# Saé 2.01 – Développement d'une application

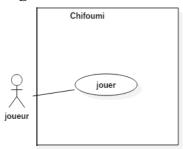
# Chifoumi – Dossier d'Analyse et conception

# 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.

#### 1

# 2. Diagramme des Cas d'Utilisation



2

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 ir	nitial.
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 sor
		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 n	ouvelle 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.

Chifoumi	_
-coupJoueur -coupMachine -scoreJoueur -scoreMachine	
+unsigned int getScoreJoueur() +unsigned int getScoreMachine() +UnCoup getCoupJoueur() +UnCoup getCoupMachine() +char determinerGagnant() +setCoupJoueur(UnCoup coup) +setCoupMachine(UnCoup coup) +setScoreJoueur(unsigned int score) +setScoreMachine(unsigned int score) +initCoups() +initScores() +majScores(char gagnant) -UnCoup genererUnCoup()	

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	unsigned	1
scoresoueur	partie courante	int	1
scoreMachine	Nbre total de points acquis par la machine durant la	unsigned	1
SCOLEMACHINE	partie courante	int	1
	Mémorise la dernière figure choisie par le joueur.		
coupJoueur	Type énuméré	UnCoup	papier
Coapoodedi	enum unCoup {pierre, ciseau, papier,	oncoup	papici
	rien};		
coupMachine	Mémorise la dernière figure choisie par la machine.	UnCoup	Ciseau

Tableau 2 : Dictionnaire des éléments - Classe Chifoumi

3

(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();
           /\star 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
                                   // dernier coup joué par le joueur
       UnCoup coupJoueur;
       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.

# Implémentation et tests

#### **Implémentation**

Liste des fichiers de cette version :

- chifoumi.h : Interface de la classe chifoumi
- chifoumi.cpp : Corps de la classe chifoumi
- main.cpp : Programme de test de la classe chifoumi

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

#### 5.2 Test

Méthodes	Resultat attendu	Résultat obtenu	Oracle
Méthode get associé	0	0	OK OK
au 'score'	V	V	OK
Méthode get associé	rien	rien	OK
au 'coup'			
Méthode	p_score	p_score	OK
setScoreJoueur	r	r —	
Méthode	P_score	p_score	OK
setScoreMachine			
Méthode	p_coup	p_coup	ОК
setCoupJoueur			
Méthode	p_coup	p_coup	ОК
setCoupMachine			
Méthode initScore	0	0	ОК
Méthode initCoups	rien	rien	OK
Méthode	p_coup	p_coup	OK
setCoupJoueur			
Méthode	p_coup	p_coup	OK
setCoupMachine			
Méthode	gagnantARetourner =	gagnantARetourner	OK
determinerGagnant	'N'	= 'N'	
(Joueur_pierre/			
Machine_pierre)			
Méthode	gagnantARetourner =	gagnantARetourner	OK
determinerGagnant	<i>'M'</i>	= 'M'	
(Joueur_pierre/			
Machine_papier)			
Méthode	gagnantARetourner =	gagnantARetourner	OK
determinerGagnant	<i>'J'</i>	= 'J'	
(Joueur_pierre/			
Machine_ciseau)			
3.67.1 1	(A.D. )	/4 P	OW
Méthode	gagnantARetourner =	gagnantARetourner	OK
determinerGagnant	<i>'J'</i>	= 'J'	
(Joueur_papier/			
Machine_pierre)			
Méthode	gagnantARetourner =	gagnantARetourner	OK
determinerGagnant	gagnaniAketourner = 'N'	gagnaniAkeiourner = 'N'	OA
(Joueur_papier/	14	1 ₹	
Machine_papier)			
тистик_раркет)			

Méthode determinerGagnant (Joueur_ciseau/ Machine_papier)	gagnantARetourner = 'J'	gagnantARetourner = 'J'	OK
Méthode determinerGagnant (Joueur_papier/ Machine_ciseau)	gagnantARetourner = 'M'	gagnantARetourner = 'M'	OK
Méthode determinerGagnant (Joueur_ciseau/ Machine_pierre)	gagnantARetourner = 'M'	gagnantARetourner = 'M'	OK
Méthode determinerGagnant (Joueur_ciseau/ Machine_ciseau)	gagnantARetourner = 'N'	gagnantARetourner = 'N'	ОК
Méthode majScores	ScoreJoueur>=0 et scoreMachine >=0	ScoreJoueur>=0 et scoreMachine >=0	OK

		Machine		
		Pierre	Papier	Ciseau
Joueur	Pierre	Égalité	Machine gagne	Joueur gagne
	Papier	Joueur gagne	Égalité	Machine gagne
	Ciseau	Machine gagne	Joueur gagne	Égalité

Test avec le programme fourni main.cpp

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

# Classe Chifoumi : Diagramme états-transitions

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

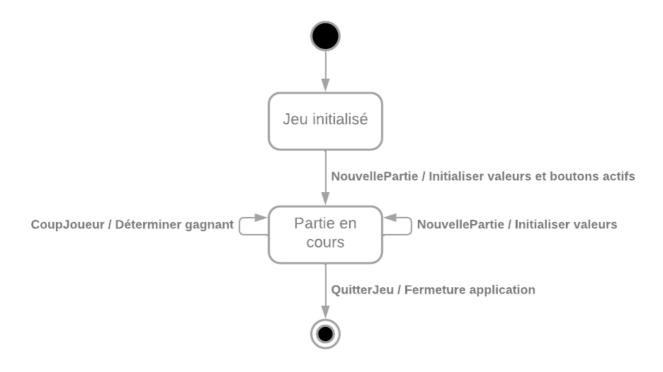


Figure 4: Diagramme états-transitions

#### Dictionnaires des états, événements et Actions

#### Dictionnaire des états du jeu

nomEtat	Signification
Jeu initialisé L'application attend que l'utilisateur lance la partie	
Partie en cours	L'utilisateur et la machine jouent tour à tour.

Tableau 3 : États du jeu

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

nomEvénement Signification		Signification
Nouvelle partie Le joueur appuie sur le bouton no		Le joueur appuie sur le bouton nouvelle partie et lance une nouvelle partie
	CoupJoueur	Le joueur choisit un signe

Tableau 4 : Evénements faisant changer le jeu d'état

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

Initialiser valeurs et boutons actifs	Les scores sont mis à 0 et les coups sont initialisés en rien. Les boutons sont activés et prêt à être sélectionné par le joueur.
Initialiser valeurs	Les scores sont mis à 0 et les coups sont initialisés en rien.
Déterminer gagnant	Afficher le coup du joueur. La machine génère un coup.

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

#### Préparation au codage :

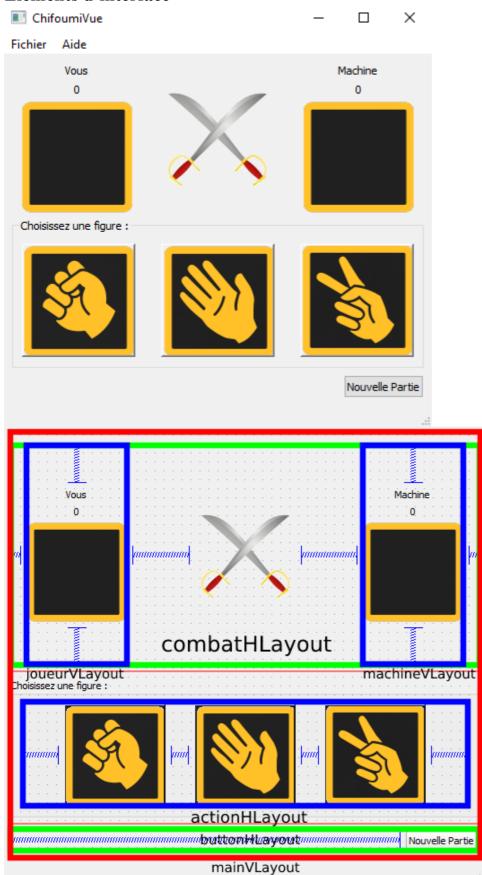
**Table T\_EtatsEvenementsJeu** correspondant à la version matricielle du diagramme états-transitions du jeu :

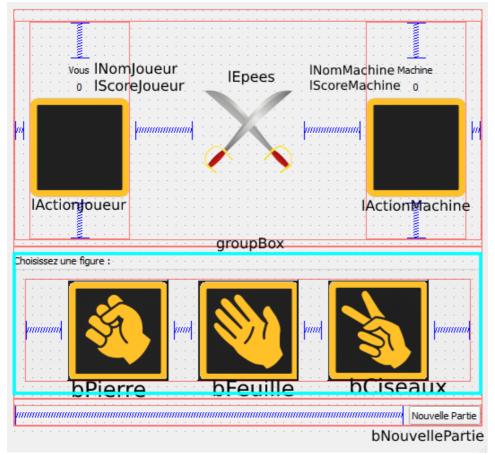
- en ligne : les événements faisant changer le jeu d'état
- en colonne : les états du jeu

Événement → nomEtatJeu	Nouvelle partie	CoupJoueur
Jeu initialisé	Partie en cours	
Partie en cours	Partie en cours	Partie en cours

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

# Éléments d'interface





L'interface est séparée en 2 parties :

- -Une partie qui affiche l'état de la partie
- -L'autre partie permet au joueur d'effectuer une action ou de lancer une nouvelle partie

La partie du haut est elle-même séparée en 2 parties, la partie joueur et la partie machine.

Dans chacune de ces parties on peut voir l'action jouée par le joueur et la machine ainsi que leurs scores.

La partie du bas est constituée de boutons d'actions (pierre, feuille et ciseaux), une fois que le joueur clique sur un de ces boutons, l'action décrite sur l'image est exécutée.

Le bouton nouvelle partie permet de remettre les scores à 0.

### Implémentation et tests

#### **Implémentation**

Chifoumi.h interface du modèle du jeu du chifoumi founi en ressource Chifoumi.cpp corps du modèle du jeu du chifoumi fourni en ressource Chifoumivue.h interface de la vue de l'application Chifoumivue.cpp corps de la vue de l'application Main.cpp programme qui initialise le modèle et la vue du chifoumi

#### 8.2 Test

Méthodes	Resultat attendu	Résultat obtenu	Oracle
bPierre	Affichage du signe dans	Affichage du signe dans	ОК
	lActionJoueur + action de	lActionJoueur + action de	
	la machine + gagnant	la machine + gagnant	
	déterminé + actualisation	déterminé + actualisation	
	des scores	des scores	
bCiseau	Affichage du signe dans	Affichage du signe dans	ОК
	lActionJoueur + action de	lActionJoueur + action de	
	la machine + gagnant	la machine + gagnant	
	déterminé + actualisation	déterminé + actualisation	
	des scores	des scores	
bFeuille	Affichage du signe dans	Affichage du signe dans	ОК
	lActionJoueur + action de	lActionJoueur + action de	
	la machine + gagnant	la machine + gagnant	
	déterminé + actualisation	déterminé + actualisation	
	des scores	des scores	
bNouvellePartie	Remise à zéro des scores	Remise à zéro des scores	ОК
	et des signes +	et des signes +	
	actualisation de	actualisation de	
	l'interface	l'interface	

# Version v2

Liste des fichiers sources de cette version (et rôle de chacun)

Chifoumi.h interface du modèle du jeu du chifoumi fourni en ressource

Chifoumipresentation.h interface de la présentation qui permet de gérer la communication entre la vue et le modèle Chifoumivue.h interface de la vue de l'application

Chifoumi.cpp corps du modèle du jeu du chifoumi de l'application

Chifoumipresentation.cpp corps de la présentation de l'application

Chifoumivue.cpp corps de la vue de l'application

Main.cpp programme qui initialise le modèle et la vue du chifoumi

Même tests que la version 1.

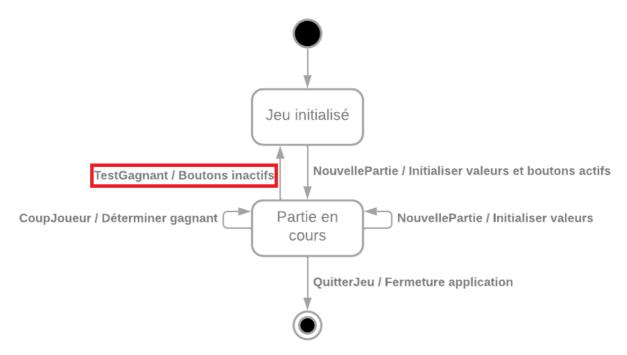
Chifoumi.h interface du modèle du jeu du chifoumi founi en ressource Chifoumi.cpp corps du modèle du jeu du chifoumi fourni en ressource Chifoumivue.h interface de la vue de l'application Chifoumivue.cpp corps de la vue de l'application Main.cpp programme qui initialise le modèle et la vue du chifoumi

Pour la mise en œuvre de la v3 le seul fichier .h modifié est le fichier Chifoumivue.h

#### Tests:

Lorsqu'on clique sur Menu-->Aide le messageBox s'affiche correctement. Lorsqu'on clique sur Fichier-->Quitter le programme se ferme. Lorsque le programme

Méthodes	Resultat attendu	Résultat obtenu	Oracle
actionA_propos_de	Affichage d'une boite de message disant "Version 3 Programme modèle de Campistron - Doyenhard – Sokhna	Affichage d'une boite de message disant "Version 3 Programme modèle de Campistron - Doyenhard – Sokhna	OK
action_Quitter	Fermeture de l'application	Fermeture de l'application	OK



#### Dictionnaires des états, événements et Actions

Dictionnaire des états du jeu

_	iculominum c aco cumo au,	jeu
	nomEtat	Signification
	Jeu initialisé	L'application attend que l'utilisateur lance la partie
-	Partie en cours	L'utilisateur et la machine jouent tour à tour.

Dictionnaire des événements faisant changer le jeu d'état

nomEvénement	Signification
Nouvelle partie	Le joueur appuie sur le bouton nouvelle partie et lance une nouvelle partie
CoupJoueur	Le joueur choisit un signe
QuitterJeu	Le joueur appuie sur le bouton quitter pour fermer l'application
TestGagnant	Test après chaque tour pour voir si un des deux joueurs a gagné. Si c'est le cas on retourne à l'état initial

Description des actions réalisées lors de la traversée des transitions

Boutons inactifs	Les boutons deviennent inactifs.
Fermeture application	L'application du chifoumi se ferme.
D <b>é</b> terminer gagnant	Afficher le coup du joueur.  La machine génère un coup.  Afficher le coup de la machine  On compare le coup du joueur et de la machine.  S'il y a un vainqueur on incrémente son score.
Initialiser valeurs	Les scores sont mis à 0 et les coups sont initialisés en rien.
Initialiser valeurs et boutons actifs	Les scores sont mis à 0 et les coups sont initialisés en rien. Les boutons sont activés et prêt à être sélectionné par le joueur.

#### 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 colollic i les <b>cius</b> aa	J = 4			
	Événement <del>-&gt;</del> nomEtatJeu	Nouvelle partie	CoupJoueur	QuitterJeu	TestGagnant
	Jeu initialisé	Partie en cours			
	Partie en cours	Partie en cours	Partie en cours		Jeu initialisé

#### Éléments d'interface

# Fichier Aide



Ajout d'un indicateur du score à atteindre pour gagner la partie.

IScoreMax s'actualise selon la va leur de scoreMax déclarée dans chifoumi.cpp

Liste des fichiers sources de cette version (et rôle de chacun)

Chifoumi.h interface du modèle du jeu du chifoumi fourni en ressource

Chifoumipresentation.h interface de la présentation qui permet de gérer la communication entre la vue et le modèle Chifoumivue.h interface de la vue de l'application

Chifoumi.cpp corps du modèle du jeu du chifoumi de l'application

Chifoumipresentation.cpp corps de la présentation de l'application

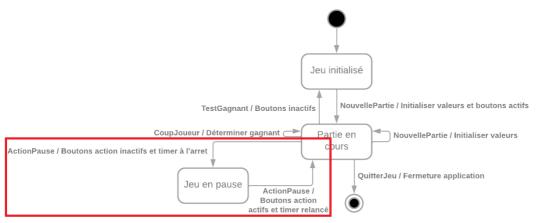
Chifoumivue.cpp corps de la vue de l'application

Main.cpp programme qui initialise le modèle et la vue du chifoumi

Les 3 fichiers .h ont été modifiés pour implémenter le système de score max.

#### Tests:

Méthodes	Resultat attendu	Résultat obtenu	Oracle
TestGagnant	Affichage d'une boîte de dialogue	Affichage d'une boîte de dialogue	OK
Avec scoreJoueur =	disant "Bravo Vous! Vous gagnez	disant "Bravo Vous! Vous gagnez	
scoreMax	avec 5 points."	avec 5 points."	
	Une fois la boîte de dialogue fermée	Une fois la boîte de dialogue fermée	
	le jeu passe en EtatInitial et donc les	le jeu passe en EtatInitial et donc les	
	boutons sont désactivés	boutons sont désactivés	
TestGagnant Avec	Affichage d'une boîte de dialogue	Affichage d'une boîte de dialogue	OK
scoreMachine =	disant "Bravo La Machine! Vous	disant "Bravo La Machine! Vous	
scoreMax	gagnez avec 5 points."	gagnez avec 5 points."	
	Une fois la boîte de dialogue fermée	Une fois la boîte de dialogue fermée	
	le jeu passe en EtatInitial et donc les	le jeu passe en EtatInitial et donc les	
	boutons sont désactivés	boutons sont désactivés	



# Dictionnaires des états, événements et Actions

Dictionnaire des états du jeu

nomEtat	Signification
Jeu initialisé L'application attend que l'utilisateur lance la partie	
Partie en cours	L'utilisateur et la machine jouent tour à tour.
Jeu en pause	La partie est en pause les boutons d'action sont inactifs et le timer en pause

Dictionnaire des événements faisant changer le jeu d'état

nomEvénement	Signification	
Nouvelle partie	Le joueur appuie sur le bouton nouvelle partie et lance une nouvelle partie	
CoupJoueur	Le joueur choisit un signe	
QuitterJeu	Le joueur appuie sur le bouton quitter pour fermer l'application	
TestGagnant	Test après chaque tour pour voir si un des deux joueurs a gagné. Si c'est le cas on retourne à l'état initial	
ActionPause	Le joueur appuie sur le bouton pause pour mettre la partie en pause	

Description des actions réalisées lors de la traversée des transitions

	Les boutons pierre feuille ciseaux sont de nouveau actifs et le timer est en marche
Boutons action inactifs et timer à l'arrêt	Les boutons pierre feuille ciseaux sont inactifs et le timer est en pause
Boutons inactifs	Les boutons deviennent inactifs.
Fermeture application	L'application du chifoumi se ferme.
D <b>é</b> terminer gagnant	Afficher le coup de la machine On compare le coup du joueur et de la machine. S'il y a un vainqueur on incrémente son score.
	Afficher le coup du joueur.  La machine génère un coup.
Initialiser valeurs	Les scores sont mis à 0 et les coups sont initialisés en rien.
Initialiser valeurs et boutons actifs	Les scores sont mis à 0 et les coups sont initialisés en rien. Les boutons sont activés et prêt à être sélectionné par le joueur.

#### 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

	Nouvelle	Coup	QuitterJeu	TestGagnant	ActionPause
nomEtatJeu	partie	Joueur			
Jeu initialisé	Partie en				
	cours				
Partie en cours	Partie en	Partie en		Jeu initialisé	Jeu en pause
	cours	cours			
Jeu en pause	Partie en				Partie en
	cours				cours

#### Éléments d'interface





lTempsRestant affiche le temps restant avant la fin de la partie bPause permet de mettre le jeu en pause ou de le relancer

Liste des fichiers sources de cette version (et rôle de chacun)

Chifoumi.h interface du modèle du jeu du chifoumi fourni en ressource

Chifoumipresentation.h interface de la présentation qui permet de gérer la communication entre la vue et le modèle Chifoumivue.h interface de la vue de l'application

Chifoumi.cpp corps du modèle du jeu du chifoumi de l'application

Chifoumipresentation.cpp corps de la présentation de l'application

Chifoumivue.cpp corps de la vue de l'application

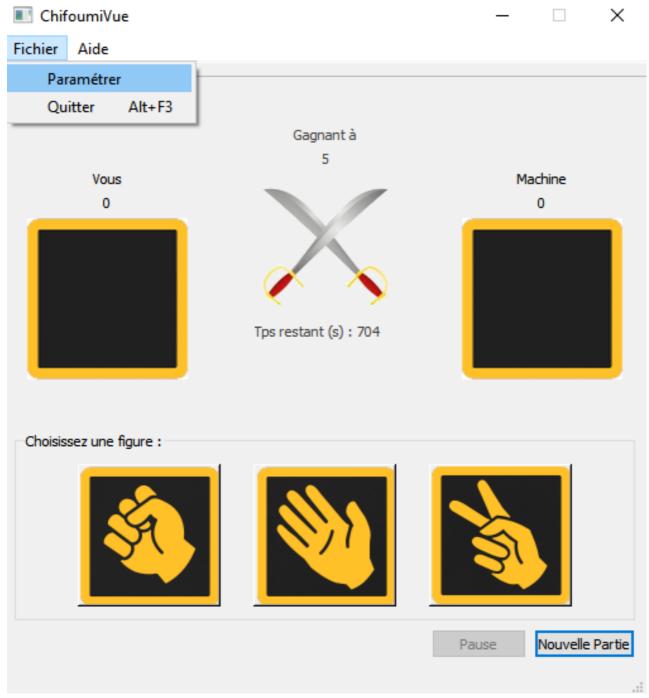
Main.cpp programme qui initialise le modèle et la vue du chifoumi

Chifoumivue et chifoumipresentation ont été modifiés.

Méthodes	Resultat attendu	Résultat obtenu	Oracle
ActionPause quand	Le timer s'arrête et les boutons	Le timer s'arrête et les boutons	OK
etat =	d'actions deviennent inactifs	d'actions deviennent inactifs	
partieEnCours			
ActionPause quand	Le timer reprend et les boutons	Le timer reprend et les boutons	OK
etat = pause	d'actions deviennent actifs	d'actions deviennent actifs	
TestVainqueur	Affichage d'une boîte de dialogue	Affichage d'une boîte de dialogue	OK
quand tempsRestant	disant "Hélas chers joueurs, temps	disant "Hélas chers joueurs, temps	
<= 0 <i>et que</i>	de jeu fini! Vous terminez toutefois	de jeu fini! Vous terminez toutefois	
scoreJoueur >	mieux, avec X points"	mieux, avec X points."	
scoreMachine	Une fois la boîte de dialogue fermée	Une fois la boîte de dialogue fermée	
	le jeu passe en EtatInitial et donc les	le jeu passe en EtatInitial et donc les	
	boutons sont désactivés	boutons sont désactivés	
TestVainqueur	Affichage d'une boîte de dialogue	Affichage d'une boîte de dialogue	OK
quand tempsRestant	disant "Hélas chers joueurs, temps	disant "Hélas chers joueurs, temps	
<= 0 <i>et que</i>	de jeu fini! La Machine termine	de jeu fini! La Machine termine	
scoreJoueur <	toutefois mieux, avec X points."	toutefois mieux, avec X points."	
scoreMachine	Une fois la boîte de dialogue fermée	Une fois la boîte de dialogue fermée	
	le jeu passe en EtatInitial et donc les	le jeu passe en EtatInitial et donc les	
	boutons sont désactivés	boutons sont désactivés	
TestVainqueur	Affichage d'une boîte de dialogue	Affichage d'une boîte de dialogue	OK
quand tempsRestant	disant "Hélas chers joueurs, temps	disant "Hélas chers joueurs, temps	
<= 0 <i>et que</i>	de jeu fini! Vous êtes à égalité."	de jeu fini! Vous êtes à égalité."	
scoreJoueur =	Une fois la boîte de dialogue fermée	Une fois la boîte de dialogue fermée	
scoreMachine	le jeu passe en EtatInitial et donc les	le jeu passe en EtatInitial et donc les	
	boutons sont désactivés	boutons sont désactivés	

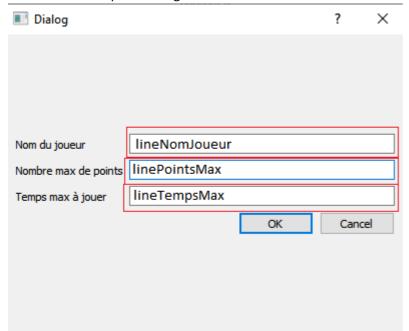
Pas de changement du diagramme d'état transition

#### Éléments d'interface



Ajout d'un bouton paramétrer dans le menu fichier.

#### Voici le menu de paramétrage :



LineNomJoueur permet d'insérer le nom du joueur LinePointsMax permet d'insérer le nombre de points maximum à atteindre LineTempsMax permet d'insérer la durée d'une partie

Liste des fichiers sources de cette version (et rôle de chacun)

Chifoumi.h interface du modèle du jeu du chifoumi fourni en ressource

Chifoumipresentation.h interface de la présentation qui permet de gérer la communication entre la vue et le modèle Chifoumivue.h interface de la vue de l'application

### ConfigPartie.h interface de la fenêtre de configuration de partie

Chifoumi.cpp corps du modèle du jeu du chifoumi de l'application Chifoumipresentation.cpp corps de la présentation de l'application Chifoumivue.cpp corps de la vue de l'application

#### ConfigPartie.cpp corps de la fenêtre de configuration de partie

Main.cpp programme qui initialise le modèle et la vue du chifoumi

#### Tests:

Méthodes	Resultat attendu	Résultat obtenu	Oracle
getNomJoueur	Chaîne de caractère "Vous"	Chaîne de caractère "Vous"	OK
getScoreMax	10	10	OK
getTempsMax	60	60	OK
configurerPartie	Les valeurs nomJoueur, scoreMax et	Les valeurs nomJoueur, scoreMax et	OK
	dureePartie se mettent à jour à	dureePartie se mettent à jour à	
	partir des champs dédiés à remplir.	partir des champs dédiés à remplir.	