INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

**Projet informatique**
**GM3-2017/2018**

# Projet de COO : La Bataille Navale


**Étudiants :**

David Albert, Henri Durozay

**A l'attention de :**

M.Itmi

# Table des matières

# Partie 1

# Présentation générale

Dans cette partie, nous allons dans un premier temps présenter l'application yED, que nous avons utiliser à plusieurs reprises dans notre projet notamment pour le schéma BPM et le diagramme de classes.

## L'application yED

yEd est un logiciel permettant de réaliser des graphiques de toutes sortes en connectant des boîtes, objets, illustrations avec des flèches. Il est très pratique pour réaliser des schémas connectés ou des diagrammes rapidement. Ce logiciel est très riche en fonctions et en méthodes d'organisation : couches hiérarchiques interactives, couches orthogonales, couches organiques, schémas UML, circulaires, en arbre, etc... Dans le cadre de ce projet, il nous servira à réaliser le diagramme d'entrée sortie, le schéma BPM ainsi que le diagramme de classes. Il propose également de nombreux types de formats pour l'exportation des graphiques. Notamment les formats JPG, SVG et même PDF ou SWF.

## Présentation du jeu

### Principe du jeu

De manière très générale, la bataille navale, appelée aussi touché-coulé, est un jeu de société dans lequel deux joueurs doivent placer des « navires », de taille différentes, sur une grille tenue secrète et tenter de « toucher » les navires adverses en indiquant la position qu'ils souhaitent attaquer (exemple : case E3). L'attaque se fait à tour de rôle. Le joueur qui se fait attaquer, répond si un navire a été touché, si il a été coulé ou s'il n'y a pas de navire à cette position. Le gagnant est celui qui parvient à couler tous les navires de l'adversaire avant que tous les siens ne le soient. On dit qu'un navire est coulé si chacune de ses cases a été touchées par un coup de l'adversaire.

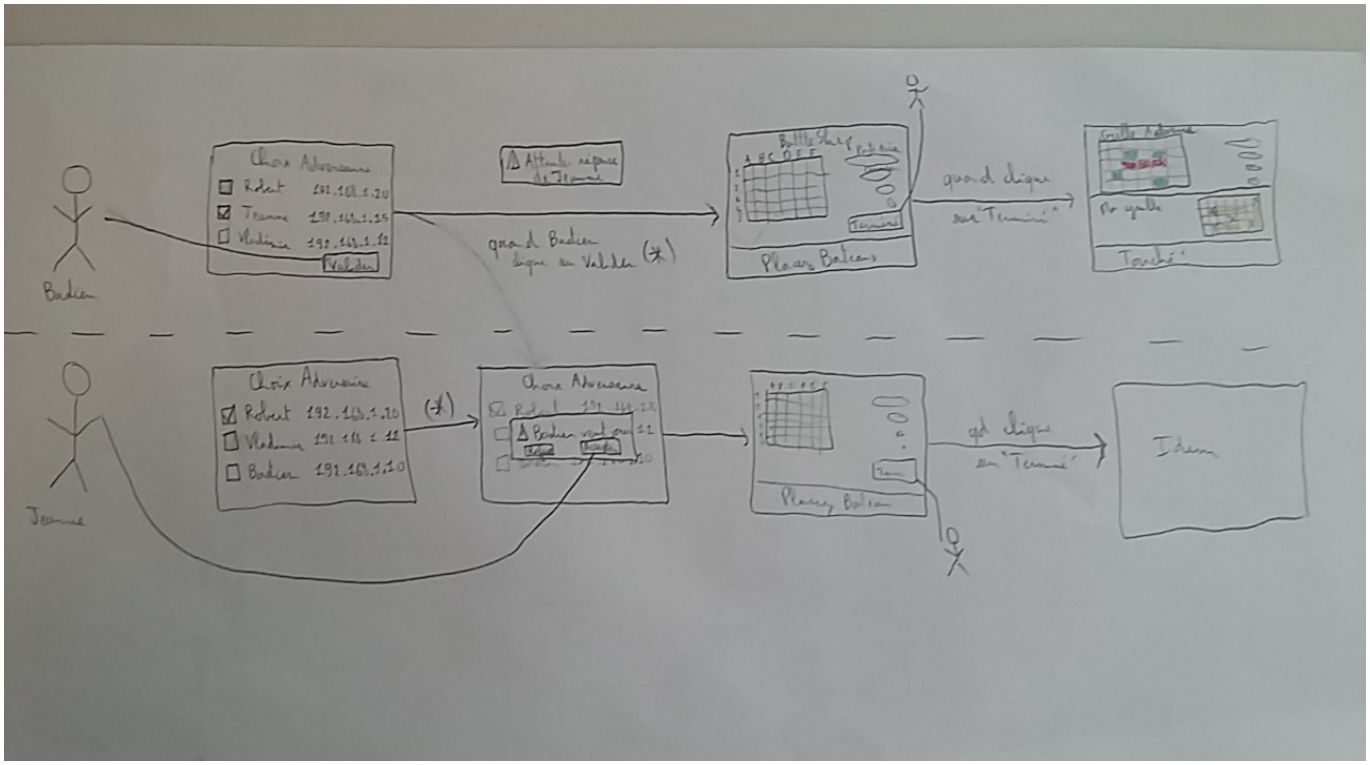
### Variante du jeu

Nous avons désiré adapté la version original du jeu, pour y intégrer la possibilité d'avoir des navires type qui ont des capacités propres à eux, comme un voilier qui est un tout petit bateau, un sous-marin qui est plus difficile à découvrir ou encore un cuirassier qui comporte une enveloppe plus solide. Concernant le développement informatique du projet, nous avons pensé à faire de ce jeu de bataille navale un jeu réseau (utilisation de sockets pour communiquer en réseau) avec une interface graphique Swing. De plus, dans un soucis de

maintenabilité, nous nous orienterons vers une conception de type modèle MVC (Modèle-Vue-Contrôleur), qui nous permettra de bien distinguer les données utilisateurs qui seront stockées et gérées sur le serveur Socket distant (un de nos ordinateurs probablement) de l'interface graphique de chaque utilisateur qui tournera sur l'ordinateur du joueur. Dans cette optique de jeu, chaque joueur pourra lancer le jeu bataille navale et y jouer contre un adversaire jouant sur un pc distant dans la limite où les 2 joueurs sont connectés aux même réseau que celui du serveur.

## Déroulement du jeu

Au démarrage du programme, le joueur arrivera sur une fenêtre dans laquelle figure la liste des joueurs en train de jouer. Il pourra sélectionner le joueur de son choix et valider. Alors, une demande sera envoyée au joueur. Cette demande sera représentée par une boîte de dialogue lui demandant s'il accepte ou non de jouer la partie. Si ce dernier accepte, les deux joueurs auront la même fenêtre sur leur ordinateur respectif et chacun devra placer à sa guise tous les bateaux sur sa grille de façon stratégique. Une fois tous les bateaux positionnés pour les deux joueurs, la partie peut commencer. Un à un, les joueurs cliquent sur une case non découverte de la grille adverse pour détruire les navires ennemis. Le clic souris (réceptionné par l'interface graphique qui implémentera l'interface MouseListener) sera envoyé au contrôleur qui vérifiera la validité de la manipulation avant d'envoyer un message aux données pour qu'elles se modifient et modifient ainsi la vue des deux joueurs. Dans le cas où le joueur tire sur un navire ennemi, le joueur recevra un message lui disant qu'un navire ennemi à été touché. Si le joueur ne touche pas de navire, le message "plouf" s'affichera. Si le navire est entièrement touché, le message « touché coulé » s'affichera. Une partie se termine lorsque l'un des joueurs n'a plus de navires.

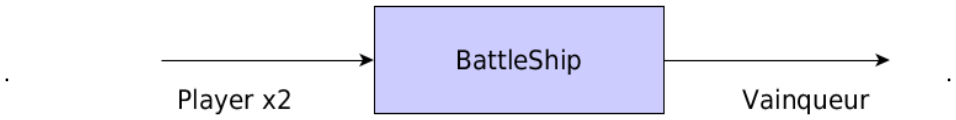


# Partie 2

# Conception du jeu

## Système entrée/sortie et acronyme

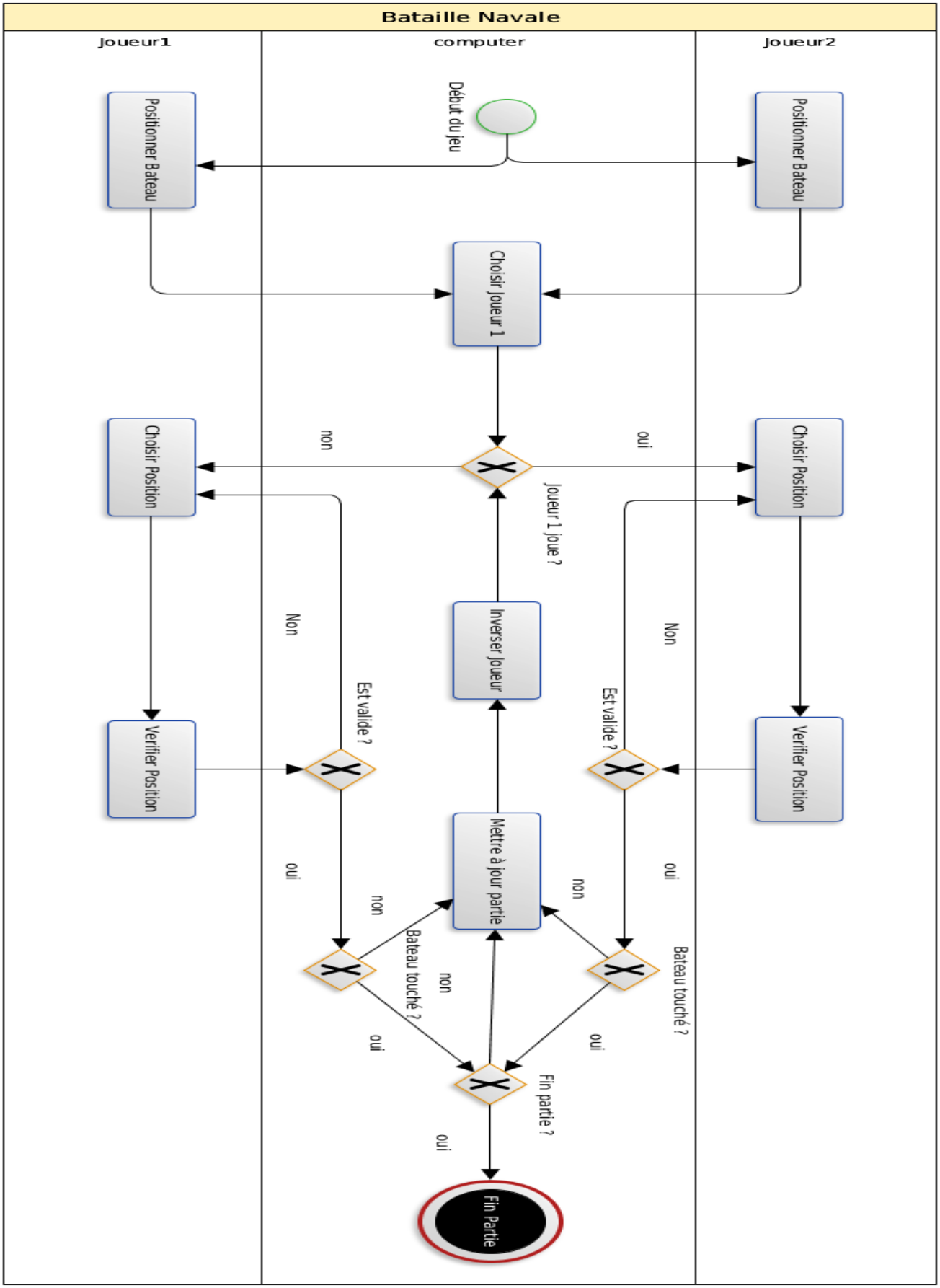
Afin de synthétiser de manière très générale le problème, nous avons choisi de représenter notre système par l'acronyme : « battleship ». Voici sa représentation en système composés d'entrées et sorties :




En effet, notre jeu a besoin en entrée de 2 joueurs pour pouvoir lancer la partie. Et en sortie de programme, on a un de ces deux joueur qui est le gagnant de la partie.

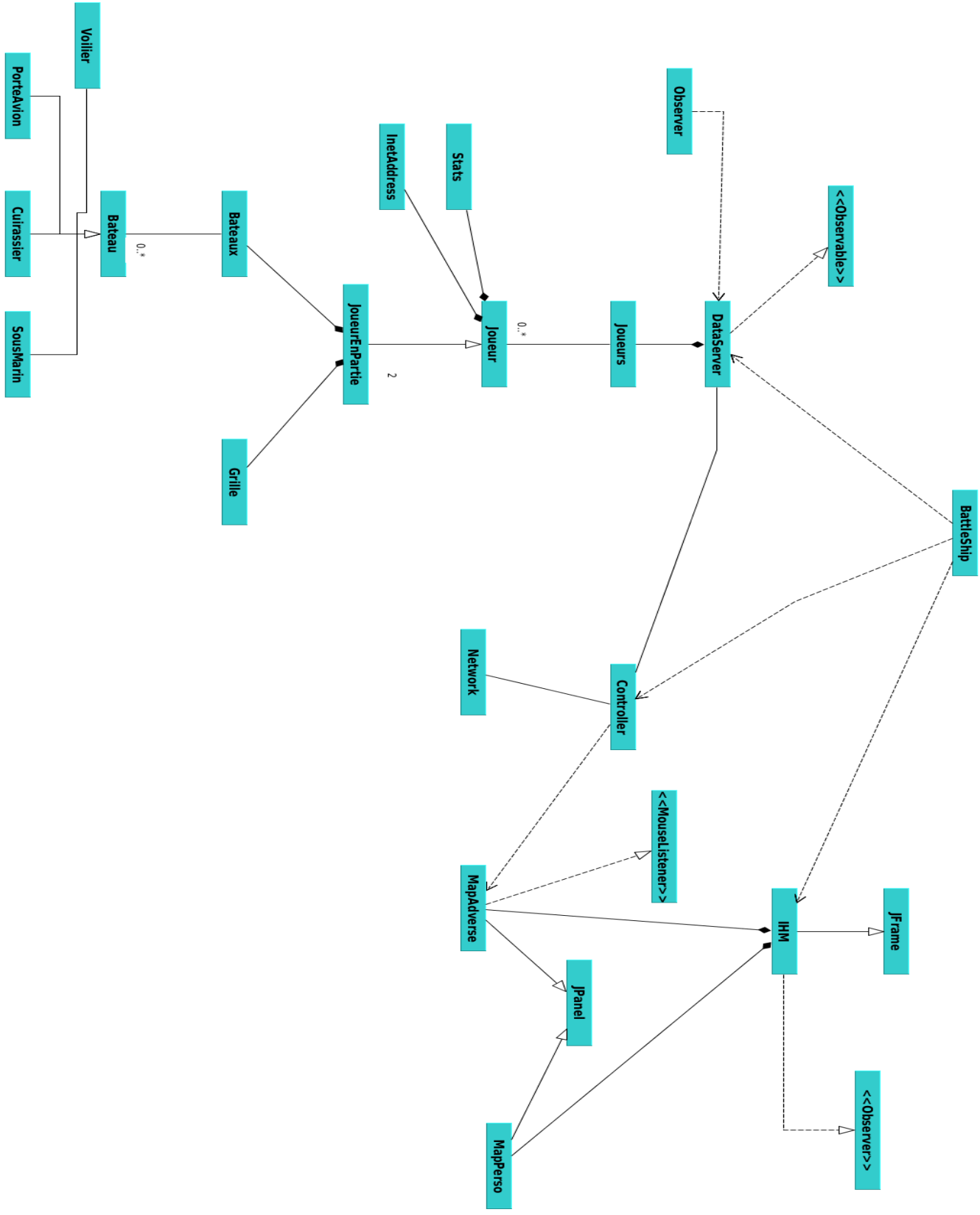
## Schémas BPM du système à réaliser

Afin de synthétiser les fonctionnalités de notre jeu ainsi que les problèmes auxquels il devra répondre, nous avons réalisé un schéma BPM. Il s'agit d'un schéma avec plusieurs rubriques dans le but « d'aboutir à une meilleure vue globale de l'ensemble des processus métiers de l'entreprise et de leurs interactions ». C'est dans ce sens , qu'il constitue un outil très efficace dans l'élaboration et le suivi d'un projet. :

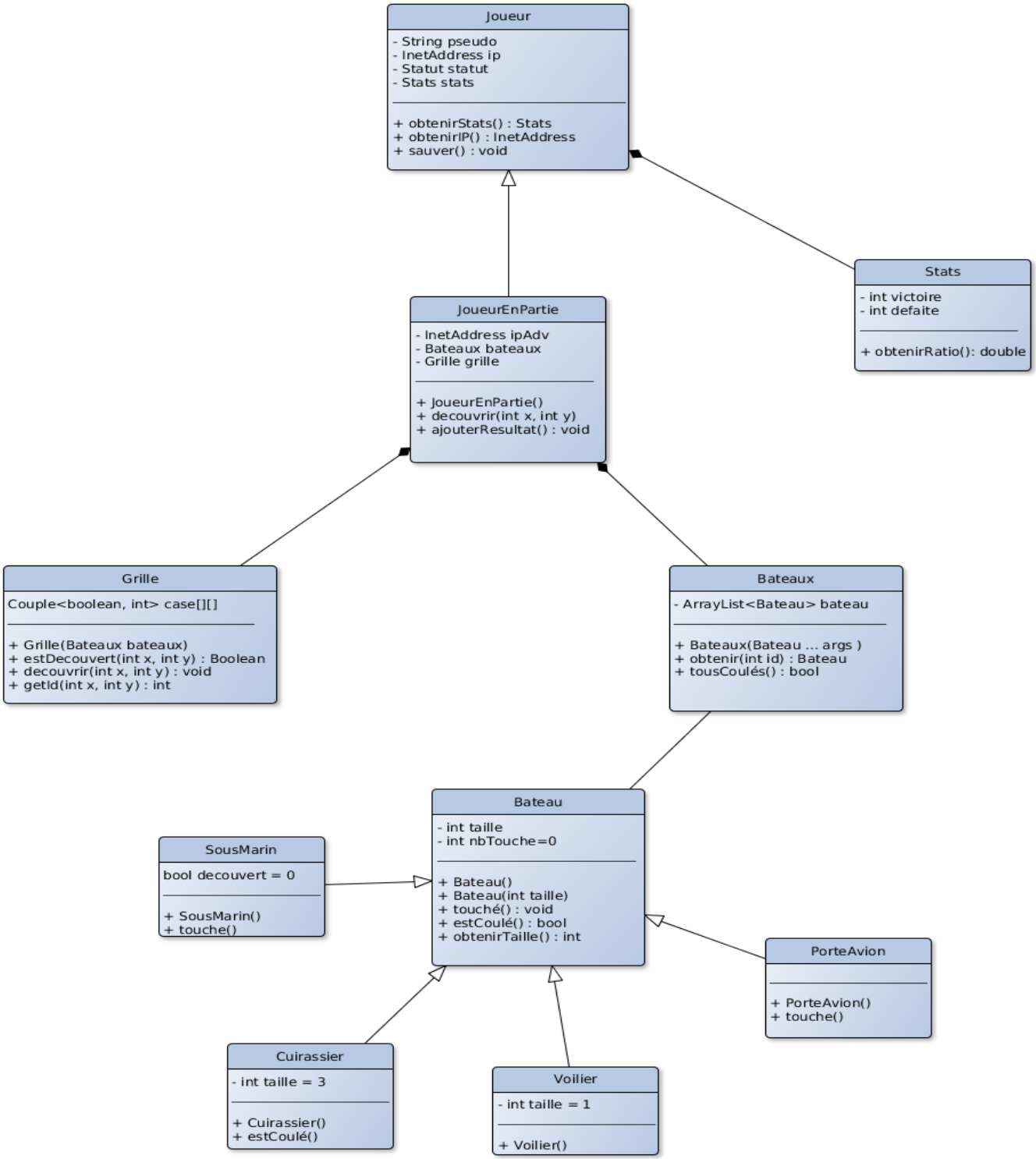



*Schémas BPM de la Bataille Navale*

# Diagramme de classes du jeu



# Description des classes



## Description de la classe Bateau et ses sous-classes

Nous avons la taille du bateau et le nombre de fois il a été touché. Grâce à ces deux attributs nous pourrons déterminer s'il est coulé (ie : $nbTouch = taille$ pour un bateau standard), et nous aurons la méthode touché qui incrémentera l'attribut nbTouche de 1 si l'adversaire touche une case sur laquelle se situe le bateau.

- Cuirassier : Redéfinissions de la méthode estCoulé() $\rightarrow nbTouche = 2 * taille$
- Voilier : Taille de 1
- Sous-Marin : Redéfinissions de la méthode touche() $\rightarrow$ si l'attribut "decouvert" est faux alors il devient vrai, sinon incrémente "nbTouche"
- Porte-Avion : Redéfinissions de la méthode touche() $\rightarrow$ dès qu'il est touché, le porte-avion lance une contre-attaque sur la même case

## Description de la classe Grille

C'est un tableau 2D de couple (ou paire) de valeur contenant un booléen et un entier. Le booléen spécifie si la case est découverte ou non et l'entier correspond à l'identifiant du bateau dans la liste des bateaux.

- estDecouverte(x,y) renvoie vrai si la première composante de case[x][y] est vrai, faux sinon
- getId(x,y) renvoie la 2 ème composante de case[x][y]
- decouvrir(x,y) appelle la méthode touche() du bateau correspondant à l'identifiant de la case (2ème composante)

# Conclusion et perspectives

Ce projet d'analyse du jeu de la Bataille Navale nous a permis d'approfondir nos connaissances en ce qui concerne la conception préliminaire d'un projet informatique. Nous pensons réutilisé cette partie théorique pour le mini-projet java et développer ce jeu de bataille navale en réseau.

Nous avons pu nous rendre compte que réaliser une analyse profonde d'un problème, par le biais du BPM et des différents diagrammes réalisés, permet d'y voir plus clair sur la conception du jeu. Ce qui nous permettra finalement de simplifier la phase d'écriture du code.

Désormais, l'objectif est de passer à la partie programmation du projet.