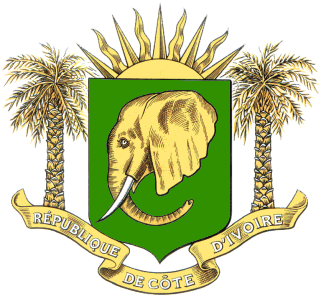
|  |
| --- |
| Ministère de l’Enseignement Supérieur et de la Recherche Scientifique |
| http://www.univ-fhb.edu.ci/fr/templates/shaper_pheonix/images/styles/style1/logo.png Université Félix Houphouët-Boigny Cocody |

République de Côte d’Ivoire



Union–Discipline–Travail

UFR de Mathématiques et Informatique



LICENCE 3

2016-2017

PROJET

Fondamentaux des réseaux et Génie Logiciel

**THEME :**

Logiciel permettant de simuler le fonctionnement d’un réseau informatique

**PROFESSEUR :**

Dr DIALLO

**LISTE DES ETUDIANTS :**

ADJEI N'GUESSAN ROMARIC

BOBIA DOGO LAMBERT

BONY YANN CEDRIC

ESSOH LATH SERGES CLOVIS

N'GBLA ELVIS EUDES

TRAORE LAMTOU KEVIN

Historique des révisions

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Auteur** |
| 01/02/2018 | 1.0 | Création du document | ADJEI N’GUESSAN ROMARIC  BOBIA DOGO LAMBERT |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table des matières

1. Introduction 4

1.1 Objectif 4

1.2 Portée 4

1.3 Références 4

2. Vue d’ensemble du projet 4

2.1 But du projet, portée, et objectifs 4

2.2 Hypothèses et contraintes 4

2.2.1 Hypothèses 4

2.2.2 Contraintes 5

2.3 Fournitures du projet 5

2.4 Evolution du plan de développement logiciel 5

3. Organisation de projet 5

3.1 Structure de l’organisation 5

3.2 Interfaces externes 5

3.3 Rôles et responsabilités 6

4. Processus de gestion 7

4.1 Estimations de projet 7

4.2 Plan de projet 7

4.2.1 Plan de Phase 7

4.2.2 Objectifs des incrémentations 7

4.2.3 Livraisons 9

5. Documentation 11

5.1 Dossier code source 11

5.2 Manuel Utilisateur 11

5.3 Conception architectural 11

5.4 Plan Qualité 11

5.5 Plan de Test 11

5.6 Calendrier de projet 11

Plan de Développement Logiciel

# Introduction

Le projet en troisième année de LICENCE informatique est important. Le choix du sujet ainsi que l’équipe de projet sont aussi importants. Outre le fait que le projet constitue une note importante pour notre moyenne du semestre, ce projet peut être un plus pour notre avenir professionnel, ou même pour notre stage. Le choix du sujet a été rapide, comme une évidence pour nous. Nous avons choisi de réaliser l’application permettant de simuler le fonctionnement d’un réseau informatique à commutation de paquets. De notre point de vue, ce projet présente deux caractéristiques fondamentales : d’une part la nouveauté du logiciel avec lequel l’application va être réalisée et d’autre part en tant que futur informaticien élargir son champ de compétences avec ce nouveau langage est un plus indéniable sur une carte de visite. L’équipe de projet est constituée de quatre membres ayant suivis tous les quatre la coloration génie logiciel au cours du semestre 1.

À partir de ce constat, nous allons donc essayer de réaliser une application fonctionnelle, remplissant les conditions imposées par le cahier des charges établi en collaboration avec notre principal utilisateur. La nouveauté réside alors dans la manière de concevoir, de développer, l’application sera donc développée à l’aide de code block de développement du progiciel de C. Nous allons donc voir à travers ce rapport dans une première partie, une présentation du projet ainsi que ces principaux objectifs. Puis dans une seconde partie, quelles méthodes avons-nous utilisées afin de mettre en œuvre le

projet, son élaboration ? Enfin, dans une dernière partie, nous verrons les résultats

obtenus ainsi que les évolutions possibles du projet et plus particulièrement de

l’application.

## Objectif

Le document permet de réunir toutes les informations nécessaires au contrôle du projet. Il décrit l’approche du développement du logiciel et est un plan haut niveau généré et utilisé par les gestionnaires pour diriger l’effort de développement.

L’équipe de projet en accord avec l’enseignant tuteur, s’est donc accordée à remplir les objectifs suivants :

Au préalable, avant de commencer le développement en lui-même, il nous a fallu nous documenter et « apprivoiser » l’environnement de travail, c'est-à-dire code block et sdl. Ensuite, deuxième étape du projet, il faut recréer Les interfaces. Pour ce faire, nous allons utiliser le logiciel sdl. Troisième objectif, construire les fonctionnalités d’origine de l’application code block et sdl , ce qui signifie la création de formulaires, rapports, etc., et améliorer l’interface graphique ainsi que les fonctions qui le permettent. Enfin, dernier objectif proposé est la gestion des utilisateurs. En effet, permettre la vue de certaines informations suivant que l’on soit enseignant, étudiant ou administrateur.

## Portée

Les personnes suivantes sont concernées par le plan de développement logiciel :

L’enseignant Dr Diallo, il lui permet de visualiser les prévisions (charges, délais) du développement logiciel.

Membre de l’équipe

Les autres étudiants

Le chef de projet utilise ce document pour planifier le calendrier de projet et les ressources nécessaires et aussi suivre l’avancement du projet dans le temps.

Les membres de l’équipe projet doivent prendre connaissance du document pour comprendre ce qu’ils ont à faire, quand le faire, et de quelles autres activités ils dépendent.

## Références

* ETUDIANT Licence 3 Sciences Informatiques
* UFR MATH\_INFOS Université FHB de COCODY (Abidjan COTE D’IVOIRE)
* Dr DIALLO

# Vue d’ensemble du projet

## But du projet, portée, et objectifs

Le but du projet est d’obtenir un outil permettant de configurer et de gérer l’exécution d’un processus dans le cadre d’un projet. Cet outil se présentera sous la forme d’un éditeur de données du projet. De plus, l’outil permettra de créer un logiciel permettant de simuler le fonctionnement d’un réseau informatique à commutation de paquet.

L’objectif principal du projet est le développement de l’application décrite précédemment par le groupe pour la mise en pratique la formation théorie que nous avons reçue tout au long de notre année académique 2016-2017.

## Hypothèses et contraintes

Plusieurs hypothèses et contraintes ont été posées pour gérer et cadrer le mieux possible le processus de développement.

### Hypothèses

Pour le déroulement du projet Fondamentaux des réseaux et Génie Logiciel L3 SI 2017, plusieurs hypothèses ont été émises :

L’équipe de développeurs connaît les langages utilisés. Les analystes possèdent les connaissances nécessaires en matière de gestion de projet et méthodologie de développement logiciel.

Le nombre de fonctionnalités à implémenter peut être réduit : les cas d’utilisations les moins prioritaires feront l’objet des dernières itérations de la phase de construction. Dans le cas où un retard serait constaté au cours du développement, l’annulation des derniers cas d’utilisation à coder serait proposée aux superviseurs du projet.

Un exécutable doit être créé afin de pouvoir utiliser le logiciel à partir de n’importe quel système d’exploitation.

### Contraintes

L’échéance du projet se trouve avant la fin du mois de Février.

Le groupe jouera le rôle d’analystes et gèrera le projet. L’équipe de développeurs sera constituée de tous les membres du groupe.

Le format des modèles de processus fourni doit être fixé en accord avec les équipes travaillant sur l’outil permettant de les réaliser.

Contraintes liées au codage : l’intégralité du logiciel doit être développée en utilisant le code block.

Contraintes liées à la disponibilité du matériel : le développement nécessite une machine possédant un logiciel code block et sdl.

## Fournitures du projet

A définir

## Evolution du plan de développement logiciel

Le Plan de Développement Logiciel sera mis à jour à chaque incrémentation, pour adapter ce document à l’avancement du projet.

# Organisation de projet

## Structure de l’organisation

Tous les membres du groupe d’étudiants de troisième année travaille sur l’ensemble du projet (gestion du projet, analyse et développement).

## Rôles et responsabilités

| Analystes | Analyste | ADJEI N'GUESSAN ROMARIC  ESSOH LATH SERGES CLOVIS  N'GBLA ELVIS EUDES  TRAORE LAMTOU KEVIN  BOBIA DOGO LAMBERT  BONY YANN CEDRIC  CED |
| --- | --- | --- |
| Développeurs | Développeurs | ADJEI N'GUESSAN ROMARIC  ESSOH LATH SERGES CLOVIS  N'GBLA ELVIS EUDES  TRAORE LAMTOU KEVIN  BOBIA DOGO LAMBERT  BONI YANN CEDRICC  LAMBER  N'GBLA ELVIS EUDES  TRAORE LAMTOU KEVIN |
| Gestionnaires | Chef de projet | N'GBLA ELVIS EUDES |
| Superviseur de projet | ADJEI N'GUESSAN ROMARIC  BOBIA DOGO LAMBERT |
| Autres | Spécialiste outils | ESSOH LATH SERGES CLOVIS  TRAORE LAMTOU KEVIN  BONY YANN CEDRIC |

# Processus de gestion

## Estimations de projet

La durée du projet est estimée à 2 mois. L’échéance du projet devant se situer avant la fin du mois de Février.

## Plan de projet

### Plan de Phase

Le projet suit un développement incrémental, et à chaque phase du processus de développement peuvent correspondre une ou plusieurs itérations. Le tableau ci-dessous présente le nombre d’itérations pour que nous avons défini pour chaque phase ainsi que les dates. L’objectif des différentes itérations sera détaillé dans une partie suivante du document.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phase** | **Incrémentations** | **Début** | **Fin** | **Durée** |
| **Préparation** | / | 26/12/2017 | 27/12/2017 | 2 jours |
| **Lancement** | IN0 | 26/12/2017 | 27/12/2017 | 3 semaines |
| **Elaboration** | IN1 | //2018 | //2018 |  |
| IN2 | //2018 | //2018 |  |
| **Construction** |  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| **Transition** |  |  |  |  |

### Objectifs des incrémentations

* **Phase de lancement**
  + *Incrémentation 0*

L’objectif de cette incrémentation est d’établir une vision globale du projet approuvée par les différents intervenants, qui décrivent les besoins utilisateur ainsi que les limites du logiciel. Cet accord entre les intervenants est concrétisé par la rédaction du document Vision, qui contient également les fonctionnalités essentielles de l’application.

Lors de l’incrémentation 0 l’équipe d’analyse doit également identifier les cas d’utilisation, déduits des fonctionnalités essentielles du produit. La liste de ces cas est conservée dans le modèle des cas d’utilisation, ainsi que les spécifications supplémentaires.

L’activité de gestion du projet tient une place importante dans l’incrémentation 0 : un calendrier du projet y est défini, ainsi que les objectifs des différentes itérations (document Plan de Développement Logiciel). Les premiers risques sont identifiés et une stratégie d’évitement envisagée dans la liste des risques.

Par ailleurs, la gestion du processus est mise en place afin d’adapter le processus de développement suivi, aux caractéristiques de notre projet. Les plans types des différents documents à produire sont également adaptés au projet, une présentation homogène est définie.

Enfin, une première étude des outils utilisés (code block et sdl) est réalisée (Guide des procédures) afin de permettre par la suite l’installation et la configuration rapide de ces outils.

* + - * **Phase d’élaboration**

La phase d’élaboration est orientée sur la définition d’une architecture stable du système : les documents produits en phase d’élaboration sont essentiels et servent de base pour l’implémentation du logiciel.

Par ailleurs, une maquette produite au début de cette phase (format papier) présente les écrans principaux de l’application, et permet d’obtenir l’approbation des clients sur l’interface homme machine du logiciel. Le prototype de l’application est également réalisé à la suite de cette maquette.

La gestion du projet continue lors de cette phase avec un éventuel ajustement des objectifs, et l’adaptation de la liste des risques.

* + *Incrémentation 1*

Les cas d’utilisation principaux de l’application doivent être spécifiés (modèle des cas d’utilisation). Une première architecture sera définie, et les composants choisis et affinés (modèle de conception). Le format des données en entrée du logiciel (modèle de processus associé au projet) devra être figé en accord avec les autres groupes participant au projet.

Les spécialistes IHM définissent l’interface de l’outil à travers une maquette au format papier. Une partie du prototype sera développée, et pourra implémenter les premiers cas d’utilisation, suivant l’avancement de la définition de l’architecture.

* + *Incémentation 2*

L’architecture pourra être revue et affinée lors de cette incrémentation, et les cas d’utilisation seront totalement spécifiés. Le prototype implémentera de nouveaux cas d’utilisation, et les éléments de base du logiciel seront codés.

* + - * **Phase de construction**

La phase de construction consiste en l’implémentation de l’application, qui s’appuie sur l’architecture définie précédemment. Cette phase est incrémental : le produit de chaque incrémentation sera une version utilisable du logiciel, même si incomplète, qui sera mise entre les mains du client pour s’assurer que l’outil réalisé est conforme aux besoins. La fin de la phase de construction est marquée par une revue lors de laquelle le client détermine si l’application peut être présentée aux utilisateurs.

Durant cette phase, la gestion du projet sera maintenue, avec une vigilance concernant les risques, et l’avancement du projet.

* + *Incrémentation 3*

Lors de cette incrémentation le développement de l’outil débutera à partir du prototype réalisé précédemment, et il s’agira d’implémenter les cas d’utilisation non encore implémentés. Et de réaliser les tests unitaires des fonctions codées.

* + *Incrémentation 4*

L’incrémentation 4 sera la continuité de l’incrémentation 3 au niveau de l’implémentation des cas d’utilisation du système. Des modifications seront éventuellement effectuées en fonction des remarques client formulées à l’issue de l’itération 3.

* + *Incrémentation 5*

Les cas d’utilisation les moins prioritaires seront codés lors de cette itération, selon l’avancement du projet. Le manuel utilisateur et l’aide en ligne du logiciel devront être complétés de façon à préparer la livraison du produit.

* + - * **Phase de construction**
  + *Incrémentation 6*

Cette incrémentation est caractérisée par la mise à disposition des intervenants d’une première version finale du logiciel . L’équipe projet procèdera alors à la correction des derniers bugs et effectuera des modifications mineures selon les remarques formulées par les intervenants.

Après la validation de l’outil, la version de référence du logiciel sera déployée, les documents du projet finalisés (notamment le manuel utilisateur) seront livrés.

### Livraisons

Le développement mis en place sur le projet est incrémental, une version utilisable (mais non complète) du logiciel sera mise entre les mains de l’enseignant à chaque fin d’incrémentation. L’incrémentation 0 ne fera pas l’objet d’une livraison de l’outil, puisqu’elle consiste à établir la vision du logiciel.

#### Plan de composition de l’équipe

Les équipes du projet et leur organisation ont été décrites précédemment dans ce document (partie « Organisation du projet ») : un groupe d’étudiants de troisième année.

Les étudiants de troisième année ont pour objectif la gestion du projet, l’analyse ainsi que la participation au développement.

#### Plan de formation

L’équipe de développement devra maîtriser rapidement les outils et méthodes de développement utilisés pour le développement du logiciel : le langage C, l’éditeur code block et sdl. Pour cela des formations peuvent être mises en œuvre et réalisées par les membres du projet.

Etant donné que les développeurs ont reçu une formation dans le cadre des enseignements, il n’y aura pas de formation complémentaire à ce sujet. Cependant, le groupe n’ayant pas de réelles compétences en sdl (pas d’enseignement reçu), ils devront s’auto former grâce à des documents.

Par ailleurs, le spécialiste outils réalisera un guide de procédures décrivant l’installation de code block et sdl et leur configuration pour le projet ainsi que l’utilisation de l’outil de gestion des sources intégré à cet éditeur.

Des documents concernant les outils et méthodes utilisés seront mis à disposition des membres de l’équipe de développement, ainsi que des liens vers des documentations en ligne.

# Documentation

## Dossier code source

CREATION DE BIBLIOTHETE(zonetext.h)

#ifndef ZONETEXTE\_H\_INCLUDED

#define ZONETEXTE\_H\_INCLUDED

#include <SDL/SDL.h>

void lireDonneeZoneTexte(SDL\_Surface \*surface, TTF\_Font \*police, char \*chaineEntree, char \*introductionInitiale, char \*texteDePreremplissage, unsigned long nombreDeCaracteresMaximal);

void detecterErreurs(SDL\_Surface \*surface, SDL\_Surface \*barre, TTF\_Font \*police, char \*chaineEntree,char\*texteDePreremplissage,unsigned long nombreDeCaracteresMaximal);

void lireCaractereZoneTexte(SDL\_Event event, int \*majuscule, char \*chaine, unsigned long \*positionCaractere, int \*continuer, unsigned long nombreDeCaracteresMaximal);

void entrerLettre(int \*lettreEntree, unsigned long numeroCaractere, char \*caractere);

void insererCaractereDansChaine(unsigned long positionCaractere, char \*chaine, char caractereAInserer);

void blitterCurseur(SDL\_Rect \*positionTexte, TTF\_Font \*police, char \*chaineEntree, SDL\_Surface \*surface, SDL\_Surface \*barre, unsigned long positionCaractere);

void detecterLettre(SDL\_Event event, int \*lettreEntree, char \*caractereEntre, int\*majuscule);

#endif // ZONETEXTE\_H\_INCLUDED

I MPLEMENTATION DES FONCTION(zonetext.cpp)

/\*

zonetexte.c

Par Rayman3640 et Troumad

Rôle : permettre au créateur de simuler des zones de texte avec la SDL

\*/

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <SDL/SDL.h>

#include <SDL/SDL\_ttf.h>

#include "zonetexte.h"

void lireDonneeZoneTexte(SDL\_Surface \*surface, TTF\_Font \*police, char \*chaineEntree, char \*introductionInitiale, char

\*texteDePreremplissage, unsigned long nombreDeCaracteresMaximal)

{

int continuer = 1, majuscule = 0;

SDL\_Surface \*texte, \*barre = NULL;

SDL\_Color noir = {0,0,0,0}; // 4ème champ inutilisé, mais présent tout de même : évite un warning.

SDL\_Color blanc = {255,255,255,0}; // 4ème champ inutilisé, mais présent tout de même : évite un warning.

SDL\_Rect positionTexte;

unsigned long positionCaractere = 0;

SDL\_Event event;

SDL\_Rect bas\_blanc;

SDL\_FillRect(surface,NULL,SDL\_MapRGB(surface->format,0,0,0));

texte = TTF\_RenderUTF8\_Solid(police,introductionInitiale,blanc);

positionTexte.x = 0;

positionTexte.y = 0;

SDL\_BlitSurface(texte,NULL,surface,&positionTexte);

positionTexte.y = texte->h +10;

bas\_blanc.y=positionTexte.y-5;

bas\_blanc.h=480-bas\_blanc.y;

bas\_blanc.x=0;

bas\_blanc.w=640;

SDL\_FillRect(surface,&bas\_blanc,65535);

barre = SDL\_LoadBMP("curseur.bmp");

detecterErreurs(surface,barre,police,chaineEntree,texteDePreremplissage,nombreDeCaracteresMaximal);

//strcpy(introduction,introductionInitiale);

strcpy(chaineEntree,texteDePreremplissage);

positionCaractere = strlen(chaineEntree);

//SDL\_FillRect(surface,NULL,SDL\_MapRGB(surface->format,255,255,0));

SDL\_EnableKeyRepeat(100,100);

SDL\_Flip(surface);

while(continuer)

{

SDL\_WaitEvent(&event);

switch(event.type)

{

case SDL\_QUIT:

continuer = 0;

break;

case SDL\_KEYDOWN:

lireCaractereZoneTexte

(event,&majuscule,chaineEntree,&positionCaractere,&continuer,nombreDeCaracteresMaximal);

break;

}

texte = TTF\_RenderUTF8\_Solid(police,chaineEntree,noir);

SDL\_BlitSurface(texte,NULL,surface,&positionTexte);

blitterCurseur(&positionTexte,police,chaineEntree,surface,barre, positionCaractere);

SDL\_Flip(surface);

SDL\_FillRect(surface,&bas\_blanc,65535);

//SDL\_FillRect(surface,NULL,SDL\_MapRGB(surface->format,255,255,0));

}

SDL\_FreeSurface(texte);

SDL\_FreeSurface(barre);

}

void detecterErreurs(SDL\_Surface \*surface, SDL\_Surface \*barre, TTF\_Font \*police, char \*chaineEntree, char \*texteDePreremplissage,

unsigned long nombreDeCaracteresMaximal)

{

if(surface == NULL)

{

fprintf(stderr,"Le pointeur sur surface vaut NULL, donc le texte ne peut pas être affiché à l'écran.\n");

exit(EXIT\_FAILURE);

}

if(barre == NULL)

{

fprintf(stderr,"Erreur lors de l'ouverture de l'image Curseur.png: %s\n",SDL\_GetError());

exit(EXIT\_FAILURE);

}

if(police == NULL)

{

fprintf(stderr,"Le pointeur sur police vaut NULL, donc aucune police ne peut être ouverte.\n");

exit(EXIT\_FAILURE);

}

if(chaineEntree == NULL)

{

fprintf(stderr,"La chaîne entrée vaut NULL, donc l'utilisateur ne peut rentrer aucune chaîne.\n");

exit(EXIT\_FAILURE);

}

if(nombreDeCaracteresMaximal < 1)

{

fprintf(stderr,"La taille maximale est nulle, donc l'utilisateur ne peut entrer aucune chaîne.\n");

exit(EXIT\_FAILURE);

}

if(texteDePreremplissage != NULL && strlen(texteDePreremplissage) > nombreDeCaracteresMaximal)

{

fprintf(stderr,"Le nombre de caractères du texte de préremplissage est supérieur au nombre de caractères maximal.

\n");

exit(EXIT\_FAILURE);

}

}

void blitterCurseur(SDL\_Rect \*positionTexte, TTF\_Font \*police, char \*chaineEntree, SDL\_Surface \*surface, SDL\_Surface \*barre,

unsigned long positionCaractere)

{

int longueurSurfaceTexte = 0, longueurSurfaceTexte2 = 0, longueurSurfaceTexte3 = 0, largeurSurfaceTexte = 0;

SDL\_Rect positionBarre;

if(strlen(chaineEntree) > 0)

{

positionBarre.x = positionTexte->x;

if(TTF\_SizeUTF8(police,chaineEntree,&longueurSurfaceTexte2,&largeurSurfaceTexte) == 0)

{

if(TTF\_SizeUTF8(police,&chaineEntree

[positionCaractere],&longueurSurfaceTexte3,&largeurSurfaceTexte) == 0)

{

longueurSurfaceTexte = longueurSurfaceTexte2 - longueurSurfaceTexte3;

}

positionBarre.x += longueurSurfaceTexte;

}

positionBarre.y = positionTexte->y;

SDL\_BlitSurface(barre,NULL,surface,&positionBarre);

}

}

void lireCaractereZoneTexte(SDL\_Event event, int \*majuscule, char \*chaine, unsigned long \*positionCaractere, int \*continuer, unsigned

long nombreDeCaracteresMaximal)

{

char caractereEntre = 0;

int lettreEntree = 0, diminuerPositionCaractere = 1;

unsigned long i = 0;

if(event.key.keysym.mod & KMOD\_SHIFT)

{

\*majuscule = 1;

}

else

{

\*majuscule = 0;

}

if (event.key.keysym.mod & KMOD\_CAPS)

\*majuscule=!\*majuscule;

switch(event.key.keysym.sym)

{

case SDLK\_LEFT:

if(\*positionCaractere > 0)

\*positionCaractere -= 1;

break;

case SDLK\_RIGHT:

if(\*positionCaractere < strlen(chaine))

\*positionCaractere += 1;

break;

case SDLK\_BACKSPACE:

//if(\*positionCaractere > 0)

{

if (\*positionCaractere==0)

{

diminuerPositionCaractere = 0;

i=0;

}

else i=\*positionCaractere-1;

while( i < strlen(chaine))

{

chaine[i] = chaine[i+1];

i++;

}

if(diminuerPositionCaractere)

\*positionCaractere-=1;

else

\*positionCaractere=0;

}

break;

case SDLK\_ESCAPE:

\*continuer = 0;

break;

case SDLK\_RETURN:

\*continuer = 0;

break;

default:

break;

}

detecterLettre(event,&lettreEntree,&caractereEntre,majuscule);

if(lettreEntree == 1 && strlen(chaine) <= nombreDeCaracteresMaximal)

{

insererCaractereDansChaine(\*positionCaractere,chaine,caractereEntre);

\*positionCaractere += 1;

}

}

void detecterLettre(SDL\_Event event, int \*lettreEntree, char \*caractereEntre, int \*majuscule)

{

long i = 0;

for(i = 0; i < 26; i++) //Lettres sans accents minuscules et majuscules

{

if(event.key.keysym.sym == SDLK\_a + i)

{

entrerLettre(lettreEntree,65 + i,caractereEntre);

if(!(\*majuscule))

{

\*caractereEntre += 32;

}

break;

}

}

for(i = 0; i < 27; i++)

{

if(event.key.keysym.sym == SDLK\_AMPERSAND + i)

{

entrerLettre(lettreEntree,38 + i,caractereEntre);

}

}

for(i = 0; i < 5; i++)

{

if(event.key.keysym.sym == SDLK\_SPACE + i)

{

entrerLettre(lettreEntree,32 + i,caractereEntre);

}

}

switch(event.key.keysym.sym) //Caractères spéciaux

{

case SDLK\_BACKSLASH:

entrerLettre(lettreEntree,92,caractereEntre);

break;

default:

break;

}

for(i = 0; i < 10; i++)

{

if(event.key.keysym.sym == SDLK\_KP0 + i) //Nombres du pavé numérique

{

entrerLettre(lettreEntree,'0' + i,caractereEntre);

}

}

}

void entrerLettre(int \*lettreEntree, unsigned long numeroCaractere, char \*caractere)

{

if(!(\*lettreEntree))

{

\*lettreEntree = 1;

\*caractere = numeroCaractere;

}

}

void insererCaractereDansChaine(unsigned long positionCaractere, char \*chaine, char caractereAInserer)

{

char \* chaine2;

chaine2=(char \*)malloc(strlen(chaine)+2); // insertion au début d'une chaîne aussi grande que souhaitée

strcpy(chaine2,&chaine[positionCaractere]);

chaine[positionCaractere] = caractereAInserer;

strcpy(&chaine[positionCaractere+1],chaine2);

}

FONCTION.H

#ifndef FONCTION\_H\_INCLUDED

#define FONCTION\_H\_INCLUDED

#define TAILLE\_BLOC 34 // Taille d'un bloc (carré) en pixels

#define NB\_BLOCS\_LARGEUR 32

#define NB\_BLOCS\_HAUTEUR 24

#define LARGEUR\_FENETRE TAILLE\_BLOC \* NB\_BLOCS\_LARGEUR

#define HAUTEUR\_FENETRE TAILLE\_BLOC \* NB\_BLOCS\_HAUTEUR

//constante du reseaux

#define GLIDE\_DELAY 5000;

#define SWITCH\_ENTRY\_AGE 60;

#define SWITCH\_CRON\_DELAY 1000;

#define ARP\_ENTRY\_AGE 60;

#define ROUTER\_CRON\_DELAY 1000;

#define IP4\_RETRY\_DELAY 2\*GLIDE\_DELAY+1000;

#define IP4\_RETRY\_MAX 4;

#define ROUTE\_HIGHLIGHT\_DELAY 5000;

#define ETHERNET\_BROADCAST 'ff:ff:ff:ff:ff:ff';

//Variable Reseaux

//

enum {VIDE, NOEUD, NOEUD1, NOEUD2, NOEUD3, NOEUD4, ROUTER, LIENS, LIENS\_UN, VOYANT\_VERT, VOYANT\_ROUGE, PARQUET, TABLE\_ROUTEUR, TABLE\_NOEUD, COMMUTATEUR};

void editeur(SDL\_Surface\* ecran);

void ecrireTexte(SDL\_Surface\* screen, int x, int y, TTF\_Font\* font, SDL\_Surface\* textSurface);

void pause();

#endif // FONCTION\_H\_INCLUDED

FONCTION.CPP

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <SDL/SDL.h>

#include <SDL/SDL\_ttf.h>

#include "fonction.h"

void editeur(SDL\_Surface\* ecran){

SDL\_Surface \*img\_rouge = NULL, \*img\_bleu = NULL, \*rougebmp = NULL, \*img\_vert = NULL;

SDL\_Surface \*vertbmp = NULL, \*table\_noeud= NULL, \*table\_routeur = NULL, \*creationPaquet = NULL;

SDL\_Surface \*elementImg = NULL, \*img\_router = NULL, \*img\_noeud = NULL;

SDL\_Surface \*img\_lien2 = NULL, \*img\_lien1 = NULL, \*img\_commut= NULL;

SDL\_Surface \*img\_station1 = NULL, \*img\_station2 = NULL, \*img\_station3= NULL, \*img\_station4= NULL;

SDL\_Rect posElement;

int i = 0, j = 0;

int carte[NB\_BLOCS\_LARGEUR][NB\_BLOCS\_HAUTEUR ] = {0};

// Chargement des sprites (décors, personnage...)

img\_bleu = SDL\_LoadBMP("bleu.bmp");

img\_vert = SDL\_LoadBMP("vert.bmp");

img\_rouge = SDL\_LoadBMP("rouge.bmp");

img\_router = SDL\_LoadBMP("router.bmp");

img\_lien1 = SDL\_LoadBMP("lien1.bmp");

img\_lien2 = SDL\_LoadBMP("lien2.bmp");

img\_noeud = SDL\_LoadBMP("pc.bmp");

img\_commut = SDL\_LoadBMP("commutateur.bmp");

rougebmp = SDL\_LoadBMP("rouge.bmp");

vertbmp = SDL\_LoadBMP("vert.bmp");

table\_noeud = SDL\_LoadBMP("table\_noeud.bmp");

creationPaquet = SDL\_LoadBMP("creationPaquet.bmp");

table\_routeur = SDL\_LoadBMP("router\_table.bmp");

img\_station1 = SDL\_LoadBMP("station1.bmp");

img\_station2 = SDL\_LoadBMP("station2.bmp");

img\_station3 = SDL\_LoadBMP("station3.bmp");

img\_station4 = SDL\_LoadBMP("station4.bmp");

//PERMET DE POSITIONNER LES ELEMENTS SUR A CARTE HORZON(G ou D)|VERTI(H ou B)

//HAUT

carte[12][6]=NOEUD;

carte[12][7]=VOYANT\_VERT;

carte[12][8]=LIENS\_UN;

carte[12][9]=VOYANT\_VERT;

carte[12][10]=LIENS\_UN;

carte[12][11]=VOYANT\_VERT;

//CENTRE

carte[12][12]=COMMUTATEUR;

//BAS

carte[12][13]=VOYANT\_VERT;

carte[12][14]=LIENS\_UN;

carte[12][15]=VOYANT\_VERT;

carte[12][16]=LIENS\_UN;

carte[12][17]=VOYANT\_VERT;

carte[12][18]=NOEUD;

//GAUCHE

carte[6][12]=NOEUD;

carte[7][12]=VOYANT\_VERT;

carte[8][12]=LIENS;

carte[9][12]=VOYANT\_VERT;

carte[10][12]=LIENS;

carte[11][12]=VOYANT\_VERT;

//carte[5][7]=VOYANT\_VERT;

//DROITE

carte[13][12]=VOYANT\_VERT;

carte[14][12]=LIENS;

carte[15][12]=VOYANT\_VERT;

carte[16][12]=ROUTER;

carte[17][12]=VOYANT\_VERT;

carte[18][12]=LIENS;

carte[19][12]=VOYANT\_VERT;

carte[20][12]=COMMUTATEUR;

carte[21][12]=VOYANT\_VERT;

carte[22][12]=LIENS;

carte[23][12]=VOYANT\_VERT;

carte[24][12]=NOEUD;

//

carte[0][19]=PARQUET;

carte[20][0]=TABLE\_ROUTEUR;

carte[0][7]=NOEUD1;

carte[9][0]=NOEUD2;

carte[9][19]=NOEUD3;

carte[25][10]=NOEUD4;

// Effacement de l'écran

SDL\_FillRect(ecran, NULL, SDL\_MapRGB(ecran->format, 255, 255, 255));

// Activation de la répétition des touches

SDL\_EnableKeyRepeat(100, 100);

for (i = 0 ; i < NB\_BLOCS\_LARGEUR ; i++){

for (j = 0 ; j < NB\_BLOCS\_HAUTEUR ; j++){

posElement.x = i \* TAILLE\_BLOC;

posElement.y = j \* TAILLE\_BLOC;

switch(carte[i][j]){

case VIDE:

break;

case COMMUTATEUR:

SDL\_BlitSurface(img\_commut, NULL, ecran, &posElement);

break;

case NOEUD:

SDL\_BlitSurface(img\_noeud, NULL, ecran, &posElement);

break;

case NOEUD1:

SDL\_BlitSurface(img\_station1, NULL, ecran, &posElement);

break;

case NOEUD2:

SDL\_BlitSurface(img\_station2, NULL, ecran, &posElement);

break;

case NOEUD3:

SDL\_BlitSurface(img\_station3, NULL, ecran, &posElement);

break;

case NOEUD4:

SDL\_BlitSurface(img\_station4, NULL, ecran, &posElement);

break;

case ROUTER:

SDL\_BlitSurface(img\_router, NULL, ecran, &posElement);

break;

case LIENS:

SDL\_BlitSurface(img\_lien2, NULL, ecran, &posElement);

break;

case LIENS\_UN:

SDL\_BlitSurface(img\_lien1, NULL, ecran, &posElement);

break;

case VOYANT\_VERT:

SDL\_BlitSurface(vertbmp, NULL, ecran, &posElement);

break;

case VOYANT\_ROUGE:

SDL\_BlitSurface(rougebmp, NULL, ecran, &posElement);

break;

case PARQUET:

SDL\_BlitSurface(creationPaquet, NULL, ecran, &posElement);

break;

case TABLE\_ROUTEUR:

SDL\_BlitSurface(table\_routeur, NULL, ecran, &posElement);

break;

case TABLE\_NOEUD:

SDL\_BlitSurface(table\_noeud, NULL, ecran, &posElement);

break;

}

}

}

// Mise à jour de l'écran

SDL\_Flip(ecran);

// Désactivation de la répétition des touches (remise à 0)

SDL\_EnableKeyRepeat(0, 0);

// Libération des surfaces chargées

SDL\_FreeSurface(img\_rouge);

SDL\_FreeSurface(img\_bleu);

SDL\_FreeSurface(img\_vert);

SDL\_FreeSurface(elementImg);

SDL\_FreeSurface(img\_router);

SDL\_FreeSurface(img\_noeud);

SDL\_FreeSurface(rougebmp);

SDL\_FreeSurface(vertbmp);

SDL\_FreeSurface(creationPaquet);

SDL\_FreeSurface(table\_noeud);

SDL\_FreeSurface(table\_routeur);

SDL\_FreeSurface(img\_lien2);

SDL\_FreeSurface(img\_lien1);

SDL\_FreeSurface(img\_station1);

SDL\_FreeSurface(img\_station2);

SDL\_FreeSurface(img\_station3);

SDL\_FreeSurface(img\_station4);

}

void pause(){

int continuer = 1;

SDL\_Event event;

while (continuer){

SDL\_WaitEvent(&event);

switch(event.type){

case SDL\_QUIT:

continuer = 0;

}

}

}

void ecrireTexte(SDL\_Surface\* screen, int x, int y, TTF\_Font\* font, SDL\_Surface\* textSurface){

SDL\_Rect textLocation = { x, y, 0, 0 };

SDL\_BlitSurface(textSurface, NULL, screen, &textLocation);

//SDL\_Flip(screen);

TTF\_CloseFont(font);

SDL\_FreeSurface(textSurface);

SDL\_FreeSurface(screen);

//pause();

}

CORPS DU LOGICIEL MAIN

#include <stdlib.h>

#include <stdio.h>

#include <string.h>

#include <SDL/SDL.h>

#include <SDL/SDL\_ttf.h>

#include "fonction.h"

#include "zonetexte.h"

int main(int argc, char \*argv[]){

//SDL\_Delay(6000);

// Le pointeur qui va stocker la surface de l'écran , message

SDL\_Surface \*ecran = NULL, \*menu= NULL, \*texteAffiche = NULL;

//variable position

SDL\_Rect positionMenu, position;

SDL\_Event event;

/\* Chargement de la police \*/

TTF\_Font \*police = NULL;

SDL\_Color couleurNoire = {0, 0, 0};

char texte[20] = "";

char chaine[100] = "";

int continuer = 1;

char chaine1[100] = "";

// Chargement de la vidéo, de l'audio et du timer

if(SDL\_Init(SDL\_INIT\_VIDEO | SDL\_INIT\_AUDIO | SDL\_INIT\_TIMER ) == -1)

{

fprintf(stderr,"Erreur lors de l'initialisation de la SDL : %s\n",SDL\_GetError());

exit(EXIT\_FAILURE);

}

//Init ecriture

if(TTF\_Init() == -1)

{

fprintf(stderr,"Erreur lors de l'initialisation de SDL\_TTF : %s\n",TTF\_GetError());

exit(EXIT\_FAILURE);

}

// Chargement de l'icône AVANT SDL\_SetVideoMode \*/

SDL\_WM\_SetIcon(SDL\_LoadBMP("pc.bmp"), NULL);

ecran = SDL\_SetVideoMode(LARGEUR\_FENETRE, HAUTEUR\_FENETRE, 32, SDL\_HWSURFACE | SDL\_DOUBLEBUF);

//Label ou nom de la fenetre

SDL\_WM\_SetCaption("Projet Simulateur Reseaux", NULL);

/\* Chargement de la police \*/

police=TTF\_OpenFont("timesbd.ttf", 11);

if(police == NULL)

{

fprintf(stderr,"Erreur police : %s",TTF\_GetError());

exit(EXIT\_FAILURE);

}

//demande IP source\*/

lireDonneeZoneTexte(ecran,police,chaine,"DONNER ADRESSE IP SOURCE","192.168.1.1",99);

//demande IP cible\*/

lireDonneeZoneTexte(ecran,police,chaine1,"DONNER ADRESSE IP CIBLE","192.168.1.1",99);

positionMenu.x = 0;

positionMenu.y = 0;

editeur(ecran);

while(continuer!=0){

if( (event.type) && (event.key.keysym.sym == SDLK\_ESCAPE) ){

continuer=0;

}

else if( (event.type) && (event.type== SDL\_QUIT) ){

continuer=0;

}

else{

continuer = 1;

sprintf(texte, chaine);

texteAffiche= TTF\_RenderText\_Solid(police, texte, couleurNoire);

SDL\_BlitSurface(menu, NULL, ecran, &positionMenu);

int carteText[NB\_BLOCS\_LARGEUR][NB\_BLOCS\_HAUTEUR ] = {0};

int im,jm;

//station 1 HAUT

carteText[9][3]=1;

//station 2 GAUCHE

carteText[0][10]=1;

//station 3 DROITE

carteText[25][13]=1;

//station 4 BAS

carteText[9][22]=1;

//ROUTER

carteText[20][3]=1;

//position texte

for (im = 0 ; im < NB\_BLOCS\_LARGEUR ; im++){

for (jm = 0 ; jm < NB\_BLOCS\_HAUTEUR ; jm++){

position.x = im \* TAILLE\_BLOC;

position.y = jm \* TAILLE\_BLOC;

switch(carteText[im][jm]){

case 0:

break;

case 1:

SDL\_BlitSurface(texteAffiche, NULL, ecran, &position);

break;

case 2:

SDL\_BlitSurface(texteAffiche, NULL, ecran, &position);

break;

case 3:

SDL\_BlitSurface(texteAffiche, NULL, ecran, &position);

break;

case 4:

SDL\_BlitSurface(texteAffiche, NULL, ecran, &position);

break;

case 5:

SDL\_BlitSurface(texteAffiche, NULL, ecran, &position);

break;

}

}

}

SDL\_Flip(ecran);

SDL\_WaitEvent(&event);

}

}

TTF\_CloseFont(police);

TTF\_Quit();

SDL\_FreeSurface(texteAffiche);

SDL\_FreeSurface(menu);

SDL\_FreeSurface(ecran);

SDL\_Quit();

return EXIT\_SUCCESS;

}

### 5.2 Manuel Utilisateur

### 5.3 Conception architectural

### 5.4 Plan et rapport de test

### 5.5 Calendrier de projet

5.5.1 **PERT**

A 3

1

3

1

3

0

0

F 20 20

G 2

0

11

17

11

17

E 7

0

0

D 5

0

10

6

10

6

5

4

5

0

C 2

0

4

0

0

1

1

1

1

B 1

38

39

37

18

0

0

57

57

57

57

K 1

J 15

42

56

42

56

0

0

0

0

39

38

0

18

37

0

40

41

I 2

40

41

0

0

38

39

H 2

38

39

0

0

NB :

Tache

Durée

D+tot F+tot

D+tard

Marge totale F+tard

Marge libre

Taches

A : analyse

B : création compte git

C : Spécification

D : Conception

E : document conception

F : réalisation

G : test

H : code produit

I : plan et rapport test

J : manuel d’utilisation

K : présentation

5.5.2 GANTT

(taches)

A

B

C

D

E

F

G

H

I

J

K

1 3 4 5 6 10 11 17 18 37 38 39 40 41 42 56 57 (durée en jour)

5.5.3 WDS (voir annexe papier excel)