

Rapport final

TER2022-025 : Validation de règles de conception dans les IHM réparties

Projet TER

Étudiants :

Yann Clodong - SI5 - IHM - yann.clodong@etu.univ-cotedazur.fr
William D'Andrea - SI5 - AL - [william.d'andrea@etu.univ-cotedazur.fr](mailto:wiliam.d'andrea@etu.univ-cotedazur.fr)
Guillaume Piccina - SI5 - AL - guillaume.piccina@etu.univ-cotedazur.fr

Encadrants :

Anne Marie Pinna Dery, encadrante, Anne-Marie.PINNA@univ-cotedazur.fr
Philippe Renevier , co-encadrant, Philippe.RENEVIER@univ-cotedazur.fr
Philippe Collet, co-encadrant, Philippe.COLLET@univ-cotedazur.fr

Sommaire

Sommaire

1. Introduction

1.1. Contexte du TER

1.2. Sujet de TER

2. État de l'art

2.1. Particularité de notre sujet

2.2. Documentation autour de la territorialité et des interfaces distribuées

2.2.1. Territorialité

2.2.2. Interfaces distribuées

2.2.3. Conception d'interfaces distribuées

2.3. Notre problématique dans ce contexte

3. Travail effectué

3.1. Introduction aux règles du jeu

3.2. Première version naïve

3.2.1. Les débuts du projet (Milestone 1)

3.2.1.1. Objectifs de cette première itération

3.2.1.2. Scénario

3.2.1.3. Implémentation

3.2.2. Ajout d'interactions et du nombre de joueurs (Milestone 2)

3.2.2.1. Description des interactions utilisateurs avec notre système

3.2.2.2. Description des modèles de données échangés et prise de recul

3.3. En connaissant les règles

3.3.1. Prise de recul sur les règles

3.3.1.1. Phase d'identification

3.3.1.2. Phase de distribution

3.3.1.3. Phase d'équilibrage

3.3.1.4. Phase de transition

3.3.2. Amélioration de notre application grâce aux règles

3.3.2.1. Amélioration de l'espace privée

3.3.2.2. Amélioration des popups au moment du choix du stack central

3.3.2.3. Amélioration de la dynamique du jeu

3.3.3. Modification des dispositifs

3.3.3.1. Suppression des téléphones

3.3.3.2. Suppression de la table

3.3.3.3. Modification de l'environnement

4. Répartition du travail

5. Conclusion

6. Bibliographie

7. Annexe

1. Introduction

1.1. Contexte du TER

La disposition d'éléments d'interface est compliquée lors de la conception d'interfaces réparties (Interface Homme Machine s'exécutant sur plusieurs dispositifs).

Des études et des papiers scientifiques ont pour vocation d'aider les concepteurs d'applications dans la définition des interactions entre utilisateurs, par l'intermédiaire de règles de conception. L'objectif de ce TER est, par l'intermédiaire d'un jeu de cartes (6 qui prend), d'utiliser ces règles afin de voir l'impact qu'elles ont sur la conception d'un système distribué.

Pour jouer au jeu, les joueurs se retrouveront autour d'une table tactile, où ils pourront interagir avec les actions générales du jeu et prendre des décisions de façon confidentielle sur leur téléphone. Cette table et les téléphones seront connectés à un serveur distant qui aura pour objectif d'exécuter les actions automatiques du jeu, tel que démarrer ou finir le jeu, vérifier et affecter les points aux joueurs à chaque tour, etc.

1.2. Sujet de TER

Dans ce projet, nous visons à enquêter sur l'efficacité d'une méthode basée sur des règles pour concevoir des applications multi-dispositifs. Le point principal de ce projet est de créer un MVP (Minimum Viable Product) du jeu "6-qui-prend" et de comparer l'expérience utilisateur des différentes versions du jeu.

Notre TER se décompose en 2 phases :

- Dans un premier temps, nous allons créer un MVP du jeu "6 qui prend" sans regarder, ni suivre les règles définies dans le papier scientifique défini dans l'article "*Territoires et IHM Distribuées : Raffinement de Règles et d'une méthode de Conception de Jeux Multi-Dispositifs*".
- Dans un deuxième temps, nous allons lire et assimiler les règles de conception, puis regarder pour chaque règle si nous l'avons prise en compte de manière intuitive ou pas.

Après discussion en début de semestre avec nos tuteurs, nous avons trouvé cette approche judicieuse : dans la mesure où la valeur ajoutée de notre TER est de permettre de comprendre si les règles sont appliquées de manière intuitive ou non, et savoir si suivre cette méthodologie à une réelle valeur ajoutée sur le développement d'une application.

2. État de l'art

2.1. Particularité de notre sujet

Notre sujet nous amène à réaliser un jeu qui est adapté d'un jeu physique, nous devons toujours garder en tête que celui-ci doit pouvoir être joué de manière semblable à la version physique. Autrement dit, nous devons essayer d'adapter ce jeu au format numérique en gardant ses règles et "l'esprit" du jeu tout en le rendant facilement jouable et intuitif pour les utilisateurs. En sachant que les utilisateurs peuvent être aussi bien des joueurs qui connaissent le jeu de carte physique ou bien des joueurs totalement novices. Pour adapter ce jeu au plus près de la version physique, nous allons le développer sur des interfaces réparties. La difficulté pour développer une application sur une interface répartie réside notamment dans le fait de définir quelles actions seront réalisées et surtout à quels endroits on pourra les réaliser. Ces décisions sont basées sur leur niveau de confidentialité, criticité, ou d'identification de l'utilisateur. Cette problématique va guider ainsi les choix de conception de notre application. Pour nous aider dans cette tâche, nous nous sommes renseignés à l'aide d'articles de recherche qui concernent le développement d'interface répartie.

2.2. Documentation autour de la territorialité et des interfaces distribuées

2.2.1. Territorialité

Comme dit auparavant, dans des problématiques d'interface répartie, il est important de pouvoir définir dans un premier temps les actions ainsi que les zones où réaliser ces actions. En ce qui concerne la gestion des zones pour les utilisateurs, on parle ici de territorialité, on peut définir la territorialité comme le principe selon lequel le champ d'application d'une règle est limitée à un espace territorial.

L'article "*Territoriality in Collaborative Tabletop Workspaces*" essaye d'analyser l'interaction entre des participants autour d'un jeu, défini en plusieurs "espaces", et regardent jusqu'à quand chaque espace devient inutilisable. Dans cet article, ils définissent trois zones territoriales avec chaque territoire ayant une utilité bien précise. Les zones sont les suivantes : les territoires personnels (espace situé proche de l'utilisateur, son espace privé), les territoires de groupe (territoire placé au centre de la table, c'est un espace public, et chaque joueur se situe à peu près à la même distance de cet espace) et les territoires de stockage (qui est la frontière des espaces privées). De plus, elle suggère que le comportement territorial s'étend au-delà des espaces physiques et affecte les interactions virtuelles.

Le centre de recherche INRIA, dans l'article "*Balancing Act: Enabling Public Engagement with Sustainability Issues through a Multi-touch Tabletop Collaborative Game*" a par ailleurs travaillé sur le sujet de la territorialité en créant un jeu, nommé "Futura", qui a été déployé, et testé durant les jeux olympiques d'hiver à Vancouver en 2010. Ce jeu a pour objectif de rassembler des personnes durant des événements, autour d'une activité ludique et enrichissante. Ils ont également utilisé une approche en différents territoires, cependant, ils n'ont pas utilisé de territoires "dynamiques". Un territoire dynamique signifie que l'on peut le bouger d'endroit, par exemple, pour un territoire privé, l'utilisateur aurait le choix de la bouger là où il se situe géographiquement. Dans le cadre de ce projet, les territoires étaient statiques, et ne pouvaient pas être déplacés. Ceci leur a permis d'être plus rapides et plus flexibles quand un grand nombre de personnes arrivaient. Ils concluent sur le fait que l'approche par territoire dynamique et statique doit être adaptée selon la situation, et qu'il faut trouver un compromis selon le projet.

2.2.2. Interfaces distribuées

De nombreuses études sur la territorialité ont eu lieu avant l'émergence au grand public de téléphones portables connectés à internet. Par exemple, dans l'article parlant du jeu déployé aux jeux olympiques, les espaces personnels étaient placés sur la table tactile, limitant donc l'isolation si des utilisateurs doivent faire des actions secrètes (car tout le monde peut voir la table).

Suite à cette émergence, un pan de la discipline, nommée Computer Supported Collaborative Learning (CSCL), a pu réaliser de nombreuses avancées. Elle traite de l'interaction entre plusieurs utilisateurs, à des fins d'apprentissages. Le but est d'analyser les interactions entre les utilisateurs et les dispositifs distribués, afin de voir quelles interactions aident le mieux à apprendre.

Un projet, nommé FLUID ("FLUID: Flexible User Interface Distribution for Ubiquitous Multi-device Interaction") a vu le jour en 2019, suite à la constatation qu'encore trop peu d'applications sur le marché étaient distribuées et multi-dispositifs. L'équipe a également écrit un papier de recherche expliquant les détails de l'implémentation, mais aussi la partie architecture logicielle, car selon les utilisations (par exemple une interface répartie qui fait de la vidéo en temps réel ou un jeu), les contraintes matérielles ne sont pas les mêmes, au niveau de l'aspect hardware du dispositif, mais aussi de l'aspect réseau. Grâce à FLUID, ils ont ainsi créé une solution logicielle sous Android, qui permet à des développeurs de créer des interfaces distribuées beaucoup plus simplement, ayant donc pour objectif de faire émerger plus d'applications multi-dispositifs.

2.2.3. Conception d'interfaces distribuées

Si dans les parties précédentes, les recherches actuelles ont été relativement nombreuses, cette partie est un peu moins riche en recherche. En effet, le but étant d'essayer de trouver de la littérature qui traite de la manière de créer des interfaces distribuées. À notre connaissance, nous n'avons trouvé que notre papier de recherche initial, traitant des règles de conceptions pour créer des interfaces réparties, qui est "*Territories and Distributed HCI: Refining Rules and a Method for Designing Multi-Device Games*".

Cet article, que nous détaillerons bien plus en détails dans la fin de notre rapport, a pour objectif de donner une suite de règles, qui, si elles sont suivies, aident à créer des interfaces réparties de manière logique, et amènent à réfléchir sur les vraies interactions nécessaires, afin de créer une implémentation fluide et cohérente pour l'utilisateur.

2.3. Notre problématique dans ce contexte

Comme dit précédemment, la littérature scientifique analyse le domaine des interfaces réparties et particulièrement les théories autour de la territorialité, et d'à quel point les interfaces réparties peuvent aider à résoudre des problèmes dans différents domaines. Concernant les règles de conceptions, nous n'avons pas trouvé énormément de littérature sur ce sujet, nous pouvons donc nous demander s'il est vraiment nécessaire de suivre des règles pour développer des interfaces réparties.

Dans ce contexte scientifique, notre objectif est ainsi de pouvoir apporter des précisions par rapport à la véracité et surtout l'efficacité de ces règles de conceptions. Globalement, nous nous posons la question de savoir si le fait de connaître la théorie et les règles de conception autour des interfaces réparties nous aide réellement à gagner du temps lors du développement, et à développer des applications qui améliorent, dès la première itération du projet, l'expérience utilisateur. Ou bien est-ce que ces règles et la théorie sont prises en compte par les développeurs naturellement et de ce fait qu'il est inutile de les apprendre ?

3. Travail effectué

Comme nous avons pu le voir en fin d'analyse de l'existant, nous allons nous concentrer sur la validation de règles. Pour cela, nous allons réaliser notre projet en deux phases séquentielles. La première étant la création naïve d'un jeu, nous n'allons ici appliquer aucune règle, nous n'en prenons même pas connaissance. Nous allons développer le jeu de manière intuitive. Ensuite, nous allons prendre connaissance des règles de conceptions, nous allons regarder ce qu'il nous manque dans notre projet, et essayer d'améliorer le jeu en suivant ces règles de conceptions, afin de voir si, oui ou non, elles sont intuitives, et s'il est vraiment nécessaire de les connaître pour pouvoir implémenter une interface distribuée.

3.1. Introduction aux règles du jeu

Le 6-qui-prend (ou 6-Nimmt!) est un jeu de cartes physique. Le jeu se compose de 104 cartes qui ont chacune un certain nombre de taureaux dessiné dessus. Le gagnant est celui qui, à la fin du jeu, a le moins de taureaux dans sa défausse.

Au début du jeu, nous distribuons 10 cartes (de manière aléatoire) à chaque joueur, et nous posons 4 cartes au centre sur une table (ces 4 cartes représentent 4 tas de cartes). À chaque tour, chaque joueur pioche une carte parmi ses cartes restantes, qui sera ensuite posée sur un des 4 tas du milieu de table. Afin de choisir sur quel tas va être posé sa carte, il y a 2 possibilités :

- **Situation 1** : Il existe un tas sur la table composé de moins de 6 cartes, ou la carte de l'utilisateur est plus grande que la carte du haut du tas, dans ce cas-là, l'utilisateur pose sa carte sur ce tas.
- **Situation 2** : Il existe un tas (numéro 1) sur la table composée de 6 cartes, et un autre tas (numéro 2) composé de N cartes. Si la valeur de la carte de l'utilisateur est entre la valeur de carte du haut du tas numéro 1 et la valeur de la carte du haut du tas numéro 2, alors l'utilisateur prend tout le tas numéro 1, le met dans sa défausse, et recommence un nouveau tas avec la carte qu'il avait initialement.
- **Situation 3** : L'utilisateur veut déposer une carte de valeur inférieure à toutes les cartes du haut des 4 tas. Dans ce cas-là, l'utilisateur peut choisir quel tas il prendra dans sa défausse, le prend et le met dans sa défausse, et finalement, il place sa carte initiale pour recréer un nouveau tas.

3.2. Première version naïve

Dans un premier temps, nous avons donc travaillé sur le développement du jeu sans avoir lu les règles de conception d'IHM réparties. Le but était globalement de voir la démarche selon laquelle nous procédons, selon laquelle nous "gérons" les interactions, afin d'ensuite avoir un recul lorsque nous appliquerons l'application en suivant les règles.

Pour mettre en place des deadlines ainsi que nous challenger sur des problématiques d'AL et d'IHM nous avons divisé ce projet en 2 milestones :

- Milestone 1 : permettre à 2 joueurs de jouer 10 tours, avec une disposition automatique des cartes sur chaque tas.

Cette première milestone nous permet de lancer le projet en nous challengeant d'un point de vue AL puisque nous devons implémenter une première version du moteur de jeu avec le minimum de joueurs. Et d'un point de vue IHM, nous devons commencer à réfléchir sur comment répartir les informations sur les dispositifs et de quelle manière ces choix vont impacter l'expérience utilisateur.

- Milestone 2 : permettre à N joueurs (maximum 8) de jouer 10 tours, avec le choix (quand nécessaire) pour l'utilisateur de choisir quel tas il aura dans sa défausse si nous sommes dans le cas de la situation 3.

Cette deuxième milestone permet d'obtenir le nombre de joueurs maximum et implémente entièrement toutes les règles du jeu physique. Cette milestone apporte son lot de challenge avec une réflexion à avoir sur l'impact du changement du nombre de joueurs aussi bien d'un point de vue architectural que d'un point de vue IHM, mais aussi l'ajout de nouvelles interactions avec le choix de tas de cartes.

Nous avons essayé de réaliser ces milestones de manière le plus transversale possible, afin qu'à chaque milestone, nous ayons un projet fonctionnel et démontrable.

Nous allons maintenant rentrer dans les détails de chaque milestone afin d'expliquer comment nous avons procédé, mais aussi les problèmes auxquels nous avons été confrontés, avec les changements potentiels effectués.

3.2.1. Les débuts du projet (Milestone 1)

3.2.1.1. Objectifs de cette première itération

Créer une partie regroupant 2 joueurs, chacun ayant une carte et où il ne sera pas possible pour l'utilisateur de choisir le tas sur lequel envoyer sa carte si celle-ci est inférieure à toutes celles présentes sur le jeu, c'est-à-dire :

- Permettre aux utilisateurs de se connecter à une partie,
- Gérer les phases de jeu,
- Afficher les informations publiques sur la table : tas, utilisateurs, etc.
- Permettre aux utilisateurs de jouer une carte,
- Implémenter les règles de placement des cartes (à l'exception du choix du tas par l'utilisateur),
- Compter le score des utilisateurs

Nous avons en conséquence dû créer une application pour la table, une autre pour le téléphone et finalement un backend pour relier les deux parties et gérer la logique.

Ces trois parties regrouperont les interactions suivantes :

- Sur la table : afficher un endroit représentant l'utilisateur sur la table pour afficher les données publiques le concernant, à savoir s'il a joué ou non, etc.
- Sur le téléphone : afficher au contraire les informations et interactions privées de l'utilisateur.
- Et donc dans le backend, gérer les phases de jeux, valider les interactions de l'utilisateur, calculer les scores des joueurs, afficher les cartes sur les bons tas, etc.

3.2.1.2. Scénario

1. Les utilisateurs rejoignent une partie en entrant un identifiant d'utilisateur disponible sur leur téléphone : ce qui permet d'assigner ce téléphone en tant que territoire personnel.
2. Ils lancent la partie quand tous les joueurs l'ont jointe.

>> *Le jeu commence*

3. Ils choisissent une carte sur leur téléphone.
4. Lorsque tous les joueurs ont terminé, ils constatent que leur carte est placée sur un tas.

>> *Fin de partie*

5. Ils peuvent lire leur résultat.

3.2.1.3. Implémentation

Diagramme de séquence

Afin de comprendre les interactions entre chaque entité de notre système, nous avons créé un diagramme de séquence.

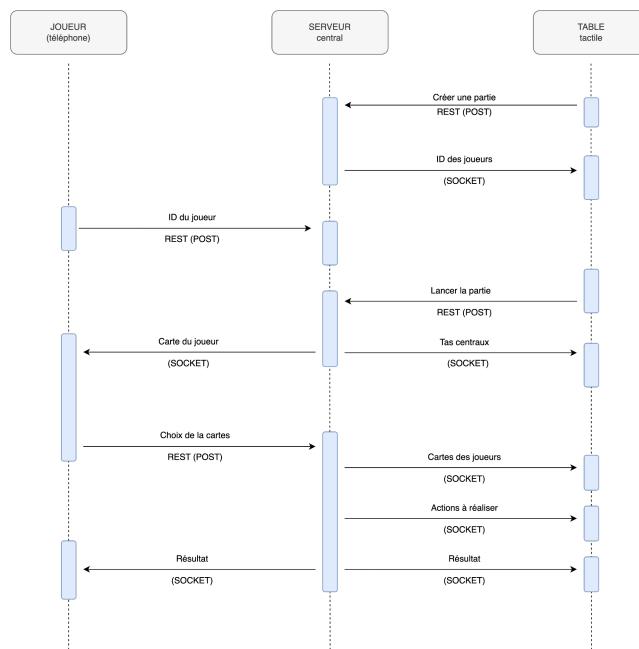


Figure 1 - Diagramme de séquence de l'architecture backend de la milestone 1

Voici les différentes étapes de ce diagramme de séquence :

1. Lors du lancement de la table, celle-ci enverra une requête POST au serveur lui demandant de créer une nouvelle partie.

2. Celui-ci renverra alors une liste de 10 ids qui seront affichés sur la table et avec lesquelles les utilisateurs pourront rejoindre la partie.
3. Les utilisateurs peuvent ainsi entrer leurs ids dans l'interface personnelle, et le valider en utilisant le bouton. Le téléphone enverra ainsi une requête au serveur pour manifester son intention de rejoindre la partie.
4. Une fois les deux utilisateurs connectés, l'un d'entre eux peut donc lancer la partie en cliquant sur le bouton "Démarrer la partie" au centre de la table, qui demandera via une requête POST de lancer la partie.
5. Le serveur enverra les informations nécessaires au début de la partie, c'est-à-dire, les cartes des joueurs, les cartes de début des tas via un WebSocket.
6. Les utilisateurs peuvent alors choisir parmi les cartes qu'ils ont reçues celles qu'ils veulent jouer.
7. Lorsque tous les utilisateurs ont joué, le serveur enverra le résultat du tour à la table, c'est-à-dire :
 - a. Les cartes jouées par les utilisateurs, afin que la table puisse les afficher.
 - b. Puis envoyer les actions que la table doit réaliser sur les cartes jouées et les tas. Par exemple, déplacer une carte jouée sur le tas n°3.
 - c. Enfin, il renverra également le score de chaque utilisateur et clôturera la partie.

Modèle de données

Nous pouvons voir sur le diagramme de séquence que nous avons 2 types de protocole de communication. Nous avons une communication via des routes REST pour tout ce qui est communication table → serveur et joueur → serveur, et une communication via websocket (dans notre implémentation, nous utilisons socket.io) pour les messages serveur → joueur et serveur → table.

Ce choix a été fait, car la table et le joueur connaissent l'URL du serveur, donc il est possible pour eux de contacter le serveur central via une communication REST puisque ce serveur est unique. Cependant, il est plus compliqué pour le serveur de contacter la table ou les joueurs, parce qu'ils n'auront pas d'URL fixe, le serveur ne peut alors pas savoir comment les contacter. Nous nous sommes ainsi orientés vers un socket, qui permet d'avoir une liaison directe entre le serveur et les joueurs (et la table). Une alternative au socket aurait été que la table et chaque application mobile envoient une requête REST au serveur toutes les secondes, afin de lui demander s'il y a des mises à jour à faire, néanmoins, ce choix n'a pas été retenu, car cela pourrait surcharger le serveur, et ce n'est pas le but, de plus, nous avons besoin d'une bonne réactivité, et si nous voudrions faire cela, il faudrait que nous fassions une requête toutes les 200 ms, cette durée correspond à une fréquence de rafraîchissement de 50 Hz, qui est la fréquence de rafraîchissement généralement admise comme suffisante, car l'œil distingue un mouvement fluide, notamment pour les jeux vidéo.

Globalement, dans notre implémentation actuelle, les données transitant entre le serveur et les périphériques externes sont assez basiques. Les modèles de données actuels reflètent relativement bien (on s'en est majoritairement inspiré) les interactions réelles que les joueurs pourraient avoir durant le jeu. Néanmoins, à un endroit, cela fut compliqué. En effet, quand les joueurs ont fini de jouer leur carte, il y a trois situations possibles (les trois situations disponibles [ici](#)).

Nous rappelons que dans cette milestone, le joueur ne choisit pas l'endroit où il veut déposer sa carte dans le cas de la situation 3, c'est fait automatiquement par notre moteur de jeu. Nous avons dû développer un modèle de données qui permettrait à la table de comprendre quelles "actions" elle a à faire.

Une action est donc représentée par un "type" qui peut être :

- “PUSH_ON_TOP” : La carte du joueur “player_id” envoyé précédemment doit être placée sur le tas représenté par la classe “StackCard”. Le “StackCard” envoyé étant le nouveau “StackCard”
- “CLEAR_PUSH” : Le joueur “player_id” doit recevoir dans sa défausse l’ancien “StackCard” (sa défausse sera mise à jour au prochain tour) et un nouveau tas (celui défini dans “stack”) remplace l’ancien

Il est à noter que ces actions ont pour objectif principal à ce que la table puisse réaliser des actions graphiques permettant aux utilisateurs de voir qui a réussi le tour et qui est le moins bon du tour.

3.2.2. Ajout d’interactions et du nombre de joueurs (Milestone 2)

3.2.2.1. Description des interactions utilisateurs avec notre système

La ou dans la première milestone, nous cherchions surtout à implémenter une première version jouable du jeu, dans cette seconde partie, nous nous sommes beaucoup plus concentrés sur les interactions. En effet, ici, nous voulions rendre le jeu beaucoup plus intuitif. L’objectif de cette partie est de vous exposer notre démarche pour évaluer quelles décisions nous avons prises au fil du temps, pour améliorer l’expérience de jeu, sans suivre les règles.

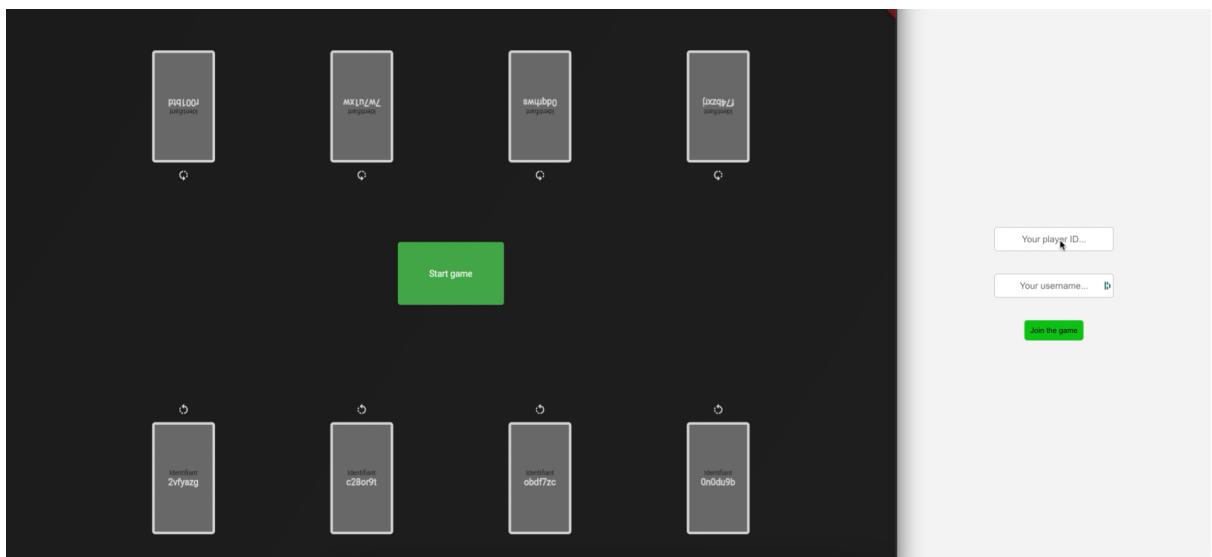


Figure 2 - Représentation de la table (à gauche) et du téléphone (à droite) au moment du login

Sur cette capture d’écran, nous pouvons voir à gauche la représentation de la table et à droite la représentation d’un téléphone. Un joueur, pour se connecter, doit choisir une carte sur la table, et y entrer l’identifiant dans la case associée sur son téléphone, avec un pseudo.

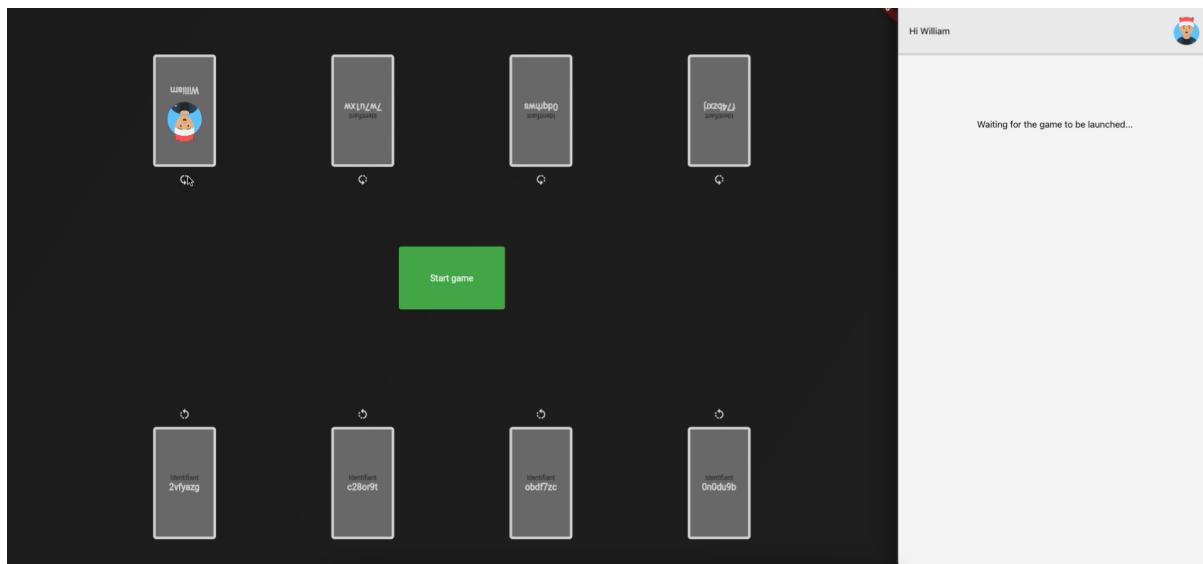


Figure 3 - Représentation de la table (à gauche) et du téléphone (à droite) quand un joueur s'est connecté

Une fois cela fait, un avatar est généré automatiquement, et l'utilisateur peut voir sur les deux périphériques qu'il est bien connecté au jeu. Nous avons fait le choix de dupliquer cette information sur les 2 périphériques, pour que l'utilisateur puisse avoir un double feedback autant sur le téléphone que sur la table.

Il est possible pour les utilisateurs de bouger leurs cartes où ils veulent et de la faire tourner. Nous avons implémenté cette fonctionnalité à la suite d'une expérience autour de la table tactile où nous nous sommes rendu compte qu'il était plus judicieux de laisser à l'utilisateur le choix de se positionner plutôt que de le fixer à une position prédéfinie. Cela permet aux utilisateurs de mieux se répartir en fonction du nombre de joueurs par exemple.

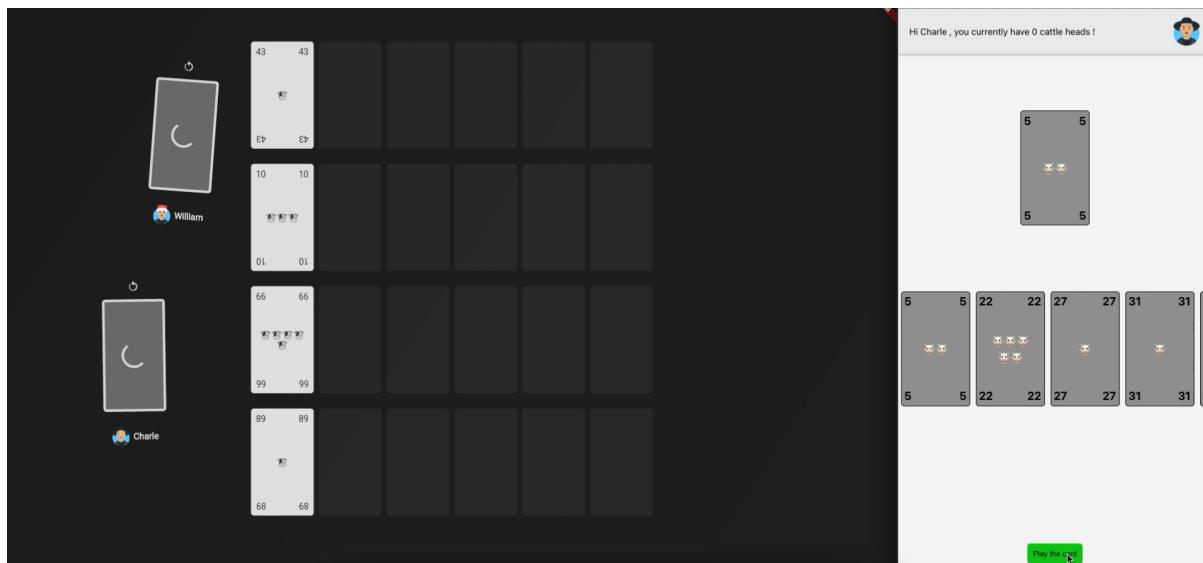


Figure 4 - Représentation de la table (à gauche) et du téléphone (à droite) quand un joueur choisit sa carte de jeu

Quand un tour commence, chaque utilisateur doit choisir une carte sur son téléphone. Pour cela, il peut slider les cartes (du bas), et dès qu'il clique sur une carte, elle est placée en haut, ensuite, il a juste à cliquer sur "Play de card" pour envoyer cette carte à la table. Cette représentation sur le téléphone nous paraissait logique au moment

de l'implémentation, car, lorsque l'on est dans la dynamique du jeu, intuitivement sous "slidons" pour regarder nos cartes, et nous cliquons sur la carte que l'on souhaite jouer.

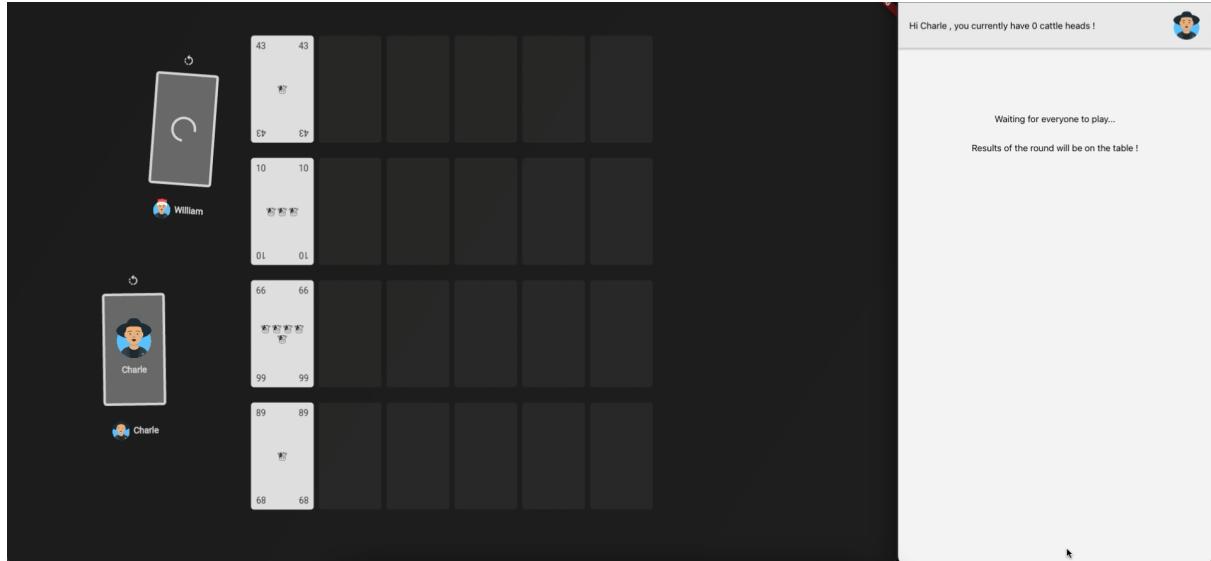


Figure 5 - Représentation de la table (à gauche) et du téléphone (à droite) quand un joueur a choisi sa carte de jeu

Une fois que le joueur a appuyé sur "Play the card", il a la double notification que sa carte a été envoyée sur son téléphone, et sur la table. Nous avons fait cela ainsi pour bien montrer à l'utilisateur que son action a bien été prise en compte, et qu'il n'a pas à s'inquiéter, il sait qu'il n'y a pas eu de bug.

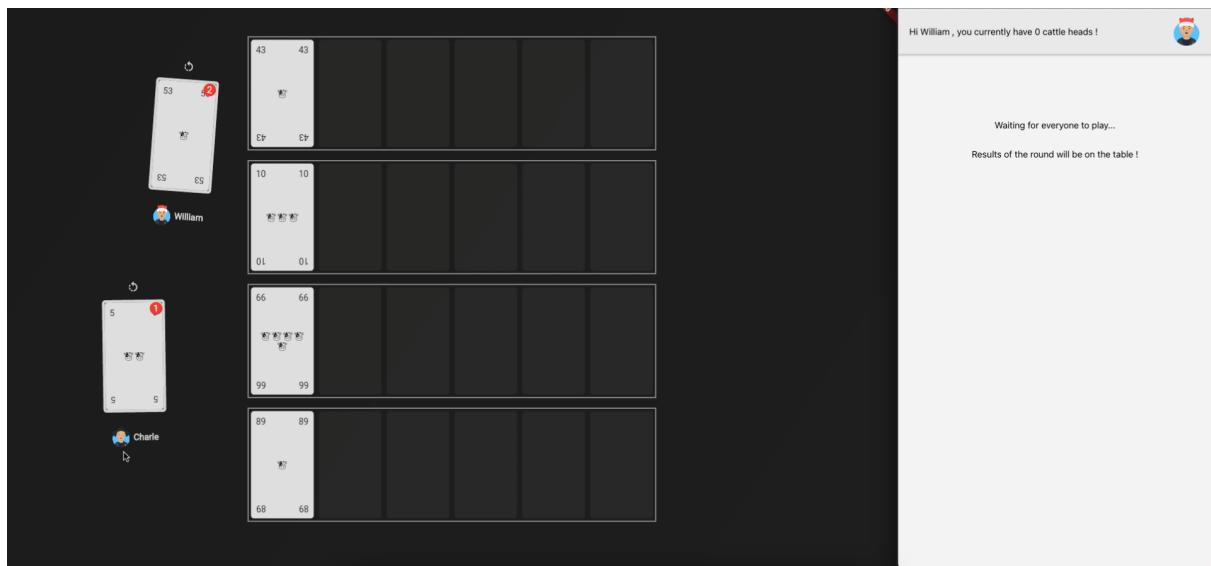


Figure 5 - Représentation de la table (à gauche) et du téléphone (à droite) quand un joueur doit sélectionner un tas central

Ici, nous pouvons voir plusieurs informations. La première étant l'ordre dans lequel les cartes seront placées au centre (ce serait le rôle d'un game-master dans le jeu réel de prendre les cartes par ordre croissant pour les placer sur le bon tas), qui est représentée par badge rouge en haut à droite de la carte. La seconde, est qu'ici, nous nous retrouvons dans la situation 3, ou un utilisateur doit choisir un tas central, car la carte qu'il a jouée est plus petite

que toutes les cartes des tas. Pour cela, un rectangle blanc s'affiche autour des tas afin de demander à l'utilisateur de cliquer sur le tas de son choix.

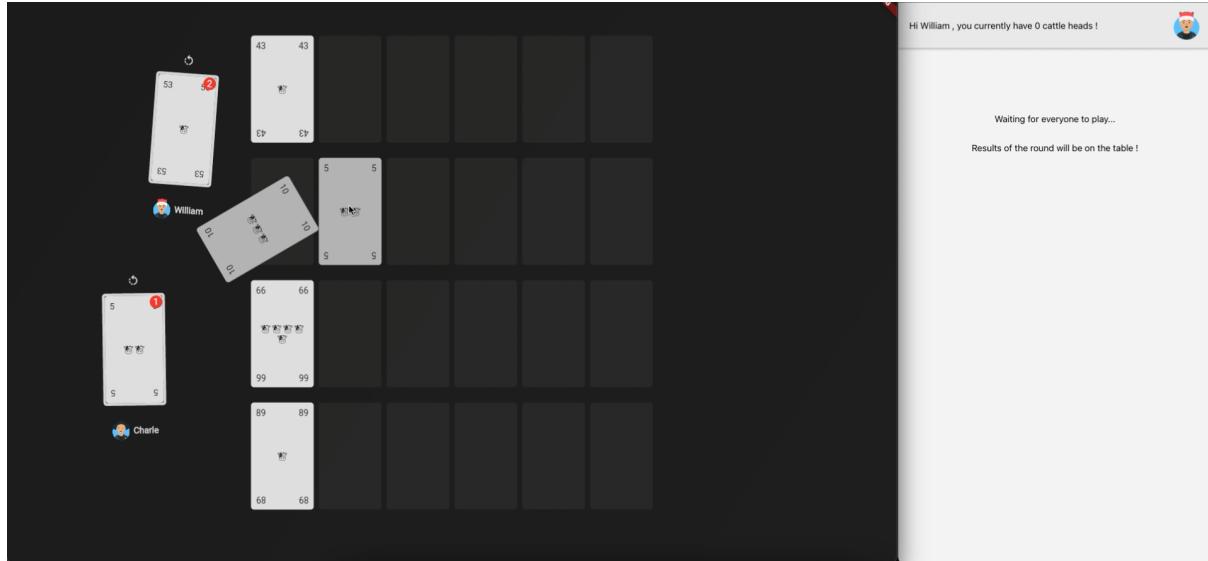


Figure 6 - Représentation de la table (à gauche) et du téléphone (à droite) avec une animation des cartes qui vont vers les tas centraux

Par la suite, via des animations, les joueurs peuvent visualiser où vont être placé leur carte, et avoir des retours pour comprendre ce qu'ils ont dans leur défausse.

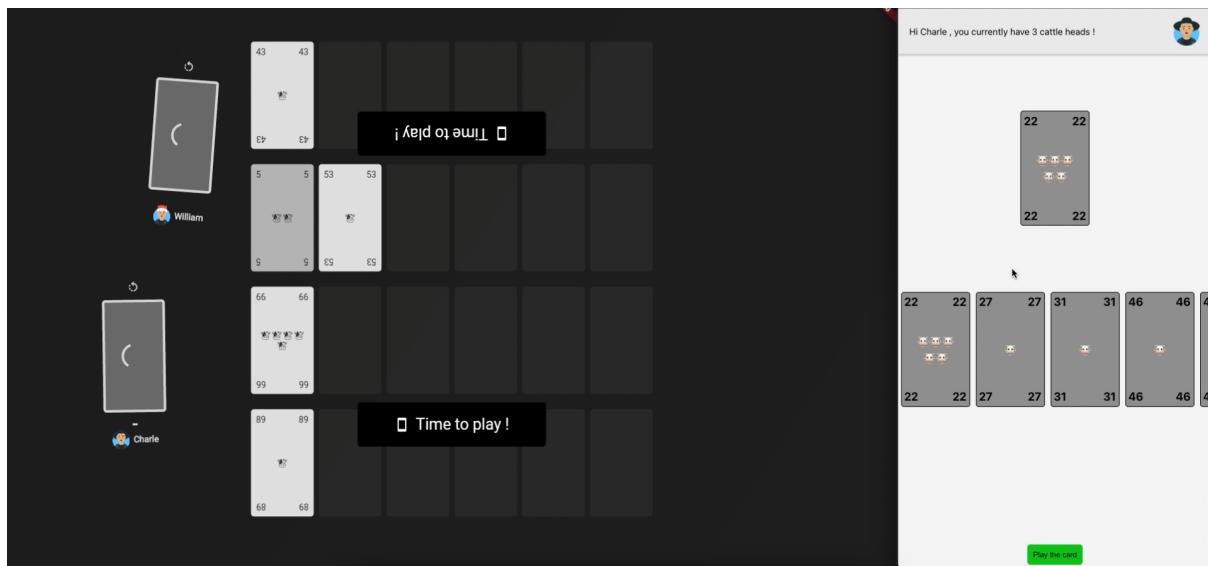


Figure 7 - Représentation de la table (à gauche) et du téléphone (à droite) quand le joueur doit choisir sa nouvelle carte de jeu

Afin de ne pas perdre l'utilisateur, nous avons mis des notifications de type popup sur la table, permettant de dire aux joueurs qu'ils doivent retourner choisir une carte. Nous avons aussi mis une notification de ce même type au moment où un joueur doit choisir le tas central.

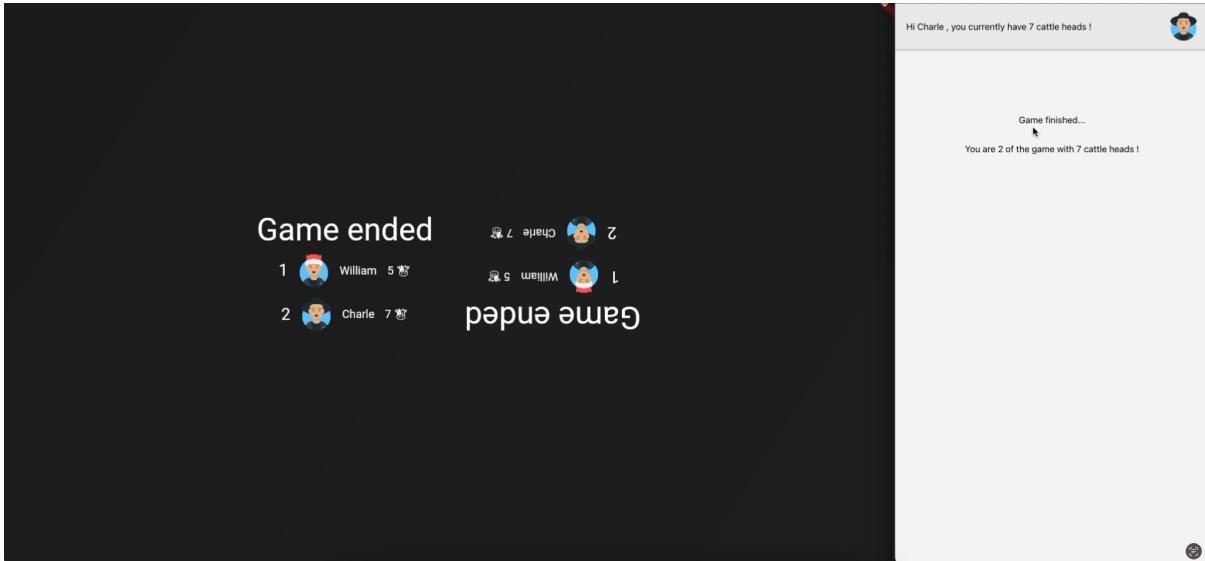


Figure 8 - Représentation de la table (à gauche) et du téléphone (à droite) à l'affichage des résultats

À la fin du jeu, nous avons affiché les résultats de jeu, à la fois sur la table (visible dans les deux sens) et à la fois sur le téléphone. Nous avons pris en compte la territorialité, dans le sens où sur la table, nous affichons un résultat global (le score et le classement de tous les joueurs), et nous gardons le téléphone en tant qu'espace privée avec que son résultat à lui, afin de ne pas le perdre.

Dans cette implémentation, nous avons fait énormément de test "à la main", dans le sens où nous jouions ensemble, et nous regardions ce qui n'allait pas, ce qui n'était pas clair, et nous l'implémentions ensuite. Par exemple, les notifications popups n'étaient pas dans la version de base, mais nous nous sommes rendus compte que cette information manquait.

Un autre exemple est la défausse de chaque joueur, ce fut une implémentation relativement compliquée. En effet, dans le jeu officiel, aucun joueur ne voit le nombre de taureaux qu'il a dans sa défausse au moment du tour de jeu, seulement à la fin. Cependant, chaque autre joueur peut voir l'épaisseur du tas de défausse de chaque joueur, leur permettant de se situer dans le jeu. Nous avons donc décidé d'optimiser cela. Par exemple, vous pouvez voir qu'en dessous de la carte de "Charle", il y a un petit rectangle blanc, ce rectangle signifie "Charle à 1 carte dans sa défausse", et il y a autant de rectangles que de carte dans la défausse. Toutefois, ce nombre de rectangles ne correspond pas au nombre de taureaux, cela donne juste une indication aux autres joueurs sur "l'épaisseur de la défausse". Nous avons ensuite décidé de donner le nombre de taureaux à chaque joueur sur son téléphone, même si cela n'est pas dans le jeu de base, nous pensions que cette implémentation avait du sens.

Nous rappelons que cette implémentation est faite sans lire les règles. À ce stade, nous ne pouvions pas encore conclure sur la véracité des règles, néanmoins nous avions observé que notre implémentation se faisait énormément de manière itérative (plus de 6). C'est-à-dire que nous ne réfléchissions pas explicitement aux interactions. Nous créions une fonctionnalité, et ensuite, nous regardions si le jeu était fluide et interactif, et nous regardions ensuite quels étaient les points de friction à la compréhension du jeu, pour ensuite réitérer. Cela prenait donc beaucoup de temps, et beaucoup de réunions, car nous regardions chaque fois ensemble ce qu'il n'allait pas, nous n'avions pas vraiment de structure de conception.

3.2.2.2. Description des modèles de données échangés et prise de recul

Pour réaliser cet avancé de projet, nous avons repensé l'architecture globale de la partie backend. L'architecture précédente n'était pas optimisée et pas homogène puisque nous avions un mélange de deux protocoles (HTTP et socket), et que nous commençons à avoir du mal à juger pourquoi un protocole est mieux qu'un autre dans certaines situations. Par exemple, si nous voulions rajouter une interaction entre le téléphone et le backend, ce n'était ni évident ni simple.

Pour choisir quel protocole nous allions utiliser, nous en sommes revenus aux interactions. Quand un joueur fait un choix sur son téléphone, si toutes les communications se font par requête HTTP, et que l'application demande au serveur des mises à jour toutes les secondes, l'utilisateur, dans le pire scénario, devrait attendre 2 secondes (1 seconde du téléphone vers la table, et une seconde de la table vers le téléphone) avant que sa décision sur son territoire privée soit affiché sur le territoire public. Ceci peut rapidement entraîner une confusion pour l'utilisateur, car ce délai n'est pas négligeable, et il pourrait se poser la question de savoir si son périphérique n'a pas un bug. À l'inverse, les communications par socket sont bien plus rapides, et nous n'avons pas à contacter de nous-même le serveur. Pour ces raisons-là, nous avons fait le choix de passer à tout WebSocket. De plus, ce choix ne nous paraît plutôt légitime par rapport à la réalité, car de nombreux gros jeux vidéo actuels fonctionnent entièrement via socket.

Dans notre première implémentation, nous avions énormément de responsabilités sur le serveur distant. En effet, il devait faire les calculs internes au jeu, et en même temps filtrer ce qu'il envoyait aux périphériques (puisque il n'envoyait que des informations très particulières). Cependant, cette approche étant très performante pour des projets de grande échelle pour des raisons de sécurité, dans notre projet, elle l'était moins. Notre but étant de créer des interactions les plus agréables possibles pour l'utilisateur, dès que nous voulions faire une modification, il fallait changer énormément de choses dans le serveur, c'était donc une perte de temps. Nous avons ainsi fait le choix d'envoyer, pour chaque requête du serveur vers la table, un payload qui contient toutes les informations de jeu, et dans la requête du serveur vers le téléphone, toutes les informations relatives au joueur spécifique. Ainsi, nos frontends respectifs peuvent faire le choix d'afficher ce dont ils ont besoin pour rendre les interactions fluides. Ce n'est que par la suite, quand le jeu sera entièrement fini, que nous filtrerons les réelles informations nécessaires à chaque requête afin d'envoyer uniquement ce qui est nécessaire.

Voici un diagramme de séquence qui montre les différentes interactions :

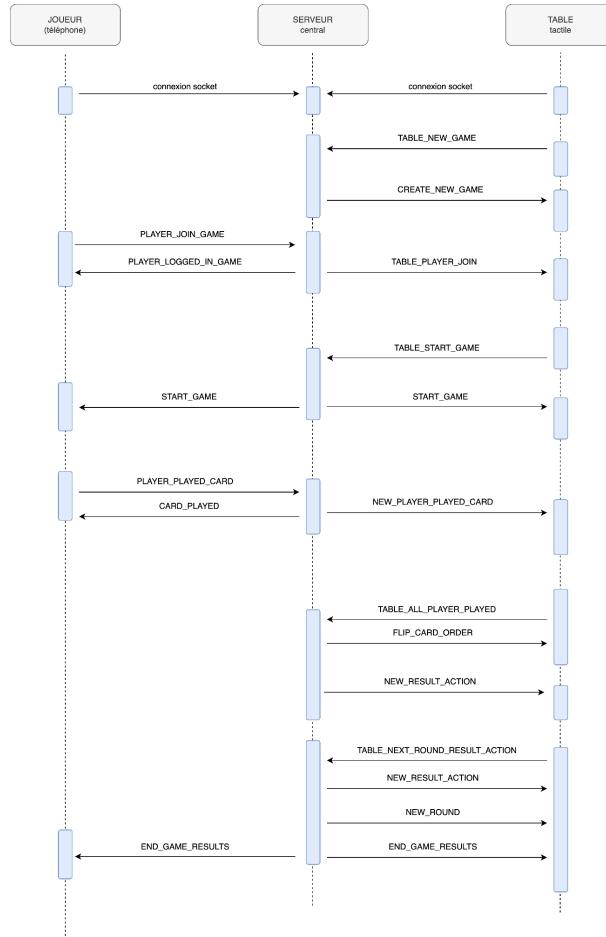


Figure 9 - Diagramme de séquence de l'architecture backend de la milestone 2

Ces choix ont entraîné certaines modifications au niveau de l'aspect "action" (les différentes règles durant un entre 2 tours). Nous avons donc transformé nos "ActionsTypeEnum" de la version précédente en 4 différents types d'actions à effectuer, qui sont envoyés via "NEW_RESULT_ACTION" (toujours à l'intérieur d'un même modèle nommé "game") qui permettent à la table de savoir quelle animation lancer :

- SEND_CARD_TO_STACK_CARD : créer une animation provenant de la carte du joueur X au tas de cartes central Y
- SEND_CARD_TO_STACK_CARD_AND_ADD_CARD_TO_PLAYER_DISCARD : créer une animation provenant de la carte du joueur X au tas de cartes central Y, mais ce tas Y se complétant intégralement, faire une animation qui ramène les cartes de ce tas-ci vers la défausse du joueur X
- CHOOSE_STACK_CARD : le player X doit choisir un tas du centre de la table et récupérer les cartes de ce tas dans sa défausse
- NEXT_ROUND : l'entre-deux-tours est fini, les joueurs peuvent revenir sur leur téléphone pour choisir une nouvelle carte.

Dans la mesure où nous voulions nous concentrer principalement sur les interactions, chaque action faite par, soit la table, soit un joueur sur son téléphone est envoyée au serveur. Ensuite, il redistribue cette information sur le périphérique adjacent (si information provenant du téléphone, on redistribue potentiellement à la table, et inversement). C'est pour cela que nous pouvons voir apparaître des routes comme :

- NEW_PLAYER_PLAYED_CARD indiquant à la table qu'un joueur a joué sa carte sur son téléphone

- TABLE_ALL_PLAYER_PLAYED_CARD qui est le déclencheur, provenant de la table, disant que tous les joueurs sont prêts à recevoir les résultats d'entre-deux-tours.
- FLIP_CARD_ORDER donnant l'ordre des joueurs dans lequel les actions de l'entre-deux tour vont se dérouler

Tous ces changements ont eu un impact sur l'architecture que nous avions développé initialement, mais elle nous a fait gagner énormément de temps sur la suite, car il a été beaucoup plus simple pour nous de se concentrer sur les interactions réelles, et sur l'optimisation de l'expérience utilisateur.

3.3. En connaissant les règles

3.3.1. Prise de recul sur les règles

3.3.1.1. Phase d'identification

- a. **IR1. Annotating all tasks with the nature of the data handled**
- b. **IR2. Annotating data input tasks (like typing text)**
- c. **IR3. Annotating major tasks, i.e., belonging to the minimal set of mandatory tasks to play**
- d. **IR4. Assigning to all devices the list of possible type of territory**

De façon corrélée, nous avons déjà fait cette étape lors de la première version. En effet, étant donné que nous sommes partis de la version originale sur laquelle le concepteur du jeu a déjà réfléchi à ce type de problèmes et a fait des milliers de tests utilisateur (commercialisation du jeu de cartes). Par exemple, le jeu utilise la table comme zone commune aux joueurs : c'est un territoire public. Le jeu de cartes en éventail devant la personne est un territoire privé : seul le joueur qui détient les cartes peut le regarder et interagir avec celles-ci. Durant la phase préliminaire de découverte du jeu, nous avons pu remarquer comment les joueurs interagissaient entre eux et avec le jeu, ce qui correspondrait donc à la phase d'identification.

3.3.1.2. Phase de distribution

- a. **DR1. For each task assignment to a device, determining whether it is within an existing suitable territory or whether to create a new one**
- b. **DR2. Assigning data input tasks to a private territory**
- c. **DR3. Assigning tasks handling private data to the preferred device for private territories**
- d. **DR4. Assigning tasks handling public data to the preferred device for public territories**
- e. **DR5. Assigning unassigned tasks to devices according to the assignment of the previous or the next task**

Après avoir observé et compris comment les joueurs jouaient au jeu, nous sommes passés à la phase plus théorique du projet.

Intuitivement, au début du projet, nous avons créé un diagramme de séquence. Nous avons pris les règles officielles du jeu, et nous avons essayé de comprendre quelles interactions nous avons, que ce soit entre les joueurs et leur tas de cartes, mais aussi les interactions entre joueurs.

Ce que nous avons fait ensuite, c'est de créer un diagramme de séquence avec 3 lignes, représentant le téléphone, la table et le serveur principal. Nous avons associé les types possibles de territoire intuitivement, en prenant l'exemple du jeu. C'est-à-dire un joueur qui a son espace privé (son tas de cartes personnel représenté par son

téléphone) et l'espace commun (la table avec les stacks représenté par la table tactile). Nous avons donc plus ou moins appliqué les règles relatives aux degrés de confidentialité, sur les données et les interactions. Cette étape se ramène aux règles DR2, DR3, DR4, DR5.

Cependant, pour la règle DR1, nous sommes partis du principe qu'il n'existant que deux territoires, l'un public (la table) et l'autre privé (téléphone), car nous n'avions pas pensé au fait qu'il pouvait y avoir d'autres territoires. Nous avons, en conséquence, réparti l'ensemble des tâches identifiées entre ces deux territoires en une fois, contrairement à ce que proposaient les règles.

Comme nous le verrons plus tard, nous avons essayé de changer l'équilibre des tâches sur l'un et l'autre des dispositifs, autrement dit de faire une version uniquement sur un téléphone et une version uniquement sur la table. La version de la table était assez simple à implémenter : nous sommes simplement partis du principe que si l'on devait faire cette interface, on ne ferait que de reporter les éléments déjà implementés sur le téléphone, il n'a de ce fait suffit que de poser le téléphone sur la table comme si celui-ci en faisait partie. Pour le téléphone en revanche, il a fallu judicieusement placer les informations, car la quantité de place est très limitée, et nous avons ainsi été forcés de créer plusieurs vues : l'une pour représenter la table, l'autre pour représenter le téléphone.

3.3.1.3. Phase d'équilibrage

- a. **BR1. If assigning certain tasks to the central device changes the gameplay, then those tasks can be moved to the personal devices**
- b. **BR2. If the public interactive device is overloaded, major tasks (see IR3) should be given priority**

Durant la phase de développement du projet, nous n'avons pas été confrontés à un problème de surcharge d'un territoire plutôt qu'un autre. En effet, dans notre implémentation actuelle, nous permettons au joueur de jouer de manière similaire aux règles de jeu officielles, nous n'avons pas rajouté d'éléments supplémentaires tels que des statistiques en temps réel, des aides à la décision, ou autre.

Cependant, après réflexion, nous nous sommes rendu compte que cette règle BR2 aurait un réel intérêt si nous étendions le jeu, ou si nous venions à créer un nouveau jeu. Si nous créons un nouveau jeu, il faudra que nous définissions chaque tâche sur chaque joueur, afin de ne pas les surcharger d'informations inutiles. Le fait que nous soyons passés au numérique implique que les cartes soient dématérialisées aussi, nous n'avons donc jamais pensé à afficher des informations inutiles telles que la valeur des cartes de la défaisse ou encore leur nombre de taureaux individuels. Si nous avions suivi le jeu original à la lettre, nous aurions dû les supprimer grâce à la règle BR2 car ces informations auraient surchargé l'espace personnel qu'est le téléphone.

De plus, nous nous sommes posé la question de la répartition qui viendrait modifier la forme du jeu de base. Dans notre situation, nous sommes partis du postulat que tous les joueurs se retrouvaient autour d'une table, avec un téléphone chacun. Toutefois, si nous étions 8 et nous n'avions que 4 téléphones, dans ce cas-là, chaque espace privé serait utilisé par 2 joueurs, et dans cette situation, il faudrait se poser la question de bouger certaines interactions d'un territoire à un autre. Nous reviendrons sur ce problème plus tard.

Avec du recul, nous nous rendons compte que nous n'avons pas appliqué intuitivement la règle BR1. En effet, lorsque l'utilisateur doit choisir un tas pour poser sa carte (situation 3), nous avons juste permis cette sélection sur la table, et non sur le téléphone. Nous sommes partis du postulat que c'était une action qui devait être réalisée sur l'espace public, mais nous n'avons pas eu l'intuition d'aussi le permettre sur l'espace privé.

Lorsque nous avons étendu le jeu, nous nous sommes inspirés de cette règle, elle nous a permis d'avoir l'idée d'implémenter une version avec le choix du tas sur le téléphone, et certains tests utilisateurs nous ont révélé que cette règle avait un réel intérêt.

3.3.1.4. Phase de transition

- a. **TR1. If there is a device change between two consecutive tasks, then the second task may be replicated on the first device**
- b. **TR2. If there is a device change between two consecutive tasks, then a reinforcement feedback may be added**
- c. **TR3. If there is a change of devices for two (frequent) consecutive tasks, then it is possible to move one of the two tasks from one device to another**

Cette phase de transition est sûrement celle qu'il nous a été le plus dur d'appliquer intuitivement. Avec le recul, nous nous rendons compte que nous les avons appliquées dans certains cas, mais pas dans d'autres. De plus, si nous avions ces règles en amont, nous aurions sûrement gagné du temps.

Nous avons appliqué la règle TR1 et TR2 de manière relativement intuitive. Un exemple est, lors d'un tour de jeu, quand un joueur joue sa carte. Dès qu'il l'a joué, il voit sur son écran de téléphone un message disant que sa carte est jouée, et, au même moment, sur la table, au niveau de son emplacement, il y a une icône "check" qui montre que ce joueur a bien joué. Cela permet à tous les utilisateurs de savoir qui a déjà joué et qui n'a pas encore joué. Nous avons également appliqué cette règle au moment du login, dès qu'un utilisateur joint le jeu, il est affiché sur la table, tout comme le résultat à la fin, l'information est disposée sur l'espace public et sur l'espace privé.

Il est à noter que ces deux règles, dans la situation précédemment décrite, n'ont pas été pensées au début de l'implémentation, ce n'est seulement qu'après quelques tests

sur la table que nous nous sommes rendus compte qu'il fallait avoir une confirmation sur les 2 périphériques au même moment, car si ce n'est pas le cas, on se demande s'il n'y a pas eu un bug, si notre action a été prise en compte ou non. Si nous avions connu cette règle dès le début, nous y aurions directement pensé, et nous n'aurions pas eu besoin de faire des tests utilisateurs pour faire ce changement.

3.3.2. Amélioration de notre application grâce aux règles

3.3.2.1. Amélioration de l'espace privée

L'amélioration de l'espace privé s'est faite en plusieurs phases. En effet, nous avions plusieurs d'idées d'améliorations, mais pas vraiment de préférence, nous avons donc testé toutes les possibilités. La première étant celle de base, où toutes les cartes sont les unes à côté des autres. La deuxième version est de mettre toutes les cartes sur le même écran, afin de permettre à l'utilisateur d'avoir une vision claire et globale de toutes ces cartes. La troisième version est de mettre les cartes en éventail.

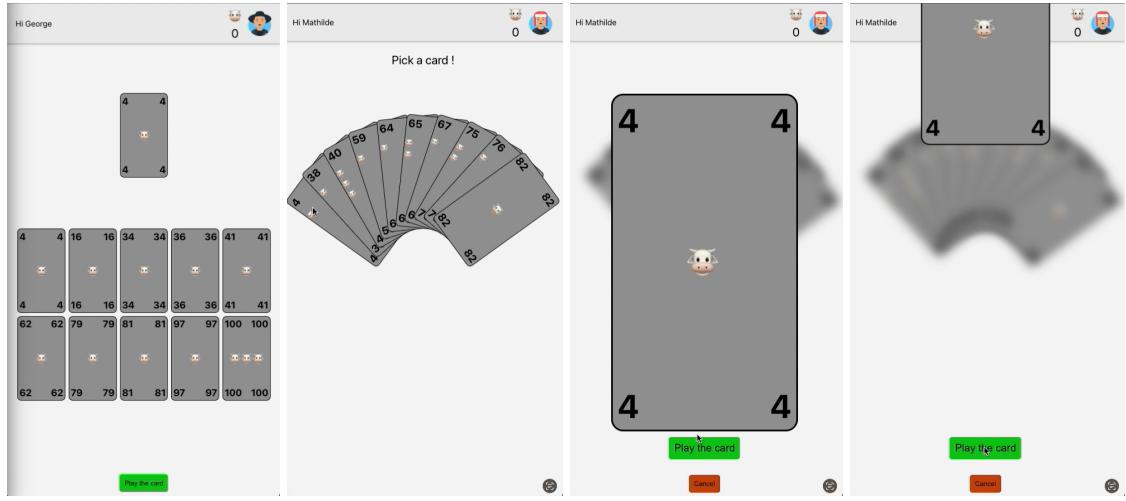


Figure 10 - Représentation des 2 variations de l'espace privée (2 représentations de gauche) et animation d'envoi de la carte à la table (2 représentations de droite)

Ces 3 différentes propositions auraient pu faire appel à un AB-Testing. Cependant, nous nous sommes rapprochés des règles et la version la plus évidente était la dernière, avec les cartes en éventail. En effet, dans cette version, le joueur appuie sur une carte de l'éventail, cette carte s'affiche en grand, et s'il clique sur "Play de card" la carte est envoyé vers la table, avec une animation de défilement vers le haut, comme si la carte se jetait sur la table. Cette animation est bien en accord avec la règle TR1 et TR2. Cela rend l'interaction intuitive, et l'utilisateur comprend bien que sa carte est envoyée, c'est bien plus interactif qu'un popup. En ajoutant cela au fait qu'il y a une confirmation que la carte a été reçue par la table, nous avons bien une fluidité de l'information.

3.3.2.2. Amélioration des popups au moment du choix du stack central

Au moment du choix du stack central, nous nous sommes rendus compte, grâce aux règles, que nous sommes tombés dans le paradigme de "je sais jouer au jeu, j'ai compris les règles, et je suis le jeu", alors qu'en réalité, il peut arriver que des utilisateurs soient débutants, et ne connaissent pas encore en détail les règles, et soient donc perdus. De plus, il peut arriver qu'un joueur pense à autre chose à un moment donné, et donc qu'il peut être perdu dans le jeu. Ces détails, nous ne nous en rendions plus compte, car nous testions le jeu à longueur de journée. Nous avons donc, notamment grâce aux règles de transition, améliorer les popups comme ceci :

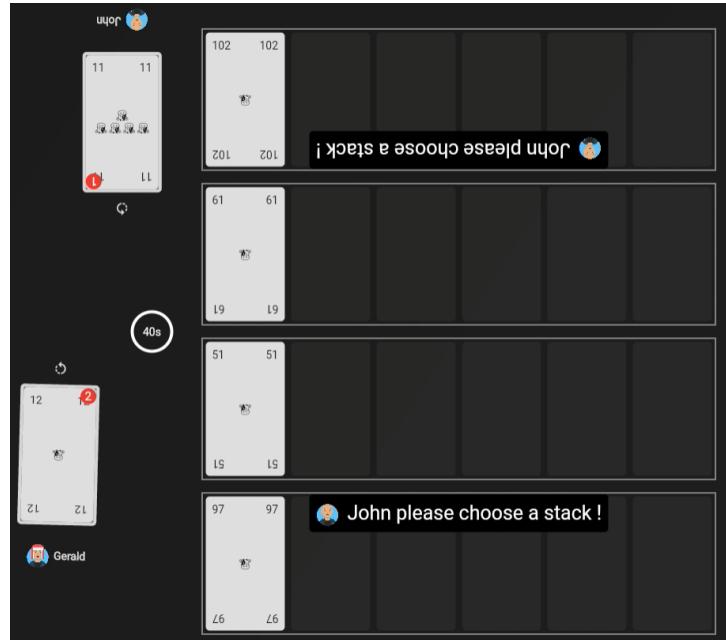


Figure 11 - Représentation de la table avec le chronomètre, et au moment du choix, par un joueur, du tas central

De plus, nous avons répliqué cette demande au niveau du téléphone, dans la version avec la table sur le téléphone. Ce qui fait que cette action, qui est importante pour la suite du jeu, est bien identifiée par le bon joueur. Nous reviendrons sur cette version plus tard.

3.3.2.3. Amélioration de la dynamique du jeu

Dans cette section, nous allons aborder un élément que nous avons ajouté permettant de rendre le jeu bien plus dynamique. Nous avons ajouté un chronomètre pour forcer le joueur à jouer rapidement, et donc avoir un jeu fluide, qui ne devient pas ennuyant.

Pour l'afficher, nous sommes donc partis du principe que cette information devait être dupliqué entre les deux dispositifs, car cette information est utile autant aux joueurs sur leurs téléphones qui sont par exemple en train de choisir une carte que les personnes qui ont par exemple fini de jouer et qui regardent la table.

Pour faire cette implémentation, les règles d'équilibrage et de transition ont été utiles, dans la mesure où quand le chrono est dépassé, cela change le jeu, donc cela doit être affiché sur l'espace public, mais aussi sur l'espace privé. Par exemple, sur le téléphone, une alerte arrive lorsque le temps arrive bientôt à sa fin, et sur la table, nous avons également une alerte qui dit aux joueurs de jouer, et la couleur du fond d'écran qui devient rouge. Dans la mesure où c'est une double information, qui est importante, elle se doit d'être située sur les 2 périphériques.

3.3.3. Modification des dispositifs

Bien que les règles aident à développer un projet, de manière plus efficiente que si nous le faisons de manière itérative, elles permettent également de remettre en question l'implémentation d'un projet. Il y a, en effet, plusieurs manières d'implémenter un produit, et le but est de trouver celle qui rend l'expérience utilisateur la plus agréable possible.

Dans cette partie, nous avons essayé de remettre en question le projet tel que nous le connaissons actuellement en lui apportant des modifications importantes. Cette partie a aussi pour vocation à mettre les règles de conceptions à l'épreuve.

Nous avons alors réfléchi à 2 variantes de jeu supplémentaires, qui permettent d'utiliser le jeu dans des environnements différents :

- Supprimer le téléphone avec un espace privé qui serait sur la table
- Supprimer la table avec un espace public qui serait disponible sur le téléphone

Ces variantes, alliées à un test utilisateur, nous ont permis de nous interroger sur la manière de faire varier l'environnement.

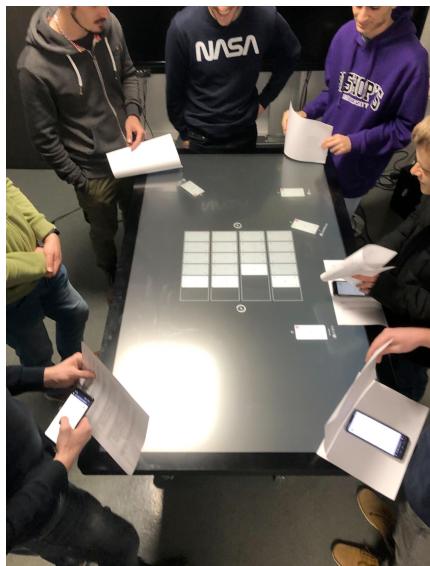
3.3.3.1. Suppression des téléphones

Tout comme dans le projet "Futura" (voir état de l'art), nous avons voulu essayer de déplacer l'espace privé présent sur le téléphone, mais sur la table. C'est-à-dire afficher les cartes des utilisateurs sur la table.

La problématique de cette variante est que, tout comme un jeu de poker, les cartes des utilisateurs doivent rester secrètes vis-à-vis des autres joueurs, si un joueur voit les cartes de son camarade, le jeu est compromis.

Au moment d'implémenter ces variantes, nous étions sur la fin du projet, nous avons donc fait une hypothèse forte.

Nous voulions tester cette variante, mais nous n'avions pas le temps de réimplémenter l'application présente sur la table. Nous avons donc demandé aux utilisateurs, durant un test utilisateur, de poser leur téléphone sur la table (visible de tous). Le but était de simuler l'espace privé présent sur la table tactile.



Pour garder l'anonymat des choix de jeu des joueurs (quelles cartes ils ont), nous leur avons demandé de cacher le téléphone avec une feuille de papier.

Les retours ont été très mitigés. Cette implémentation limite le nombre de joueurs autour de la table (cette disposition prend énormément de place), et chaque joueur avait chaque fois peur que son espace privé soit visible par un autre joueur. Certains ont même, au bout d'un moment, repris leur téléphone en main, car cela était trop dur de jouer comme cela.

Figure 13 - Photographie, durant le test utilisateur, de la variante 1 (territoire privé situé sur la table tactile) en faisant la supposition que le téléphone représente l'espace privé qui devrait être affiché sur la table.

De cette expérience, nous en arrivons à la conclusion que, s'il y a un espace privé qui contient des données sensibles, il faut vraiment mettre ces éléments sur un périphérique externe individuel. Les règles de distribution (notamment DR3 et DR4) doivent être appliquées, et surtout, prendre le temps de bien y réfléchir dès le début, afin de ne pas perdre du temps à implémenter un jeu qui ne sera pas pratique à jouer.

3.3.3.2. Suppression de la table

Tout comme nous nous sommes demandés si nous pouvions jouer sans les téléphones, nous voulions savoir s'il était possible de jouer au jeu sans la table tactile. Cette implémentation a pour avantage de pouvoir permettre à des gens qui ne se situent pas dans la pièce de jouer. Supposons par exemple que nous voulions jouer en famille, mais que nos parents sont à l'autre bout de la France, peut-on quand même faire une partie. C'est ce que nous avons essayé de faire.

Pour cela, nous avons décidé de modifier la page d'attente sur le téléphone par une représentation de la table.

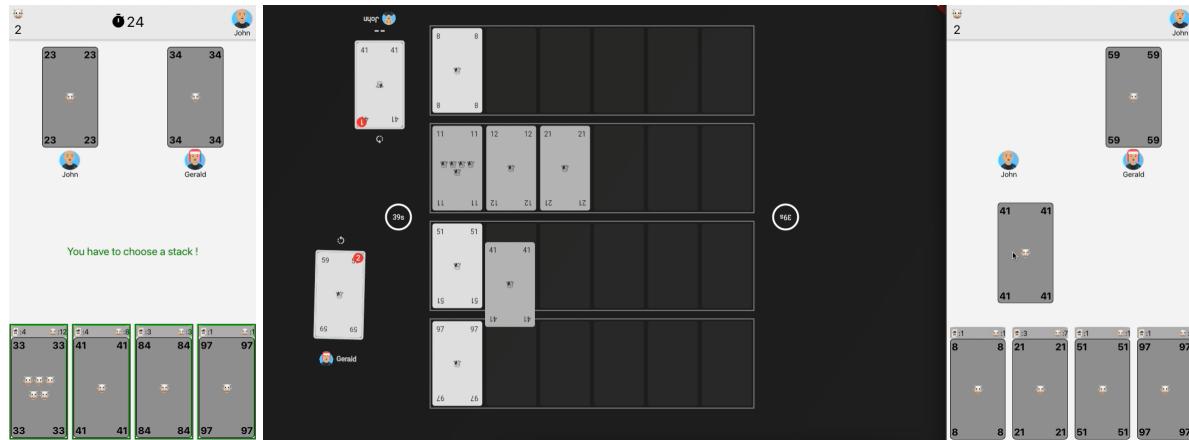


Figure 14 - Représentation des téléphones et de la table, au moment où un joueur doit faire son choix de tas central (version de la personne à distance)

Globalement, nous allons y retrouver les mêmes informations que celles de la table. Nous pouvons voir les cartes jouées par les différents joueurs, le mouvement de ces cartes vers le stack, et, lorsqu'il y a un choix à faire, une notification sur le téléphone demandant de choisir un tas. Cette version nous a permis d'appliquer en détail les règles d'équilibrage et de transition. Par exemple, la règle d'équilibrage BR1 à vraiment du sens ici, car la décision change le roleplay, il est donc intéressant que cette action soit présente sur le téléphone également.

Lors de notre test utilisateur, cette version fut relativement bien acceptée, mais elle n'est pas optimale selon les joueurs, puisqu'elle rend le jeu moins convivial, il est plus dur de jouer à distance, et cela diminue les interactions humaines.

Au moment du test utilisateur, nous n'avions pas eu le temps d'implémenter l'affichage des 4 stacks sur le téléphone quand le joueur doit choisir une carte. Il devait donc retenir par cœur les 4 stacks, ce qui ne rendait pas l'expérience utilisateur agréable. Nous avions anticipé ce problème, notamment grâce aux règles de distributions, qui disent qu'il faut afficher sur chaque périphérique ce qui est nécessaire à prendre les bonnes décisions. Nous avons donc implémenté l'affichage des stacks sur le téléphone afin d'avoir un jeu fluide. Dans cette situation, les règles nous ont permis d'anticiper un problème que nous n'aurions pas pensé avant, c'est notamment sur cette implémentation que nous nous sommes rendus compte de l'efficacité des règles.

3.3.3.3. Modification de l'environnement

Pour cette partie, nous avons essayé de prendre du recul sur l'expérience réelle de jeu, nous avons donc pensé à faire jouer les utilisateurs assis ce qui favorise le confort au cours de la partie. Cependant, le nombre d'utilisateurs maximal que l'on peut asseoir autour de la table est d'à peu près 8 (3 par côté et 1 par bout). Nous avons alors

essayé d'optimiser l'espace, afin de faire jouer un maximum d'utilisateurs sur la table. Si la partie se fait debout, nous pouvons facilement faire jouer les 10 joueurs (le jeu est conçu pour permettre un maximum de 10 joueurs) puisqu'ils peuvent participer au jeu à une certaine distance de la table et se rapprocher lorsque cela est nécessaire.

Nous avons remarqué que notre version avec la table sur le téléphone était très adaptée à une utilisation avec beaucoup de personnes. En effet, de cette manière, la table tactile devient juste un élément de représentation. On peut toujours y faire ses choix, mais nous pouvons aussi juste l'utiliser pour voir l'état du jeu actuel, et dans cette situation, jouer à 10 n'est pas très compliqué, car tout le monde peut rester à une légère distance de la table, voir même rester assis et ne pas être perdu dans le jeu.

Nous avons conclu sur le fait que cette implémentation était la plus pratique de toutes, d'un point de vue expérience utilisateur, car elle permet à des joueurs assis de pouvoir jouer facilement, et de toujours être intégré au jeu. De plus, avec cette version, nous pouvons jouer sur une TV (tactile ou non), c'est ce qui s'est passé au moment de nos journées de travail.



Figure 15 - Photographie d'un test (durant la conception collaborative) qui nous a permis de tester la version sur écran vertical.

Sur la figure 15, nous étions autour d'une TV tactile, et nous pouvions jouer au jeu sans se lever, et après tous nos tests utilisateurs*, nous nous sommes rendus compte que cette version hybride était une des meilleures, car cela permettait d'avoir une expérience utilisateur agréable.

* Les tests utilisateurs avaient pour objectif de voir comment le jeu se comportait s'il n'était pas à plat, comme sur ce tableau. Ici, nous étions dans des conditions optimales pour faire ce genre de test puisque nous étions dans une salle de classe dans laquelle des dizaines de chaises étaient face à ce tableau. Ici, nous pouvons facilement imaginer avoir une vingtaine d'utilisateurs (si le jeu le permettait).

En effet, celle-ci semblait être aussi conviviale que le jeu original tout en ne nécessitant pas de matériel particulier. De plus, chaque utilisateur peut se placer où il le désire du moment qu'il peut voir l'écran.

4. Répartition du travail

Durant tout le projet, nous avons globalement travaillé de manière verticale, nous avons tous travaillé sur chaque élément du jeu, mais chacun était chef de projet d'une partie spécifique :

- Yann Clodong de l'interface sur la table tactile,
- Guillaume Piccina de l'interface distante sur téléphone
- D'Andréa William de l'implémentation du moteur de jeu et de l'architecture du serveur distant.

Ceci nous a permis à tous de pouvoir avancer efficacement, nous étions tous spécialisés sur un domaine, cependant, nous travaillons souvent sur plusieurs éléments différents. Par exemple, en milieu de projet, il y avait énormément de travail sur la table tactile, nous avons donc tous travaillé dessus et pareil pour le téléphone quand nous avons implémenté les variantes.

5. Conclusion

Dans ce TER, nous avons cherché à valider des règles de conception d'interfaces distribuées. Ces règles étant extraites de la littérature scientifique et validées par des projets d'études. Notre objectif ici était de savoir si ces règles étaient intuitives ou non, et si nous les appliquons sans les connaître, ou non. C'est pour cela que toute la première partie du projet, nous n'en avions pas connaissance. Nous avons en conséquence avancé comme nous l'aurions fait dans un autre projet. Une fois que nous sommes arrivés à la fin de cette implémentation, nous avons lu les règles pour voir ce que nous avions manqué.

Nous en concluons sur le fait que, sans lire les règles, nous avons quand même réussi à aller loin dans le jeu. En effet, après lecture des règles, il n'y a pas eu de changement (en-dehors des variantes), vraiment complexe, il y avait des optimisations, mais pas de changement drastique. Cependant, il est à noter que, dans la version itérative, nous avons fait énormément de tests (utilisateurs ou non). Nous n'avions pas réellement de structure. Donc même si à la fin, nous avions un jeu fonctionnel, nous avons perdu beaucoup de temps à tester ou à itérer. Là où, et nous l'avons remarqué après lecture des règles, nous aurions gagné énormément de temps si nous avions connu les règles dès le début, car nous aurions directement mis les bonnes informations au bon endroit, sans se dire "je pense que cette information serait mieux sur ce territoire plutôt qu'un autre" (et le tester, se rendre compte que c'est vrai/faux, itérer...).

Pour être plus précis, les règles d'identification et de distribution ont été appliquées intuitivement. Dès le début, il nous a paru logique d'identifier chaque tâche (c'est globalement ce que l'on a fait avec les diagrammes de séquence), les assigner au périphérique adapté (territoire privé, territoire public). Globalement, après lecture des règles, nous n'avons pas eu à revenir dessus.

Cependant, les règles d'équilibrage et de transition n'ont été appliquées qu'après de nombreuses itérations. Même si dans certaines situations, il paraissait logique de ne pas surcharger les différents territoires, et de bien isoler ou dupliquer les informations importantes sur les territoires adaptés, dans certaines situations, nous sommes passés à côté. Par exemple, pour les popups d'informations disant à un utilisateur de choisir sur quel stack il veut poser sa carte, nous ne l'affichions que sur la table, alors que c'est une information importante, qui change le gameplay, se devrait d'être dupliqué.

De plus, ces règles permettent une remise en question structurelle du jeu. Par exemple, nous considérons la variation avec la représentation de la table sur le téléphone comme une remise en question structurelle. Ces règles, notamment celles de transitions, ont été utilisées pour arriver à cette version, nous ne l'aurions pas forcément appliquée intuitivement. Avec le recul, nous nous rendons compte que ces règles doivent être appliquées en accord avec la version itérative. Par exemple, commencer par lire les règles, définir les spectres, les frontières, etc, implémenter un début de projet, et ensuite reprendre les règles, les lire et essayer de prendre du recul sur le projet actuel. C'est globalement ce que nous avons fait et qui nous a permis d'arriver à la dernière variation.

Prendre connaissance en amont de ces règles, les appliquer avant de se jeter dans la création du jeu, permettra de faire gagner du temps, et beaucoup de tests utilisateurs, car cette méthode est prouvée (même si ce n'est pas le but de notre TER, nous avons pu prouver que les règles sont vraiment utiles). Cependant, il ne faut pas non plus vouloir juste appliquer les règles bêtement en se disant que les tests utilisateurs ne seront pas utiles, il faut tout de même en faire. En effet, connaître uniquement les règles n'est pas suffisant pour créer la meilleure expérience utilisateur possible. Notre exemple étant celui de mettre des joueurs à distance, cela est intéressant, car cela permet d'appliquer les règles, mais les joueurs n'aiment pas cette version parce qu'elle enlève de la convivialité.

Ce projet nous a permis de prendre du recul sur cela, néanmoins, une suite logique voudrait que l'on organise plusieurs équipes, dont la moitié travaillerait sur l'approche de prototypage "naïve" que nous avons mis-en-place, et

la seconde moitié qui développerait le même projet, mais seulement avec les règles. Ce projet ne serait entièrement nouveau afin d'éviter le biais de la "copie de la version matérielle". Bien sûr, ces deux équipes ne pourraient pas réaliser de test utilisateurs avant la finalisation du projet pour voir quelles erreurs ont été évitées par les développeurs eux-mêmes. Enfin, nous ferions tester les différents projets avec plusieurs groupes d'utilisateurs qui les testeraient dans des ordres différents.

6. Bibliographie

- Anne-Marie Déry-Pinna, Alain Giboin, Sophie Lepreux, Philippe Renevier Gonin, Territoires et IHM Distribuées : Raffinement de Règles et d'une Méthode de Conception de Jeux Multi-Dispositifs, [here](#)
- Jean-Charles Marty, Audrey Serna, Thibault Carron, Philippe Pernelle, David Wayntal, Multi-Device Territoriality to support Collaborative activities
- Alissa N. Antle, Joshua Tanenbaum, Allen Bevans, Katie Seaborn, Sijie Wang. Balancing Act: Enabling Public Engagement with Sustainability Issues through a Multi-touch Tabletop Collaborative Game. 13th International Conference on Human-Computer Interaction (INTERACT), Sep 2011, Lisbon, Portugal. pp.194-211, ff10.1007/978-3-642-23771-3_16ff. fffhal-01590835f, [here](#)
- Sangeun Oh, Ahyeon Kim, Sunjae Lee, Kilho Lee, Dae R. Jeong, Steven Y. Ko, and Insik Shin. 2019. FLUID: Flexible User Interface Distribution for Ubiquitous Multi-device Interaction. In The 25th Annual International Conference on Mobile Computing and Networking (MobiCom '19), Oct. 21–25, 2019, Los Cabos, Mexico. ACM, New York, NY, USA, 16 pages. <https://doi.org/10.1145/3300061.3345443>
- James R. Wallace, Stacey D. Scott, Carolyn G. MacGregor, Collaborative Sensemaking on a Digital Tabletop and Personal Tablets: Prioritization, Comparisons, and Tableaux, [here](#)

7. Annexe

Question	Remarques	Moyenne
Comment évaluez-vous la facilité d'utilisation de notre jeu ?	jeu assez intuitif	4
Les actions que vous deviez réaliser étaient-elles claires et compréhensibles ?	- pas très clair quand sacrifice demandé (le chrono empire) / la plupart des actins sont compréhensible (indiqu" visuellement)	4
Les interactions avec la table tactile et le téléphone étaient-elles intuitives et faciles à comprendre ?	- pas très clair quand sacrifice demandé (le chrono empire) / pour un débutant le choix du sacrifice pourrait être plus clair / Oui car on a 2 entité distinctes	4
Avez-vous été perdus à un moment dans le jeu ? Vous ne saviez plus trop que faire ou si votre action a bien été prise en compte (vous ne saviez pas quoi regarder entre le téléphone ou la table)	Du moment de choisir de tas a remplacer s'il n'y a plus d'emplacement disponible	4,25
Y avait-il des éléments ou des fonctionnalités qui vous ont posé des problèmes ?	jouer une carte sur le téléphone choisir un tas à l'entre-deux jeux sélectionner une carte sur le téléphone se connecter choisir sa carte au moment du tour	4,6 3,4 4,6 4,8 4,8
Y a-t-il des suggestions ou des améliorations que vous pourriez apporter à l'interface utilisateur ou à la jouabilité du jeu ?	jouer une carte sur le téléphone choisir un tas à l'entre-deux jeux Sélectionner une carte sur le telephone se connecter choisir sa carte au moment du tour récapitulatif de fin de tour	Plus de place entre les cartes - choisir un tas entre 2 tour : instructions plus clair serait bien / alertes plus claires, meilleur alerte ? quand on a beaucoup de cartes il faut être précis sur le sélectionner meme mot pour id sur tel et table 4,75 3,75 4 4,75 4,75 4,75
Le fait de jouer sur une table tactile a-t-il contribué à rendre l'expérience de jeu plus agréable pour vous ?	très agréable avec un grand écran	4,4
Le fait de jouer sur une table tactile a-t-il ajouté une dimension sociale ou collaborative à l'expérience de jeu ?		4,6
Avez-vous eu besoin d'instructions supplémentaires pour comprendre les actions que vous deviez effectuer dans le jeu ?	Que faire quand j'ai des cartes inférieures au tas (simplement la connaissance du jeu) / sélectionner le tas, et quand on doit le sélectionner	3,4
Y avait-il des moments où vous étiez incertain quant à ce que vous deviez faire ensuite dans le jeu ?	choix du tas, sacrifice / placer sa carte quand il faut choisir quel tas prendre / sélectionner le tas, et quand on doit le sélectionner	3,4
Comment avez-vous trouvé la présentation des actions que vous deviez réaliser sur l'interface utilisateur ? (est ce que la disposition des différents éléments étaient bien faite, compréhensible)		4,8
Les actions que vous deviez réaliser étaient-elles présentées de manière logique et séquentielle ?		4,8
Y avait-il des actions que vous deviez effectuer régulièrement qui vous semblaient inutilement compliquées ou difficiles à réaliser ?		4,6
Comment avez-vous trouvé la présentation des informations sur l'interface utilisateur ?		4,4
Y avait-il des fonctionnalités ou des éléments du jeu que vous auriez aimé voir ajoutés ou supprimés ?	Avoir le nombre de taureaux sur la table (mais pas raccord avec le vrai jeu)	5

Résultats tests utilisateurs