Lucas GERARD
Laura TRIVINO
Loïc LEPRIEUR
Louis ZWAVIAK
S3B

Rapport de l’étude préalable :

S22 - Jeu d’aventure textuel

Table des matières

TODO :

- créer un lexique (moteur != éditeur / emplacement qui peut être une salle ou un lien...)
- vérifier que les termes utilisés ne sont pas des synonymes et les supprimer le cas échéant
- rédiger l’introduction
- changer les solutions techniques 3)
- Ajouter les sources sur les pages où elles sont utilisés

Introduction

1.1) Consigne

L'intitulé du sujet du projet tutoré est "jeu d'aventure textuel",

On définira les termes utilisés dans le rapport pour clarifier leur sens dans le lexique ci-dessous.

Lexique :

- **Éditeur** : L'éditeur construit un comprenant des cartes où chacune des cartes contienne un niveau.
- **Script** : texte structuré par un langage formé d'un verbe et d'un complément. Le script sera écrit dans l'éditeur pour créer des niveaux. Chaque niveau aura un ensemble de règles
- **Règle** : structure d'un script, donnant les conditions nécessaire au déclenchement d'une conséquence. Chaque règle régis une salle. On aura une règle général : liste de verbes + liste de compléments → conséquence
- **Verbe** : condition définit dans l'éditeur qui implique une conséquence
- **Complément** : précision d'un verbe, le complément donne un plus grand choix de conséquences que le verbe seul
- **Conséquence** : résultat d'un verbe donné par le moteur. Les conséquences peuvent être graphiques (changement de l'image), liées à l'inventaire (ajout/suppression d'un objet) ou à la position dans le niveau (la salle
- **Niveau** : interprétation du script généré par l'éditeur. Le niveau correspond au fichier de sortie de l'éditeur lisible par le moteur.
- **Carte** : ensemble des salles et des liens entre salles d'un niveau
- **Lien** : jointure entre deux salles de la carte. Lorsque deux salles sont reliées par un lien, alors le déplacement du joueur par le moteur est possible, sinon aucun déplacement n'est effectué.

- **Salle** : lieu formé d'une description, d'une image et d'une liste d'objets. La salle sera l'emplacement courant du joueur. Le joueur parcourt la carte de salle en salle.
- **Objet** : entité ramassable ou jetable présente dans une salle. Chaque objet peut être utilisé dans une phrase par le joueur afin d'interagir avec les salles des niveaux.
  Dans l'éditeur, un objet peut être utilisé dans une condition ou une conséquence.
- **Joueur** : personne physique interagissant avec un niveau. Le joueur peut rentrer une phrase dans un formulaire présent dans chaque salle. Les verbes et les compléments de la phrase rentrée seront analysé selon les règles de la salle. Une conséquence sera exécutée si une des règles de la salle courante correspond à la phrase.

1.2) Description du sujet :

Le sujet "jeu d'aventures textuelles" est un jeu dans lequel le joueur interagit avec le jeu grâce a une zone de texte dans laquelle il décrit des actions qu'il souhaite exécuter.

Nous développerons le jeu avec : Un éditeur de jeu et un moteur de jeu (qui exécutera un niveau édité par l'éditeur).

L'éditeur de jeu consistera à une interface simpliste et intuitive dans laquelle l'utilisateur (ici appelé concepteur) crée un niveau. C'est à dire définir l'environnement (La carte), les salles (se trouvant dans la carte), les objets, les verbes (à ajouter si les verbes disponibles ne suffisent pas) et les règles (composées de conditions et de conséquences qui agissent sur l'environnement et/ou un objet et/ou sur l'inventaire du joueur).

Le moteur de jeu consiste à charger un niveau crée a partir de l'éditeur. C'est le moteur de jeu qui est réellement jouable car il permet à l'utilisateur (maintenant joueur) d'évoluer dans le niveau grâce a une zone de saisie de texte (dans laquelle il entre les actions qu'il souhaite exécuter).

1.3) Étude de l'existant :

Les jeux d'aventures textuel furent un genre très populaire dans les années 80. Ces jeux consistaient en l'incarnation d'un personnage se déplaçant de lieu en lieu. Les commandes se résumaient à des phrases naturelles interprétées par un moteur d'analyse syntaxique. Le jeu pouvait ainsi comprendre les intentions du joueur et modifier le jeu en conséquence.

Applications similaires :

- **SRAM 1 & 2 (1986) :**

  Jeu aventure textuel ou de fiction interactive par analyse syntaxique.
  Via le clavier le joueur est invité à taper des chaînes de caractères sous forme de phrases contextuelles (demander quelque chose, aller vers, prendre...)

  Chaque phrase déclenchera une action si celle-ci est compréhensible par l'analyseur syntaxique, à défaut une réponse humoristique sera donnée.

  Le jeu se déroule sous forme de tableaux, une ou plusieurs actions est à réaliser dans un ordre qu'il faudra découvrir, les déplacements sont effectués en suivant les points cardinaux (nord, sud, est, ouest).
- **Le passager du temps (1987) :**

2.1) Fonctionnalités (use cases) : TODO différencier acteurs et introduire + expliquer tableau

Les cas d'utilisation sont découpés en trois packages : l'**interprétation des expressions** (soit la récupération des verbes et des compléments des phrases entrées au clavier et définit la conséquence à exécuter par le moteur du jeu), **le moteur du jeu** (

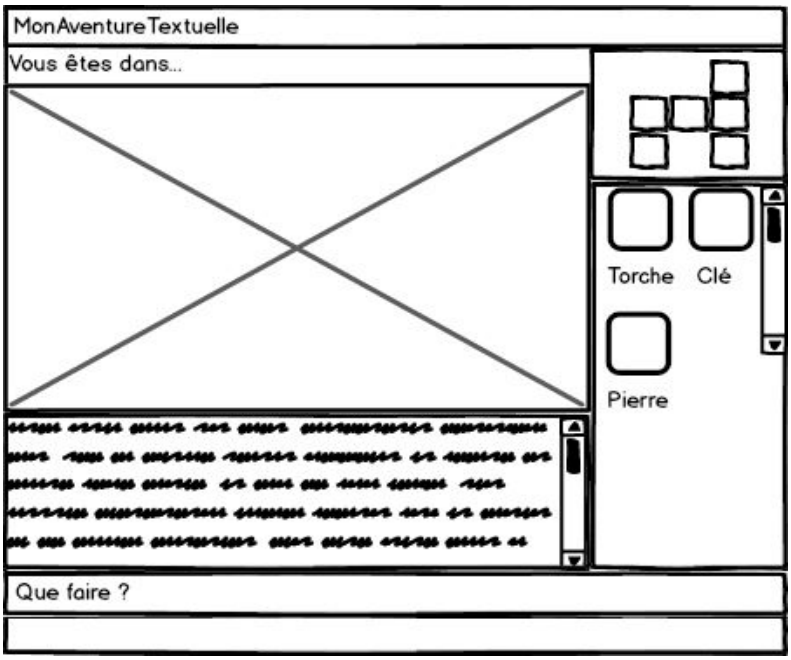
Fonctionnalité	Sous-Fonctionnalité	Description de la fonctionnalité
Interprétation des expressions		
Trouver expression	**Pas de fonctionnalités secondaire**	Le moteur graphique va lire la phrase écrite par le joueur, le moteur graphique va ensuite découper la phrase et regarder la signification de chaque mot. Le moteur graphique va ensuite rechercher un verbe en premier, puis un complément et s'arrêtera.
Interpréter expression	**Pas de fonctionnalités secondaire**	Si des verbes et des compléments ont été trouvés alors le moteur du jeu fera appel à l'action correspondante au couple verbe/complément.
Moteur du jeu		
Déplacer joueur	**Déplacer au nord**	L'utilisateur se déplacera sur la carte du haut / bas / droite / gauche en fonction de la commande entrée. Il pourra se déplacer si une carte est existante et si le passage est possible. Le moteur placera le joueur dans le lieu demandé.
	Déplacer au sud	
	Déplacer à l'est	
	Déplacer à l'ouest	
Générer carte*	**Dessiner salles**	Les salles et les liens sont des les deux types d'emplacements possibles sur la carte. Si il existe des salles dans la carte, alors le moteur va afficher le dessin des salles au joueur.
	Dessiner liens	Les liens entre les salles sont indépendant des salles. Les liens sont affichés entre deux salles qui comporte un emplacement vide. Ainsi deux salles côte à côte sans lien ne permettent pas au moteur de déplacer le joueur entre elles.
Modifier salle	**Gérer inventaire**	Le moteur ajoute ou supprime de l'inventaire du joueur les objets présent dans la salle qui y sont supprimés ou ajoutés également.

		Exemple : lorsque le joueur veut ajouter un objet de la salle dans son inventaire alors le moteur retire l'objet de la salle et l'ajoute à l'inventaire du joueur.
	Donner description	● Lorsque le joueur rentre dans une salle, la description de la salle est affiché. ● Si le le concepteur du jeu a défini une description à un objet et une conséquence donnant la description de l'objet alors elle est affichée.
	~~Changement d'état*~~	~~Le moteur met à jour les états selon la situation. Par exemple l'état de porte ouverte si le joueur a ouvert une porte.~~
Afficher inventaire	**Lister objets**	Le moteur affichera graphiquement ou textuellement les objets dans l'inventaire
	Chercher objet	Le moteur parcourra l'inventaire du joueur et indiquera si l'objet recherché est présent ou non dans l'inventaire.
Gérer l'inventaire	**Insérer objet**	Le moteur place dans l'inventaire un objet il peut être utilisé par le joueur et remplir les conditions pour certaines de ses actions
	Retirer objet	Le moteur supprime de l'inventaire un objet, il n'est plus utilisable depuis l'inventaire par le joueur
Créer script	**Créer liens entre les salles**	Le concepteur du jeu donnera dans le script les liens entre les salles.
	Associer objets à une salle	Le concepteur du jeu définira les objets qui existent dans les salles.
	Ajouter condition	Le concepteur du jeu donnera la liste des objets nécessaire pour déclencher une conséquence existante
	Ajouter conséquence	Le concepteur du jeu donnera le comportement que le jeu devra adopter si les conditions liées à cette conséquence sont vérifiées. Le jeu peut
Éditeur de niveaux		
Créer élément	**Créer salle**	Le concepteur du jeu crée une nouvelle salle et lui affecte une image d'arrière plan.

	Créer objet	Le concepteur du jeu peut créer des objets et définira un nom, une description et une image pour chaque objet.
	Créer conséquence	Le concepteur du jeu définit le comportement du jeu (un nouvel affichage, une modification de l'inventaire, la fin du jeu...) lorsqu'une nouvelle conséquence est appelée.
	Créer salle	Le concepteur définit le nom, la description, l'image d'une salle. La salle occupera un emplacement sur la carte.
Sauvegarder script		Le concepteur enregistre la carte écris avec l'éditeur contenant la carte dans un fichier lisible par le moteur du jeu.
Charger script		Le concepteur du jeu peut charger dans l'éditeur le fichier d'un script existant afin de le modifier.

2.2) "Implémentation" (mal dit !) : TODO introduire et expliquer

- **Modèle** : liste des objets du lieu courant, liste des objets de l'inventaire du joueur
- **Vues** :
  - VueInventaire : affiche les objets en inventaire
  - VueCarte : emplacement actuel et salles adjacentes et marquées comme déjà visités dans un même niveau
  - VuePiece : lieu courant où se trouve le joueur, affiche les objets de la pièce et le décor
- **Contrôleur** : ControleurSaisie : zone de saisie des actions du joueur géré par un **contrôleur** (lexique)



2.3) Déroulement attendu (exemple):

A partir des use cases développés ci-dessus, on cherche à obtenir un comportement donné avec le moteur du jeu.
L'éditeur de niveaux doit généraliser la création d'un jeu textuel.
Les caractéristiques (nom, description, image, conséquence) des objets et les conséquences qui leur sont associées seront définis , mais plutôt permettre l'ajout de conditions nécessaires pour que des conséquences (modification de l'inventaire, changement de lieu...) soient déclenchées.
Il s'agit en résumé d'une génération dynamique d'objets et de conditions amenant à des conséquences.
On aura par exemple l'énigme d'une porte verrouillée qu'on souhaite franchir, les objets qu'on aura à notre disposition seront : porte - poubelle - feuille - clé - serrure.
Si on trouve la suite correcte de conditions, on terminera le jeu. L'exemple suivant permet de visualiser une résolution type de ce genre d'énigme et comment le jeu devra gérer le cas.

- Regarder serrure (voir clé)
  - Interaction avec l'environnement par une syntaxe observer/regarder/...
- Fouiller poubelle (--> feuille)
  - gestion de l'inventaire : ajout
  - environnement : la feuille n'y est plus
- Placer feuille sous porte
  - Condition : avoir la feuille dans l'inventaire + faire action de placer la feuille
- Secouer porte (clé tombe)

    - répond à la condition d'avoir la feuille placée et d'écrire l'action secouer
    - environnement : la clé change d'état
- Tirer feuille (-->clé)
    - Récupération de la clé dans l'inventaire et changement d'état de la clé
    - Répond aux conditions : avoir fait tomber la clé sur la feuille et d'écrire l'action de tirer cette feuille
- Ouvrir serrure avec clé (porte ouverte)
    - Changement d'état dans l'environnement : porte ouverte et donc nouveau lieu possible d'accès
- Aller vers Nord
    - Le jeu change de lieu le personnage, vers le nouveau lieu souhaité par le joueur
    - Répond à la condition d'avoir ouvert la porte et de demander l'action

→ **FIN DU JEU**

3) Solutions techniques : TODO à revoir !

- **Analyseur syntaxique simplifié :**

A partir d'un fichier JSON répertoriant un verbe principal et ses synonymes, l'analyseur recherche le verbe donné et il renvoie le verbe principal correspondant. Une fois la phrase interprétée, elle est envoyer à un vérificateur de règle qui va comparer les conditions à la phrase interprétée, et appliquera les conséquences si les conditions sont vérifiées.

4) Sources : TODO à replacer dans les pages

- Sujet du projet : https://drive.google.com/file/d/0B-95anfbJhzEaFFOOGcxOGtrcjA/view

- Analyseur syntaxique :

- http://stackoverflow.com/questions/8204914/syntax-analysis-and-syntax-tree
- http://www.commentcamarche.net/forum/affich-13587602-analyse-syntaxiqueGén-en-java
- http://blog.soat.fr/2013/12/08-jvm-hardcore-part-7-mon-premier-analyseur-syntaxique-12/
- https://www.enseignement.polytechnique.fr/informatique/ARCHIVES/IF/poly/main004.html
- Wiki de Analyse_syntaxique

- Jeux textuels :
  - http://www.grospixels.com/
  - http://www.planetemu.net/article/le-passager-du-temps