

## Saé 2.01 – Développement d’une application

### Chifoumi – Dossier d’Analyse et conception

---

#### Table des matières

1. Compléments de spécifications externes.....	2
2. Diagramme des Cas d’Utilisation .....	2
3. Scénarios .....	2
4. Diagramme de classe (UML) .....	2
Version v0.....	6
5. Implémentation et tests.....	6
Version v1.....	7
6. Classe Chifoumi : Diagramme états-transitions .....	7
7. Éléments d’interface .....	9
8. Implémentation et tests.....	10
Version v2.....	11
9. Implémentation et tests.....	11
Version v3.....	12
10. Implémentation et tests.....	12
Version v4.....	13
11. Diagramme État-transition de cette version.....	13
12. Implémentations et tests .....	16
Version v5.....	17
13. Diagramme État-transition de cette version.....	17
14. Implémentations et tests .....	22
Version v6.....	24
15. Diagramme État-transition de cette version.....	24
16. Implémentations et tests .....	29
Version v7.....	30
17. Diagramme État-transition de cette version.....	30
18. Implémentations et tests .....	35

## 1. Compléments de spécifications externes.

On précise **uniquement** les points qui vous ont semblé flous ou bien incomplets. Rien de plus à signaler dans cette étude.

## 2. Diagramme des Cas d'Utilisation

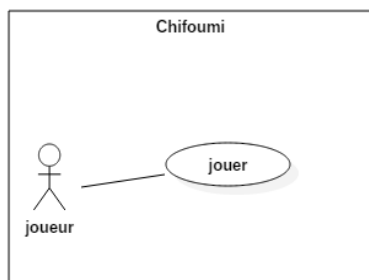


Figure 1 : Diagramme des Cas d'Utilisation du jeu Chifoumi

## 3. Scénarios

### (a) Exemple Scénario

Cas d'utilisation	JOUER	
Résumé	Le joueur joue une partie.	
Acteur primaire	Joueur	
Système	Chifoumi	
Intervenants		
Niveau	Objectif utilisateur	
Préconditions	Le jeu est démarré et se trouve à l'état initial.	
Postconditions		
Date de création		
Date de mise à jour		
Créateur		
Opérations	Joueur	Système
1	Démarre une nouvelle partie.	
2		Rend les figures actives et les affiche actives.
3	Choisit une figure.	
4		Affiche la figure du joueur dans la zone d'affichage du dernier coup joueur.
5		Choisit une figure.
6		Affiche sa figure dans la zone d'affichage de son dernier coup.
7		Détermine le gagnant et met à jour les scores.
8		Affiche les scores. Retour à l'étape 3.
Extension		
3.A	Le joueur demande à jouer une nouvelle partie.	
3.A.1	Choisit une nouvelle partie	
3.A.2		Réinitialise les scores.
3.A.3		Réinitialise les zones d'affichage des derniers coups.
3.A.4		Retour à l'étape 3.

Tableau 1 :  
Scénario  
nominal

### (b) Remarques :

- *Le scénario est très simple.*
- *L'objectif est de mettre en évidence les actions de l'utilisateur, celles du système, sachant que ces actions sont candidates à devenir des méthodes du système*

## 4. Diagramme de classe (UML)

(a) Le diagramme de classes UML du jeu se focalise sur les classes **métier**, cad celles décrivant le jeu

indépendamment des éléments d’interface que comportera le programme.

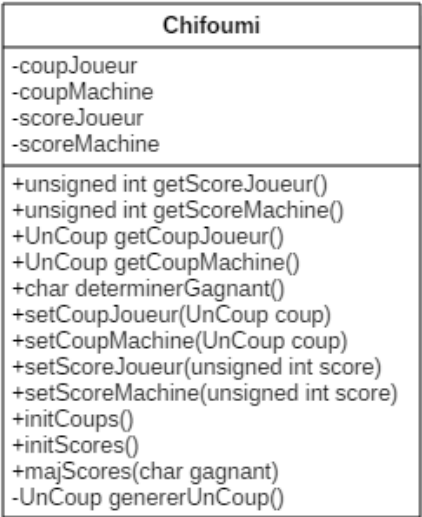


Figure 2 : Diagramme de Classes UML du jeu Chifoumi

(b) Dictionnaire des éléments de la Classe Chifoumi

Nom attribut	Signification	Type	Exemple
scoreJoueur	Nbre total de points acquis par le joueur durant la partie courante	unsigned int	1
scoreMachine	Nbre total de points acquis par la machine durant la partie courante	unsigned int	1
coupJoueur	Mémoire la dernière figure choisie par le joueur. Type énuméré enum unCoup {pierre, ciseau, papier, rien};	UnCoup	papier
coupMachine	Mémoire la dernière figure choisie par la machine.	UnCoup	Ciseau

Tableau 2 : Dictionnaire des éléments - Classe Chifoumi

(c) Dictionnaire des méthodes : intégrées dans l'interface de la classe : cf Figure 3

```
using namespace std;
class Chifoumi
{
    ///  
---- PARTIE MODÈLE -----  

    ///  
Une définition de type énuméré  

public:
    enum UnCoup {pierre, papier, ciseau, rien};

    ///  
Méthodes publiques du Modèle  

public:
    Chifoumi();
    virtual ~Chifoumi();

    // Getters  

    UnCoup getCoupJoueur();
        /* retourne le dernier coup joué par le joueur */
    UnCoup getCoupMachine();
        /* retourne le dernier coup joué par le joueur */
    unsigned int getScoreJoueur();
        /* retourne le score du joueur */
    unsigned int getScoreMachine();
        /* retourne le score de la machine */
    char determinerGagnant();
        /* détermine le gagnant 'J' pour joueur, 'M' pour machine, 'N' pour match nul
        en fonction du dernier coup joué par chacun d'eux */

    ///  
Méthodes utilitaires du Modèle  

private:
    UnCoup genererUnCoup();
    /* retourne une valeur aléatoire = pierre, papier ou ciseau.
    Utilisée pour faire jouer la machine */

    // Setters  

public:
    void setCoupJoueur(UnCoup p_coup);
        /* initialise l'attribut coupJoueur avec la valeur
        du paramètre p_coup */
    void setCoupMachine(UnCoup p_coup);
        /* initialise l'attribut coupMachine avec la valeur
        du paramètre p_coup */
    void setScoreJoueur(unsigned int p_score);
        /* initialise l'attribut scoreJoueur avec la valeur
        du paramètre p_score */
    void setScoreMachine(unsigned int p_score);
        /* initialise l'attribut coupMachine avec la valeur
        du paramètre p_score */

    // Autres modificateurs  

    void majScores(char p_gagnant);
        /* met à jour le score du joueur ou de la machine ou aucun
        en fonction des règles de gestion du jeu */
    void initScores();
        /* initialise à 0 les attributs scoreJoueur et scoreMachine
        NON indispensable */
    void initCoups();
        /* initialise à rien les attributs coupJoueur et coupMachine
        NON indispensable */

    ///  
Attributs du Modèle  

private:
    unsigned int scoreJoueur;    // score actuel du joueur
    unsigned int scoreMachine;  // score actuel de la Machine
    UnCoup coupJoueur;          // dernier coup joué par le joueur
    UnCoup coupMachine;         // dernier coup joué par la machine
};
```

Figure 3 : Schéma de classes = Une seule classe Chifoumi

**(d) Remarques concernant le schéma de classes**

1. On ne s'intéresse qu'aux attributs et méthodes métier. Notamment, on ne met pas, pour l'instant, ce qui relève de l'affichage car ce sont d'autres objets du programme (widgets) qui se chargeront de l'affichage. Par contre, on n'oublie pas les méthodes `getXXX()`, qui permettront aux objets métier de communiquer leur valeur aux objets graphiques pour que ceux-ci s'affichent.
2. On n'a mis ni le constructeur ni le destructeur, pour alléger le schéma.
3. D'autres attributs et méthodes viendront compléter cette vision ANALYTIQUE du jeu. Il s'agira des attributs et méthodes dits DE CONCEPTION nécessaires au développement de l'application.

### 5. Implémentation et tests

#### 5.1 Implémentation

Liste des fichiers de cette version :

- chifoumi.h : interface la classe Chifoumi
- chifoumi.cpp : Corps de la classe Chifoumi
- main.cpp : Moteur + tests de la classe
- acChifoumiAlvesJouve\_TP4.pdf : Document d'analyse

Respectivement spécification et corps de la classe Chifoumi décrite au paragraphe 4.

#### 5.2 Test

Test avec le programme fourni main.cpp

Valeurs fournies / attendues... comme montré dans la ressource R2.03 (partie tests)

Méthode	Valeur rentrée	Résultat attendu	Résultat obtenu	Commentaire
determinerGagnant()	coupJoueur=Pierre coupMachine=Pierre	'N'	'N'	ok
determinerGagnant()	coupJoueur=Pierre coupMachine=Papier	'M'	'M'	ok
determinerGagnant()	coupJoueur=Pierre coupMachine=Ciseau	'J'	'J'	ok
determinerGagnant()	coupJoueur=Papier coupMachine=Pierre	'J'	'J'	ok
determinerGagnant()	coupJoueur=Papier coupMachine=Papier	'N'	'N'	ok
determinerGagnant()	coupJoueur=Papier coupMachine=Ciseau	'M'	'M'	ok
determinerGagnant()	coupJoueur=Ciseau coupMachine=Pierre	'M'	'M'	ok
determinerGagnant()	coupJoueur=Ciseau coupMachine=Papier	'J'	'J'	ok
determinerGagnant()	coupJoueur=Ciseau coupMachine=Ciseau	'N'	'N'	ok

Méthode	Valeur rentrée	Résultat attendu	Résultat obtenu	Commentaire
majScore(char)	'M'	scoreJoueur++	scoreJoueur++	Ok
majScore(char)	'J'	scoreMachine++	scoreMachine++	Ok

Méthode	Valeur rentrée	Résultat attendu	Résultat obtenu	Commentaire
setScoreJoueur(unsigned int)	1	scoreJoueur=1	scoreJoueur=1	ok
setScoreMachine(unsigned int)	2	scoreMachine=2	scoreMachine=2	ok
getScoreJoueur()	scoreJoueur	0	0	Ok
getScoreMachine()	scoreMachine	0	0	Ok

Méthode	Valeur rentrée	Résultat attendu	Résultat obtenu	Commentaire
setCoupJoueur(unCoup)	pierre	coupJoueur=pierre	coupJoueur=pierre	ok
setCoupMachine(unCoup)	ciseau	coupMachine=ciseau	coupMachine=ciseau	ok
getCoupJoueur()	coupJoueur	pierre	pierre	Ok
getCoupMachine()	coupMachine	ciseau	ciseau	ok

## Version v1

### 6. Classe Chifoumi : Diagramme états-transitions

#### (a) Diagramme états-transitions -actions du jeu

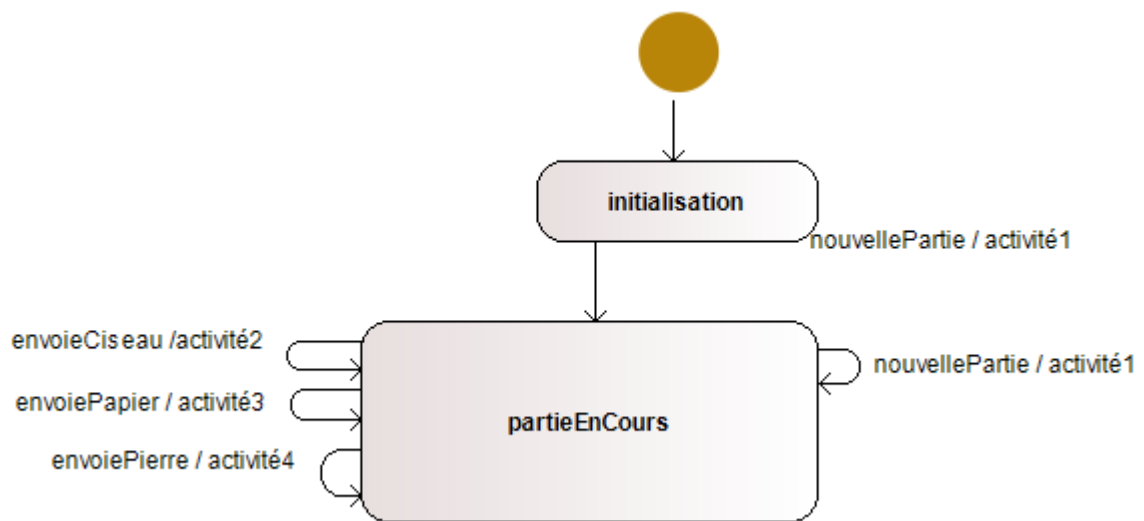


Figure 4 : Diagramme états-transitions

(b) Dictionnaires des états, événements et Actions

**Dictionnaire des états du jeu**

<i>nomEtat</i>	<i>Signification</i>
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer

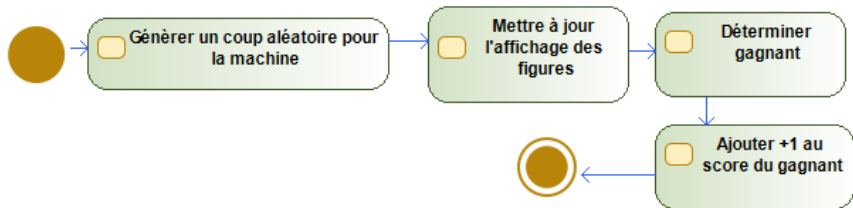
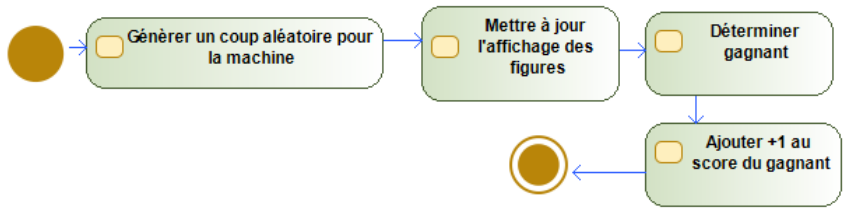
Tableau 3 : États du jeu

**Dictionnaire des événements faisant changer le jeu d'état**

<i>nomEvénement</i>	<i>Signification</i>
clickPierre	Le joueur clique sur le bouton bPierre
clickCiseau	Le joueur clique sur le bouton bCiseau
clickPapier	Le joueur clique du le bouton bPapier
clickNouvPart	Le joueur clique sur le bouton bNewGame

Tableau 4 : Evénements faisant changer le jeu d'état

**Description des actions réalisées lors de la traversée des transitions**

<i>nomAction</i>	<i>Signification</i>
<b>envoiCiseau</b>	<p>Réalise les opérations de la manche si le joueur a joué Ciseau.</p> 
<b>envoiPapier</b>	<p>Réalise les opérations de la manche si le joueur a joué Papier.</p> 



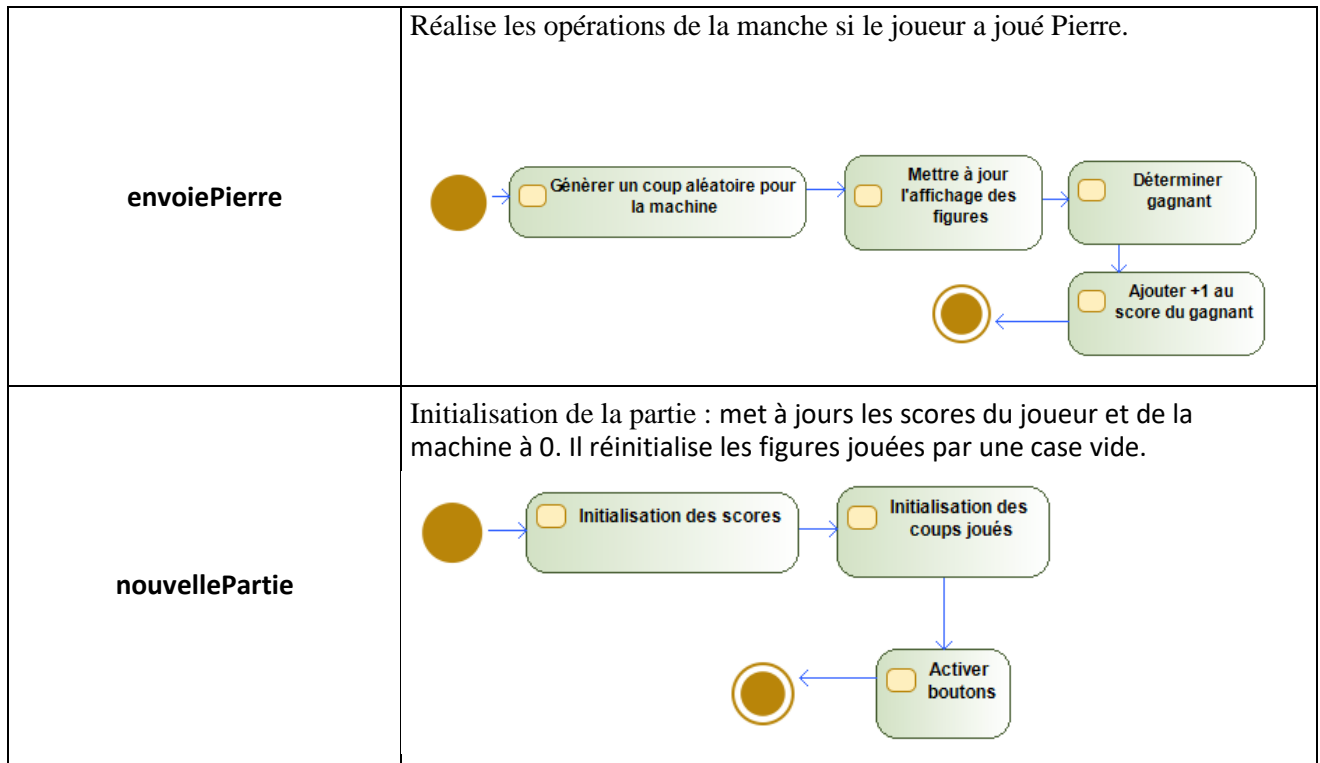


Tableau 5 : Actions à réaliser lors des changements d'état

(c) Préparation au codage :

Table **T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état
- en colonne : les *états* du jeu

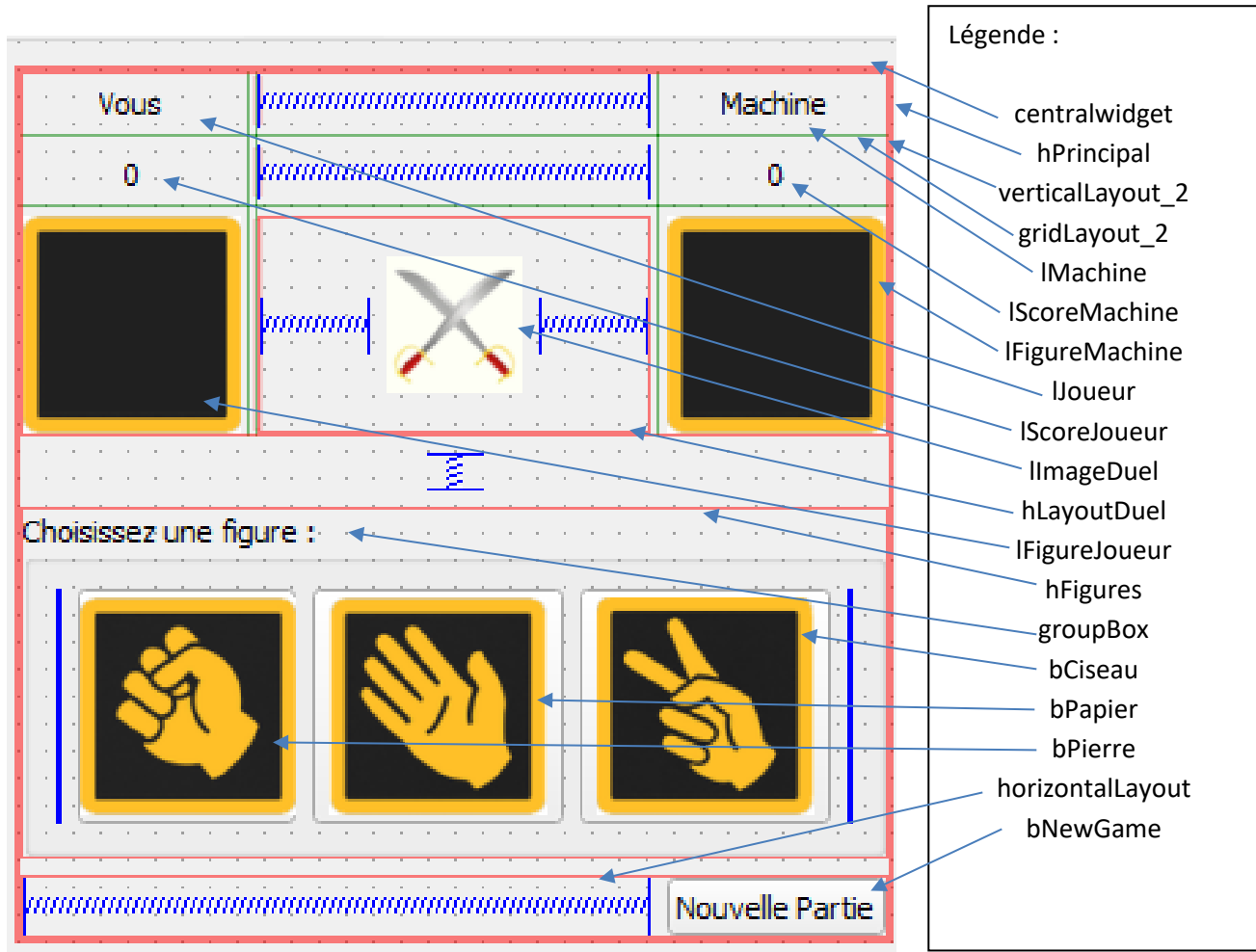
Éléments d'interface →	bCiseau	bPapier	bPierre	bNewGame
Événement → nomEtatJeu	clickCiseau	clickPapier	clickPierre	clickNouvPart
initialisation	---	---	---	partieEnCours/ activité1
partieEnCours	partieEnCours/ activité2	partieEnCours/ activité3	partieEnCours/ activité4	partieEnCours/ activité1

Tableau 6 : Matrice d'états-transitions du jeu chifoumi

*L'intérêt de cette vue matricielle est qu'elle permet une préparation naturelle et aisée de l'étape suivante de programmation.*

## 7. Éléments d'interface

*A faire ici : description sommaire des éléments de l'interface, par exemple, avec une copie d'écran sur laquelle sont nommés les variables/objets graphiques et où les layouts sont positionnés et nommés.*



## 8. Implémentation et tests

### 8.1 Implémentation

A faire :

Lister les fichiers impliqués dans cette version (répertoire, nom de fichier, rôle de chaque fichier)

Commenter brièvement les choix importants d'implémentation réalisés, comme par exemple, les signals/slots

- ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT
- chifoumiVue.h : Module de la classe de la vue du chifoumiVue
- chifoumiVue.cpp : Corps de la classe chifoumiVue
- chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi
- main.cpp : Affichage de la fenêtre principale
- images : Répertoire contenant les images des figures du Chifoumi
- chifoumi.h : Module de la classe chifoumi
- chifoumi.cpp : Corps de la classe chifoumi

### 8.2 Test

A faire :

Décrire les tests prévus / réalisés pour montrer :

Le comportement fonctionnel du programme

Le comportement de l'interface non lié aux aspects fonctionnels du programme

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
<i>envoiePierre()</i>	<i>clickPierre</i> <i>coupMachine = Ciseau</i>	<i>scoreJoueur++</i>	<i>scoreJoueur++</i>	<i>ok</i>
<i>envoiePierre()</i>	<i>clickPierre</i> <i>coupMachine = Pierre</i>			<i>ok</i>
<i>envoiePierre()</i>	<i>clickPierre</i> <i>coupMachine = Papier</i>	<i>scoreMachine++</i>	<i>scoreMachine++</i>	<i>ok</i>
<i>envoiePapier()</i>	<i>clickPapier</i> <i>coupMachine = Pierre</i>	<i>scoreJoueur++</i>	<i>scoreJoueur++</i>	<i>ok</i>
<i>envoiePapier()</i>	<i>clickPapier</i> <i>coupMachine = Papier</i>			<i>ok</i>
<i>envoiePapier()</i>	<i>clickPapier</i> <i>coupMachine = Ciseau</i>	<i>scoreMachine++</i>	<i>scoreMachine++</i>	<i>ok</i>
<i>envoieCiseau()</i>	<i>clickCiseau</i> <i>coupMachine = Papier</i>	<i>scoreJoueur++</i>	<i>scoreJoueur++</i>	<i>ok</i>
<i>envoieCiseau()</i>	<i>clickCiseau</i> <i>coupMachine = Ciseau</i>			<i>ok</i>
<i>envoieCiseau()</i>	<i>clickCiseau</i> <i>coupMachine = Pierre</i>	<i>scoreMachine++</i>	<i>scoreMachine++</i>	<i>ok</i>
<i>nouvellePartie()</i>	<i>clickNouvPart</i>	<i>scoreJoueur = 0</i> <i>scoreMachine = 0</i>	<i>scoreJoueur = 0</i> <i>scoreMachine = 0</i>	<i>ok</i>
<i>miseAJourInterface(Chifoumi::UnCoup, Chifoumi::UnCoup);</i>	<i>Ciseau, Papier</i>	<i>lfigureJoueur = ciseau_115.png</i> <i>lfigureMachine = papier_115.png</i>	<i>lfigureJoueur = ciseau_115.png</i> <i>lfigureMachine = papier_115.png</i>	<i>ok</i>

## Version v2

### 9. Implémentation et tests

#### 9.1 Implémentation

Liste des fichiers de cette version :

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaire à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi  
presentation.h : Module de la classe présentation  
presentation.cpp : Sources de la classe présentation  
chifoumi.cpp : Corps de la classe chifoumi  
chifoumi.h : Module de la classe chifoumi

## 9.2 Présentation des .h

La classe presentation représente le modèle de la classe Chifoumi. (Fonctionnement général du jeu)  
La classe chifoumi représente la présentation de la classe Chifoumi. (Fonctionnement complet du jeu)  
La classe chifoumivue représente la vue de la classe Chifoumi. (Aspect graphique)

## 9.3 Tests

Test avec le programme fourni main.cpp

*Valeurs fournies / attendues... comme montré dans la ressource R2.03 (partie tests)*

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
majScoreJoueur()	5	lScoreJoueur = 5	lScoreJoueur = 5	ok
majScoreMachine()	2	lScoreJoueur = 2	lScoreJoueur = 6	ok

## Version v3

# 10. Implémentation et tests

## 10.1 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaires à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et l'exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

Les seuls fichiers .h modifiés pendant la mise en œuvre de la v3 sont « chifoumiVue.h » et « presentation.h ».

Le slot de la présentation appelle la méthode de la vue qui affiche la message Box.

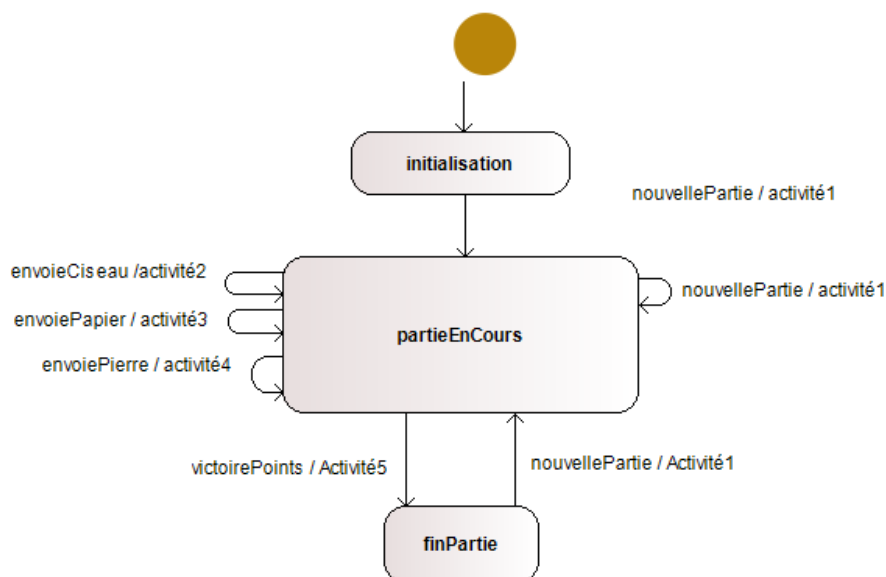
## 10.2 Tests

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
actionQuitter	triggered(actionQuitter)	Fin du programme	Fin du programme	ok
(actionA_propos_de	triggered(actionA_propos_de)	Afficher Message Box	Affiche Message Box	ok
actionQuitter	F1	Afficher Message Box	Affiche Message Box	ok
(actionA_propos_de	Alt + F3	Fin du programme	Fin du programme	ok

## Version v4

### 11. Diagramme État-transition de cette version

#### (a) Diagramme états-transitions -actions du jeu



(b) Dictionnaires des états, événements et Actions

**Dictionnaire des états du jeu**

<i>nomEtat</i>	<i>Signification</i>
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer
finPartie	La partie est terminée, une fenêtre s'ouvre indiquant le résultat de la partie et les figures sont désactivées.

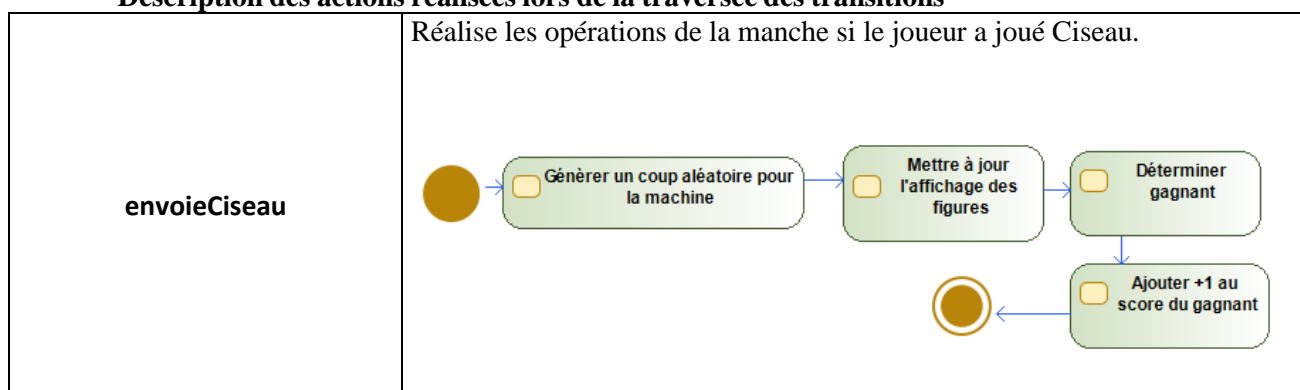
Tableau 7 : États du jeu

**Dictionnaire des événements faisant changer le jeu d'état**

<i>nomÉvénement</i>	<i>Signification</i>
clickPierre	Le joueur clique sur le bouton bPierre
clickCiseau	Le joueur clique sur le bouton bCiseau
clickPapier	Le joueur clique du le bouton bPapier
clickNouvPart	Le joueur clique sur le bouton bNewGame
clickPierre	Le joueur clique sur le bouton bPierre
scoreReachMax	Le score du joueur ou de la machine atteint le score max fixé.

Tableau 8 : Événements faisant changer le jeu d'état

**Description des actions réalisées lors de la traversée des transitions**



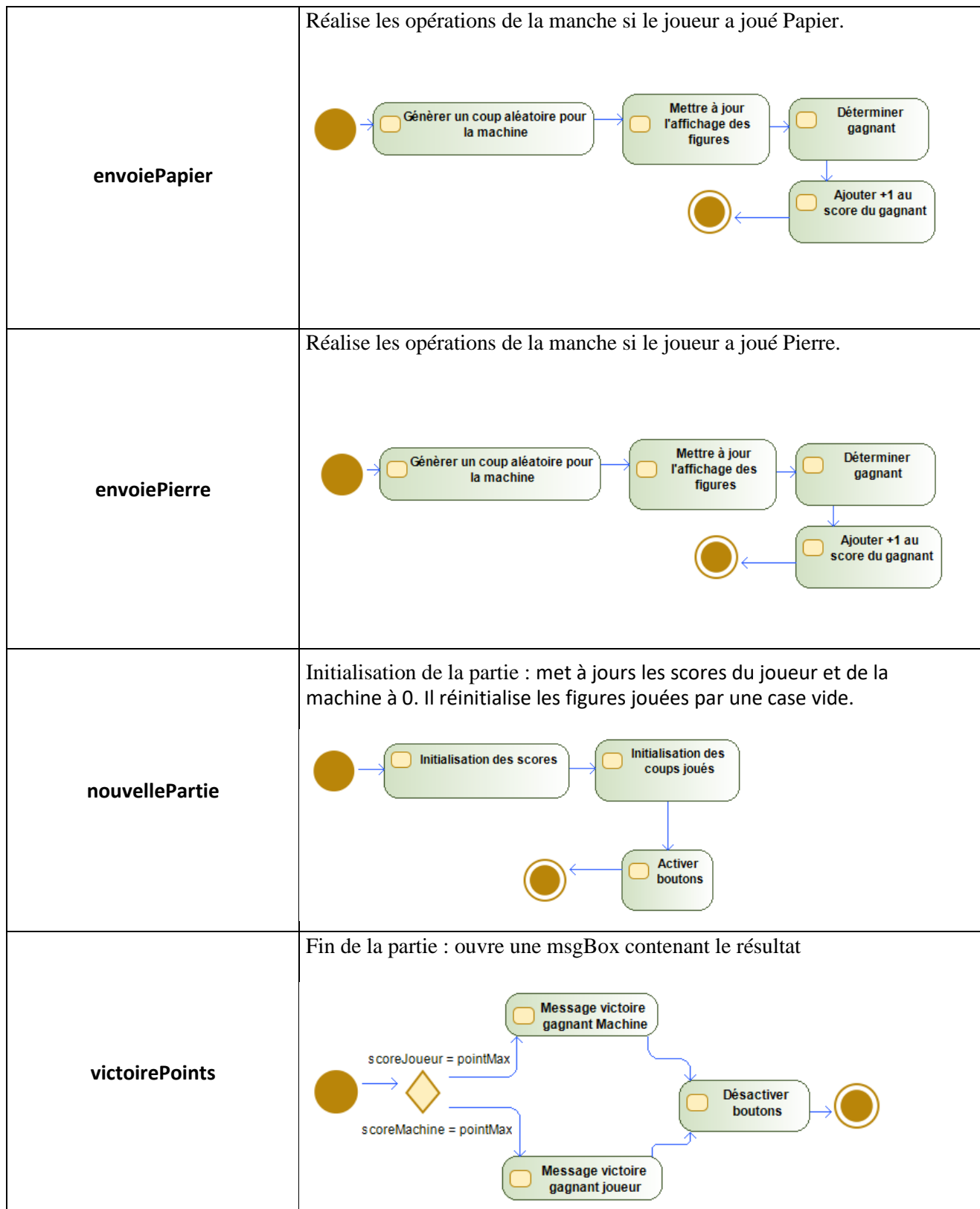


Tableau 8 : Actions à réaliser lors des changements d'état

(d) Préparation au codage :

Table **T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état
- en colonne : les *états* du jeu

-

<i>Éléments d'interface →</i>	<b>bCiseau</b>	<b>bPapier</b>	<b>bPierre</b>	<b>bNewGame</b>	---
<i>Événement → nomEtatJeu</i>	<b>clickCiseau</b>	<b>clickPapier</b>	<b>clickPierre</b>	<b>clickNouvPart</b>	<b>scoreReachMax</b>
initialisation	---	---	---	partieEnCours/ activité1	---
partieEnCours	partieEnCours/ activité2	partieEnCours/ activité3	partieEnCours/ activité4	partieEnCours/ activité1	partieEnCours/ activité5
finPartie	---	---	---	partieEnCours/ activité1	---

Tableau 9 : Matrice d'états-transitions du jeu chifoumi

## 12. Implémentations et tests

### 12.1 Nouveaux éléments d'interface

Le nouvel élément graphique ici est la Message Box qui affiche les résultats de la partie.

Le label « IGagne » qui contient le texte « Gagnant à » et le label IPointsGagnant qui contient lui le contenu de la variable pointsMax à atteindre pour terminer la partie.

### 12.2 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaire à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

Le fichier chifoumiVue.h a été le seul modifié dans cette version pour permettre l'affichage de la Message Box.

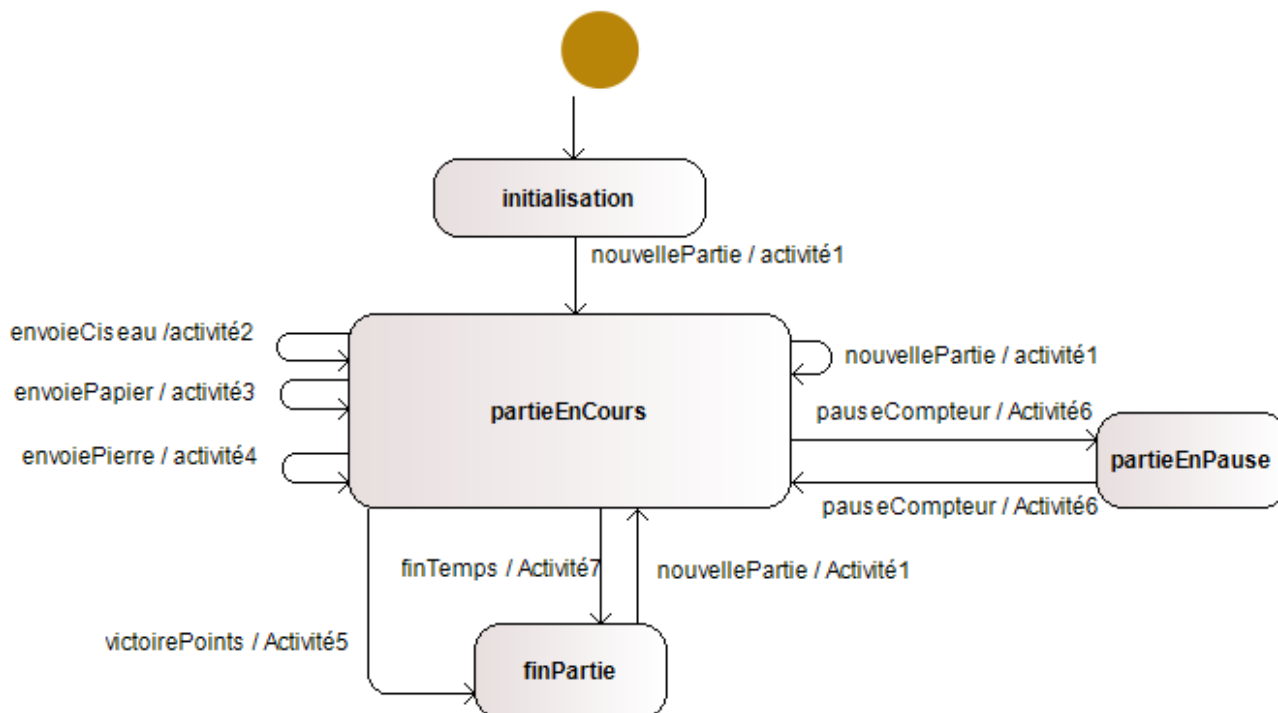
### 12.3 Tests

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
messageVictoire(char)	'J'	Message victoire joueur	Message victoire joueur	ok
messageVictoire(char)	'M'	Message victoire machine	Message victoire machine	ok



### 13. Diagramme État-transition de cette version

#### (a) Diagramme états-transitions -actions du jeu



#### (b) Dictionnaires des états, événements et Actions

##### Dictionnaire des états du jeu

<i>nomEtat</i>	<i>Signification</i>
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer
finPartie	La partie est terminée, une fenêtre s'ouvre indiquant le résultat de la partie et les figures sont désactivées.
partieEnPause	La partie est en pause, le joueur ne peut pas sélectionner de figure ni lancer de nouvelle partie dans cet état. Le chrono du temps restant est arrêté le temps de cet état.

Tableau 10: États du jeu

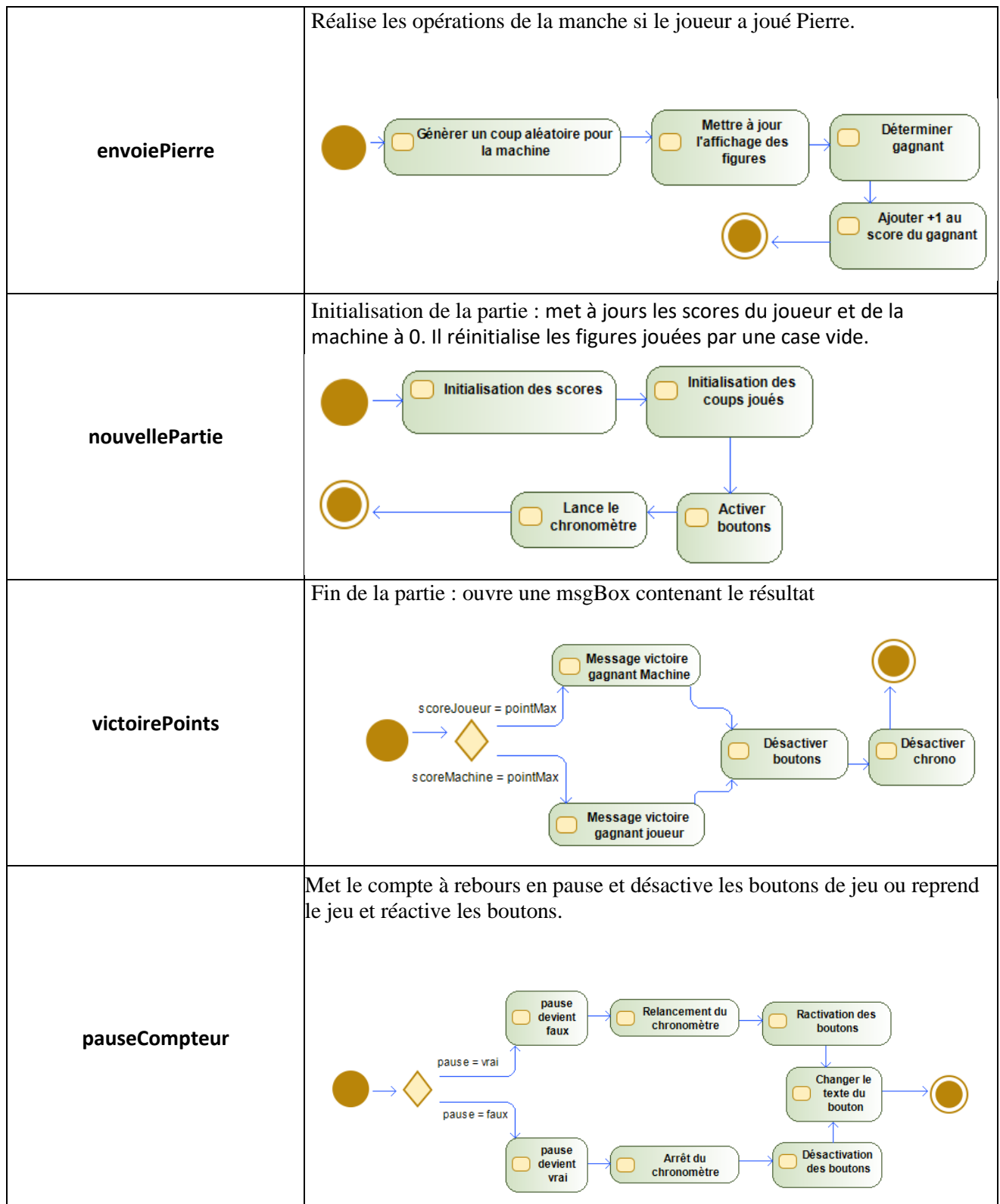
### Dictionnaire des événements faisant changer le jeu d'état

<i>nomEvénement</i>	<i>Signification</i>
clickPierre	Le joueur clique sur le bouton bPierre
clickCiseau	Le joueur clique sur le bouton bCiseau
clickPapier	Le joueur clique du le bouton bPapier
clickNouvPart	Le joueur clique sur le bouton bNewGame
clickPierre	Le joueur clique sur le bouton bPierre
scoreReachMax	Le score du joueur ou de la machine atteint le score max fixé.
clickPause	Le joueur clique sur le bouton bPause
tempsEcoule	Le compte à rebours atteint 0.

Tableau 11 : Événements faisant changer le jeu d'état

### Description des actions réalisées lors de la traversée des transitions

<b>envoiCiseau</b>	<p>Réalise les opérations de la manche si le joueur a joué Ciseau.</p>
<b>envoiPapier</b>	<p>Réalise les opérations de la manche si le joueur a joué Papier.</p>



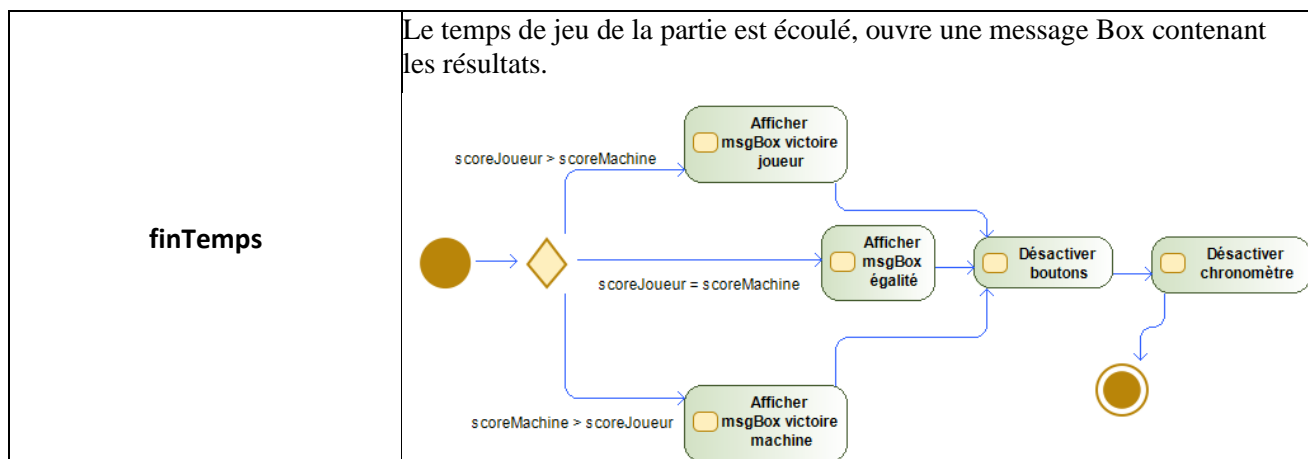


Tableau 12 : Actions à réaliser lors des changements d'état

(c) Préparation au codage :

**Table T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état

- en colonne : les *états* du jeu

Éléments d'interface →	bCiseau	bPapier	bPierre	bNewGame	---	bPause	---
Événement → nomEtatJeu	clickCiseau	clickPapier	clickPierre	clickNouvPart	scoreReachMax	clickPause	tempsEcoule
initialisation	---	---	---	partieEnCours/ activité1	---	---	---
partieEnCours	partieEnCours/ activité2	partieEnCours/ activité3	partieEnCours/ activité4	partieEnCours/ activité1	partieEnCours/ activité5	partieEnPause/ activité6	finPartie/ activité7
finPartie	---	---	---	partieEnCours/ activité1	---	---	---
partieEnPause	---	---	---		---	partieEnCours/ activité6	---

Tableau 13 : Matrice d'états-transitions du jeu chifoumi

## 14. Implémentations et tests

### 14.1 Nouveaux éléments d'interface

Le bouton « bPause » qui permet d'arrêter le compte à rebours de la partie et de le reprendre le jeu quand on le veut.

Un label « lSecondes » qui affiche le temps restant du compte à rebours.

Un label « lTemps » qui affiche la phrase « Tps restant (s) : ».

Un layout horizontal « horizontalLayout\_3 » dans lequel on trouvera lTemps et lSecondes.

Un layout vertical « verticalLayout » qui différencie l'image de duel de la partie consacrée au temps.

### 14.2 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaires à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et l'exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

Le fichier chifoumiVue.h et presentation.h ont été modifiés pendant la V5.

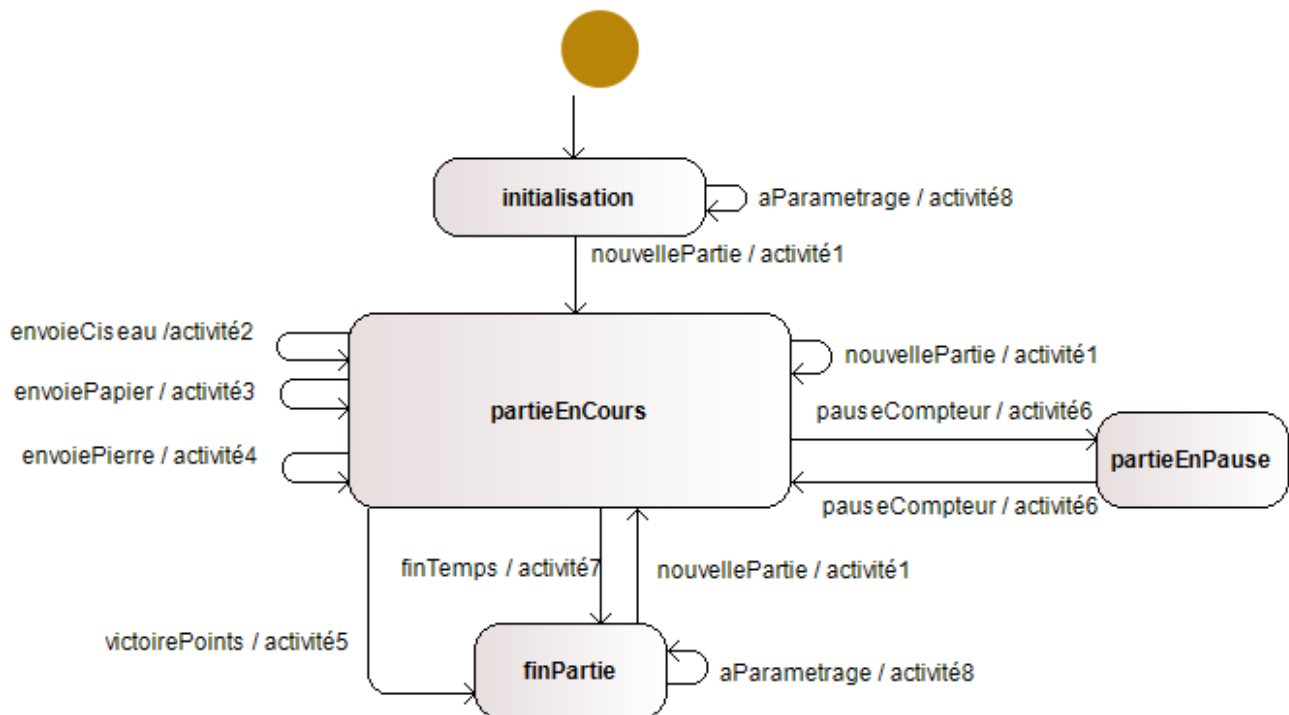
### 14.3 Tests

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
<i>tempsCompteur(unsigned int)</i>	<i>18</i>	<i>lSecondes -&gt; « 18 »</i>	<i>lSecondes -&gt; « 18 »</i>	<i>ok</i>
<i>messageFinTemps(char, unsigned short int)</i>	<i>char = 'J' Int = 8</i>	<i>Message victoire joueur de fin de temps</i>	<i>Message victoire joueur de fin de temps</i>	<i>ok</i>
<i>messageFinTemps(char, unsigned short int)</i>	<i>char = 'M' Int = 8</i>	<i>Message victoire machine de fin de temps</i>	<i>Message victoire machine de fin de temps</i>	<i>ok</i>
<i>messageFinTemps(char, unsigned short int)</i>	<i>char = 'N' Int = 8</i>	<i>Message égalité de fin de temps</i>	<i>Message égalité de fin de temps</i>	<i>ok</i>

## Version v6

### 15. Diagramme État-transition de cette version

#### (a) Diagramme états-transitions -actions du jeu



#### (b) Dictionnaires des états, événements et Actions

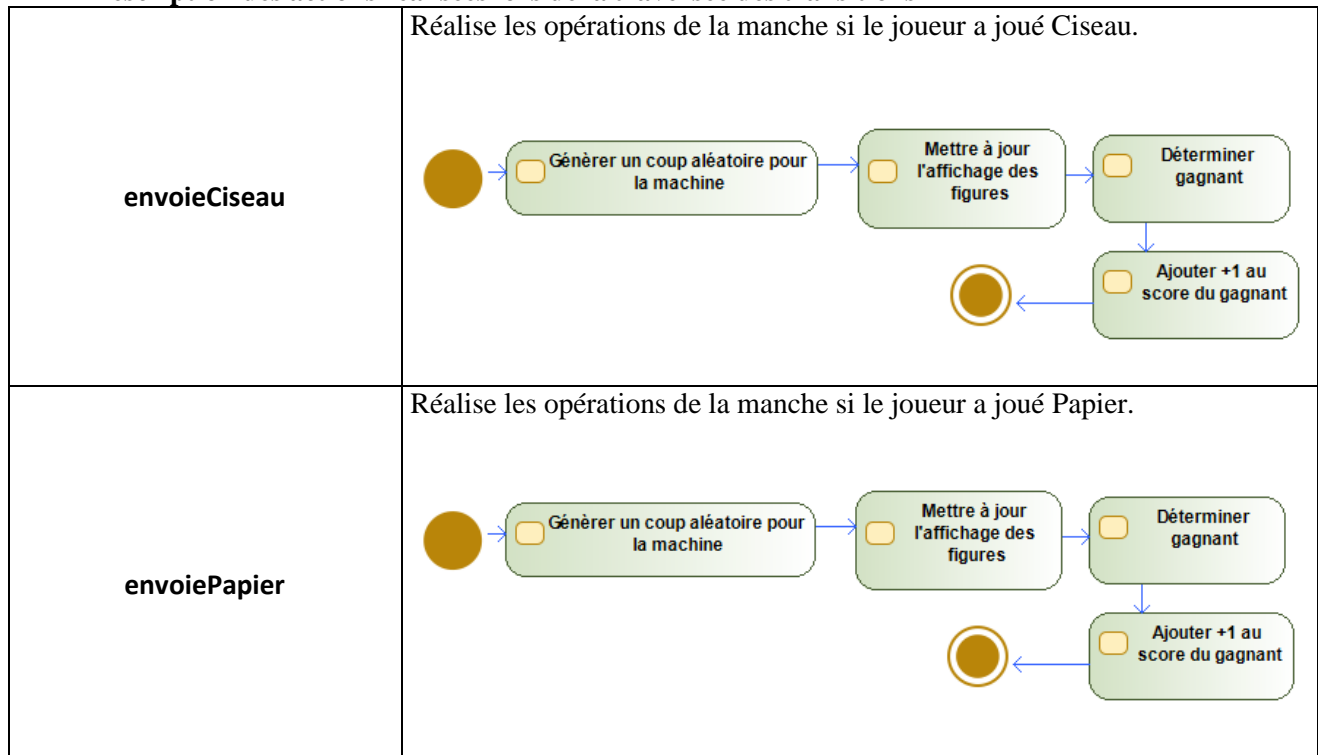
<i>nomEtat</i>	<i>Signification</i>
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer
finPartie	La partie est terminée, une fenêtre s'ouvre indiquant le résultat de la partie et les figures sont désactivées.
partieEnPause	La partie est en pause, le joueur ne peut pas sélectionner de figure ni lancer de nouvelle partie dans cet état. Le chrono du temps restant est arrêté le temps de cet état.

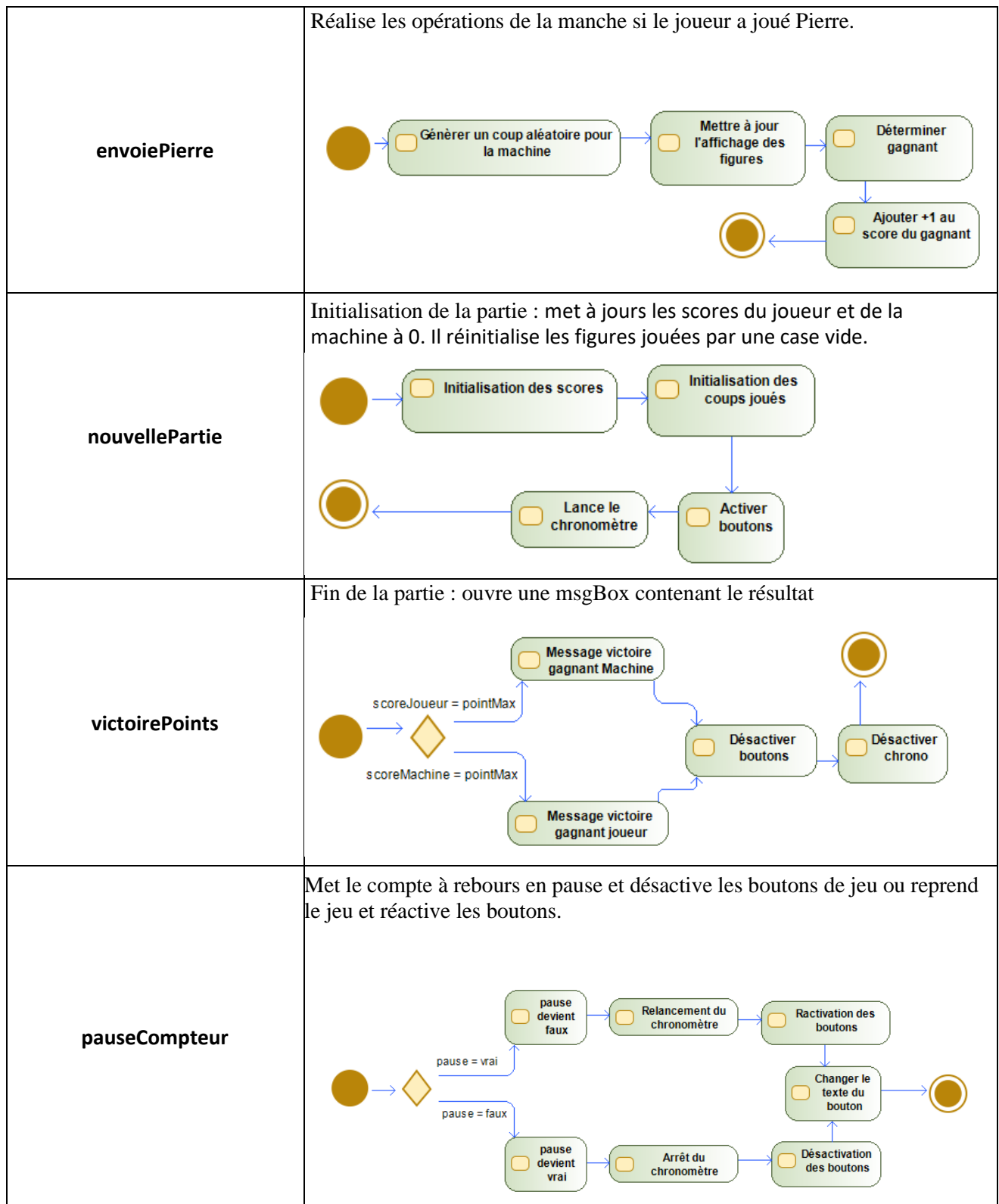


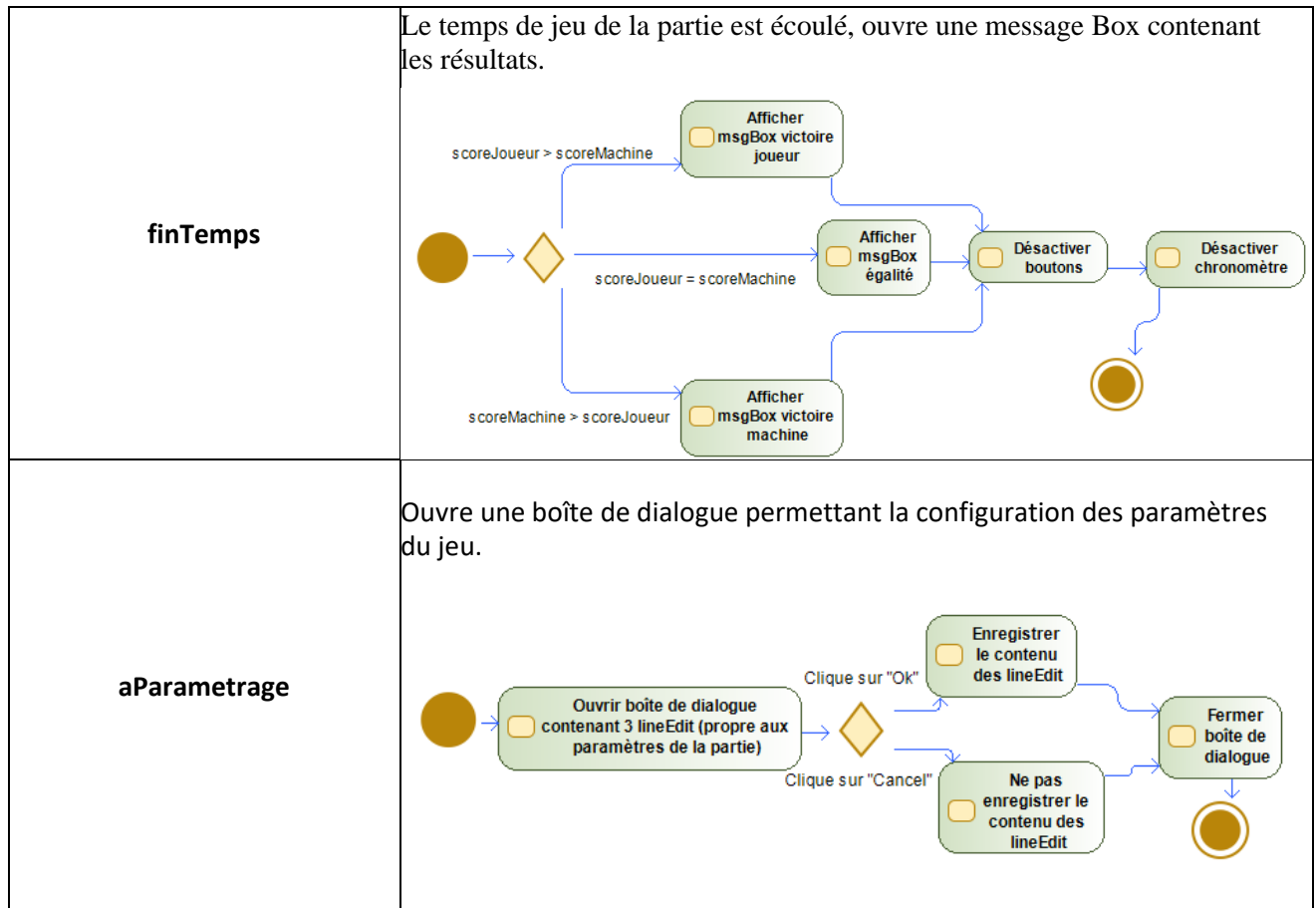
### Dictionnaire des événements faisant changer le jeu d'état

<i>nomEvénement</i>	<i>Signification</i>
clickPierre	Le joueur clique sur le bouton bPierre
clickCiseau	Le joueur clique sur le bouton bCiseau
clickPapier	Le joueur clique du le bouton bPapier
clickNouvPart	Le joueur clique sur le bouton bNewGame
clickPierre	Le joueur clique sur le bouton bPierre
scoreReachMax	Le score du joueur ou de la machine atteint le score max fixé.
clickPause	Le joueur clique sur le bouton bPause
tempsEcoule	Le compte à rebours atteint 0.
clickParametrage	Le joueur clique sur le menu actionParam_trer

### Description des actions réalisées lors de la traversée des transitions







(d) Préparation au codage :

**Table T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état

- en colonne : les *états* du jeu

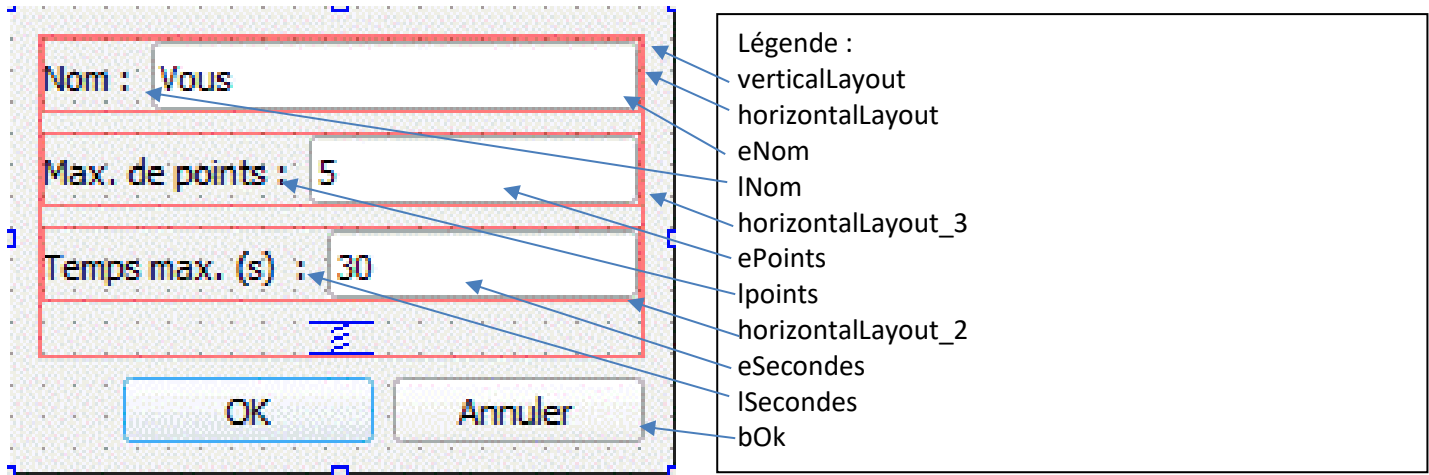
Éléments d'interface →	bCiseau	bPapier	bPierre	bNewGame	---	bPause	---	actionParam_trer
Événement → nomEtatJeu	clickCiseau	clickPapier	clickPierre	clickNouvPart	scoreReachMax	clickPause	tempsEcoule	clickParametrage
initialisation	---	---	---	partieEnCours/ activité1	---	---	---	initialisation /activité8
partieEnCours	partieEnCours/ activité2	partieEnCours/ activité3	partieEnCours/ activité4	partieEnCours/ activité1	partieEnCours/ activité5	partieEnPause/ activité6	finPartie/ activité7	---
finPartie	---	---	---	partieEnCours/ activité1	---	---	---	finPartie/ activité8
partieEnPause	---	---	---		---	partieEnCours/ activité6	---	---

Tableau 13 : Matrice d'états-transitions du jeu chifoumi

## 16. Implémentations et tests

### 16.1 Nouveaux éléments d'interface

Le menu paramétrer « actionParam\_trer » et sa fenêtre de dialogue.



### 16.2 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaires à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et l'exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

parametres.h : Module de la classe parametres qui ouvre une fenêtre de dialogue modale.

parametres.cpp : Corps de la classe parametres

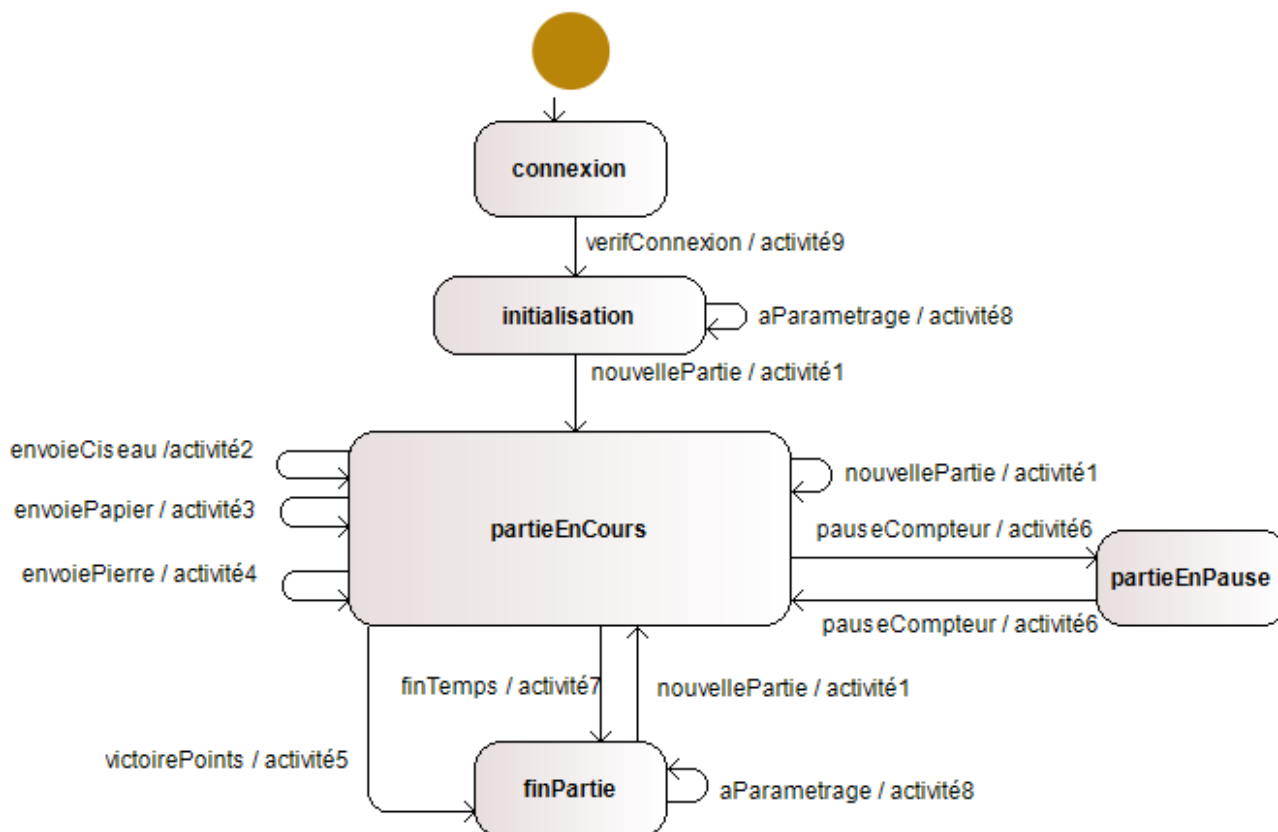
Les fichiers « parametres.h », « presentation.h » et « chifoumiVue.h » ont été modifiés pour la v6.

### 16.3 Tests

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
parametrage()	eNom -> « Test » ePoints -> « 8 » eSecondes -> « 60 » Clique sur OK	pointMax = 8 nom = « Test » secondesMax = 60	pointMax = 8 nom = « Test » secondesMax = 60	ok
parametrage()	eNom -> « Test » ePoints -> « 8 » eSecondes -> « 60 » Clique sur Cancel			ok

## 17. Diagramme État-transition de cette version

### (a) Diagramme états-transitions -actions du jeu



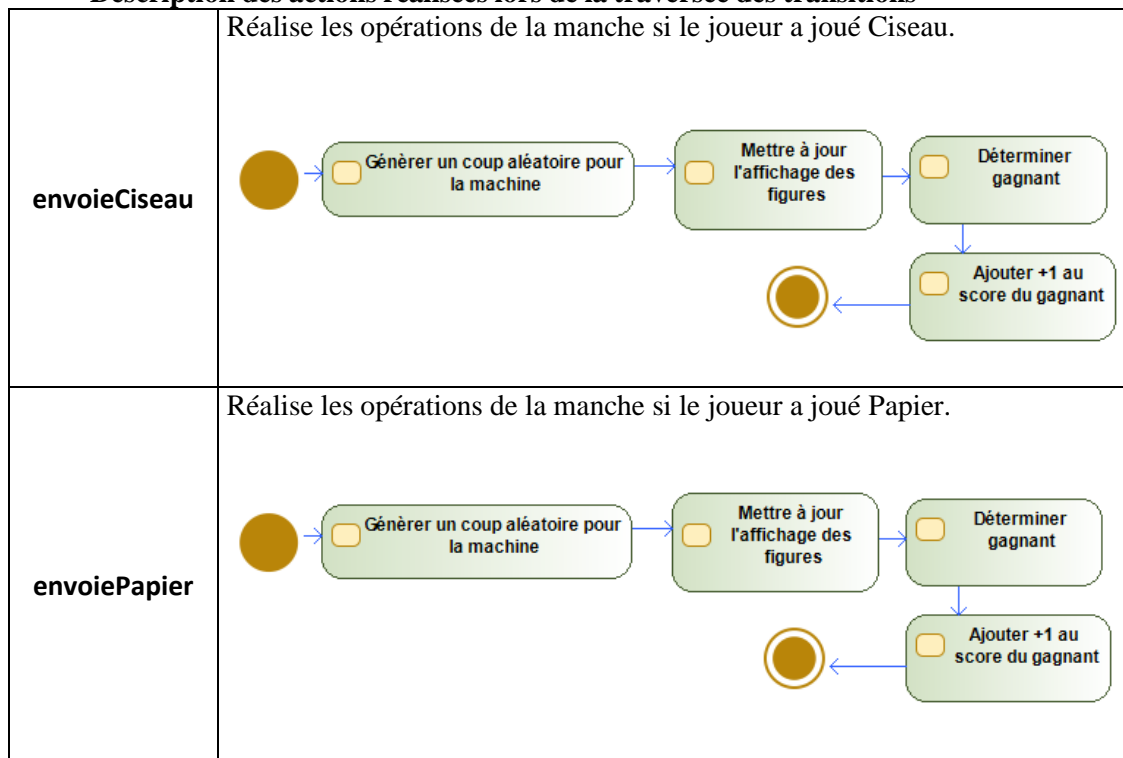
### (b) Dictionnaires des états, événements et Actions

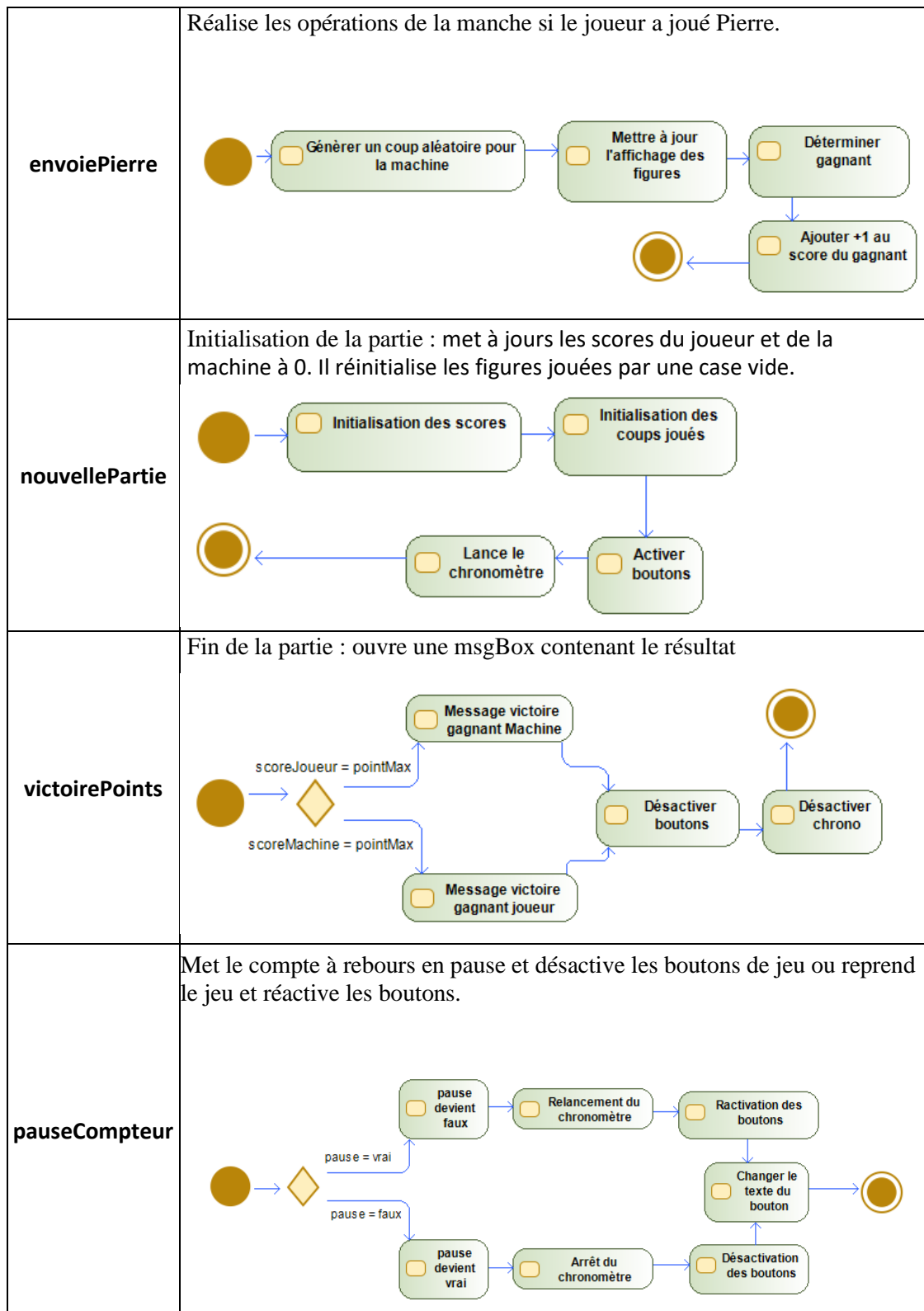
<i>nomEtat</i>	<i>Signification</i>
<i>connexion</i>	Le joueur doit s'authentifier pour pouvoir jouer au jeu du Chifoumi.
initialisation	État initial de la création du jeu
partieEnCours	La partie est en cours, le joueur sélectionne la figures pour jouer
finPartie	La partie est terminée, une fenêtre s'ouvre indiquant le résultat de la partie et les figures sont désactivées.
partieEnPause	La partie est en pause, le joueur ne peut pas sélectionner de figure ni lancer de nouvelle partie dans cet état. Le chrono du temps restant est arrêté le temps de cet état.

### Dictionnaire des événements faisant changer le jeu d'état

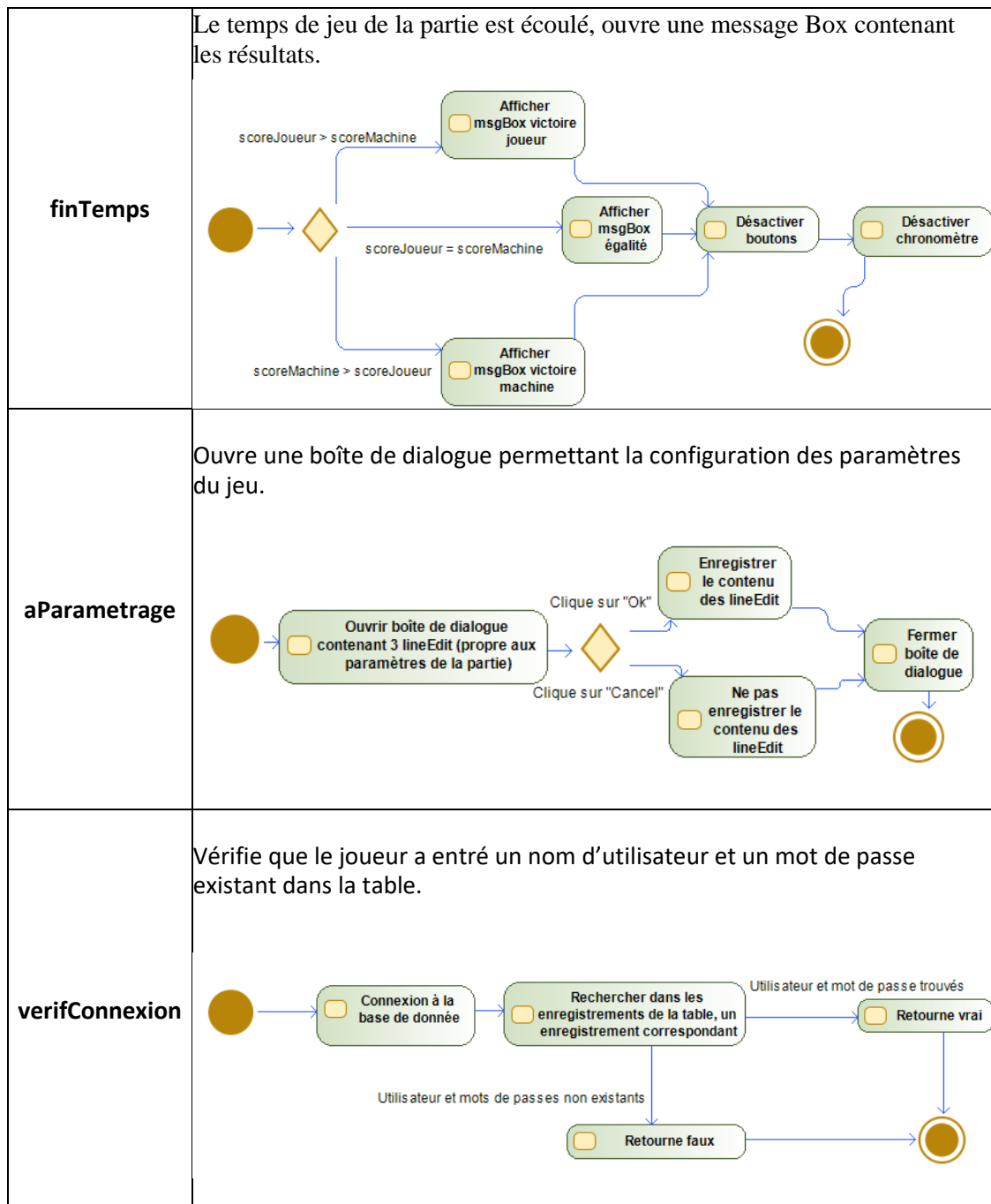
<i>nomEvénement</i>	<i>Signification</i>
clickPierre	Le joueur clique sur le bouton bPierre.
clickCiseau	Le joueur clique sur le bouton bCiseau.
clickPapier	Le joueur clique du le bouton bPapier.
clickNouvPart	Le joueur clique sur le bouton bNewGame.
clickPierre	Le joueur clique sur le bouton bPierre.
scoreReachMax	Le score du joueur ou de la machine atteint le score max fixé.
clickPause	Le joueur clique sur le bouton bPause.
tempsEcoule	Le compte à rebours atteint 0.
clickParametrage	Le joueur clique sur le menu actionParam_trer .
clickEntrer	Le joueur clique sur le bouton bConfirm.

### Description des actions réalisées lors de la traversée des transitions









(e) Préparation au codage :

**Table T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

- en ligne : les *événements* faisant changer le jeu d'état

- en colonne : les *états* du jeu

Éléments d'interface →	bCiseau	bPapier	bPierre	bNewGame	---	bPause	---	actionParam_trer
Événement → nomEtatJeu	clickCiseau	clickPapier	clickPierre	clickNouvPart	scoreReachMax	clickPause	tempsEcoule	clickParametrage
connexion	-	---	---	---	---	---	---	---
initialisation	---	---	---	partieEnCours/ activité1	---	---	---	initialisation /activité8
partieEnCours	partieEnCours/ activité2	partieEnCours/ activité3	partieEnCours/ activité4	partieEnCours/ activité1	partieEnCours/ activité5	partieEnPause/ activité6	finPartie/ activité7	---
finPartie	---	---	---	partieEnCours/ activité1	---	---	---	finPartie/ activité8
partieEnPause	---	---	---		---	partieEnCours/ activité6	---	---

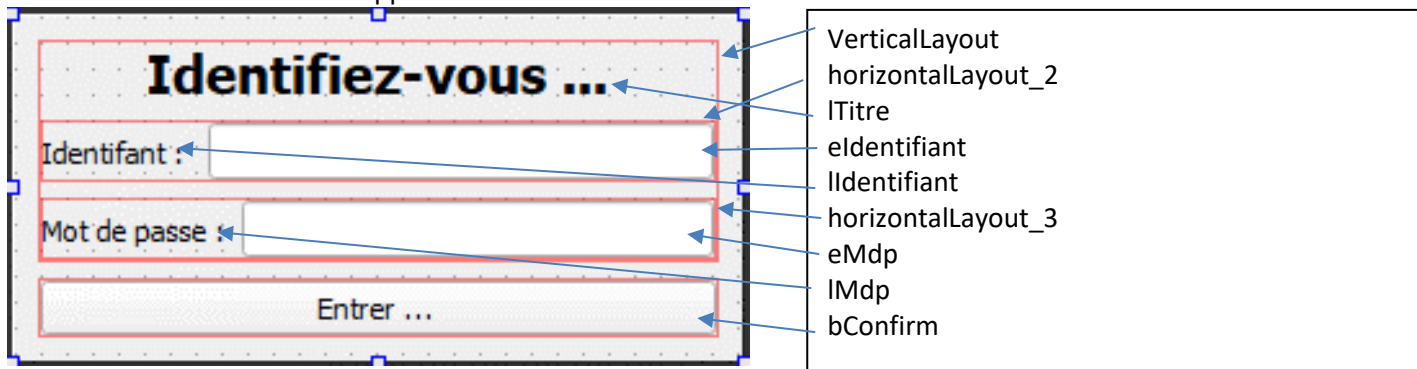
Éléments d'interface →	bConfirm
Événement → nomEtatJeu	clickEntrer
connexion	[identifiants_OK]initialisation/ activité9 [identifiants_NOK]connexion/ activité9
initialisation	---
partieEnCours	---
finPartie	---
partieEnPause	---

Tableau 13 : Matrice d'états-transitions du jeu chifoumi

## 18. Implémentations et tests

### 16.1 Nouveaux éléments d'interface

Une fenêtre de connexion à l'application.



### 16.2 Implémentation

.gitignore : permet de ne pas devoir enregistrer à chaque commit les fichiers non nécessaire à l'exécution du projet

ChifoumiInterface.pro : fichier permettant l'ouverture et exécution du projet QT

chifoumiVue.h : Module de la classe de la vue du chifoumiVue

chifoumiVue.cpp : Corps de la classe chifoumiVue

chifoumiInterface.ui : Interface graphique de la fenêtre du Chifoumi

main.cpp : Affichage de la fenêtre principale

images : Répertoire contenant les images des figures du Chifoumi

presentation.h : Module de la classe présentation

presentation.cpp : Sources de la classe présentation

chifoumi.cpp : Corps de la classe chifoumi

chifoumi.h : Module de la classe chifoumi

parametres.h : Module de la classe parametres qui ouvre une fenêtre de dialogue modale.

parametres.cpp : Corps de la classe parametres

database.h : Module de la classe database qui permet la connexion à la base de donnée

database.cpp : Corps de la classe database

connexion.h : Module de la classe connexion qui permet de vérifier l'authentification au jeu du Chifoumi

connexion.cpp : Corps de la classe connexion

connexion.ui : Interface graphique de la fenêtre de connexion

Les fichiers « connexion.h », « database.h » et ont été modifié pour la v7.

### 16.4 Tests

Méthode	Valeur rentrée	Valeur attendue	Valeur affichée	Commentaire
verifConnexion()	eIdentifiant = "invite" eMdp = invite	true	true	ok
verifConnexion()	eIdentifiant = erreur eMdp = erreur	false	false	ok