1/16 éme dans une image où chaque pixel correspondra à un pavé du modèle, en fonction de sa couleur. Voici par exemple, le monde principal dans la version la plus récente du projet (à gauche) :



Treenor – Implémentation du Tiling, image définissant le monde et image des « pavés impraticables »

Est défini également dans le système une « carte des pavés impraticables ». En effet, il paraît évident pour le joueur qu'un personnage ne peut pas passer à travers les murs ou marcher dans l'eau. Cette carte est définie dans le système par l'image de droite de la figure précédente. Les pixels non blanc indiquent que le pavé correspondant n'est pas praticable – non traversable par les personnages. La gestion des collisions sera davantage développée dans le paragraphe 7.3.3.

Enfin, une option de démarrage réservée aux développeurs, lance le jeu en mode « Map Viewer » qui facilite la construction de la carte – un fastidieux travail de fourmi ! – en rendant à l'écran les modifications apportés en temps réel aux images cartes.

6.3.2. Interface Homme-Machine et jouabilité

L'IHM est un point important qui peut très facilement rendre le jeu désagréable à jouer. Cet aspect rentre dans une notion plus vaste appelée « jouabilité » par les testeurs et journalistes spécialisés. La jouabilité fait toujours partie des systèmes de notation adoptés dans les magazines spécialisés et a le même poids que les considérations esthétiques ou purement techniques. Nombre de jeux, magnifiques et techniquement impressionnants, n'ont remporté qu'un succès médiocre à cause d'une jouabilité jugée mauvaise.

Menu principal

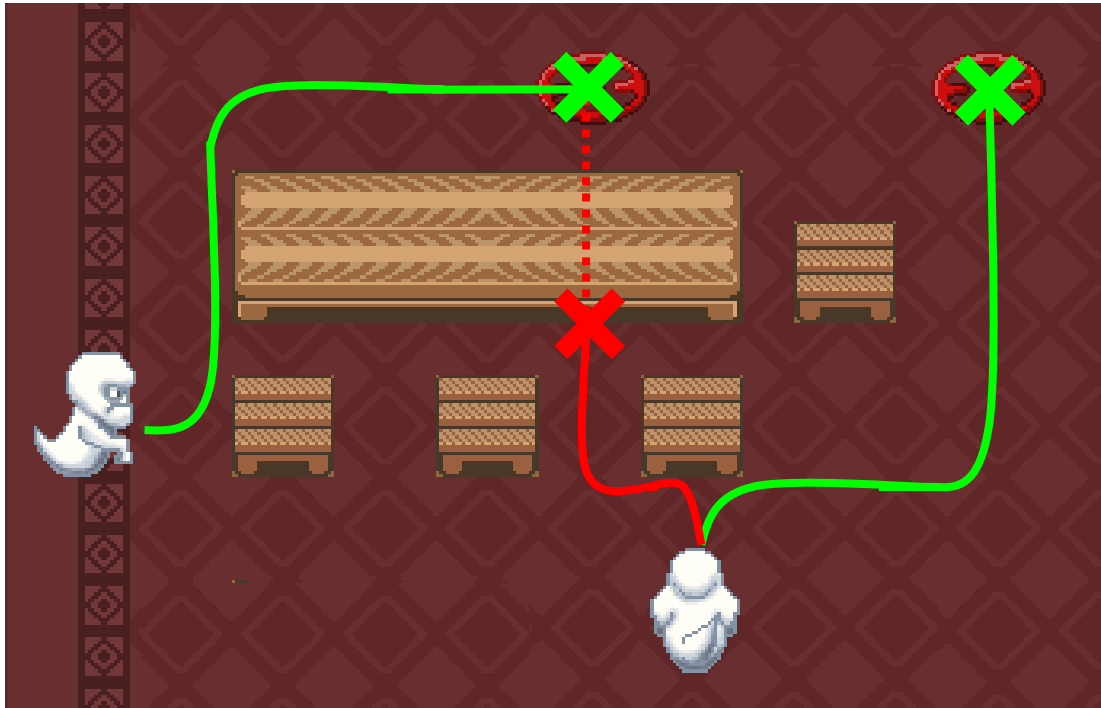
N'ayant pas personnellement suffisamment d'expérience en la matière, l'IHM du projet *Treenor* est calquée le plus possible sur celles, déjà éprouvées, de jeux célèbres existants. Par exemple, le menu principal, très épuré, reprend le principe des menus que l'on retrouve dans toutes les versions du jeu *Final Fantasy* de *SquareSoft*, déjà évoqué à maintes reprises dans ce document.

Déplacement de l'avatar

Le système de déplacement de l'avatar est calqué sur celui que l'on retrouve dans les célèbres *Starcraft* et *Warcraft* de *Blizzard* où on donne un ordre de déplacement en indiquant avec la souris - ou le crayon - la destination souhaitée. À noter que l'algorithme de décision de trajectoires n'est pas implémenté à l'heure actuelle en A*. Il se contente pour le moment, à chaque instant, de réduire la distance avec le point destination dans une direction (Nord, Sud, Est, Ouest), choisie pour avoir l'écart le plus important. Quand une direction aboutit à un pavé impraticable ou à la position adéquate sur l'axe vertical (Nord / sud) ou horizontal (Est / Ouest), une autre direction est élue jusqu'à la destination ou blocage du personnage.

Avec cet algorithme, dans la plupart des cas, les ordres donnés par l'utilisateur aboutissent à une trajectoire cohérente pour le personnage. Seul les cas où il est nécessaire de contourner un obstacle qui éloignerait complètement le personnage momentanément de sa destination pose problème. C'est là que A* fait la différence dans les autres jeux.

La figure suivante présente trois cas différents. Tous les ordres sont fructueux à l'exception du cas rouge qui nécessitera de la part de l'utilisateur un click intermédiaire à gauche de la table.



Treenor – les décisions de trajectoires prises dans trois cas particuliers.

Maniement et Interactions en phase planétaire

Le joueur dispose pour interagir avec son avatar :

- D'un bouton « action » – bouton *calendar* sur *PocketPC* et espace sur *PC* – qui permet d'entreprendre différentes actions comme lire un panneau, ouvrir une porte ... Et d'autres actions non encore implémentées, comme ouvrir le dialogue avec un robot, tirer dans un ballon, ...
- Des flèches pour déplacer la position de la caméra centrée par défaut sur le personnage. A partir d'un certain délai, un recentrage automatique de la caméra autour du personnage est réalisé.
- Comme présenté dans le paragraphe précédent, on déplace l'avatar en touchant l'écran sur *PocketPC* ou avec la souris sur *PC*.
- Avec la touche *Esc* sur *PC* ou le bouton latéral gauche sur *PocketPC*, un menu s'ajoute en sur-impression sur l'écran permettant d'interrompre, de quitter la partie ou de quitter complètement le jeu.

Pour informer ou assister le joueur, sont présents toujours à l'écran :

- Une icône clavier faisant apparaître/disparaître un clavier virtuel – représentation graphique d'un clavier dont toutes les touches sont cliquables – indispensable pour dialoguer sur *PocketPC*.

- Une icône faisant apparaître/disparaître le tableau informatif donnant les informations (horaires, quai, destinations, « à défendre » ...) des trains actuellement posés sur *Treenor*.
- Une horloge digitale donnant l'heure actuelle sur *Treenor* qui est synchronisée avec l'heure terrestre. En jeu off-line c'est l'heure système qui est utilisée, alors qu'en réseau c'est l'heure de la stations orbitale – application fonctionnant dans les nœuds faiblement mobiles (cf : 6.5) – prévue à cette effet.
- Une carte réduite en bas à droite du monde avec une croix symbolisant le joueur, et des points munis d'un code couleur pour les autres joueurs et robots présent dans le monde. En cliquant sur la mini-carte, celle-ci s'agrandit pour prendre tout l'écran. Elle n'interdit pas de continuer à jouer puisqu'elle apparaît en semi-transparence.
D'un point de vue technique, cela signifie que l'objet *Map* dispose des positions de tous les personnages à chaque instant. Cette centralisation de l'information est réutilisée pour la détection des collisions inter-personnage.
- Le nombre de « vies » (chance de rejouer après destruction du vaisseau) est symbolisé par des cœurs pleins ou vides.
- En mode debug, le *frame rate* actuel est affiché à tout moment.

6.3.3. La gestion des collisions

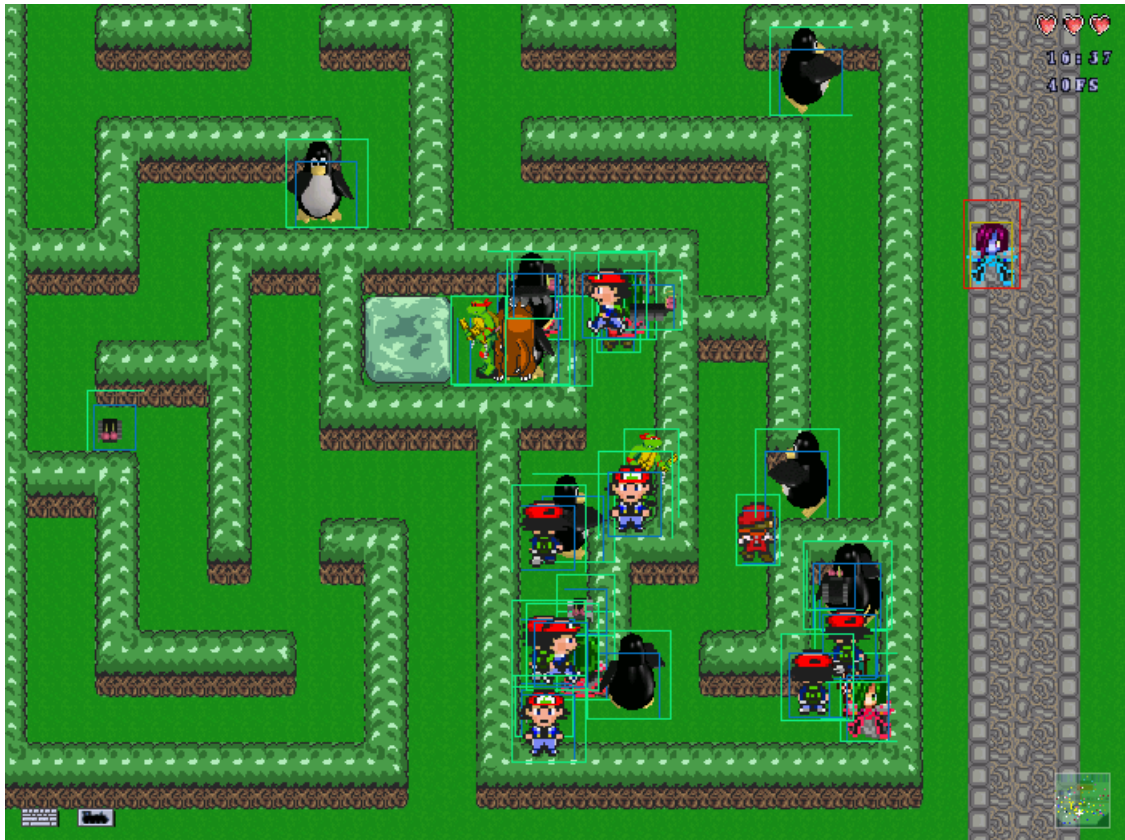
Le système de terrain praticable ou impraticable a déjà été présenté dans le chapitre 6.3.1. Dans l'algorithme de détermination de trajectoire, on prend en compte cette information pour éviter des « phénomènes paranormaux » comme la traversée d'un mur par une poule ou la course d'un cheval sur la surface de l'océan.

Le problème revient à devoir détecter les collisions entre le sprite actuel du personnage et les zones pavées du décors. Remarquons que la problématique est identique lorsqu'il s'agit de déterminer si un missile percute ou non un vaisseau.

Ces besoins sont communs à tous les jeux vidéo en deux dimensions et les techniques ont été formalisées, les algorithmes spécifiés et optimisés depuis fort longtemps. Dans le domaine de la 3D, on se contente d'ailleurs d'adapter les mêmes idées. Le *TREngine* ne réinvente donc rien et propose actuellement deux des algorithmes les plus largement utilisés : la détection de collision inter-sprite par « Bounding Box », et la détection dite « Parfaite ».

Rapidement, la détection « Parfaite » consiste à vérifier pour chacun des pixels composant le sprite A qu'aucun autre pixel du sprite B ne se trouve aux mêmes coordonnées à l'instant du test. Le principe des « Bounding Box » consiste à définir au préalable un rectangle imaginaire sur l'image des sprites. Le chevauchement est alors testé entre les deux rectangles de A et B et non pour chacun des pixels.

L'algorithme « Parfait » est bien entendu plus coûteux en puissance de calcul, c'est pourquoi l'algorithme « Bounding Box » a été retenu pour le projet *Treenor*. La figure suivante présente un ensemble de robot dans un espace où les collisions avec le décors sont fréquentes – le labyrinthe de *Treen*. Le rectangle vert sur chacun des personnages représente les limites de la surface de l'image stockée pour le sprite. Le rectangle bleu quant à lui représente la « bounding box » définie pour le sprite.



Treenor – le système de collision personnages / décors par la technique des bounding Box

Remarquer que les bounding box « coupent un peu la tête » des personnages. En effet, elle ne couvre que 75% de la hauteur du personnage. Cette dissymétrie permet en fait d'autoriser la superposition de la tête des personnages avec des éléments du décors et donne ainsi un léger effet de perspective.

6.3.4. Des robots « intelligents »

Pour enrichir et mettre un peu de vie sur *Treenor*, ont été ajoutés des animaux sauvages et des joueurs robots gérés **en local** par l'ordinateur. Ces « robots » se contentent de se balader sur l'espace de jeu. Il est prévu qu'il soit possible d'engager une conversation (prédéfinie dans un script) avec eux mais cette fonctionnalité n'est pas encore implémentée.

Le script gérant actuellement leur déplacement est très simple. Toutes les 2 à 5 secondes, les robots se fixent une nouvelle destination à atteindre dans un rayon d'action paramétrable – par exemple les poules ont un rayon d'action assez restreint et

restent ainsi d'avantage groupées qu'un cheval qu'on verra galoper d'un bout à l'autre de son prés.

6.3.5. Les trains spatiaux

Conformément au scénario spécifié dans le chapitre 4, des trains de l'espace se posent et décollent de *Treenor* aux mêmes heures que les trains arrivent et partent de la gare Paris Montparnasse. Techniquement, ceci se met en place par un mécanisme de transfert « d'informations trains » entre les stations orbitales – application serveur fonctionnant dans les nœuds faiblement mobiles du réseau ad-hoc – et les terminaux mobiles. Cet aspect sera d'avantage développé dans les paragraphes 3.5 et 3.6.

6.3.6. Immersion sonore et autres détails

Afin de rendre l'immersion plus totale, on utilise la couche son du *TRengine* pour ajouter une ambiance sonore. Le principe est de déclencher de manière plus ou moins aléatoire la lecture d'extraits sonores lus en boucle, pour une durée également aléatoire, introduits par un *fade-in* et stoppés par un *fade-out* très lents.

Un certain nombre d'extraits « passe-partout » sont utilisés : Bruit d'oiseaux, de vent ... Un jeu d'extrait de bord de mer n'est utilisé que lorsque le joueur se situe dans une large zone proche de l'océan en bas de carte. Egalement, des extraits sont définis pour la zone supérieure de la carte donnant une ambiance « gare ».

Enfin, l'heure influe également sur l'univers sonore puisque des extraits spécifique sont utilisés dès la tombée de la nuit (criquets, grenouilles, ...).

Le *TRengine* est également mit à contribution pour simuler les couchés/levés de soleil et tombées de la nuit. Si le principe est simple – on renforce progressivement le poids de la composante bleu des pixels de l'espace de jeu pour faire tomber la nuit de 10h à 5h du matin et on augmente plus vivement la composante rouge pour l'aube et l'aurore – la mise en oeuvre n'a pas été si évidente. En effet, il a fallu définir empiriquement une fonction « réaliste » dirigeant R,G et B en fonction de l'heure du jour.

Enfin, à noter la présence de *cheats-codes* cachés permettant quelques fantaisies aux tricheurs en ayant connaissance – et aux développeurs qui travail, eux ! Il est possible, en vrac, de détraquer l'horloge en la faisant tourner plus vite (faites dire « mad clock » ou « nitro clock » à votre personnage), de faire passer X heures instantanément (dire « X hours is not enough »), de gagner ou perdre une vie, de se téléporter vers des points donnés (dire « teleport quai 11 », « teleport bar », « teleport ocean » ...). Mais pour activer les *cheats-codes*, il faut dire le mot magique !

6.4. Problématiques en phase spatiale et solutions retenues

6.4.1. Représentation de l'espace interstellaire

Les phases spatiales sont également en deux dimensions et en vue aérienne. Pour représenter l'espace interstellaire, sont superposés deux fonds étoilés – *TRStarfield* – dont le plus éloigné a une densité d'étoiles plus importante. Lorsque le vaisseau du joueur se déplace, les plans se déplacent également mais à une vitesse inversement proportionnelle à leur éloignement pour donner une illusion de profondeur.

Un troisième plan – *TRLayer* – se superpose aux étoiles et contient différentes planètes disséminées dans l'espace de combat. Leurs positions respectives sont tirées aléatoirement en local, mais pourtant identiques pour chaque arène de jeu sur toutes les plateformes et d'un joueur à l'autre, sans pour autant être transmises sur le réseau. L'astuce utilisée ici consiste à initialiser le générateur pseudo-aléatoire (*srand* en C) avec le numéro du train au cœur de l'arène.

6.4.2. Interface Homme-Machine

L'IHM en face spatiale est le plus possible proche de celle de la phase planétaire pour donner un tout cohérent. Sont donc également affichés à tout moment à l'écran : le nombre de vies, l'heure treenorienne, la carte de la zone de combat (rétractable avec chaque vaisseau représenté par un code couleur) et le *frame rate* en mode debug.

En revanche, la jouabilité ne s'effectue qu'avec le *joypad* du *PocketPC* ou les flèches du clavier. La caméra suit au plus près les déplacements du vaisseau mais avec une petite inertie pour renforcer l'impression de vitesse donnée par le défilement du décor interstellaire. Le bouton « action » de la phase planétaire devient le bouton de tir.

6.4.3. Combattre dans l'espace

Rappelons que le but des joueurs pirates est de détruire le train au centre de l'arène de jeu et que le but des chasseurs de primes est de détruire tous les vaisseaux pirates. Les munitions à disposition du joueur sont illimitées. Pour le moment, chaque vaisseau à une seule et même arme et les améliorations de l'arsenal ne sont pas encore implémentées.

Les collisions entre les projectiles et les vaisseaux ou trains sont détectées avec l'algorithme « Bounding Box » présenté précédemment. L'algorithme de détection « parfait » est souvent utilisé dans les jeux de tir mais il n'a pas paru nécessaire – au regard du coût qu'il représente – dans notre cas. En effet, les vaisseaux sont de forme assez rectangulaire, de taille réduite, et la grande vitesse des projectiles dissimule assez bien l'approximation des détections.

6.4.4. Robots et trains de l'espace

L'IA des vaisseaux pilotés par l'ordinateur en jeu off-line et par la station orbitale – nœud faiblement mobile – en réseau ad-hoc est clairement un point faible de la version actuelle de *Treenor*. Pour le moment, le script se contente de faire naviguer les vaisseaux spatiaux aléatoirement dans l'espace de combat à la manière des animaux sauvages qui se promènent sur la planète. La seule dimension stratégique du script repose sur la tendance des pirates à se déplacer d'avantage vers le train de l'espace.

Cette première approximation de l'IA suffit déjà pour disputer quelques parties, mais on est loin des « bots » de *Quake* ou des unités de *Starcraft* qui élaborent une stratégie de groupe et qui donnent réellement du fil à retordre.

Enfin, en phase spatiale, les trains de l'espace sont représentés de la même manière que sur la planète mais agrandis d'un coefficient deux - zoom implémenté dans le *TREngine*. Ils se déplacent de manière très lente sur l'arène de jeu. Mis à part en off-line, la position et le niveau de vie du train est géré par la station orbital – application serveur fonctionnant sur le nœud faiblement mobile –.

Penchons nous à présent sur l'architecture réseau déployée dans les deux types de phases.

6.5. Architecture réseau

6.5.1. Les différents modes de jeu

Si le mode réseau ad-hoc est le mode de jeu par défaut, il est également possible de jouer à *Treenor* seul en off-line. Tout est alors géré en local et l'absence de partenaires de jeu est compensé par la présence des robots.

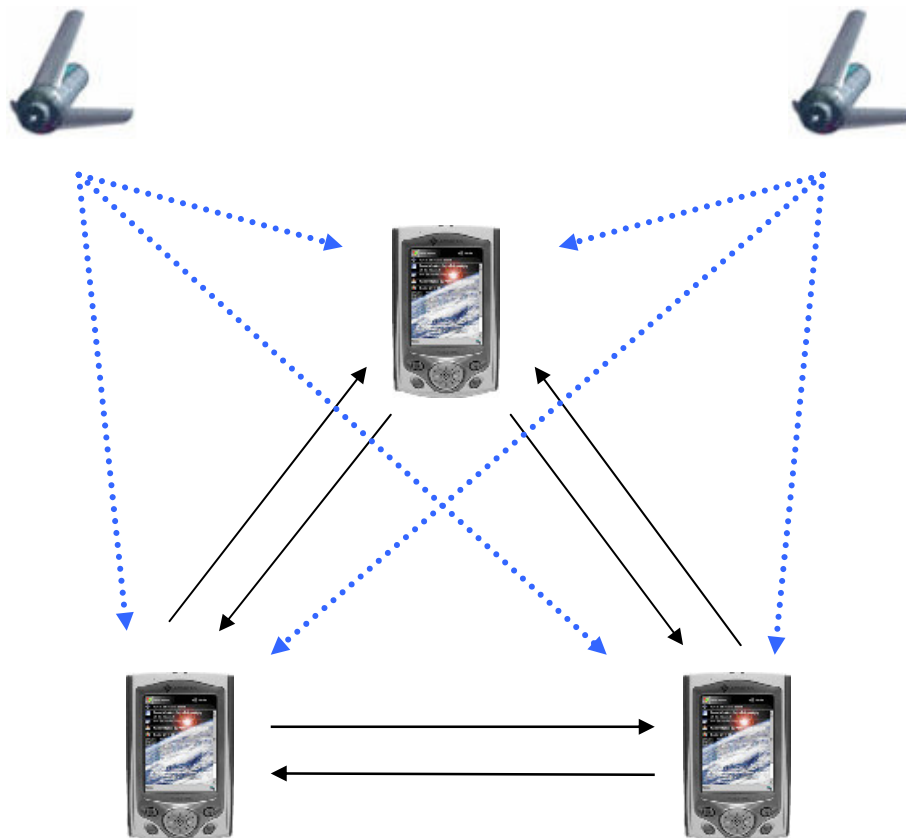
Il est également possible de jouer en réseau classique (WiFi / LAN). Pour rendre ce mode pertinent, il faudra s'appuyer sur les techniques de *broadcast* sur réseau local ou créer un serveur centralisé (mais cela pas ni l'objectif du stage, ni celui du projet *SAFARI* ...). Pour le moment dans ce mode, *Treenor* fait du *Peer-to-Peer* sur réseau local. Un des joueurs de chaque pair de joueurs possible doit spécifier l'IP de son partenaire en cours de partie en le faisant dire à son avatar précédé du mot « IP ».

Enfin le mode principal en réseau ad-hoc déploie deux architectures distinctes en fonction de la phase de jeu. Rappelons que le réseau ad-hoc est constitué de terminaux mobiles – les joueurs – et de terminaux faiblement mobiles, fixé à des points précis de la gare. Ces derniers hébergent une application spécifique au projet *Treenor* baptisée *Treenor Orbital Station* ou *TOS*. Les stations orbitales sont responsables de la gestion des trains du groupe de quais pour lesquels elles sont affectées et font office de serveur pour les arènes de jeu spatiaux des trains qu'elles gèrent.

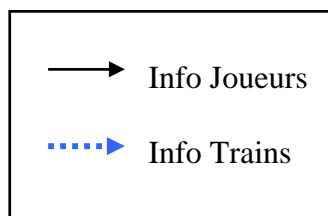
6.5.2. En phase planétaire

Chaque joueur du réseau ad-hoc « broadcast » - c'est à dire diffuse - les informations correspondantes à l'état actuel de son avatar aux autres nœuds du réseau. *OLSR*, l'implémentation actuelle de la technologie ad-hoc *SAFARI* n'autorisant pas encore le « broadcasting », le jeu tient donc une liste des joueurs du réseau et transmet les paquets à chacun d'eux pour le simuler.

Les *TOS* « broadcast » quand à eux les informations des trains qui leurs ont été affectés : heure d'arrivé, de départ, quai, look, numéro et provenance. A l'heure actuelle, les stations orbitales possèdent un fichier de configuration dans lequel on spécifie la liste des trains et leurs informations. Pour les besoins des démonstrations, les horaires d'une journée type de la gare Montparnasse sont utilisés. Le mécanisme d'actualisation automatique de ces informations avec les véritables informations SNCF en temps réel n'est pas encore implémenté.



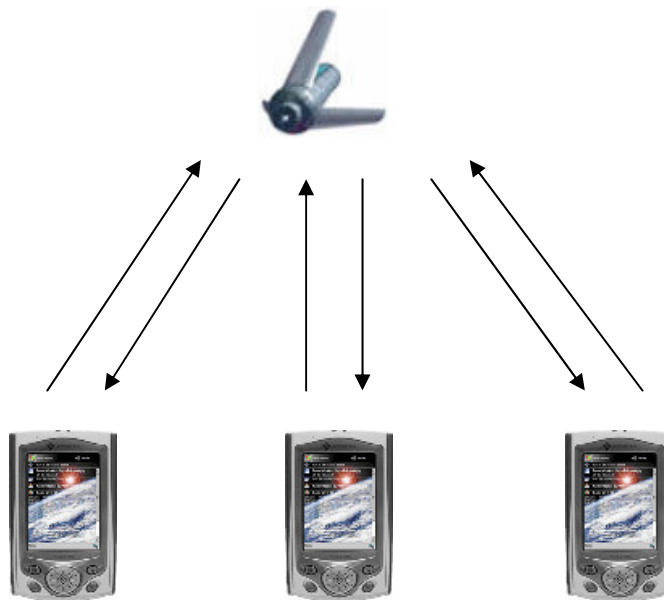
Treenor – les flux d'informations entre les joueurs et les TOS en phase planétaire en mode ad-hoc



6.5.3. En phase spatiale

En phase spatiale, on rentre dans une architecture beaucoup plus classique de type client/serveur. Les joueurs envoient tous leurs informations à la *TOS* gérant le train qu'ils défendent ou attaquent. La station orbitale a le rôle de faire transmettre les informations reçues à tous les autres joueurs qui défendent où attaquent le même train – et sont donc dans la même arène. La *TOS* gère également la position et le niveau de vie des trains spatiaux.

Les difficultés dans l'implémentation des stations orbitales sont nombreuses puisqu'elles doivent être en mesure de gérer plusieurs arènes de jeu en simultanée, diffuser les bonnes informations aux bons intervenant de l'espace, tout en gérant les autres trains se posant en phases planétaires.



Treenor – les flux d'informations entre les joueurs et les TOS en phase spatiale en mode ad-hoc

Pour diminuer la charge de calcul nécessaire aux stations orbitales, celles-ci ne tiennent pas une représentation graphique des arènes qu'elles gèrent mais bien une représentation logique. Ce concept améliore grandement les résultats mais a nécessité un travail supplémentaire de conception et d'implémentation non négligeable.

6.6. Protocole de communication

6.6.1. En phase planétaire

Chaque terminal mobile envoie :

- A une fréquence donnée un paquet « I AM HERE » - avec look, nom et position.
- A chaque commande de déplacement, « MOVE TO AND POSITION » - avec position et destination de la commande de déplacement.
- A chaque phrase énoncée par le personnage (saisie par le joueur, lecture de panneau, réaction automatique ...), « SENTENCE » - avec phrase énoncée.

Chaque terminal mobile reçoit :

- « I AM HERE » : seul le look et le nom sont interprétés, modifiant l'avatar du joueur associé ou le faisant apparaître le cas échéant. A partir de trois secondes sans réception de paquet « I AM HERE » de la part d'un autre terminal, celui-ci entre en état *link dead* (avatar en transparence). A partir de dix secondes, l'avatar est supprimé de la représentation locale du monde.
- « MOVE TO AND POSITION » et « SENTENCE » : directement interprété comme commande pour l'avatar associé.

Chaque station orbitale envoie :

- Les TrainInfo (horaires de départ, d'arrivée, voie, nom, look du train) à chaque départ et arrivée de train. Les TrainInfo sont ensuite stockés et gérés en local dans les terminaux mobiles.
- « YOU CAN TAKE OFF » : Les autorisations de décollage lorsque : l'avatar du joueur se situe dans la zone « entry point » d'un quai ou un train est en phase « opening » (signifiant qu'il peut être défendu ou attaqué) ET que le terminal du joueur est à un bond au sens ad-hoc du nœud faiblement mobile hébergeant la station.
Le paquet contient également la nouvelle position de l'avatar pour son retour de la phase spatiale.

Chaque station orbitale reçoit :

- « I AM HERE » : seule la position est interprétée, modifiant l'avatar du joueur en mémoire (pas de représentation graphique). Elle permettra de déterminer s'il faut, ou non, envoyer une autorisation de décollage.
- « MOVE TO AND POSITION » et « SENTENCE » : ignorée.

6.6.2. En phase spatiale

Tous les paquets envoyés par les joueurs contiennent l'IP – v4 ou v6 - de leur terminal juste après les champs PACKET_TYPE (8bit). Tous les paquets joueurs sont envoyés

directement à la station orbitale qui gère la partie et qui les transférera à tous les autres joueurs de la même partie. Comme expliqué précédemment, ce procédé permet de tirer parti du fait que tous les terminaux sont à un bond du nœud faiblement mobile. En effet, on stabilise les retards entre les différentes transmissions.

Chaque terminal envoie :

- au *frame rate*, des paquets « SHIP DESCRIPTION » : contiennent nom, look, position, vecteur déplacement courant du vaisseaux et des projectiles et EventId actuel du sprite vaisseaux.
- « SHOOT » : lorsque la commande de tirer est activée.
- « EXPLODED » : lorsque le vaisseau est détruit.

Chaque terminal reçoit (par l'intermédiaire de la station orbitale) :

- au *frame rate*, les paquets « SHIP DESCRIPTION » des autres joueurs.
- le cas échéant, les paquets « SHOOT » et « EXPLODED » des autres joueurs.
- très fréquemment les paquets « SPACE TRAIN DESCRIPTION » et « BOT DESCRIPTION » de la station orbitales, et, le cas échéant, les paquets « PIRATES WIN » ou « HUNTER WIN ».

Chaque station orbitale envoie, dans le cadre de chaque arène :

- « SPACE TRAIN DESCRIPTION » : la position du train, son niveau de vie.
- « BOT DESCRIPTION » : descriptions – idem SHIP DESCRIPTION – des vaisseaux pirates et/ou chasseurs de primes robots gérés par la station.
- « PIRATES WIN » : lorsque le train est totalement détruit.
- « HUNTERS WIN » : lorsque tous les pirates sont détruits.

Chaque station orbitale reçoit, dans le cadre de chaque arène :

- « SHIP DESCRIPTION », de tous les joueurs.
- « SHOOT » et « EXPLODED » de tous les joueurs.

Dans les deux cas, les représentations logiques des arènes sont actualisées après réception afin d'orchestrer les combats interstellaires les plus palpitants jamais perpétrés dans l'enceinte d'une gare !

6.7. Références

6.7.1. Références indexées

Réf 1 : Test de *Another World*

<http://www.macreators.com/testspreviews/00026.html>

Réf 2 : Eric Chahi – concepteur, programmeur, graphiste

<http://atari.games.free.fr/atarist/authors/chahi.htm>

6.7.2. Références bibliographiques

Réf : *Networked Virtual Environments, Design and Implementation*
Sandeep SINGHAL et Michael ZYDA, Edition Addison Wesley

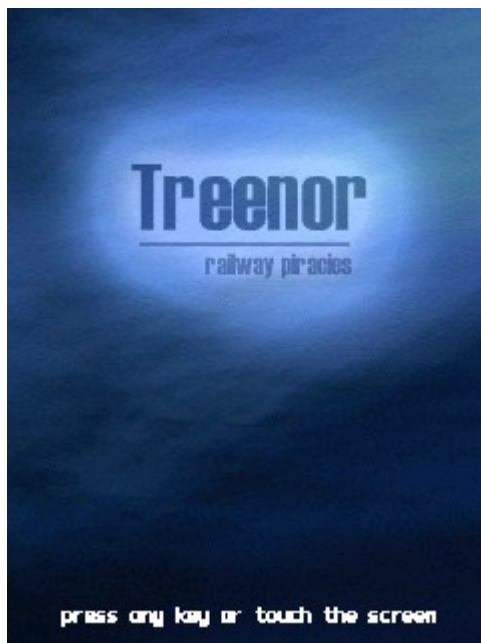
Réf : *Le langage C++*
Bjarne Stroustrup, edition Pearson Education

Réf : *Computer Graphics, principles and practice*
Foley, Van Dam, Feiner et Hughes, edition Addison Wesley

7. Résultat final

Pour clôturer ce rapport, on se propose dans ce chapitre de faire un tour d'horizon en images du jeu *Treenor railway piracies*. On laissera le lecteur se fier à son appréciation personnelle.

7.1. *Treenor : introduction, menu et création d'avatar*



Treenor en images - Logo, introduction et menu. Se dessine en arriere plan un décor montagneux brumeux et en déplacement.



Treenor en images – La création de son personnage. Nom, caste, apparence générale de l'avatar et du vaisseau puis choix d'un colori

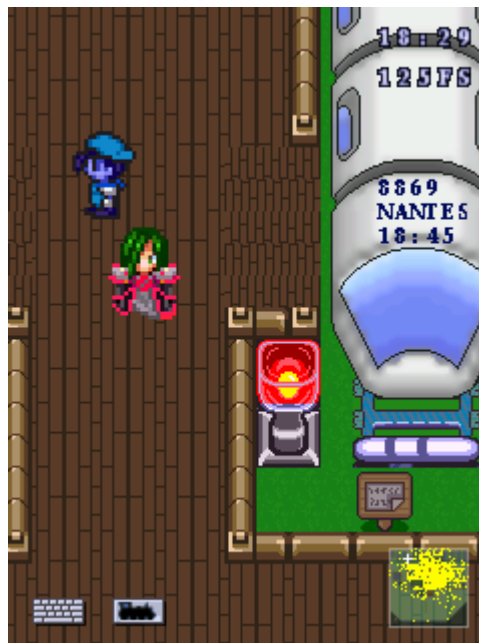
7.2. Treenor : les phases planétaires



Treenor en images – Ballade bucolique au milieu des poules et folle journée au comptoir.



Treenor en images – Nos amis les chevaux et un petit garçon visiblement perdu



Treenor en images – La frontière océane et le quai 11 de la gare spatiale vu par deux joueurs en réseau



Treenor en images – Le quai 11 au couché du soleil et une maisonnette à la nuit tombée.

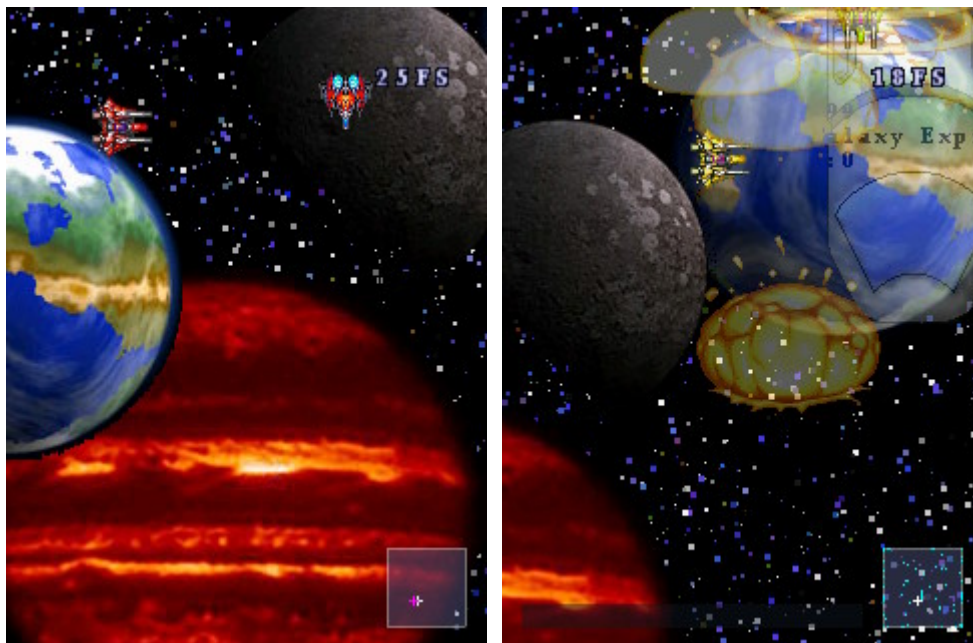


Treenor en images – sur PC – le panneau d'information des trains à l'arrivée disponible à travers l'icône "Train".

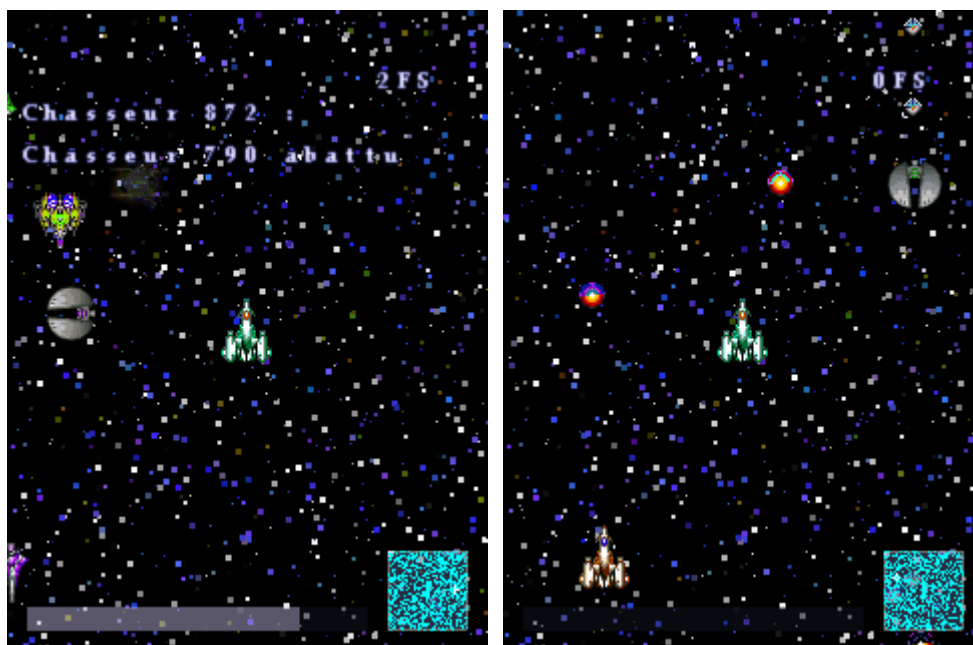


Treenor en images – sur PC –Mais que fait-il là ??

7.3. Treenor : les phases spactiales



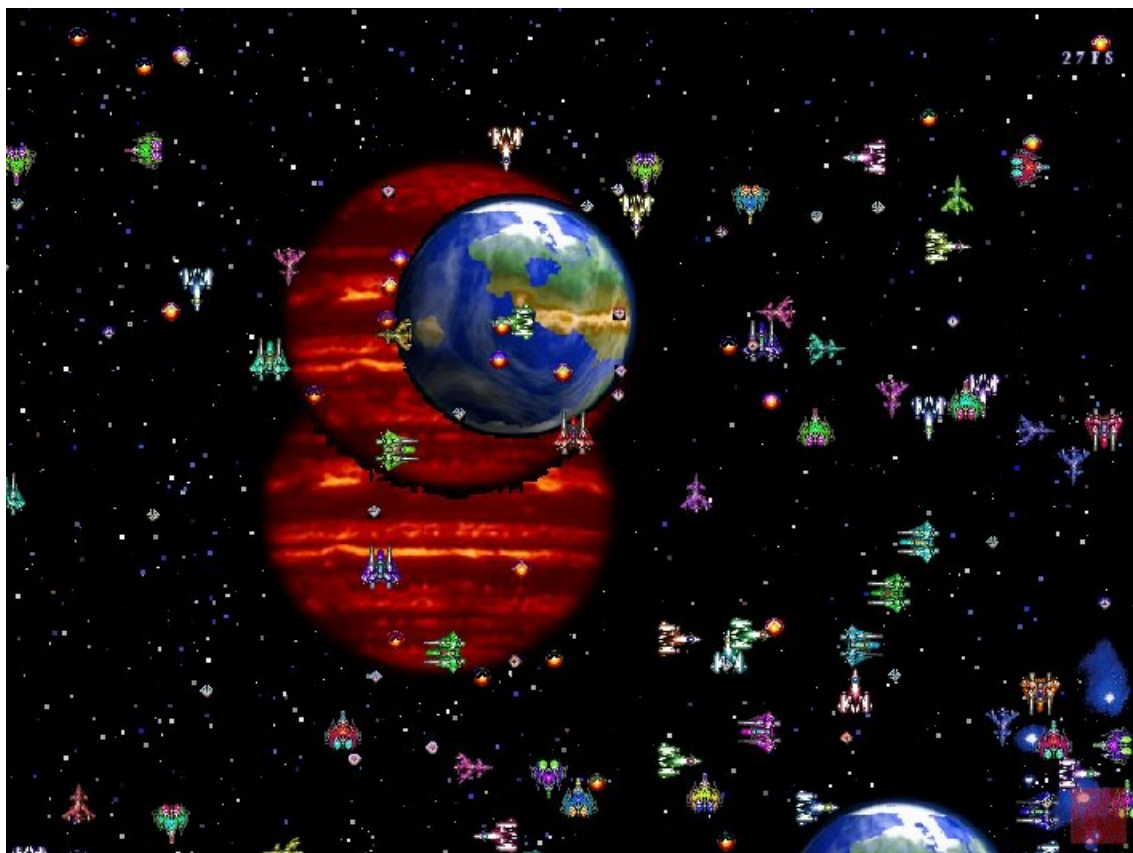
Treenor en images - rencontre de deux joueurs en réseau et disparition d'un train explosé



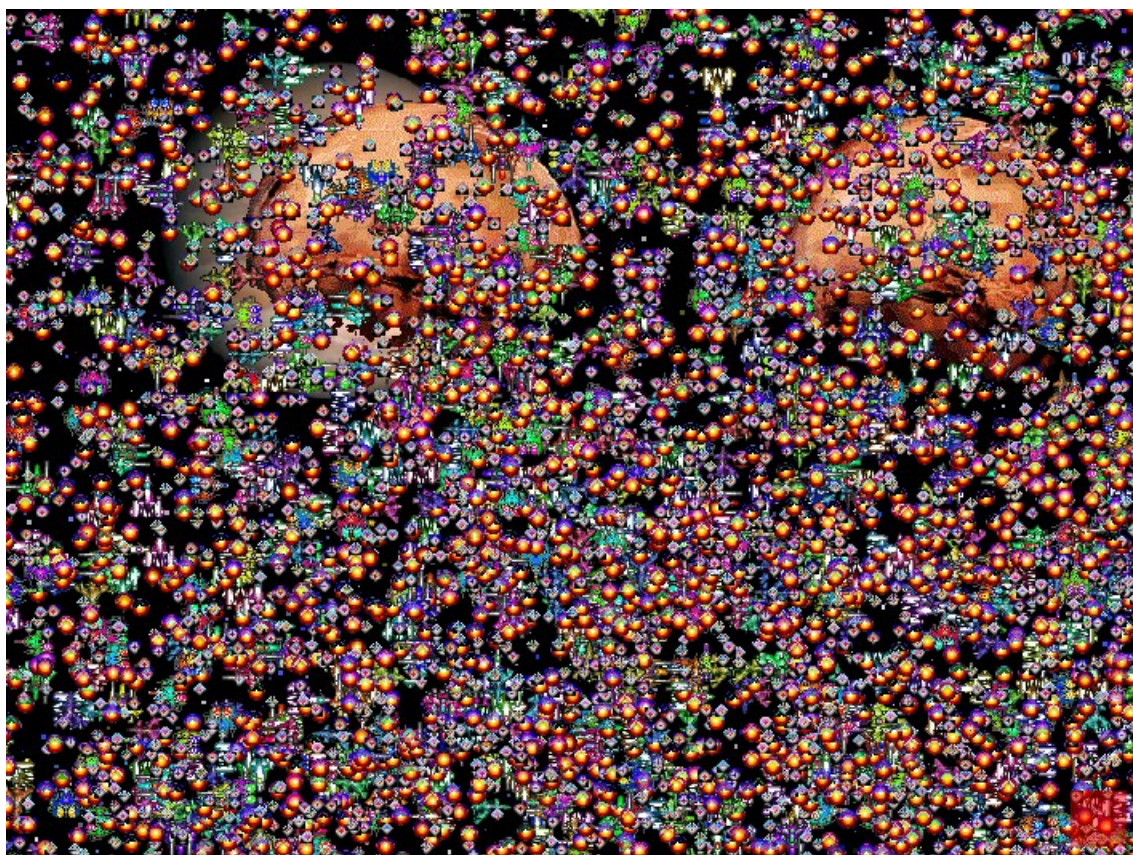
Treenor en images – la bataille fait rage dans le vide interstellaire



Treenor en images – Une victoire pirate avec explosion final du train de l'espace ... un feu d'artifice !



Treenor en images – sur PC – space jam



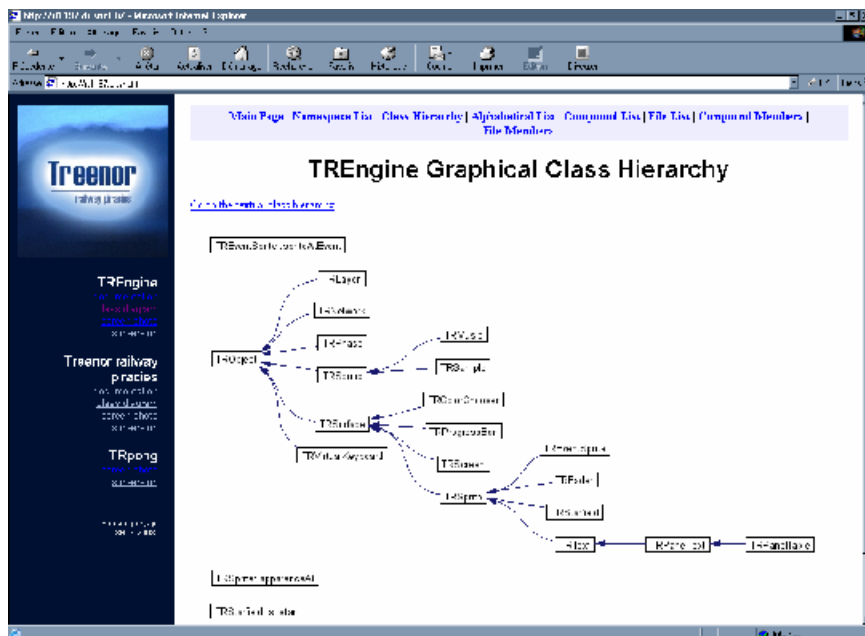
Treenor en images – sur PC – Incredible space jam !

7.4. Treenor : les stations orbitales



Treenor en images – que les stations orbitales ne représentent les combats que logiquement ne les empêches pas d’être graphique.

7.5. The Treenor Project : site Internet (non publié)



Treenor – site Internet – Diagramme de classes et documentation de TREngine, TRPong et Treenor.

8. Conclusion

Ce document a eu pour objectif de retracer l'ensemble des études et travaux réalisés dans le cadre de ce stage qui s'est avéré très complet. Après avoir présenté au lecteur l'entreprise *SNCF* et son unité de recherche en nouvelles technologies, le cadre de recherche et les objectifs du projet ont été exposés. Afin de mieux appréhender l'offre actuelle dans le domaine du ludisme sur terminaux nomades communicants, un état de l'art a été réalisé.

A la suite d'un processus de sélection d'un scénario répondant à toutes les contraintes et attentes de la *SNCF* et des acteurs du projet *RNRT SAFARI*, et à l'issue de trois mois de développement intenses et très riches en enseignements, *Treenor railway piracies* a vu le jour. Ce jeu, basé sur un concept original et une technologie nouvelle, permet de mettre en valeur les caractéristiques des réseaux ad-hoc et d'en démontrer les performances.

Qui plus est, un *frame work* facilitant la réalisation d'applications ludiques multi-plateformes en deux dimensions est né : le *TREngine*. L'éventuelle publication que je propose du *TREngine* en licence GPL, permettrait à la *SNCF* de valoriser le travail réalisé pendant ce stage mais surtout de bénéficier d'une poursuite des travaux d'amélioration du système grâce à la mise à contribution de centaines de développeurs de la communauté du logiciels libres.

Le démonstrateur en gare en 2005 pourrait marquer le lancement de la *SNCF* dans le ludisme ? Et dès 2006 *Treenor* dans toutes les poches ?

This document was created with Win2PDF available at <http://www.daneprairie.com>.
The unregistered version of Win2PDF is for evaluation or non-commercial use only.