Saé 2.01 – Développement d'une application

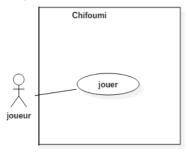
Chifoumi – Dossier d'Analyse et conception

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.

1.1

2. Diagramme des Cas d'Utilisation



1.2

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 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 soi
		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

1.3

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
-coupJou -coupMa -scoreJou -scoreMa	chine ueur
+unsigne +UnCoup +UnCoup +char de +setCoup +setScor +setScor +initCoup +initScor +majScor	· ·

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
scoreoodedi	partie courante	int	1
ScoreMachine Nbre total de points acquis par la machine durant la		unsigned	1
SCOTEMACHTHE	partie courante	int	1
	Mémorise la dernière figure choisie par le joueur.		
coupJoueur	Type énuméré	UnCoup	papier
	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

(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.

1.3.1

Version v0

5. Implémentation et tests

5.1 Implémentation

Liste des fichiers de cette version :

- chifoumi.h : Header la classe Chifoumi
- chifoumi.cpp : Sources de la classe Chifoumi
- main.cpp : Corps du programme de tests
- 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)

Données en entrée	Valeur attendue	Valeur résultante	Description
Coup joueur Pierre	Score joueur + 1 et score	Score joueur + 1 et score	Victoire
Coup machine Ciseau	machine	machine	
Coup joueur Pierre	Score joueur et score	Score joueur et score	Égalité
Coup machine Pierre	machine	machine	
Coup joueur Pierre	Score joueur et score	Score joueur et score	Défaite
Coup machine Papier	machine + 1	machine + 1	
Coup joueur Papier	Score joueur + 1 et score	Score joueur + 1 et score	Victoire
Coup machine Pierre	machine	machine	
Coup joueur Papier	Score joueur et score	Score joueur et score	Égalité
Coup machine Papier	machine	machine	
Coup joueur Papier	Score joueur et score	Score joueur et score	Défaite
Coup machine Ciseau	machine + 1	machine + 1	
Coup joueur Ciseau	Score joueur + 1 et score	Score joueur + 1 et score	Victoire
Coup machine Papier	machine	machine	
Coup joueur Ciseau	Score joueur et score	Score joueur et score	Égalité
Coup machine Ciseau	machine	machine	
Coup joueur Ciseau	Score joueur et score	Score joueur et score	Défaite
Coup machine Pierre	machine + 1	machine + 1	

Version v1

6. Classe Chifoumi : Diagramme états-transitions

(a) Diagramme états-transitions -actions du jeu

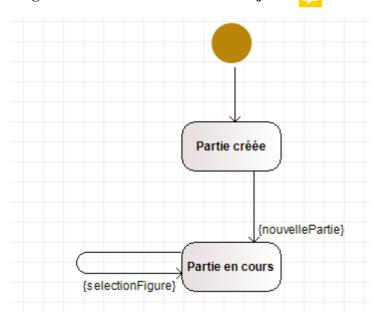


Figure 4 : Diagramme états-transitions

(b) Dictionnaires des états, événements et Actions

Dictionnaire des états du jeu

nomEtat	Signification
Partie créée	État initial de la création du jeu
Partie en cours	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

Dienoman e des evenements laisant changer le jeu a etat		
nomEvénement	Signification	
nouvellePartie	Le joueur clique sur le bouton pour lancer une nouvelle partie	
selectionFigure	Le joueur choisit une figure	

Tableau 4 : Evénements faisant changer le jeu d'état

Description des actions réalisées lors de la traversée des transitions



ouverturePgm	L'utilisateur ouvre le programme
appuyerBoutonNouvPart	L'utilisateur appuie sur le bouton de nouvelle partie
choisirFigure	L'utilisateur sélectionne une figure

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

Événement → nomEtatJeu	nouvellePartie	selectionFigure
Partie créée	Partie en cours	
Partie en cours		Partie en cours

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

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