

EPIGRAPHE

« Faites-nous de bonne politique et je vous ferai de bonnes finances qui vous rendrons célèbres et riche. »

Le Baron Louis

DEDICACE

A nos parents pour leur soutien total et une éducation efficace.
Trouvez-en le fruit de tous les efforts ainsi que tous les sacrifices consentis.

Particulièrement à notre très cher Papa MUSHESHA Raymond et
maman BAKUNGU Viviane, pour leur amour, soutien, solidarité et conseils qui
nous ont permis de bien évoluer dans ce domaine que la paix, l'amour, la grâce
du Christ soit entre eux.

KUABO MOSANTU Fortuné

REMERCIEMENTS

Nous ne pouvons pas entrer dans le vif de notre travail sans pour autant remercier les personnes qui nous ont soutenues Durant notre parcours d'étude jusqu'à la réalisation de ce travail de fin de cycle.

Nous tenons à remercier d'abord le Dieu tout puissant source de toute sagesse et intelligence, donateur de tout Don excellent et parfait.

Que le professeur KASORO MULENDA Nathanaël veuille trouver ici nos remerciements les plus sincères pour avoir accepté de diriger le présent travail, sa disponibilité, malgré ses multiples occupations, ses remarque et suggestions ainsi que sa lucidité d'esprit ont été déterminantes pour nous donner une orientation définitive.

Nos sincères remerciements vont droit à nos parents MUSHESHA Raymond et BAKUNGU Viviane d'avoir assuré notre éducation en voulant toujours le meilleur de notre vie et pour leur soutien moral, financier et spirituel avec lequel nous nous sommes conduisons avec prudence et sagesse qu'ils trouvent dans ce travail, l'expression de nos sentiments de gratitude.

Notre profonde reconnaissance s'adresse à nos frères et sœurs de la famille : MUBYALE GLORE, LOLA BAIBONGE, IMOJA MULAWAKO pour l'amour qu'ils ont prouvé à égards et leur contribution matérielle, financière et morale qu'ils trouvent également notre reconnaissance.

Nos sentiments de reconnaissances s'adressent également à tous nos compagnons de lutte : MAJIBU RUKIRA Adolphe, BAMPENDE BAMPENDE Adolphe, MBAYA TAMBWE David, BOYONGO BO-LOBONGA Rodriguez, KADIABUE MITEO Jonathan, KASHEMWA BINJA Christian, KALIVANDA KISONI Prisca, MUKENDI BANZA Giress, NGAJA NKANDA Naomi que tous ceux dont le nom ne figure pas ci-haut, qu'ils trouvent ici l'expression de notre parfaite gratitude.

C'est aussi pour nous un devoir d'adresser nos sincères remerciements aux autorités académiques et au corps professoral de l'université de Kinshasa, particulièrement celui de la section mathématique et informatique pour la formation de qualité que nous avons reçue.

0. INTRODUCTION GENERALE

0.1. PROBLEMATIQUE

La problématique désigne l'ensemble des questions posées dans un domaine de la science en vue d'une recherche des solutions.

Dans le cas de notre domaine d'application, il se posait un problème lorsqu'une personne souhaite entrer dans un local où l'accès demande une permission de l'occupant du local. En pratique le système actuel marche selon un protocole qui veut à ce que, la personne voulant entrer doit frapper souvent trois fois sur la porte du bureau, pour qu'en suite l'occupant du bureau l'ayant entendue puissent lui accorder l'accès.

Les résultats obtenus de ce système ne sont pas fiables, en dépit de certaines difficultés rencontrées tel que :

- Il arrive que l'occupant du local n'entend pas frapper soit parce qu'il est trop pris ou soit la personne ne frappe pas assez fort ;
- Parfois l'occupant du bureau est absent, alors la personne voulant entrer passe une éternité à frapper sur la porte, mais sans réponse ;
- De fois l'occupant du local donne accès, mais la personne voulant entrer ne l'attend pas
- Il y a même des personnes frappent et entre directement sans attendre la permission de l'occupant du local, même quand elle n'est pas disponible ;
- Et parfois certaine personne frappe très fort au risque même d'effrayer l'occupant du local ;

Par conséquent, relativement à la préoccupation de la haute hiérarchie, nous avons décidé à pallier à ces différentes difficultés ayant trait à cet ancien système qui jusqu'à présent accuse certaines défaillances.

C'est pourquoi, notre problématique se résumera avec la question générale suivante :

Que faire pour instaurer un système qui soit à la fois souple et fiable pour la supervision d'entrée dans un bureau.

Image illustrative de l'ancien système :

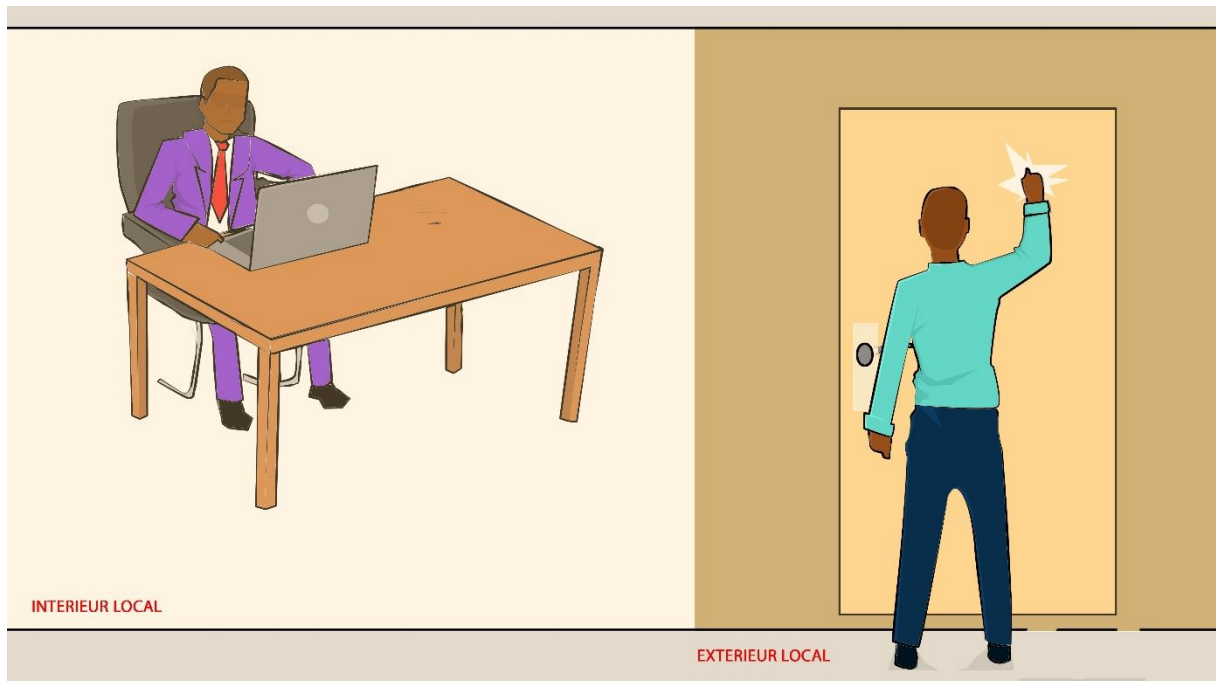


Figure 1 : illustration du fonctionnement de l'ancien système.

0.2. HYPOTHESE

Une hypothèse est une proposition liée à l'explication des phénomènes naturels et doit se vérifier par le fait.

Sur base des différents problèmes évoqués ci-dessus, nous posons comme hypothèse la mise en place d'un système automatique, pour une bonne supervision des entrées dans un local à accès limité. Ici le protocole change, la personne voulant entrée va appuyer sur un bouton poussoir se trouvant près de la porte, qui fera sonner un dispositif se trouvant à l'intérieur du local, puis l'occupant du local va décider à partir de son ordinateur si elle est disponible ou pas en cas d'absence le dispositif extérieur pourra le signaler. Le signalement du dispositif extérieur se fera grâce à des indicateurs lumineux avec des labels à côté (Entrer, patienter et indisponible).

Cette nouvelle technologie facilitera :

- ✓ Une communication facile entre l'occupant du local et la personne voulant entrer.

- ✓ La réduction du temps de traitement et du temps de réponse du système.
- ✓ La fiabilité des résultats du système.
- ✓ L'accès bien contrôlé.
- ✓ Le contrôle sur le système

Image illustrative du nouveau système :



Figure 2 : illustration du fonctionnement du nouveau système.

0.3. PRESENTATION DU SUJET

Le sujet dont il est question dans ce travail est intitulé :

« Conception et réalisation d'un system de supervision des entrées dans un local à accès limité ».

0.4. CHOIX ET INTERET DU SUJET

1) Choix du sujet

Le choix porté sur le sujet, provient du souci de la recherche d'une solution lié à la technologie aux problèmes des accès à un local qui

depuis très longtemps était trop mécanique, avec tout ce que cela entraîne comme inconvénients.

2) Intérêt du sujet

Ce sujet présente un grand intérêt à savoir :

- L'implantation d'un nouveau système pour une bonne supervision des accès à un bureau pour résoudre leurs problèmes en temps réel, voulu et de manière optimal.
- L'expérimentation des connaissances acquises sur l'analyse d'un problème jusqu'à la conception de la solution prise (nouveau système).
- Le contenu de ce travail pourra constituer une référence pour les chercheurs intéressés à l'implantation de la technologie dans les problèmes similaires.

0.5. DELIMITATION DU SUJET

Selon une exigence scientifique, une étude doit être délimitée sous deux aspects thématique, dans l'espace et dans le temps :

- Dans le temps, notre étude va se limiter pour une période allant de 2019 à 2020.
- Dans l'espace, notre sujet va se limiter au niveau de l'accès à un local à accès limité.

0.6. METHODE ET TECHNIQUE UTILISEES

1) Méthodes utilisé

Une méthode est une démarche à suivre pour atteindre un objectif.

Pour notre travail, nous nous sommes basés sur des méthodes ci-après :

- Méthode analytique : Elle procède par décomposition du sujet, celle-ci nous a permis d'analyser les différents problèmes existant et proposer des pistes de solution.
- Méthode descriptive : Nous a aidées dans l'étude préalable de donner les faits selon certaines normes universelles.
- Méthode structuro-fonctionnelles : Qui permet de présenter la structure ainsi que le fonctionnement du système étudié.

2) Technique utilisées

Une technique est un outil qu'on emploie pour atteindre un but ou un objectif bien défini.

Pour un bon accomplissement de notre travail, nous avons utilisé les techniques suivantes :

- La technique d'observation : à partir de nos observations approfondies nous avons découvert d'autres informations manquantes pour le déroulement de notre travail.
- la technique de questionnement : à partir de différentes questions composées après observation nous avons découvert une grande partie des informations manquantes.

0.7. Canevas du travail

Hormis l'introduction et la conclusion, le présent travail est subdivisé en deux grandes parties qui sont les suivantes :

- **La première partie** : les généralités conceptuelles :
 - Chapitre I : les concepts théoriques de base
 - Chapitre II : cadrage du projet
- **La deuxième partie** : modélisation et de réalisation du nouveau système
 - Chapitre IV : Modélisation du nouveau système d'information
 - Chapitre V : Réalisation du nouveau système.

Liste des figures

Figure 1 : illustration du fonctionnement de l'ancien système.

Figure 2 : illustration du fonctionnement du nouveau système.

Figure 3 : processus d'un système.

Figure 4 : système d'information.

Figure 5 : représentant les acteurs.

Figure 6 : liaison entre cas d'utilisation et un acteur.

Figure 7 : représentation d'une classe.

Figure 8 : Formalisme diagramme de séquence.

Figure 9 : formalisme diagramme d'activité.

Figure 10 : Quelques symboles des générateurs.

Figure 11 : Quelques symboles des récepteurs.

Figure 12 : symbole de la masse.

Figure 13 : symbole de terre.

Figure 14 : illustration de la loi de nœud.

Figure 15 : illustration de la loi de maille.

Figure 16 : représentation des dipôles.

Figure 17 : Schéma récapitulatif.

Figure 18 : symboles résistances.

Figure 19 : différents symboles des diodes.

Figure 20 : symbole condensateur.

Figure 21 : différents symboles des transistors.

Figure 22 : Quelques composants électroniques.

Figure 23 : Quelques symboles de composants électroniques.

Figure 24 : Présentation du Graphe PERT.

Figure 25 : représentation d'un acteur.

Figure 26 : représentation des interactions entre acteur et cas d'utilisation.

Figure 27 : Représentation du diagramme de cas d'utilisation

Figure 28 : Représentation du diagramme de séquence

Figure 29 : Représentation du diagramme de classe.

Figure 30 : Formalisme diagramme d'activité

Figure 31 : Représentation du diagramme d'activité.

Figure 32 : schémas block

Figure 33 : Schémas block détailler 1.

Figure 34 : Schémas block détailler 2.

Figure 35 : Schémas block détailler au complet.

Figure 36 : Partie intérieur du montage (partie1).

Figure 37 : Partie extérieur du montage (partie2).

Figure 38 : Schémas du montage au complet.

Figure 39 : Dessin d'implantation 1.

Figure 40 : Dessin d'implantation 2.

Figure 41 : Plan de câblage 1.

Figure 42 : Plan de câblage 2.

Figure 43 : Architecture logiciel.

Figure 44 : Architecture matérielle

Liste des tableaux

Tableau 1 : comparaison post a post et client-serveur.

Tableau 2 : famille de matériaux

Tableau 3 : Planning d'exécution des taches, estimations de données

Tableau 4 : liste des composants

1 CONCEPTS THEORIQUE DE BASE

1.1 Système

1.1.1 Définition

Il existe plusieurs définitions du mot système. Nous retiendrons cependant ici celle de Jean Patrick Matheson qui considère le système comme un ensemble d'éléments matériels et immatériels (homme, machines, procédures) en interaction, transformant par un processus des éléments en entrée pour avoir d'autres en sortie.

Les trois éléments récents, à savoir les entrées, les sorties et le traitement forment les trois composantes interactives du système.

En plus de ces trois composantes, le système en comprend deux autres, à savoir : la rétroaction (comprenant les données relatives au rendement du système) et le contrôle (comprenant les données de vérification du fonctionnement du système).

Schématiquement, on a :

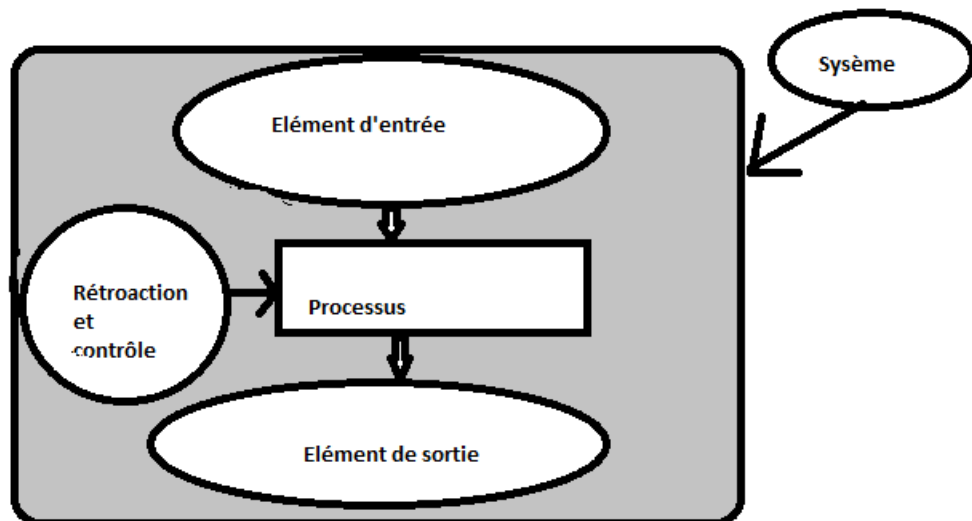


Figure 3 : processus d'un système.

1.1.2 Caractéristique d'un système

Il découle de cette définition que le système agit, se régule, est identifiable, s'informe, décide et mémorise.

1.1.3 Système d'information

Un système d'information est un ensemble de personnes, objet, procédures et des ressources qui recueillent de l'information, la transforment et la distribuent au sein d'une organisation.

✓ **Entreprise comme système**

La systémique facilite la compréhension de l'entreprise, objet complexe, actif et organisé, Tout corps social organisé, en particulier les entreprises ou les administrations, pourra être modélisé comme un système. L'entreprise comme système renferme en lui trois sous-systèmes.

- Le système de pilotage (organe de régulation)
- Le système opérant (organe actif)
- Le système d'information (organe de mémorisation)

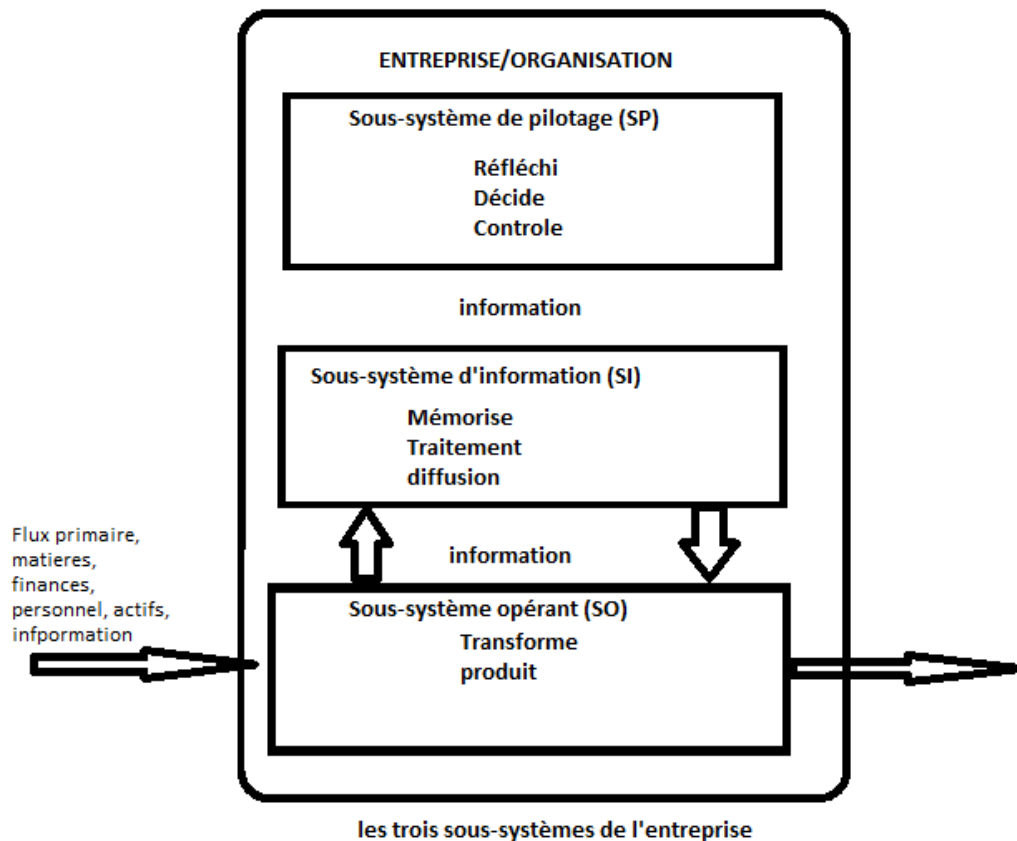


Figure 4 : système d'information.

a) Système de pilotage(SP)

Il est le siège de l'activité décisionnelle de l'entreprise, cette activité décisionnelle de l'entreprise, cette activité est très large et est assurée par tous les acteurs de l'entreprise, a des niveaux divers, depuis les acteurs agissant plutôt dans l'activité productrice de l'entreprise. Elle permet la régulation, le pilotage mais aussi l'adaptation de l'entreprise a son environnement.

C'est cette activité qui conduira l'évolution décidera notamment de l'organisation et de l'évolution des systèmes opérant et d'information. L'activité décisionnelle du système de pilotage de façon classique concerne entre autre l'allocation des ressources impliquées (prévision, planification, etc.) ainsi que leur suivi (contrôle de gestion, contrôle budgétaire).

b) Système opérant (SO)

Il est le siège de l'activité productive de l'entreprise. Cette activité consiste en une transformation des ressources ou flux du primaire, ces flux peuvent être des flux de matière, des flux financiers, des flux du personnel, des

flux d'actifs, ou enfin des flux d'information matière première non directement utilisable par le système d'information ou par le système de pilotage.

c) Système d'information (SI)

Un système d'information peut se définir comme la partie du réel constituée d'information organisées, d'évènement ayant un effet sur ces informations et d'acteur qui agissent sur ces informations ou à partir de ces informations, selon des processus visant une finalité de gestion et utilisant les technologies de l'information. Il faut considérer le système de pilotage d'assurer ses fonctions, notamment en assurant son couplage avec le système opérant. Ceci devient possible grâce aux quatre fonctions primaires qu'assure le système d'information dans l'entreprise : la génération des informations, la mémorisation des informations dans l'entreprise, la communication et la diffusion des informations et le traitement des informations.

Les systèmes d'information peuvent jouer un rôle capital dans le succès d'une entreprise, il fournit l'information dont l'entreprise a besoin pour une exploitation efficiente et une gestion efficace, et pour obtenir ou maintenir son avantage sur les concurrents.

1.1.4 Système informatique

a) Définition d'un système informatique

Le système informatique peut être défini comme étant l'ensemble des moyens humains, matériels et procédures permettant le traitement automatique des informations, au moyen de l'ordinateur qui en constitue l'élément central.

b) Qualité d'un système informatique

Dans la pratique, un bon système d'information doit posséder les qualités suivantes :

- Rapidité ;
- Fiabilité;
- Pertinence
- Sécurité;

c) Typologies des systèmes informatiques

Il existe plusieurs critères de classification des systèmes informatiques.

a) Selon le degré d'organisation

On distingue :

- **Le système indépendant**

Chaque service dispose de son propre système informatique, c'est-à-dire, ses propres applications, matériels et logiciels.

L'avantage de ce système est son Indépendance. Toutefois, il comporte un inconvénient : la multiplicité des matériels ; d'où possibilité d'incompatibilité entre matériels.

- **Le système intégré**

On recourt à l'approche base des données et réseaux. On dispose en effet d'un site de traitement où a été implantée une base des données qu'elle peuvent accéder les différents services reliés au serveur centrale dans un environnement client-serveur. Son avantage est qu'il offre une intégration des données dans une base de données entre services utilisateurs ainsi que la comptabilité des matériels et des informations.

- **Selon le degré d'automatisation**

On distingue :

- le système à traitement manuel ;
- le système à traitement mécanique où on utilise les auxiliaires mécaniques ;
- le système à traitement automatique où tout travail est réalisé à l'aide d'une machine électronique programmable. (Ex. ordinateur, machine à lessiver, etc.)

1.2 Présentation de la démarche UML

1.2.1 Introduction

UML « UNIFIED MODELING LANGUAGE » se définit comme un langage de modélisation graphique et textuel destiné à comprendre et décrire des besoins, spécifier et documenter des systèmes, esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue.

UML 2 s'articule autour de 13 diagrammes. Chacun de ses diagrammes permet de faire une représentation de système logiciel selon différents points de vue. Ces diagrammes sont divisés en grandes structures qui sont :

- La structure statique qui possède 7 diagrammes et
- La structure dynamique qui possède 6 diagrammes

Mais nous n'allons pas entrer en détail d'expliquer chaque diagramme, seul les diagrammes que nous allons utiliser seront explicités pendant la conception.

L'intérêt d'UML dans notre travail est suite aux certaines limites que les méthodes de modélisation classique ont rapidement montrées (telle MERISE) et par la principale avancée des quinze dernières années qui résident dans la programmation orientée objet (POO) afin de faire face à la complexité croissante des systèmes d'information.

1.2.2 Objectif d'UML

- Représenter des systèmes entiers ;
- Etablir un couplage explicite entre les concepts et les artefacts exécutables ;
- Prendre en compte les facteurs d'échelle ;
- Créer un langage de modélisation utilisable à la fois par les humains et les machines.

1.2.3 Caractéristique d'UML

UML est un langage de modélisation qui est :

- Prêt à l'emploi ;

- Simple ;
- Expressif ;
- Précis ;
- Extensible ;
- Indépendant d'une implémentation ;
- Indépendant d'un processus.

1.2.4 Les différents types de diagramme en UML

a) Diagramme de cas d'utilisation

Un cas d'utilisation « use case » représente un ensemble de séquence d'action qui sont réalisées par le système et qui produisent un résultat observable intéressant pour un user (acteur) particulier.

Chaque cas d'utilisation correspond à une fonction métier d'un système selon le point de vue de ses user(acteurs).

Tous les cas d'utilisation sont présents par un verbe a l'infinitif suivi d'un complément du point de vue de l'acteur.

Dans le diagramme de cas d'utilisation tous les acteurs sont externes au système d'information étudié.

Identification des acteurs du système

Identification des acteurs est une étape très capitale qu'on ne peut pas s'en passer car le système sera utilisé par les utilisateurs que nous devons bien connaître. D'où a cette étape nous devons identifier les acteurs qui vont interagir avec les systèmes.

D'où il faut savoir QU'EST-CE QU'UN ACTEUR ?

Un acteur représente un rôle joué par une entité externe (utilisateur humain, dispositif matériel ou autres système) qui interagit directement avec le système étudié.

Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant de messages susceptibles d'être porteurs de données.

Il existe quatre catégories d'acteurs :

- **Les acteurs principaux** : les personnes qui utilise la fonction du système.

- **Les acteurs secondaires** : les personnes qui effectuent des tâches administratives ou maintenance.
- **Le matériel externe** : les dispositifs matériels incontournables qui font partie du domaine de l'application et qui doivent être utilisés.
- **Les autres systèmes** : les systèmes avec lesquels le système doit interagir.

L'acteur est représenté par le formalisme suivant :

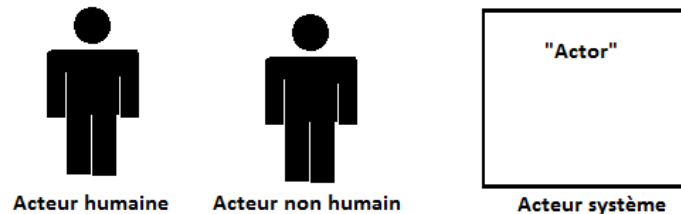


Figure 5 : représentant les acteurs.

QU'EST –CE QU'UN CAS D'UTILISATION ?

Un cas d'utilisation (user case) représente un ensemble de séquences d'actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier.

Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée « notable » à l'acteur concerné chaque cas d'utilisation spécifie un comportement attendu du système considéré comme un tout, sans imposer le mode de réalisation de ce comportement.

Il permet de décrire ce que le futur système devra faire, sans spécifier « comment ? » il le fera. Dans le cadre de la branche fonctionnelle, le cas d'utilisation doit mettre en valeur les interactions métier entre les acteurs et le système.

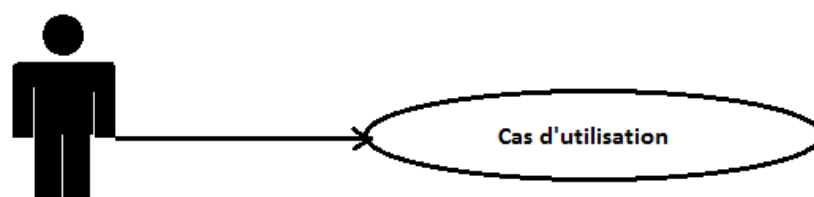


Figure 6 : liaison entre cas d'utilisation et un acteur.

b) Diagramme de classe :

On appelle classe la structure d'un objet c'est-à-dire l'ensemble des entités qui composent un objet. C'est la description abstraits d'un ensemble d'objets de même structure et de même comportement extraits du monde à modéliser.

Le diagramme de classe met en œuvre des classes contenant des attributs et des opérations, et reliées des associations ou généralisation.

- **Attribut** : représente un type d'information contenue dans une classe. C'est une donnée élémentaire servant à caractérisées les classe et les relations.

- **Méthodes** : ce sont des opérations programmées sur les objets d'une classe.

Note :

UML définit trois niveaux de visibilité pour les attributs.

- 1) **Public (+)** : l'élément est visible pour tous les objets de la classe ;
- 2) **Protège (#)** : l'élément est visible pour les sous-classes de la classe ;
- 3) **Prive (-)** : l'élément est visible pour tous les objets de la classe dans laquelle il est déclaré.

- **Identifiant** : est un attribut particulier qui permet de repérer de façon unique chaque objet, instance de classe.

- **Association** : représente une relation sémantique durable entre deux classes. Même si le verbe qui nomme une association entre concepts dans un modèle du domaine est par défaut bidirectionnelle.

Une association doit être accompagnée d'une multiplicité, une multiplicité est une information portée par le rôle, qui quantifie le nombre de fois ou un objet participe à une instance de relation.

On distingue les types de multiplicités ci-après :

1 : un et un seul

0...1 : zéro ou un

* : plusieurs

0...* : de zéro a plusieurs

1..* : de 1 a plusieurs

- **Agrégation et composition** : une agrégation est un cas particulier d'association non symétrique expriment une relation de contenance. L'agrégation n'a pas besoin d'être nomme.

- **Généralisation, superclasse et sous-classe** : une superclasse est une classe plus généralement reliée à une ou plusieurs autres classes plus spécialisée(sous-classes) par une relation de généralisation.

- Ainsi, les sous-classes hérite des propriétés de leur superclasse et peuvent porter aussi des propriétés spécifiques élémentaires.

Formalisme : une classe est représentée par un rectangle compartimenté.

- Le premier compartiment représente le nom de la classe ;
- Le deuxième compartiment représente les attributs de la classe ;
- Le troisième compartiment représente les opérations de la classe

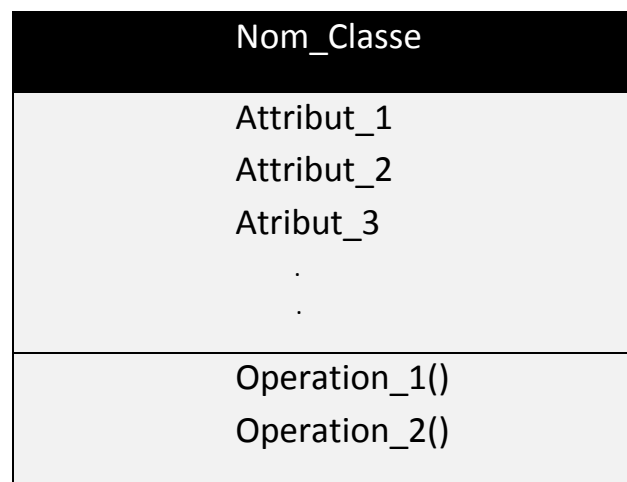


Figure 7 : représentation d'une classe.

c) Diagramme d'objets

Le diagramme d'objets illustre les objets et leurs relations. Les objets sont les instances des classes.

d) Diagramme de séquence

Le diagramme de séquence permet de décrire les actions à entreprendre pour le scénario de cas d'utilisation.

Formalisme :

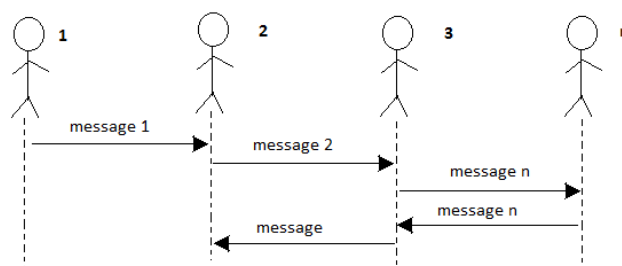


Figure 8 : Formalisme diagramme de séquence.

e) Diagramme d'activités

Le diagramme d'activités est attaché à une catégorie de classe et décrit le déroulement des activités de cette catégorie.

Formalisme :

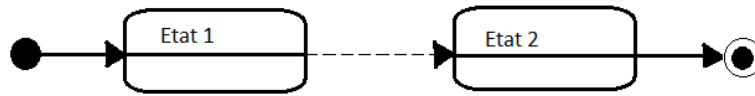


Figure 9 : formalisme diagramme d'activité.

f) Diagramme de collaboration

Le diagramme de collaboration permet de mettre en évidence les interactions entre les différents objets du système.

g) Diagramme de composants

Les diagrammes de compositions décrivent les composants et leurs dépendances de l'environnement de réalisation.

h) Diagrammes d'états-transition

Les diagrammes d'état-transitions ont pour rôle de représenter les traitements (opérations) qui vont gérer le domaine étudié. Ils définissent l'enchaînement des états de classe et font donc apparaître l'ordonnancement des travaux.

i) Diagramme de déploiement

Les diagrammes de déploiement montrent la disposition physique des différents matériels (nœuds) qui entrent dans la composition d'un système. En ceci intègre le réseau.

1.2.5 L'approche orientée objet

L'approche orientée objet considère le logiciel comme une collection d'objets dissociés, et identifiés, définis par des propriétés. Une propriété est soit un attribut (une donnée caractérisant l'état de l'objet), la fonctionnalité du logiciel émerge alors de l'interaction entre les différents objets qui le constitue. L'une des particularités de cette approche est qu'elle rapproche les données et leurs traitements associés au sein d'un unique objet.

Comme nous venons de le lire, un objet est caractérisé par plusieurs notions :

- L'identité : un objet possède une identité, qui permet de le distinguer des autres objets, indépendamment de son état.

- Les attributs : il s'agit des données caractérisant l'objet. Ce sont des variables stockant des informations sur l'état de l'objet.
- Les méthodes : ils caractérisent le comportement d'un objet, c'est-à-dire l'ensemble des actions (appelées opérations) que l'objet est à même de réaliser. Ces opérations permettent de faire réagir l'objet aux sollicitations extérieures (ou d'agir sur les autres objets).

La conception orientée objet (COO) est la méthode qui conduit à des architectures logicielles fondées sur les objets du système, plutôt que sur la fonction qu'il est censé réaliser.

a) Approche fonctionnelle vers l'Approche objet

Selon la thèse de Church-Turing tout langage de programmation non trivial équivaut à une machine de Turing. Il en résulte que tout programme qu'il est possible d'écrire dans un langage pourrait également être écrit dans n'importe quel autre langage. Ainsi, tout ce que l'on fait avec un langage de programmation par l'objet pourrait être fait en programmation impérative. La différence entre une approche fonctionnelle et une approche objet n'est donc pas d'ordre logique mais pratique.

L'approche structurée privilégie la fonction comme moyen d'organisation du logiciel. Ce n'est pas pour cette raison que l'approche objet est une approche non fonctionnelle. En effet, les méthodes d'un objet de l'approche fonctionnelle, c'est que les fonctions obtenues à l'issue de la mise en œuvre de l'une ou l'autre méthode sont distinctes.

L'approche objet est une approche orientée données. Dans cette approche, les fonctions se déduisent d'un regroupement de champs de données formant une entité cohérente, logique, tangible et surtout stable quant au problème traité.

L'approche structurée classique privilégie une organisation des données postérieure à la découverte de grandes, puis petites fonctions qui les décomposent, l'ensemble constituant les services qui répondent aux besoins.

b) Concepts importants de l'approche objet

- **Une classe** : décrit un groupe d'objets ayant les même propriétés (attribut), un même comportement (opérations), et une sémantique commune (domaine de définitions).
- **Polymorphisme** : est la capacité donnée à une même opération de s'exécuter différemment suivant le contexte de la classe où elle se trouve.
- **Opération** : action exécutée par un objet ou transformation subie par l'objet.
- **Méthode** : implémentation d'une opération par une classe plusieurs méthodes pour une même opération une méthode par classe pour une opération donnée.
- **Encapsulation** : masquage de l'information, séparation des aspects externes d'un objet, accessibles aux autres objets, des détails de l'implémentation, cachés à l'autre objet.
- **Abstraction** : représente une relation entre plusieurs classes. Elle correspond à l'abstraction des liens qui existent entre les objets dans le monde réel. Les multiplicités (ou cardinalités) et les rôles des objets participant aux relations complètent la description d'une association. Les exemples d'associations sont donnés directement dans les diagrammes de classe d'UML.
- **Association** : description d'un groupe de liens qui partagent une structure et sémantique commune.
- **Héritage** : partage de propriétés entre classes sur la base d'une relation hiérarchique
 - Super-classe (classe mère)
 - Sous-classe (classe fille) : spécialisation de la superclasse héritage des propriétés de la super-classe.
- **Classe abstraite** : classe ne pouvant pas être instanciée en tant que telle.
 - Sous-classes d'une classe abstraite : obligatoirement contraintes
 - Possibilité d'utiliser {abstract}
- **Un objet** : représente une entité du monde réel (ou monde virtuel pour les objets immatériels) qui se caractérise par un ensemble de propriétés (attributs), des états significatifs et un comportement.
- **Agrégation** : est une forme particulière d'association entre plusieurs classes. Elle exprime le fait qu'une classe est composée ou la relation structurelle représentant l'organigramme d'une entreprise sont des exemples types de la relation d'agrégation.

- **Interface** : permet de décrire la vue externe d'une classe. La classe d'interface, identifiée par un nom, comporte la liste des opérations accessibles par les autres classes. Le compartiment des attributs ne fait pas partie de la description d'une interface.

1.3 Définition de quelques notions électroniques

1.3.1 Notions de base

a) Pourquoi l'électronique a-t-elle été inventée ?

Avec les guerres, l'électronique fut un domaine favorisé car très utilisé dans la fabrication d'engins de combat et de matériel militaire, mais aussi la communication par radio.

Aujourd'hui, elle vient s'imposer dans nos foyers en améliorant notre confort quotidien. La physique, la médecine et l'électronique elle-même sont favorisées par les applications possibles de l'électronique grâce au matériel de plus en plus sophistiqué et performant.

b) Qu'est-ce que l'électronique ?

L'électronique est une science qui permet l'étude de structures qui traitent des signaux électriques, le but étant de transmettre ou de recevoir des données ou de l'énergie afin de créer des systèmes plus ou moins complexes et autonomes.

Ces systèmes utilisent des sources d'énergie pour pouvoir fonctionner. En général, il est question d'association de sous-systèmes pour en créer de gros qui présentent la caractéristique de réaliser plusieurs tâches différentes et parfois en même temps.

On peut trouver deux branches en électronique : l'électronique de signaux et l'électronique de puissance. En guise de ce projet nous parlerons uniquement de l'électronique qui utilise des signaux.

c) Grandeur physique utiliser en électronique

En électronique, on manipule diverses grandeurs physiques. On appelle « grandeur physique », ou simplement grandeur, toute propriété de la science de la nature qui peut être mesurée ou calculée. On va essentiellement utiliser trois grandeurs physiques qui sont : la tension, le courant et la puissance.

La source d'énergie

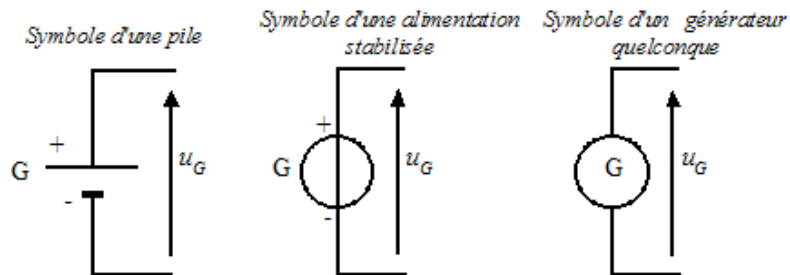
Nous allons d'abord déterminer d'où viennent les grandeurs physiques avant de voir ce que c'est exactement.

Citons plusieurs de ces sources que vous connaissez certainement déjà : la pile ; la batterie ; l'accumulateur ; le secteur électrique (attention, ce n'est pas le même type de source que les autres), en électronique ils sont appelé des générateurs.

Le générateur

Un générateur est un système qui fournit de l'énergie.

Quelques symboles :



Le récepteur

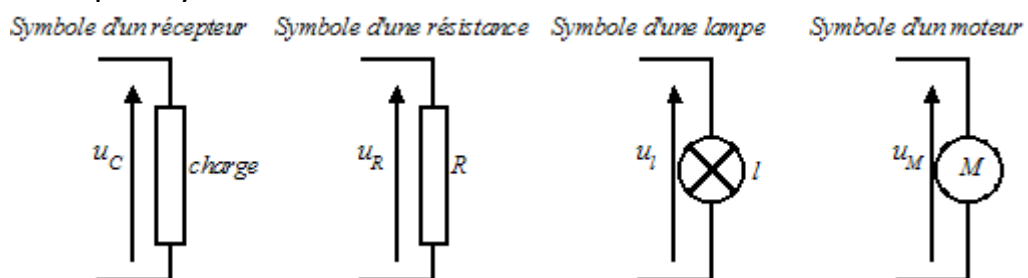
Un récepteur est un système qui reçoit de l'énergie.

Citons plusieurs exemples que vous connaissez également :

- Le chauffage ;
- Le grille-pain ;
- Le téléphone portable ;
- L'ordinateur ;
- etc.

Pour faire simple, un récepteur en électronique est tout ce qui fonctionne avec de l'électricité.

Quelques symboles :



Le courant électrique

Un courant électrique est un déplacement ordonné de charges électriques dans un conducteur.

Soit un courant constant et une quantité d'électricité qui parcourt une section d'un fil électrique. La formule qui suit permet de savoir quelle est la quantité de charges électriques qui ont circulé dans ce conducteur pendant une seconde :

$$Q = I \times T$$

Explications

I : c'est l'intensité, en ampère.

T : c'est le temps, en seconde.

Q : c'est la charge électrique d'un électron, exprimée en coulomb.

LA TENSION :

Dans un fluide, ce qui met en mouvement le carburant, c'est la pression. Dans un circuit électrique, c'est la tension. La formule qui suit permet de calculer la tension électrique par la loi d'ohm :

$$U = I \times R$$

Explications

U : est la tension en volt (v) ;

I : est le courant en Ampère (A) ;

R : est la résistance en ohm.

La puissance

La puissance : c'est à la fois l'énergie consommée par un système (en général sous forme de chaleur, mais aussi sous forme mécanique ou lumineuse) ; c'est aussi l'énergie maximale qui peut être fournie par un système.

Sachez que l'énergie peut prendre une multitude de formes. Par exemple, elle peut devenir une énergie thermique, lumineuse, mécanique, électrique, cinétique, chimique, etc. Dans un système, il n'est pas rare que l'énergie se transforme et passe d'une forme à une autre. C'est ce qui se passe dans une pile : celle-ci crée une tension dans un circuit. Pour cela, elle utilise des réactions chimiques qui vont libérer un

peu d'énergie et l'utiliser pour créer un déséquilibre de charges électriques afin de charger ses bornes « + » et « - ».

d) La notion de conductibilité

Tous les matériaux ne se comportent pas de la même façon en présence d'un courant électrique. Leur comportement diffère en fonction d'un paramètre que l'on appelle la conductibilité.

La conductibilité, c'est la capacité qu'ont les matériaux à se faire traverser par un courant électrique.

Du point de vue de la conductibilité, il existe quatre grandes familles de matériaux :

Famille de matériaux	Propriétés
Les isolants	Conductibilité nulle. Le courant ne passe pas à travers ces matériaux. Citons par exemple le plastique ou le verre.
Les semi- conducteurs	Conductibilité faible, mais variable suivant la situation. Ils ne se laissent pas traverser facilement par un courant, sauf dans certaines situations. Là encore, tout dépend de certains paramètres. Par exemple, certains sont complètement isolants, mais laissent passer le courant quand on les éclaire ou quand on les chauffe. Citons quelques exemples : le silicium ou le germanium.

Les conducteurs métalliques	<p>Conductibilité forte.</p> <p>Ils se laissent traverser par le courant quelles que soient les circonstances.</p> <p>Le seul problème est qu'une partie de l'énergie des charges électriques du courant va se dissiper dans le métal sous forme de chaleur. Citons le cuivre, l'aluminium, l'or, le fer...</p> <p>Tous les métaux en somme.</p>
Les supraconducteurs	<p>Conductibilité infinie.</p> <p>Ils se laissent traverser par un courant sans opposer la moindre résistance. Un courant qui rentre dans le matériau ressortira sans aucune perte : pas de création de chaleur. Enfin, ce n'est vrai qu'en théorie : dans la réalité, un supraconducteur possède toujours quelques impuretés qui seront la cause de pertes, aussi infimes soient-elles, par effet Joule (chaleur).</p>

Tableau 1 : familles de matériaux.

Dans les circuits électroniques, on utilise aussi bien des semi-conducteurs que des conducteurs métalliques. Sur un montage électronique, on utilise aussi des isolants.

e) La masse et notion de référentiel

a) Le référentiel

Lorsque l'on prend une mesure, on fixe un point qui va nous permettre de prendre cette mesure. C'est le référentiel.

Un référentiel est un point que l'on définit comme étant la référence. La référence est donc un choix arbitraire, c'est pour cela que l'on parle d'une référence et pas de la référence.

b) La masse

La masse est le point de référence qui permet de mesurer la tension. C'est le 0 volt du circuit. Représentation

La masse est représentée dans un circuit par ce symbole qui peut légèrement différer selon les schémas.

Représentation :

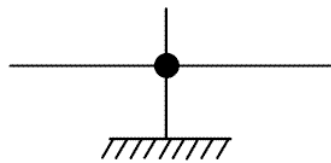


Figure 12 : symbole de la masse.

c) La terre

Le symbole de terre représente la mise à terre du réseau de distribution électrique 230v.

Représentation :

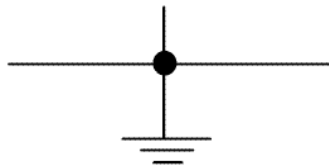


Figure 13 : symbole de terre.

Effet joule

Lorsqu'un conducteur métallique ou un semi-conducteur est traversé par un courant, il chauffe, et ça, c'est inévitable. Une partie de l'énergie électrique transportée par le courant est transformée en chaleur. C'est ce qu'on appelle l'effet joule (nom de son inventeur).

La puissance est par conséquent la quantité d'énergie que peut fournir un système à un autre système durant un temps donné.

En électronique, la puissance est définie par la relation entre l'intensité et la tension par cette formule :

$$P = U \times I$$

f) Notions de Schémas électrique

Le schéma d'un circuit électronique est une représentation conventionnelle de ses fonctions et interconnexions pour faciliter la compréhension et la lecture.

Pour réaliser une carte ou autre dispositif électronique, on part d'un schéma bloc ou de principe.

Le schéma sera composé des composant ou élément actifs et passifs ainsi que les accessoires représentés par des symboles en vigueur.

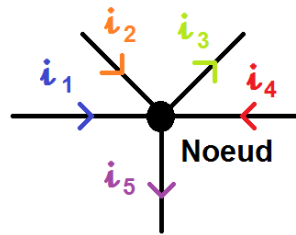
Type des schémas :

Nous avons :

- Le schéma de principe ou bloc ;
- Le dessin d'implantation ;
- Le plan de câblage ;
- Le dessin de définition ;
- Le dessin de définition ;
- Le dessin d'opération ;

a) Lois sur les fils et leurs liaisons

- ❖ Un nœud est une connexion qui relie au moins trois fils
- Illustration :



Loi des noeuds :

$$i_1 + i_2 + i_4 = i_3 + i_5$$

Figure 14 : illustration de la loi de nœud.

- ❖ Une branche est une portion de circuit (un fil) comprise entre deux nœuds consécutifs ;
- ❖ Une maille est un chemin fermé, formé d'un ou de plusieurs fils (ou branches) et de dipôles dans un circuit électrique.

Illustration :

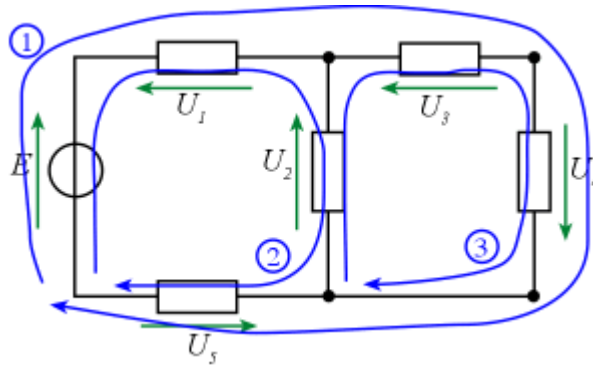


Figure 15 : illustration de la loi de maille.

b) Lois concernant les dipôles

Un dipôle est un composant électronique ayant deux borne ou pattes.

Représentation :

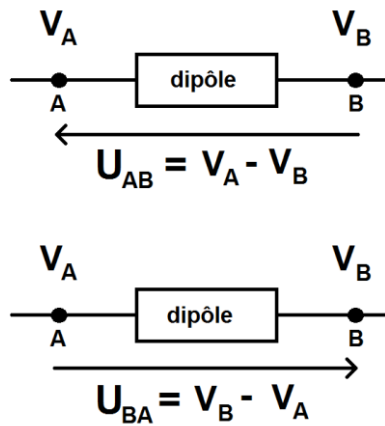


Figure 16 : représentation des dipôles.

- ❖ Deux dipôles sont en série lorsqu'ils appartiennent à la même branche ;
- ❖ Deux dipôles sont en dérivation (ou en parallèle) lorsqu'ils forment une maille.

Pour résumer, voici un schéma illustratif du vocabulaire présenté ci-dessus :

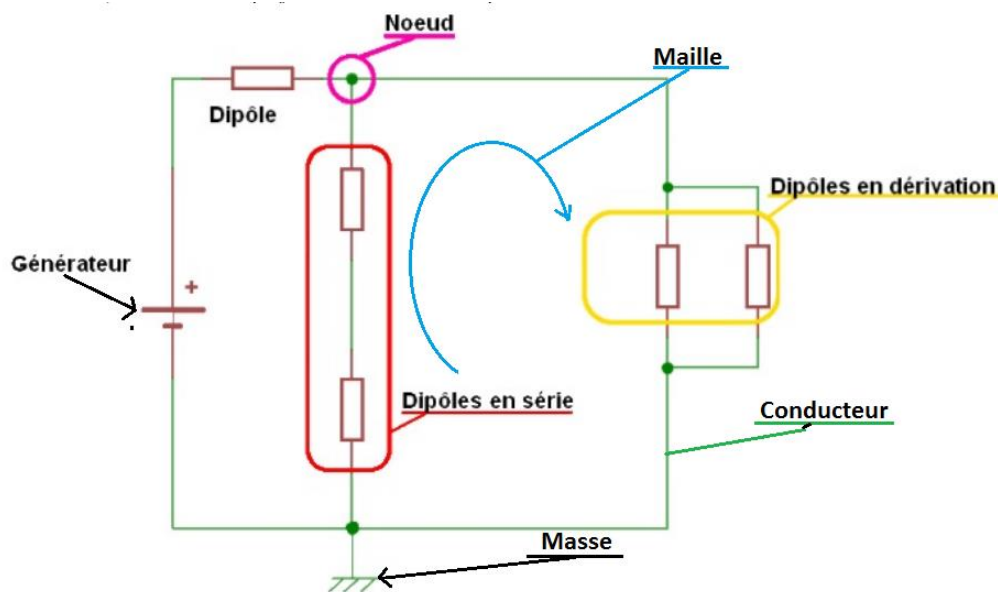


Figure 17 : Schéma récapitulatif.

1.3.2 Les composants électroniques

C'est un élément, qui, assemblé avec d'autres selon un schéma électronique, permet de réaliser une ou plusieurs fonctions électroniques.

Vous noterez que l'on parle ici d'assemblage de composants. En effet, si quelques composants peuvent fonctionner seuls, c'est très rare, et bien souvent, différents composants sont associés pour répondre aux besoins de l'électronicien.

Prenons quelques exemples, lorsque vous utilisez un amplificateur audio, la fonction électronique est l'amplification du signal d'entrée.

Il existe des composants électroniques de toute forme et de toute taille : du condensateur haute tension, cylindrique, la résistance rectangulaire de 0,5 x 1 mm ! Certains sont plats, d'autres creux. D'autres encore émettent de la lumière (LED, écrans LCD, etc.) ou du son (buzzer, par exemple). Enfin, beaucoup présentent un marquage permettant de les identifier, de les orienter et de respecter leur polarité.

a) Quelques composant de base :

La résistance :

C'est un composant qui permet de résister au passage des électrons. Cette propriété laisse à ce que l'on utilise ces composants partout où on a besoin de limiter le courant.

Toutes les résistances ne s'opposent pas de la manière, elles sont caractérisées par leurs valeurs, que l'on appelle valeur ohmique. Une résistance est donc exprimée en fonction de cette valeur, tout simplement appelé « Ohm » avec comme symbole (Ω).

Lorsqu'on place une résistance dans un circuit, elle provoque toujours une chute de tension parce qu'elle freine le passage des électrons. En électronique les résistances sont utilisées pour réduire la pression des électrons que l'on appelle la tension électrique, exprimée en Volts (V).

Il existe plusieurs sortes de résistances des :

- Résistances fixes ou linéaire : dont la valeur ohmique ne varie pas.

- Photorésistances : dont la valeur ohmique dépend ou varie avec la lumière, oui ça varie avec la lumière.
- Thermistances : dont la valeur ohmique dépend ou varie avec la température, cela nous permettra de réaliser beaucoup des choses.
- Trimmers : dont la valeur ohmique varie
- Varistances : dont la valeur ohmique dépend ou varie avec la tension.
- Potentiomètres : dont la valeur ohmique varie suivant un angle donné.

Toutes ses résistances ont des symboles différents

Symbole :

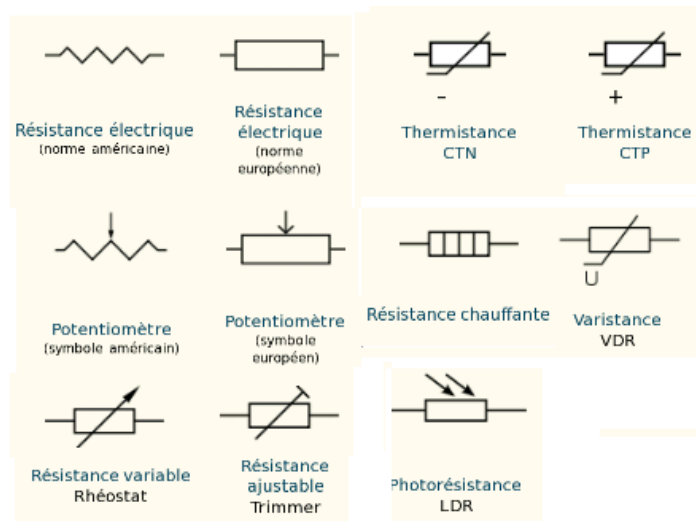


Figure 18 : symboles résistances.

La diode

Une diode est un composant électronique qui a une propriété importante qui est celle de ne laisser passer le courant que dans un sens.

Une diode possède deux parties : une partie que nous appelons l'anode et une autre la cathode. Je vous invite également de noter l'anode par la lettre A et la cathode par la lettre K comme le montre son symbole.

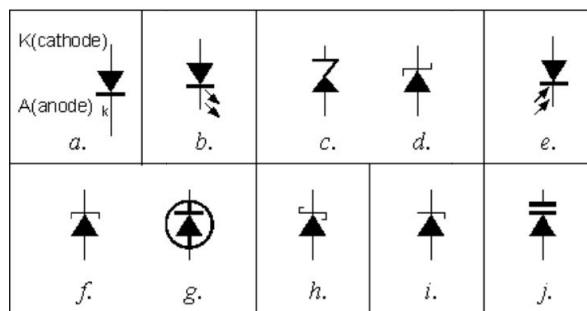
Le sens normal de conduction est celui-ci : le courant va de l'Anode à la Cathode. Cela veut dire que l'anode sera reliée au potentiel haut et la cathode au potentiel bas pour permettre ou imposer ce sens de

courant. Si la cathode est au potentiel Haut que l'anode, la diode restera bloquée. D'où elles sont caractérisées par une tension dite inverse, c'est-à-dire une tension à partir de laquelle le courant peut aller de la cathode à l'anode et qui peut causer également une destruction du composant.

Il existe plusieurs sortes des diodes :

- Simples ou au silicium
- ZENER permettant de réguler la tension tunnel Schottky
- Électroluminescentes qui produisent de la lumière.
- Des photodiodes qui dépendent de la lumière pour conduire.
- Des diodes à capacité variable appelé diode VARICAP pour la radio ou la télévision.

Symbole :



Symbole de Diode : a – diodes standard, b - LED, c, d - Zener, e - photodiode, f, g –diode tunnel, h - diode Schottky, i – diode à avalanche (breakdown), j – diode varicap

Figure 19 : différents symboles des diodes.

Le condensateur

Le condensateur sert à stocker ou emmagasiner une énergie électrique durant un temps bien déterminé, sa capacité de stockage s'exprime en Farads mais souvent on utilise le microfarad (μF) car dans la fabrication il est difficile de fabriquer un condensateur de quelques Farads.

Les condensateurs serviront plus lors du filtrage, une opération qui permet d'obtenir une vraie tension continue à partir d'une autre variable en amplitude.

Il existe plusieurs sortes des condensateurs, dont :

- Des condensateurs au céramique
- Des condensateurs électrolytiques
- Des condensateurs variables
- Des condensateurs polyester

Symbole :

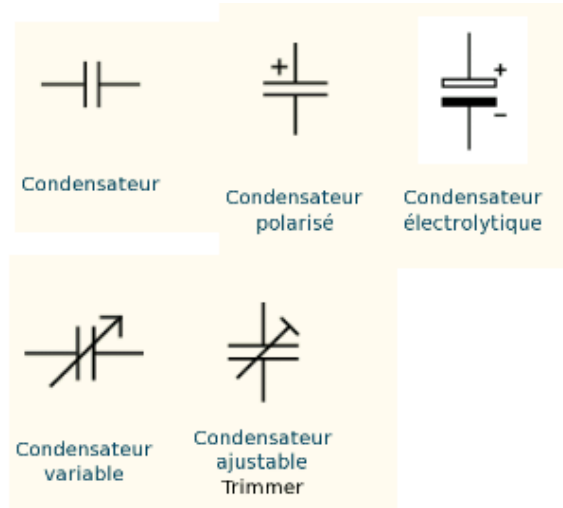


Figure 20 : symbole condensateur.

La bobine

Une bobine est un composant sur base des enroulements que nous appellerons des spires. Il permet aussi de stocker l'énergie sous forme de courant électrique, contrairement au condensateur qui stocke en tension.

On exprime sa capacité de stockage en Henry (H) mais souvent on utilisera le micro Henry (μH) et le milli Henry (mH). Ce composant sera utilisé lors du lissage du courant.

Le transistor

Ce composant a permis à fabriquer des circuits que nous appellerons des amplificateurs opérationnels. Des circuits qui ont vraiment révolutionné l'électronique et amélioré la conception des systèmes électroniques.

C'est un composant électronique à 3 pattes, ou 3 broches dont une borne de :

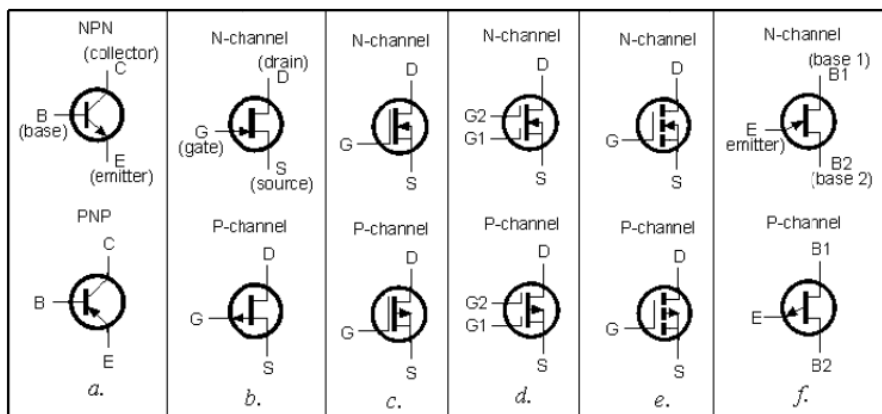
- Commande : base (B)
- Collecter : collecteur (C)
- D'émission : émetteur (E)

Types des transistors :

Il existe plusieurs types de transistors :

- Des transistors bipolaires : commandé par le courant (deux catégories : NPN et PNP)
- Des transistors MOS (Métal Oxyde Silicium) : commandé par la tension (deux catégories Canal N et Canal P).
- Des transistors MOSFET : deux catégories Canal N et Canal P
- Des transistors Bipo-MOS : que la technologie développe actuellement

TRANSISTORS



Symbole de Transistor : a -transistor bipolaire, b –TEC (FET), c - MOSFET, d –MOSFET à double gâchette (dual gate MOSFET), e - MOSFET à canal induit (enrichissement), f – transistor uni jonction

Figure 21 : différents symboles des transistors.

Le transistor possède plusieurs caractéristiques qui sont toujours données dans des livres, et souvent sur le site officiel des composants électroniques.

En conclusion, il existe plusieurs composants électroniques qui sont utilisés dans des circuits ou des montages électroniques spécialisés.

Cette image illustre quelques composants électroniques de base :

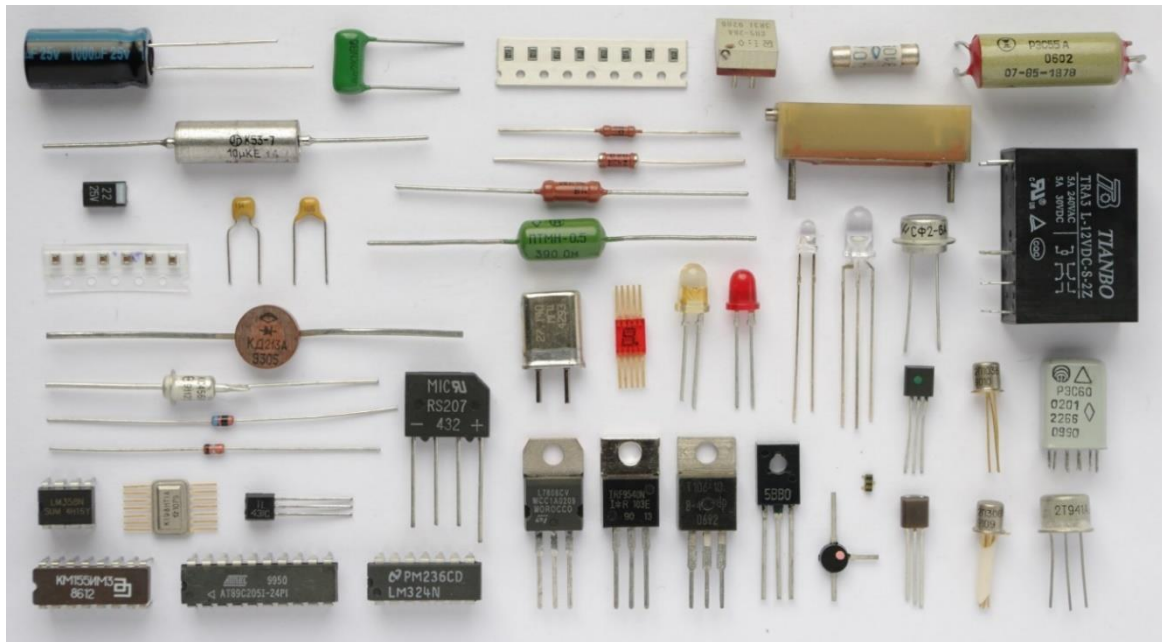


Figure 22 : Quelques composants électroniques.

b) Quelques symboles utiliser en électronique :

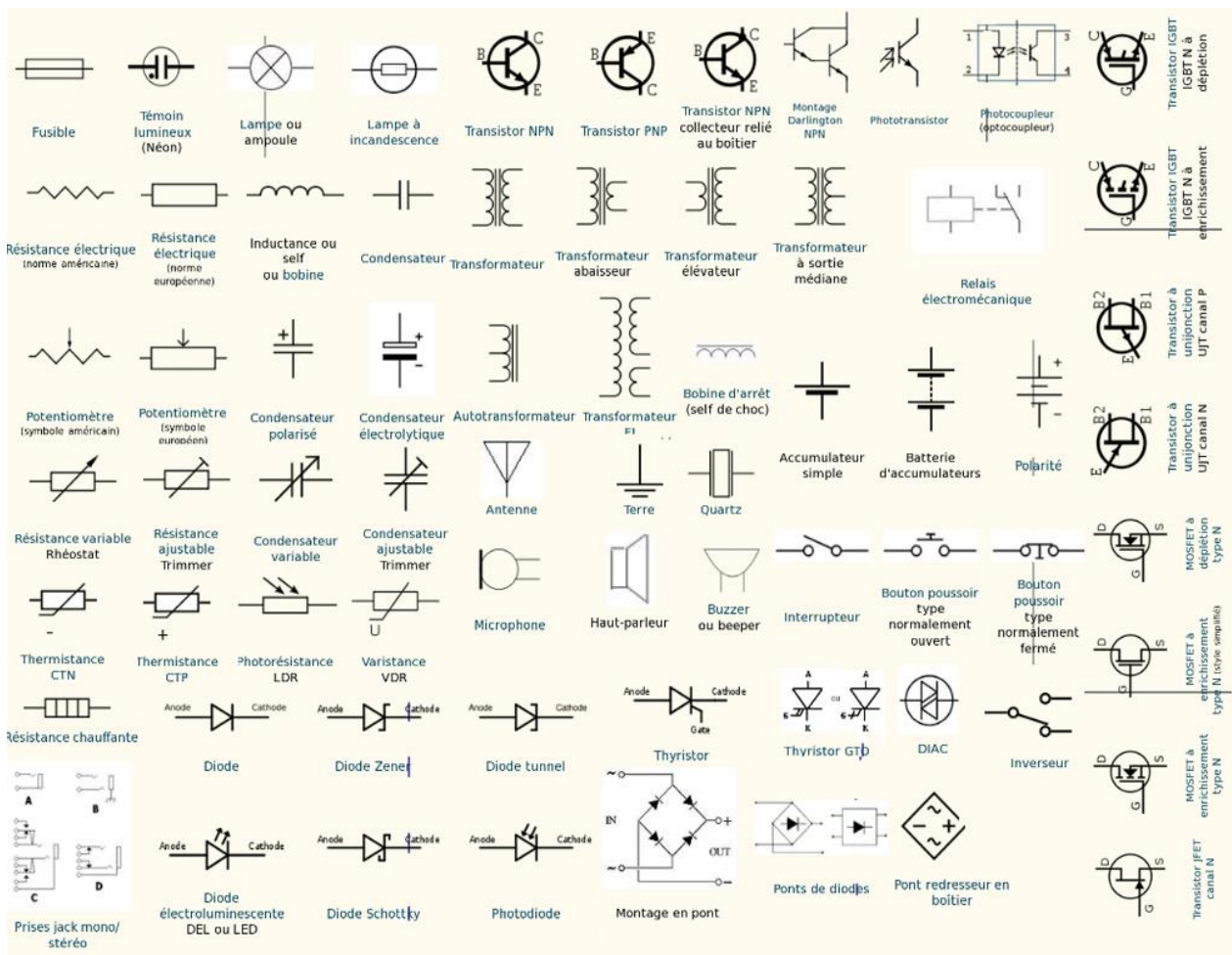


Figure 23 : Quelques symboles de composants électroniques.

2 CADRAGE DU PROJET

2.1 Présentation de la méthode d'ordonnancement

Les problèmes d'ordonnancement sont apparus au départ dans la planification des grands projets. Le but était de gagner du temps sur leur réalisation. Des tels projets sont constitués de nombreuses étapes, également appelées tâches et des réalisations temporelles existent entre ces dernières. Par exemple :

- Une étape doit commencer à une date précise ;
- Un certain nombre de tâches doivent être déterminées pour pouvoir démarrer d'autres ;
- Deux tâches ne peuvent être réalisées en même temps (elles utilisent une même machine par exemple) ;
- Chaque tâche nécessite une certaine quantité de main d'œuvre. Il faut donc éviter, à chaque instant, de dépasser la capacité totale de main d'œuvre disponible.

Toutes ces contraintes ne sont pas à prendre en compte dans la résolution des problèmes. Ici, nous allons nous intéresser uniquement aux types de contraintes.

On cherchera à déterminer une planification, un ordonnancement des étapes qui minimise le temps total de la réalisation du projet. A partir de cette réalisation, nous verrons que le temps de certaines étapes peut

éventuellement être modifier sans entrainer un retard du projet, alors que d'autres étapes dites « critiques », retardent entièrement le projet au moindre retard local.

Actuellement il existe deux méthodes scientifiques et techniques basées sur la théorie de graphe pour résoudre le problème central d'ordonnancement (P.C.O) :

- La méthode française : méthodes de potentiel Métra (MPM) en 1958.
- La méthode américaine PERT: « program Evaluation and Research Task » en 1958.

Dans le cadre de notre projet, nous utiliserons la méthode américaine program Evaluation and Research Task "PERT".

2.2 Identification des taches du projet

L'écart de réalisation de projet s'effectue par l'analyse du projet en précisant les principales caractéristiques de taches à savoir :

- La durée de l'exécution de l'ouvrage ;
- Les couts de l'exécution ;
- les matériels nécessaire à l'exécution ;
- l'enchainement logique des taches.

2.3 Diagramme de la méthode relative à l'ensemble des opérations

2.3.1 Présentation de la méthode PERT

La méthodes PERT (program ou projet Evaluation and Review Technique), est une méthode conventionnelle utilisable en gestion de projet, développée par la marine américaine dans les années 1950. Elle est censée être capable de représenter et d'analyser de manière logique les taches et le réseau des taches à réaliser dans un projet.

Le terme PERT est l'acronyme de program (ou projet) évaluation and review technique, ce qui signifie « technique d'évaluation et d'examen de programme » ou « de projet », ou encore « technique d'élaboration et de mise à jour de

programme » ; c'est également un jeu de mots avec l'adjectif anglais « pert », signifiant « malicieux », mutin ».

Dans le vocabulaire de tous le jour, un projet désigne une action future. Dans l'ingénierie (activité des ingénieurs et techniciens), projet désigne l'ensemble des actions en cours d'élaboration. On utilise un graphe de dépendances. Pour chaque tâche, on indique une date de début et de fin au plus tôt et au plus tard. Le diagramme permet de déterminer le chemin critique qui conditionne la durée minimale du projet.

Le but est de trouver la meilleure organisation possible pour qu'un projet soit terminer dans le meilleur délai, et d'identifier les tâches critiques, c'est-à-dire les tâches qui ne doivent souffrir d'aucun retard sous peine de retarder l'ensemble du projet.

Cette méthode d'organisation est sans doute l'une des plus exigeantes en rigueur mais aussi l'une des plus puissantes.

2.4 Planning d'exécution des tâches, estimations des durées

Après le recensement des tâches qui seront exécutées, nous pouvons estimer la durée de l'exécution de notre projet :

Tache	Désignation	Durée	Cout en USD	Prédécesseur
D	Début du projet	0	-	-
T2	Recueil de données	8	0.5	D
T2	Analyse de l'existant	5	0.2	T1
T3	Bilan critique de l'existant	3	5	T2
T4	Choix et proposition des solutions	2	25	T3
T5	Conception	30	100	T4

T6	Développement de l'application	45	70	T5
T7	Achat des composant électronique	14	50	T11
T8	Achat de matériel électronique	10	60	T6
T9	Acquisition matériels	6	5	T7
T10	Aménagement locaux	14	20	T9
T11	Elaboration du cahier de charge	6	10	T6
T12	Installation des matériels	7	10	T8 et T10
T13	Configuration du nouveau système d'information	15	3	T12
T14	Elaboration manuel d'utilisation	8	20	T12
T15	Formation du personnel	6	25	T13 et T14
T16	Test du nouveau système	2	0.2	T15
T17	Livraison du système	1	5	T16
F	Fin du projet	0	-	T17

Tableau 2 : Planning d'exécution des tâches, estimations des durées.

2.5 Graphe PERT

2.5.1 Détermination des niveaux du graphe du projet

Pour construire le graphe, nous allons partir du tableau de dictionnaires des successeurs, c'est-à-dire les tableaux qui renseignent des descendants. Ce dictionnaire sert à déterminer le niveau de graphe de manière ordonnée.

Notre graphe a seize niveaux que voici :

N1 = {T1}	N11 = {T12}
N2 = {T2}	N12 = {t13 et T14}
N3 = {T3}	N13 = {T15}
N4 = {T4}	N14 = {T16}
N5 = {T5}	N15 = {T17}
N6 = {T6}	N16 = {f}
N7 = {T18 ET T11}	
N8 = {T7}	
N9 = {T9}	
N10 = {T10}	

2.5.2 Présentation du graphe PERT

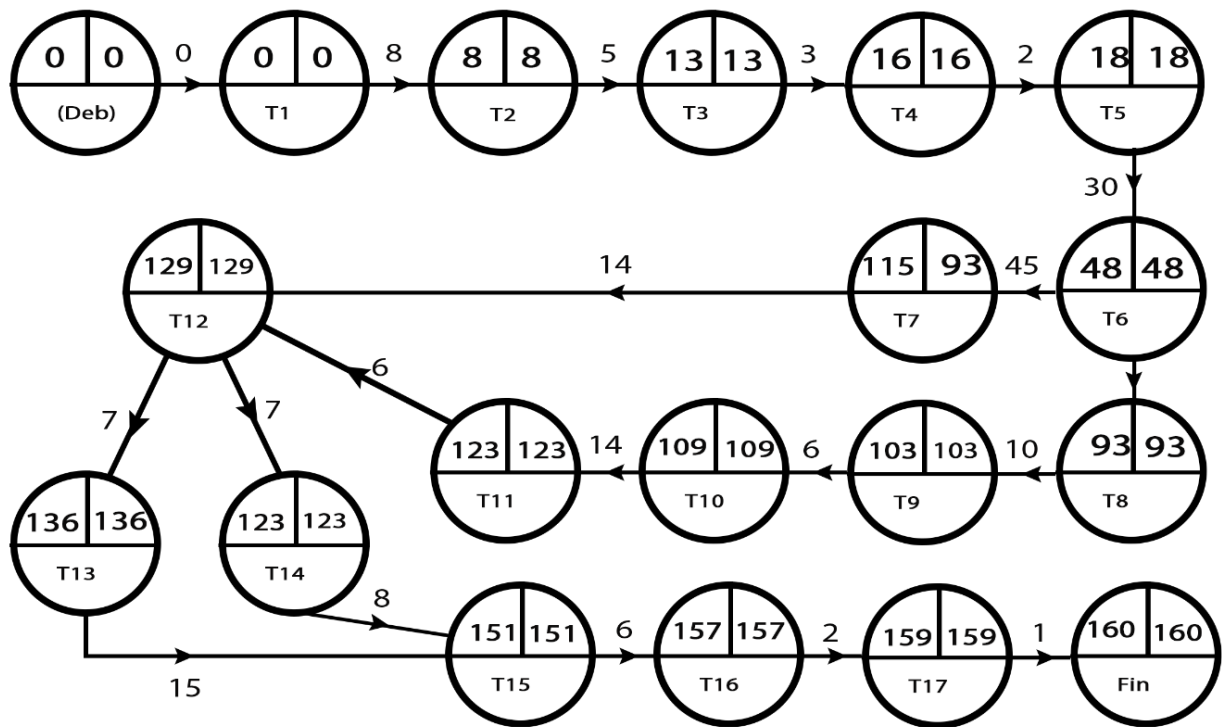


Figure 24 : Présentation du Graphe PERT.

2.6 Délai de réalisation du projet

2.6.1 Calcul des dates au plus tôt et dates au plus tard

Note : au niveau de ce point, nous allons nous attarder sur les considérations théoriques avant de nous consacrer infra, dans un tableau synthèse au calcul des différentes valeurs.

a) Détermination de la date au plus tôt (DTO)

Dans cette démarche, on choisit pour T_x (DTO) la valeur du plus long chemin (la valeur la plus grande) entre toutes les contraintes d'antériorités.

b) Détermination de la date au plus tard (DTA)

Ici il est question de choisir par T^*x (DTA) la valeur minimale entre toutes les contraintes de succession.

2.6.2 Calcul des marges libres et marges locales

a) Détermination de marge libre

La marge libre est le retard maximum que l'on peut se permettre dans la mise en route de la tâche sans remettre en cause la date au plus tôt d'une autre tâche. Elle est pour X a :

$$\text{Min} = (TY - TX - dx)$$

b) Détermination de marge totale

La marge totale constitue le retard maximum que l'on peut prendre dans la mise en exécution de la tâche sans remettre en cause les dates au plus tard de succession ou suivante. Cette marge est donc :

$$MT = T^*x - TX$$

2.6.3 Chemin critique

On appelle chemin critique, un chemin passant par les tâches critiques. Celle-ci n'accepte aucun délai flottage c'est-à-dire aucun retard.

2.7 Calcul de cout du projet :

$$0.5 + 0.2 + 5 + 25 + 100 + 70 + 50 + 60 + 5 + 20 + 10 + 10 + 3 + 20 + 25 + 0.2 + 5 = 408,9\$$$

$$\text{Imprévues (10\% du cout du projet)} \quad 408,9 * 10/100 = 40.89\$$$

$$\text{Cout total du projet} = \text{cout du projet} + \text{imprévues}$$

$$\text{Cout total du projet} = 408,9 + 40.89 = 449,79\$$$

3 MODELISATION DU NOUVEAU SYSTEME D'INFORMATION

3.1 Modélisation du logiciel

3.1.1 But de la modélisation

Le modèle permet de mieux répartir les tâches et d'automatiser certaines d'entre elles. C'est également un facteur de réduction des coûts et des délais.

3.1.2 Cycle de vie d'un logiciel

Les modèles de cycle de vie du logiciel décrivent à un niveau très abstrait et idéalisent les différentes manières d'organiser la production, les étapes, leur ordonnancement, et parfois les critères pour passer d'une étape à l'autre sont

explicités (critère de terminaison d'une étape revue de documents, critères de choix de l'étape suivante, critères de démarrage d'une étape).

Il faut souligner la différence entre étapes (découpage temporel) et activités (découpage selon la nature du travail). Il y a des validations et la vérification), voir dans toutes les étapes (ex : la documentation).

Rappelons aussi la différence entre vérification et validation (B. Boehm, 76) :

- Verification: « are we building the product right? » (« construisons nous le produit correctement ? » : correction interne du logiciel, concerne les développeurs).
- Validation : « are we building the right Product » (« construisons-nous le bon produit ? » : adaptation vis à vis des besoins des utilisateurs, concerne les utilisateurs.

3.1.3 Diagramme de cas d'utilisation (DCU)

Un cas d'utilisation « Use Case » représente un ensemble de séquence d'action qui sont réaliser par le système et qui produisent un résultat observable intéressante pour un user (acteur) particulier.

Chaque cas d'utilisation correspond à une fonction métier d'un système selon le point de vue de ses user (acteurs).

Tout le cas d'utilisation sont représentés par un verbe à l'infinitif suivi d'un complément du point de vue de l'acteur.

Dans le diagramme de cas d'utilisation tous les acteurs sont externes au système d'information étudié.

❖ Relation entre cas d'utilisation

Trois type de relation standard entre cas d'utilisation sont proposés par UML :

- « Include » : le cas d'utilisation incorpore explicitement et de manière obligatoire un autre cas d'utilisation à l'endroit spécifié.
- « Extend » : le cas d'utilisation incorpore implicitement de manière facultative un autre cas d'utilisation à l'endroit spécifié.

- Généralisation : les cas d'utilisation descendants héritent des propriétés de leur parent.

❖ Identification des acteurs du système

Identification des acteurs est une étape très capitale qu'on ne peut pas s'en passer car le système sera utilisé par les utilisateurs que nous devons bien connaître. D'où à cette étape nous devons identifier les acteurs qui vont interagir avec le système.

D'où il faudra savoir QU'EST-CE QU'UN ACTEUR ?

Un acteur représente un rôle joué par une entité externe (Utilisateur humain, dispositif matériel ou un autre système) qui interagit directement avec le système étudié.

Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages susceptibles d'être porteurs de données.

Il existe quatre catégories d'acteurs :

- Les acteurs principaux : les personnes qui utilisent les fonctions du système.
- Les acteurs secondaires : les personnes qui effectuent des tâches administratives ou de maintenance.
- Les acteurs externes : les dispositifs matériels incontournables qui font partie du domaine de l'application et doivent être utilisés.
- Les autres systèmes : les systèmes avec lesquels le système doit interagir.

L'acteur est représenté par le formalisme suivant :

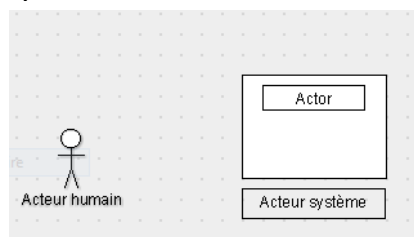


Figure 25 : représentation d'un acteur.

Pour notre système nous avons identifié les acteurs suivants :

- 🚦 Visiteur
- 🚦 Administrateur

QU'EST-CE QU'UN CAS D'UTILISATEUR ?

Un cas d'utilisateur (use case) représente un ensemble de séquences d'actions réalisées par le système et produisant un résultat observable intéressant pour un acteur particulier.

Un cas d'utilisation modélise un service rendu par le système. Il exprime les interactions acteurs/système et apporte une valeur ajoutée "notable" à l'acteur concerné.

Chaque cas d'utilisation spécifie un comportement attendu du système considéré comme un tout, sans imposer le mode de réalisation de ce comportement. Il permet de décrire ce que le futur système devra faire, sans spécifier « comment ? » il le fera. Dans le cadre de la branche fonctionnelle, le cas d'utilisation doit mettre en valeur les interactions métier entre les acteurs et le système.

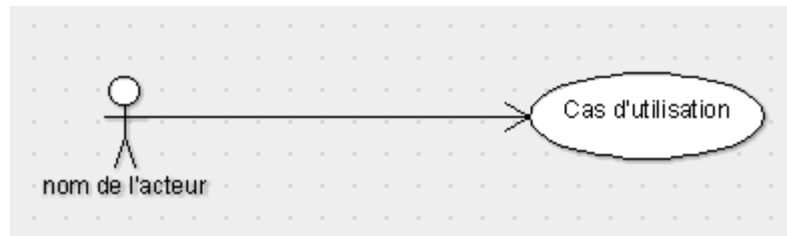


Figure 26 : représentation des interactions entre acteur et cas d'utilisation.

- Identification de cas d'utilisation :
Processus dans le tableau suivant :

Acteurs	Cas d'utilisateur
Visiteur	
Administrateur	

Tableau 3 : Tableau des acteurs du cas d'utilisation

- Les différents cas d'utilisation

Représentation du diagramme de cas d'utilisation :

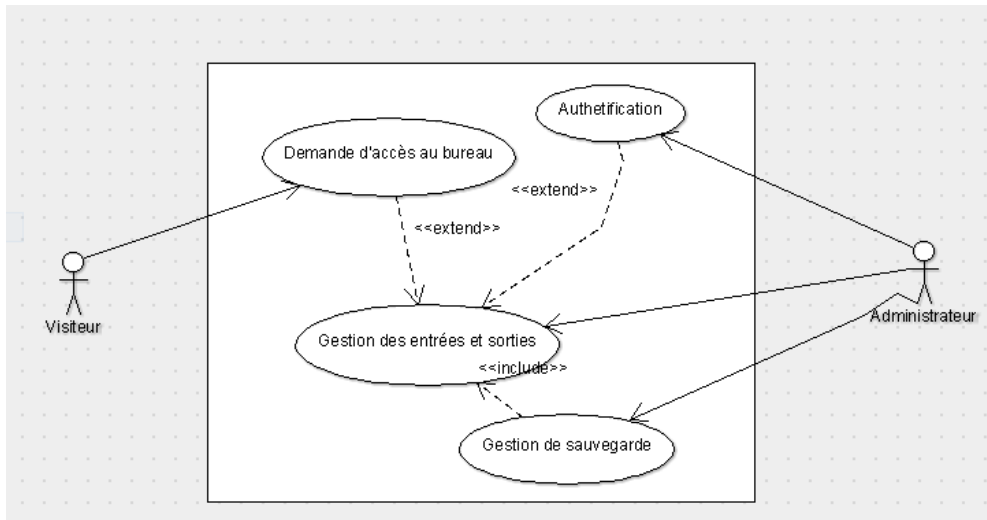


Figure 27 : Représentation du diagramme de cas d'utilisation

3.1.4 Diagramme des séquences

❖ Rappel

Les principales informations contenues dans le diagramme de séquence sont les messages échangés entre les lignes de vie présentés dans un ordre chronologique. Une ligne de vie se présente par un acteur contenant une étiquette, auquel est attaché, une ligne vertical pointillée.

Représentation du diagramme de séquence :

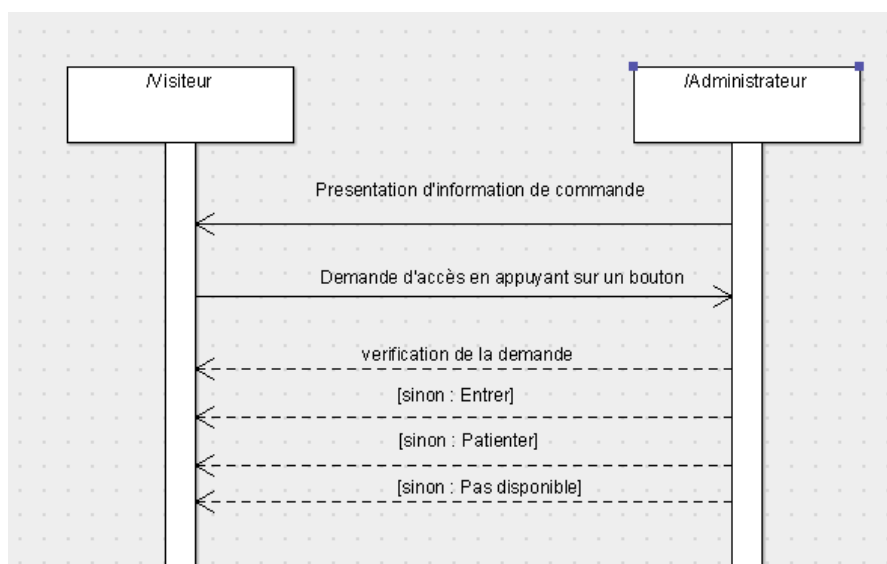


Figure 28 : Représentation du diagramme de séquence

3.1.5 Diagramme des classes

❖ Rappel

Une classe est une représentation abstraite des objets ayant les caractéristiques similaires. Ce diagramme de classe montre la structure d'un système d'une manière générale en termes de classe et relation entre cette classe. Il décrit également les liens être les objets.

Cette classe est représentée par un rectangle à trois compartiments :

En haut : le nom de la classe en gras ;

Au milieu : la liste des attributs ;

En bas : la liste des méthodes.

❖ Recensement des classes et relation entre ses classes :

Nous avons recensé les classes suivantes :

- Administrateur
- Rendez-vous
- Note

Représentation du diagramme de classe :

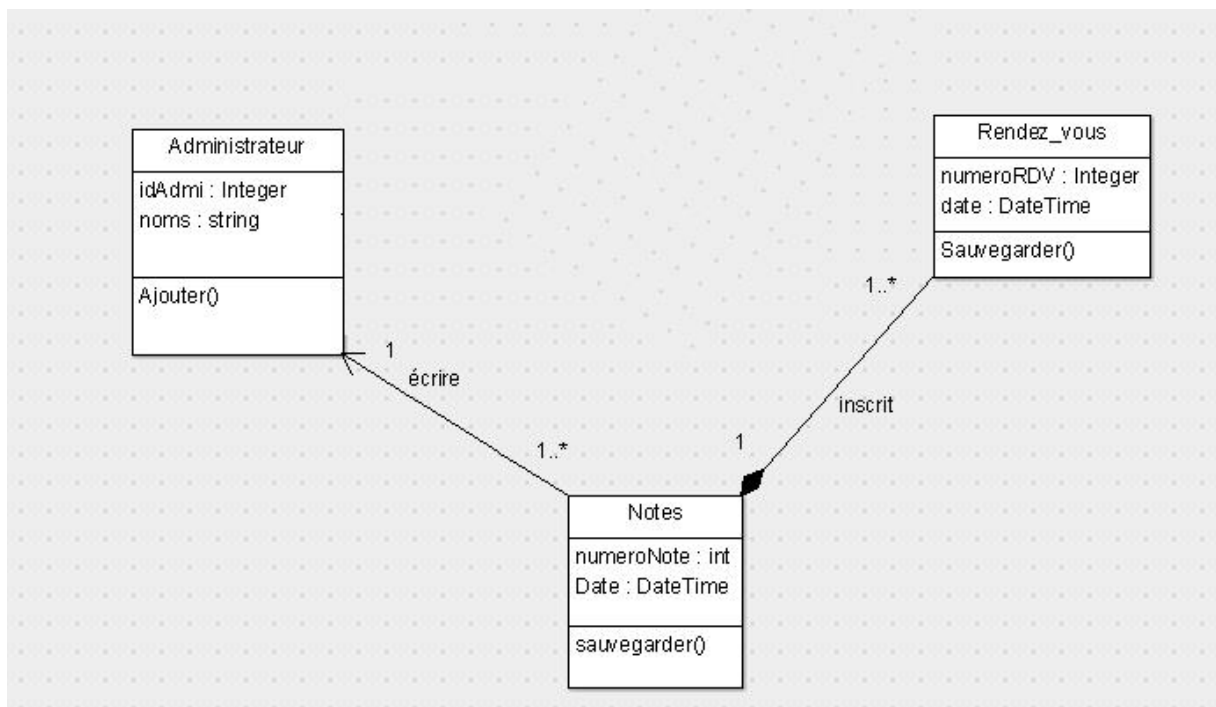


Figure 29 : Représentation du diagramme de classe.

3.1.6 Diagramme d'activité

❖ Rappel

Le diagramme d'activité est attaché à une catégorie de classe et décrit le déroulement des activités de cette catégorie.

Formalisme :

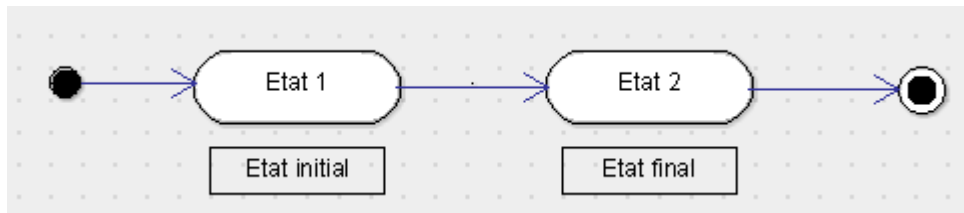


Figure 30 : Formalisme diagramme d'activité

Représentation du diagramme d'activité :

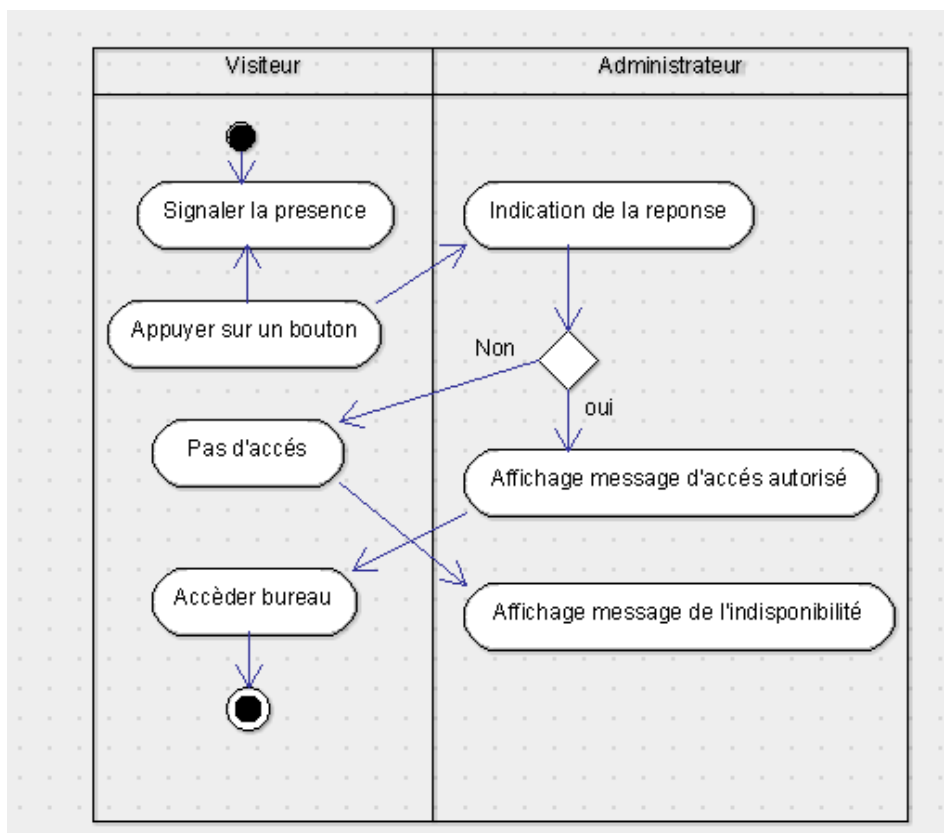


Figure 31 : Représentation du diagramme d'activité.

3.2 Modélisation du montage

3.2.1 Schéma

Le schéma d'un circuit électronique est une représentation conventionnelle de ses fonctions et interconnexions pour faciliter la compréhension et la lecture.

Pour réaliser une carte ou autre dispositif électronique, on part d'un schéma bloc ou de principe.

Le schéma sera composé des composants ou éléments actifs et passifs ainsi que les accessoires représentés par des symboles en vigueur.

3.2.2 Type des schémas

Nous avons :

- Le schéma de principe ou bloc ;
- Le dessin d'implantation ;
- Le plan de câblage ;
- Le dessin de définition ;
- Le dessin de définition ;
- Le dessin d'opération ;

a) Schéma de principe ou bloc

Appelé aussi schémas fonctionnel ou organique, il indique la fonction de chaque étage dans un rectangle, sans déterminer la technologie interne. Le schéma block est le suivant :

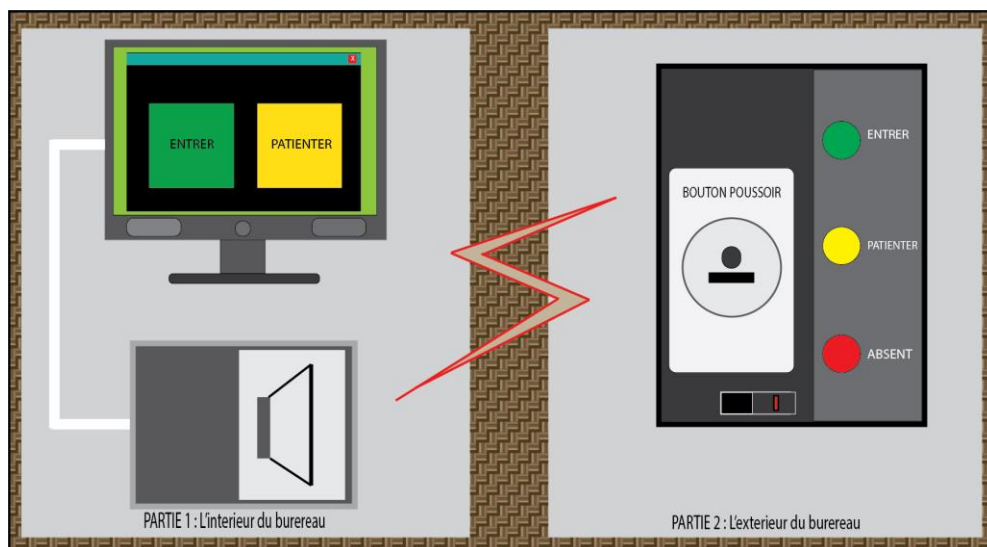


Figure 32 : schémas block

Schémas block détailler :

- La partie intérieure du montage (partie 1) :

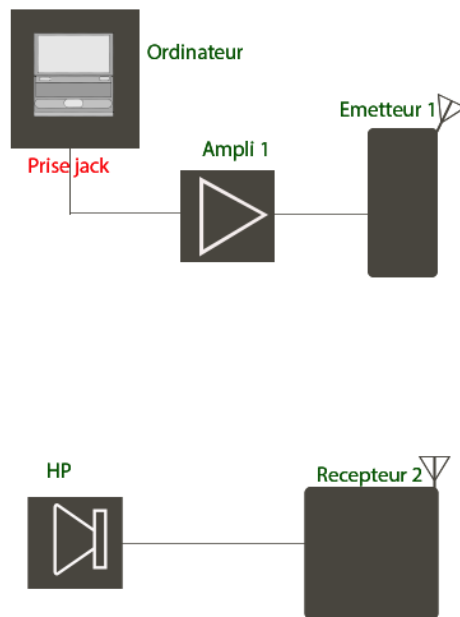


Figure 33 : Schémas block détailler 1.

- La partie extérieure du montage (partie 2) :

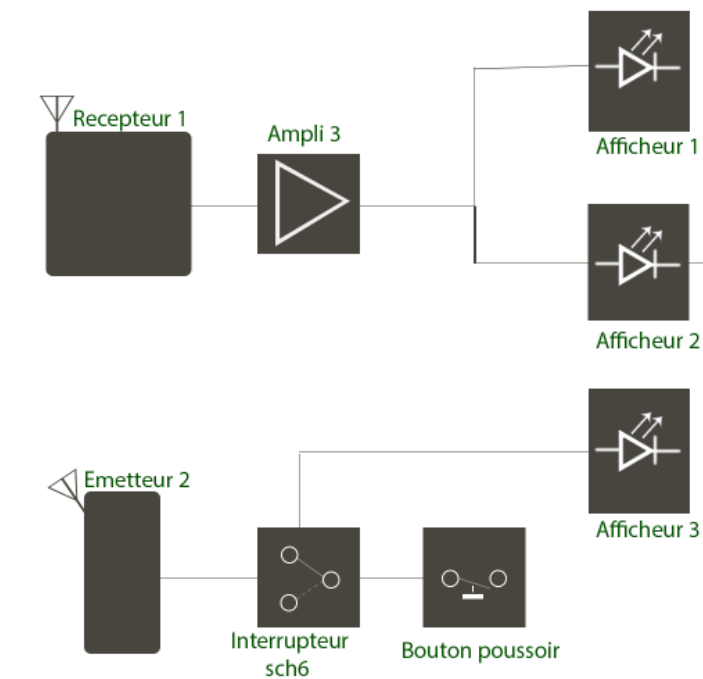


Figure 34 : Schémas block détailler 2.

Schémas block détailler au complet :

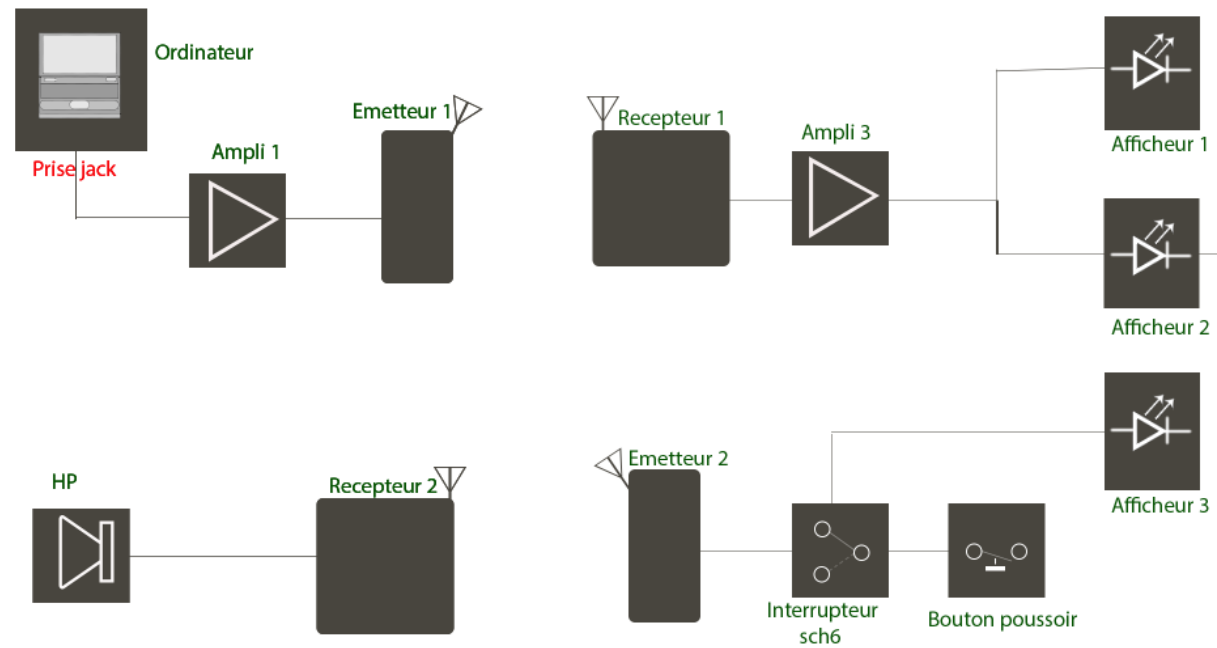


Figure 35 : Schémas block détailler au complet.

b) Schéma développé

Ici, en utilisant les symboles conventionnels à partir des blocs, on développe la technologie interne.

Liste de composant

Composant	Dimension	Quantité
transistor	BC 547	2
	2N 2222 (NPN)	5
	2N 2222 (PNP)	2
résistance	68K	1
	56K	2
	12K	1
	330	1
	10K	1

	220	2
	1	1
condensateur	2A472J	1
	2.2u	1
	100u	1
	220u	1
Diode LED	Rouge	1
	vert	1
	bleu	1
Diode ordinaire	1N4007	2
Emetteur-récepteur	Radio	2
Générateur DC	9v	2

Tableau 4 : Liste de composant

Schémas du montage

- Partie intérieur du montage (partie1) :

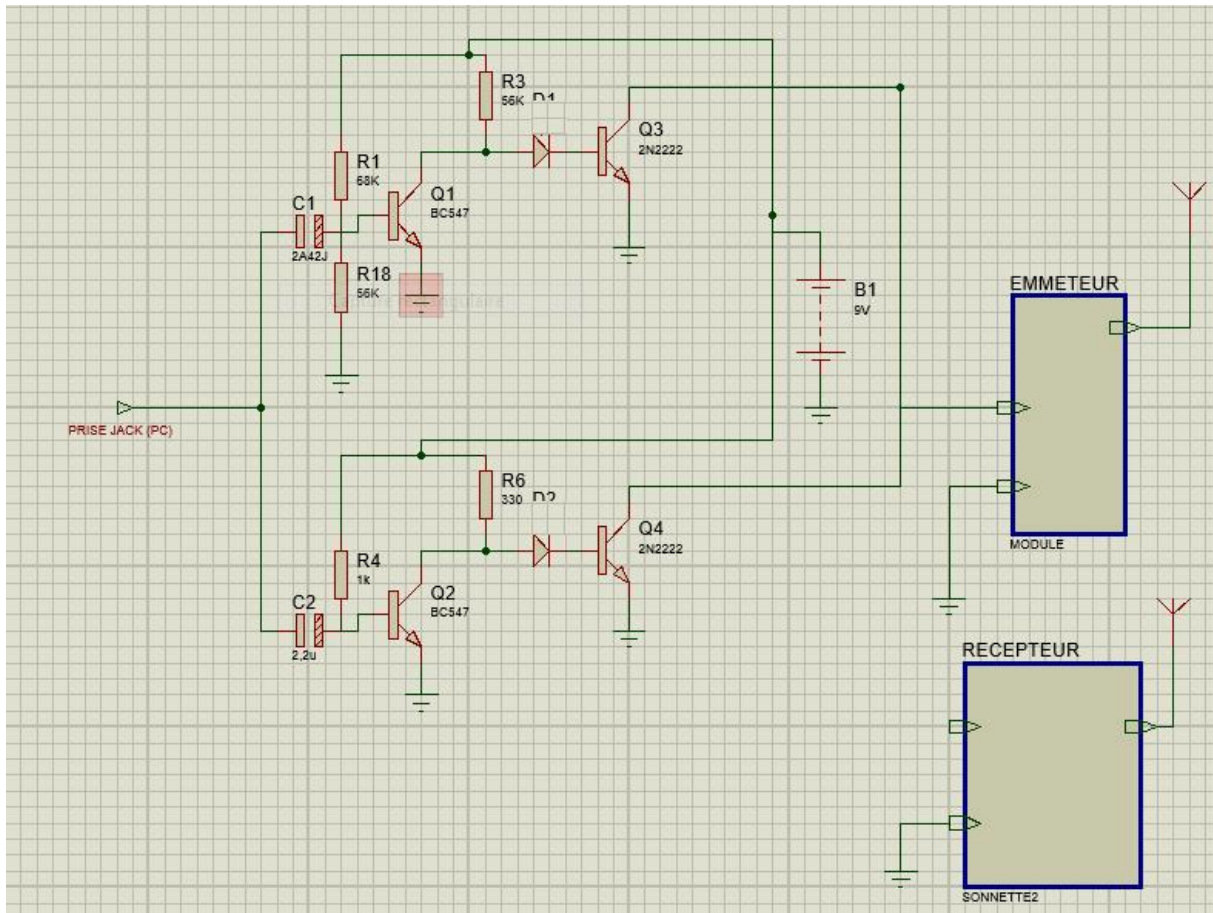


Figure 36 : Partie intérieur du montage (partie1).

- Partie extérieur du montage (partie2) :

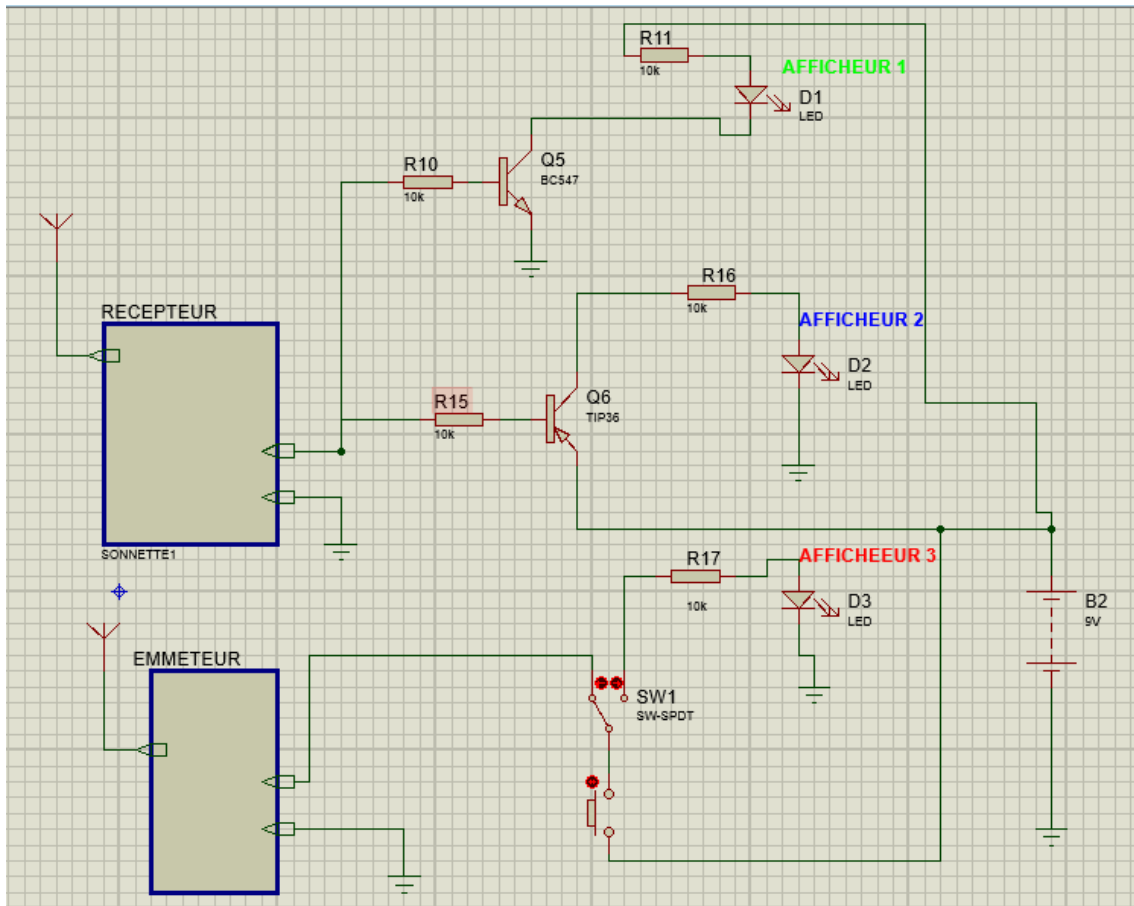


Figure 37 : Partie extérieur du montage (partie2).

❖ Schémas du montage au complet

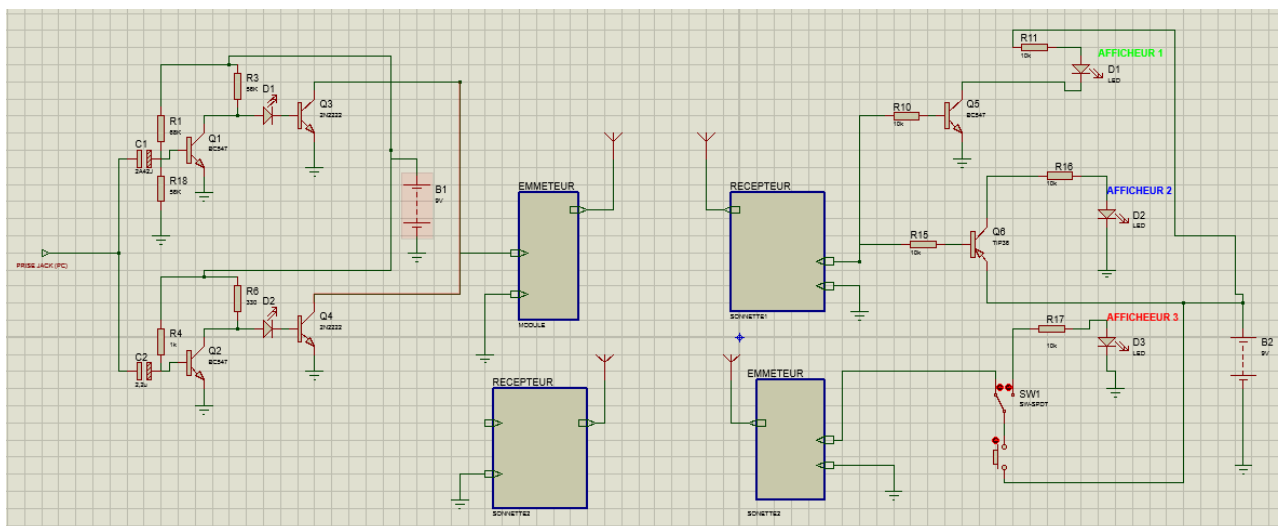


Figure 38 : Schémas du montage au complet.

c) Dessin d'implantation

Ce dessin montre la disposition des éléments sur la plaquette.

- Partie intérieur (partie1) :

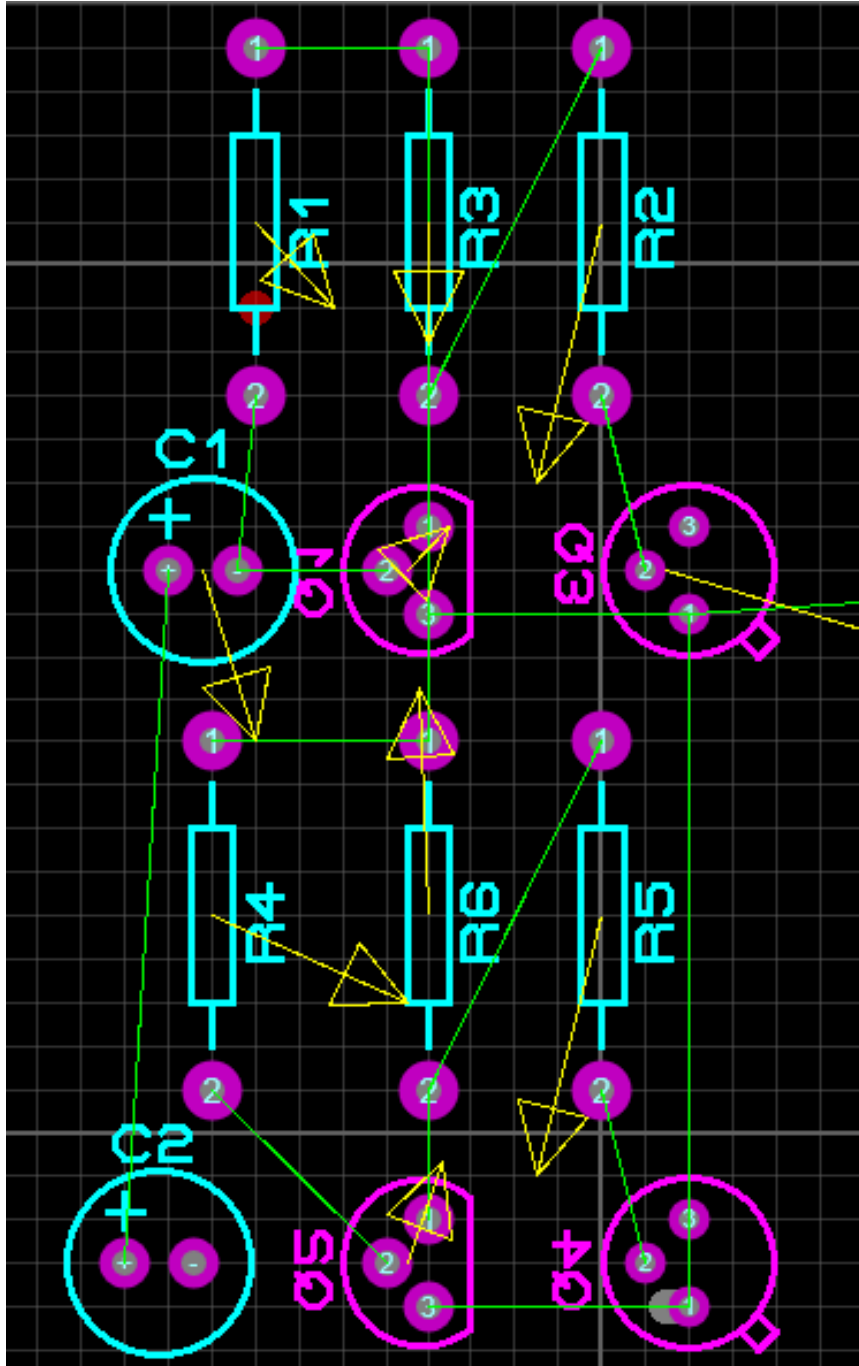


Figure 39 : Dessin d'implantation 1.

- Partie extérieur (partie 2) :

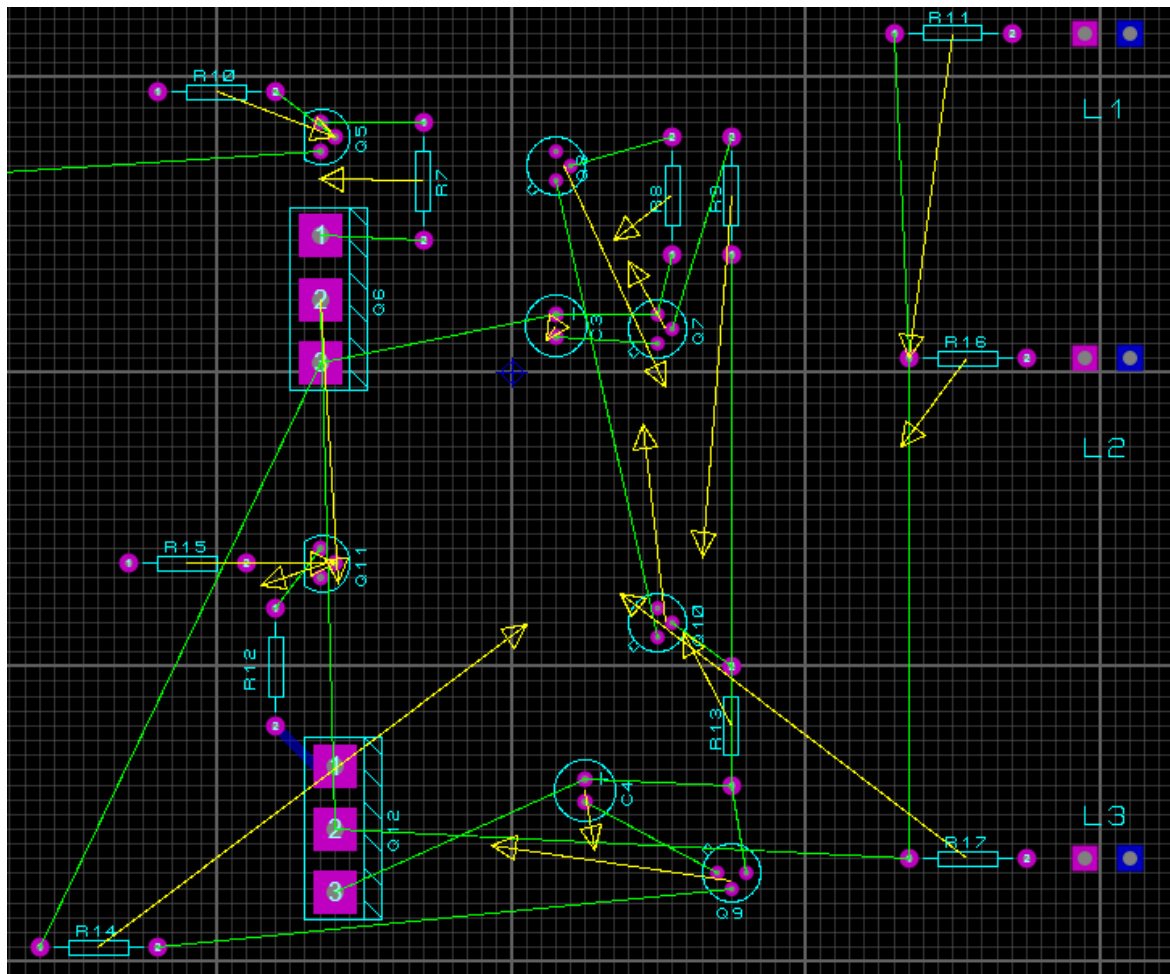


Figure 40 : Dessin d'implantation 2.

d) Plan de câblage

C'est la disposition des fils de câblage et des circuits imprimés. C'est le typon.

- Partie intérieur (partie 1)

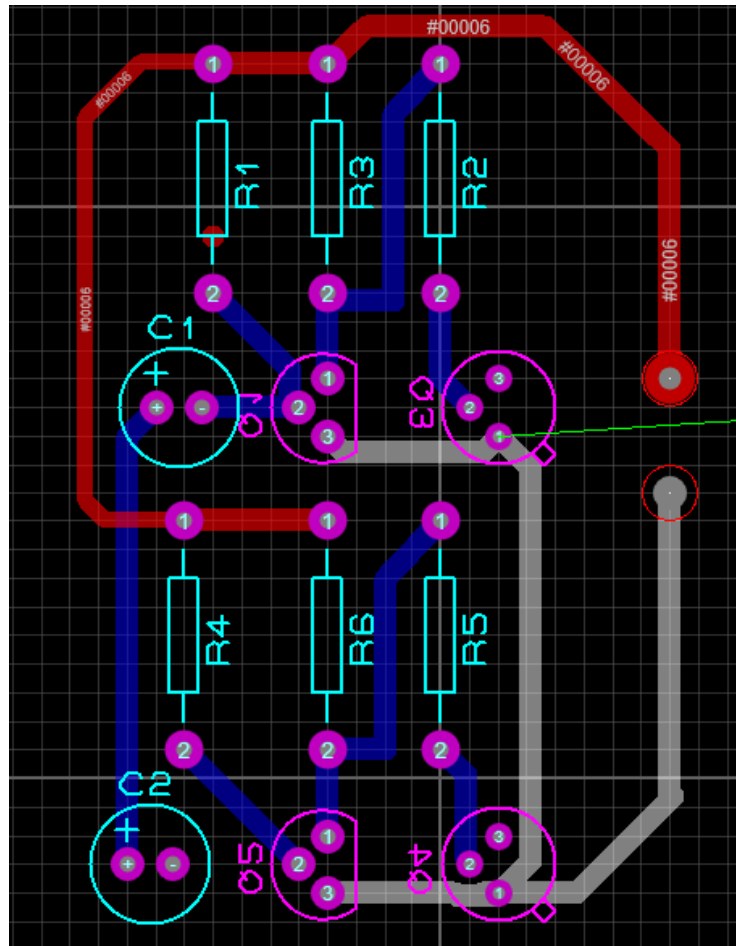


Figure 41 : Plan de câblage 1.

- Partie extérieur du montage (partie 2)

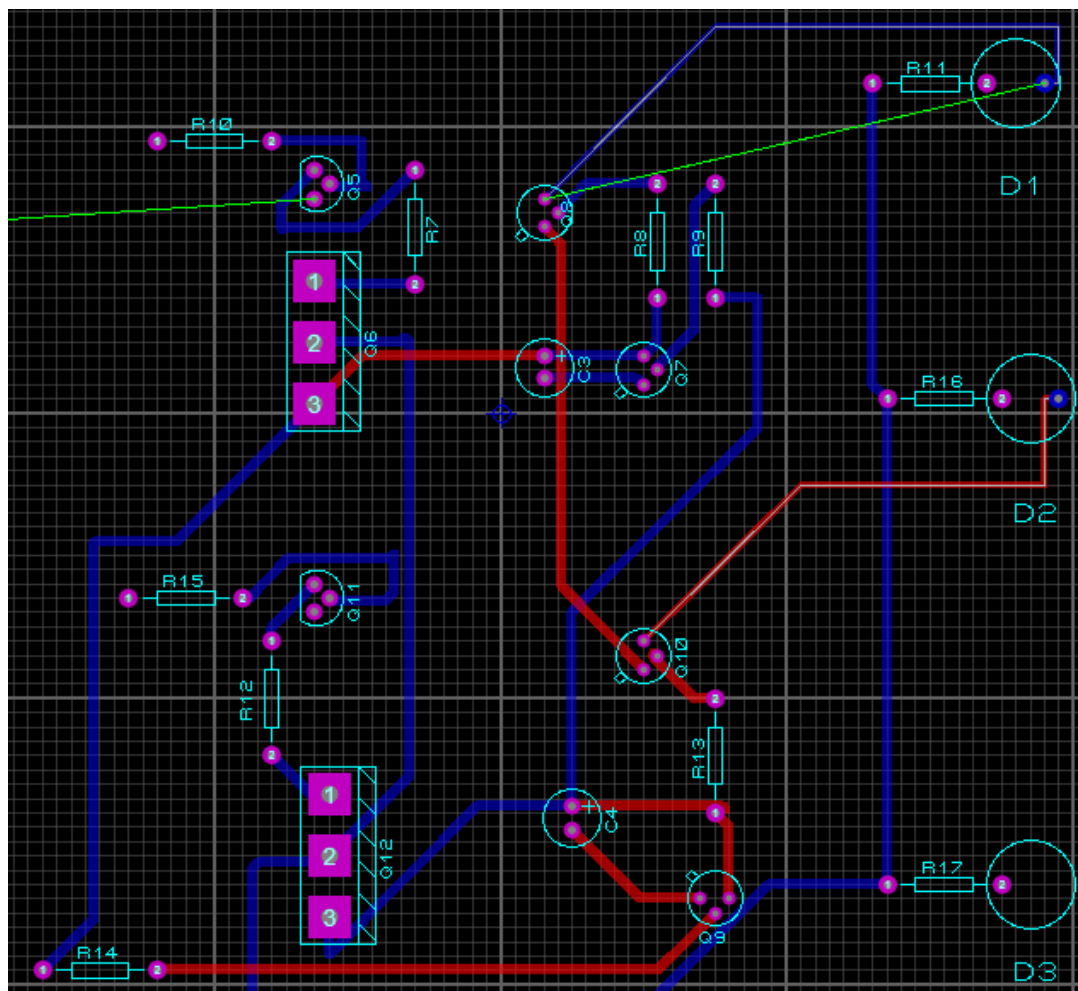


Figure 42 : Plan de câblage 2.

4 REALISATION DU NOUVEAU SYSTEME

4.1 Développement technique

4.1.1 La stratégie

a) La stratégie prévisionnelle

Ce projet étant personnel, notre première mission fut de définir nous-mêmes une stratégie prévisionnelle ainsi que les objectifs à atteindre. Bien entendu, cette stratégie a évolué au cours du temps afin de satisfaire nos exigences, mais aussi les contraintes auxquelles nous avons fait face. Nous avons ainsi tissé notre réflexion, avec pour fil conducteur la relation entre l'utilisateur et le nouveau système. Tout d'abord l'utilisateur : même si celui-ci ne fait pas partie à proprement parler du développement technique, il est primordial d'identifier le public concerné. Quel est le profil de l'utilisateur type ? Voilà le fondement de notre réflexion. Nous avons finalement privilégié une clientèle novice en développant une solution facile d'utilisation, la plus intuitive possible.

b) Architecture logicielle

À la manière d'une interface homme-machine, l'utilisateur 1 envoie un signal (onde radio) de demande d'accès, suite à un dispositif de signalisation l'utilisateur 2 reçoit la demande de celui-ci grâce à un signal sonore, puis à son tour répond par un signal de retour et cela grâce au logiciel développé en C#.

Le diagramme ci-dessous résume notre paragraphe précédent sous la forme d'un schéma :

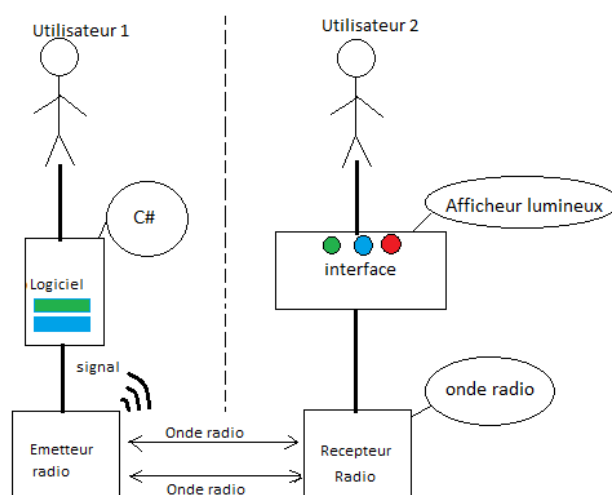


Figure 43 : Architecture logiciel.

c) Architecture matérielle

Toujours À la manière d'une interface homme-machine, l'utilisateur 1 appui sur un bouton poussoir qui fera retentir une sonnerie dans le locale de l'utilisateur 2 puis celui-ci grâce à un programme se trouvant dans son ordinateur va décider de la réponse à envoyer à l'utilisateur 1 par un interface équipée d'un afficheur a trois DEL dont chacune porte sa signification (entrer, patienter), en cas d'indisponibilité ou absence le dispositif extérieur est équipé d'un mini-switch à déplacer vers le haut, la DEL rouge doit s'allumer.

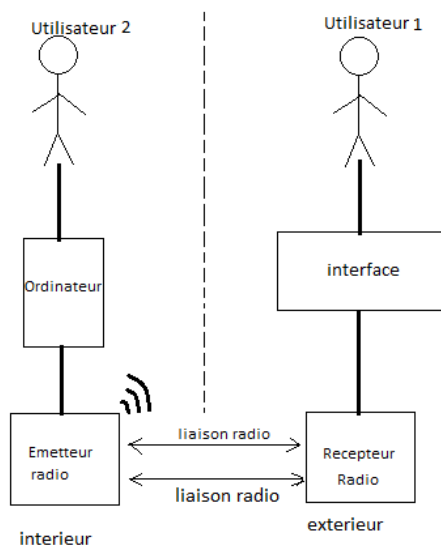


Figure 44 : Architecture matérielle

4.2 Les solutions logiciels et technologies utilisé

a) Langage de programmation

Notre choix est porté sur la programmation orienté objet, car elle permet d'avoir des programmes fiables. En ce qui concerne notre application nous avons opté pour le langage de programmation C#.

b) Environnement de développement

Environnement de développement ou IDE, pour faire simple disons qu'un IDE est un programme qui rassemble un ensemble d'outils facilitant le développement d'autres programmes.

Nous utiliserons Visual studio (vs), t un environnement de développement bien adapté pour le c#. Créer par Microsoft, Il existe en plusieurs

version différentier par l'année de création (vs2008, vs2010, vs2012, vs2013, vs2015, 2017...), selon l'évolution du c.

c) Stockage des données

Pour le stockage de données nous avons opté pour le fichier car le programme présent stocke moins de données et non complexe.

4.3 Présentation de différentes interfaces

0. Créer compte

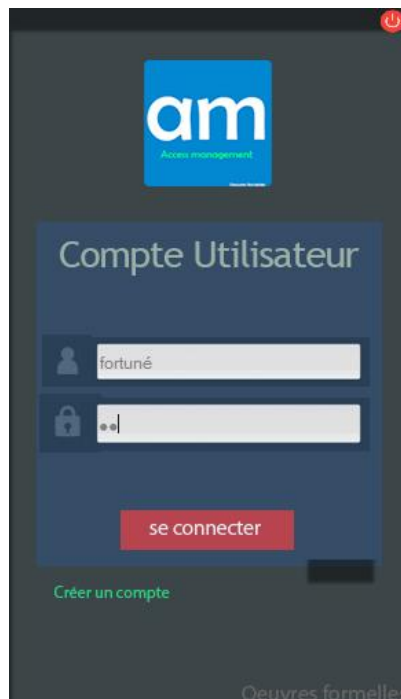
C'est grâce à cet interface qu'un utilisateur peut créer un compte afin d'utiliser l'application.



The screenshot shows a mobile application interface titled 'Configuration'. Below the title, there is a section for 'Durée de la signalisation' (Signalization duration) with a value of '5000' and a unit of 'Milli-seconde'. Below this, there is a section titled 'Créer un Compte' (Create Account). This section contains three input fields: 'Nom d'utilisateur' (Username) with the value 'fortuné', 'Mot de passe' (Password) with the value '12', and 'Confirmer mot de passe' (Confirm password) with the value '12'. A green 'Sauvegarder' (Save) button is located at the bottom of the form.

1. Connexion

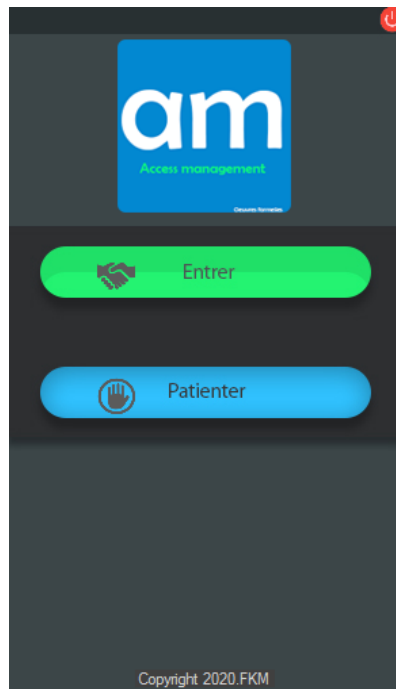
Après avoir créé un compte le nouvel utilisateur pourra se connecter à l'application via cette interface afin d'utiliser les fonctionnalités de l'application



The screenshot shows a mobile application interface titled 'Compte Utilisateur' (User Account). At the top, there is a logo for 'am' (Accès management). Below the logo, there are two input fields: one for the username 'fortuné' and one for the password, which is masked with dots. A red 'se connecter' (Connect) button is located below the password field. At the bottom, there is a green link that says 'Créer un compte' (Create account). The footer of the screen displays 'Oeuvres formelles'.

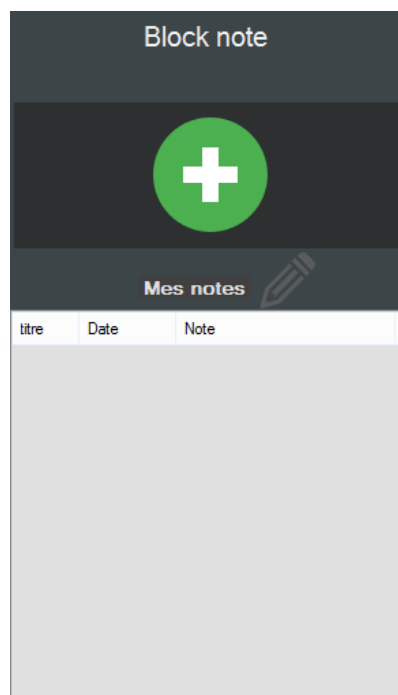
2. Commande

C'est cette interface que l'utilisateur intérieur utilisera pour communiquer avec l'utilisateur extérieur via les deux boutons présents.



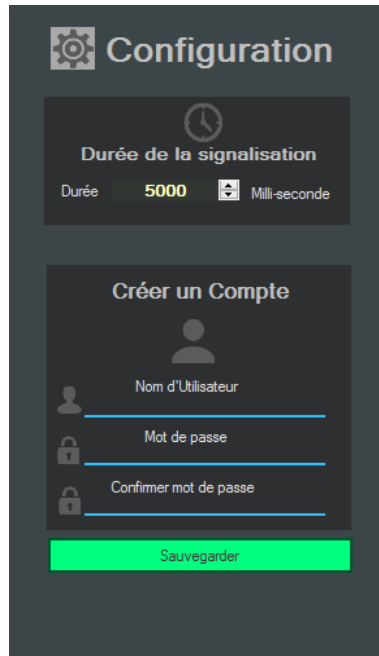
3. Block note

Cette interface supplémentaire permettra à l'utilisateur de faire des petites notes, qui seront sauvegarder dans son ordinateur et afficher sur cette interface.



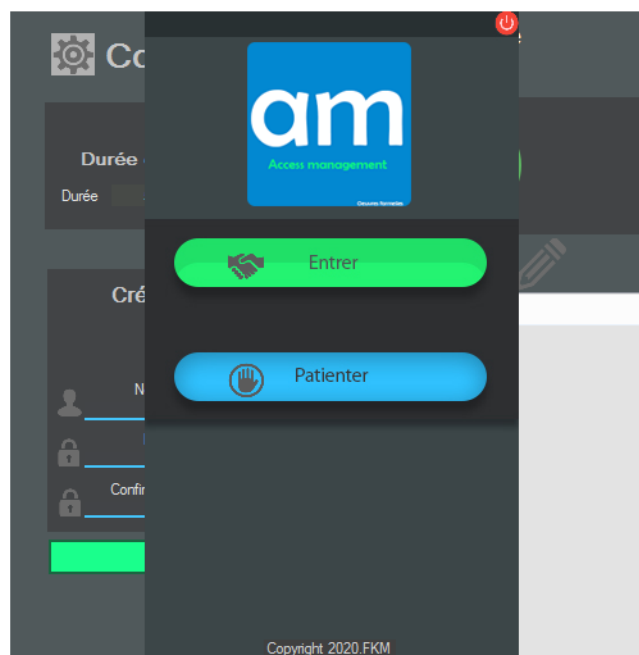
4. Configuration

C'est dans cette interface que l'utilisateur pourra faire une configuration de l'application tel que créer un compte et modifier la durée de signalisation.



5. Application complet :

Voici la vraie présentation de l'application, avec ses trois interfaces, le choix d'interface se fait en cliquant sur une partie de l'interface voulue.



4.4 Quelques codes de la programmation

0. Créer compte

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.IO;

namespace tfc_design
{
    public partial class Form3 : Form
    {
        public Form3()
        {
            InitializeComponent();
        }
        int Intit=0;
        Form formu;
        public Form3(int nb, Form fd)
        {
            Intit = nb;
            formu = fd;
            InitializeComponent();
        }

        private void Form3_Load(object sender, EventArgs e)
        {
        }

        int compt;
        private void timer1_Tick(object sender, EventArgs e)
        {
            if (compt == 30)
            {
                timer1.Stop();
            }
            int px = this.Location.X - compt;
            int py = this.Location.Y;
            this.Location = new Point(px, py);

            compt = compt + 5;
        }

        private void Form3_Click(object sender, EventArgs e)
        {
            Gestion_des_forms G = new Gestion_des_forms(this);
            G.testPosition();
        }

        private void numericUpDown1_ValueChanged(object sender, EventArgs e)
        {
            Gestion_des_forms.duree = Convert.ToInt16(numericUpDown1.Value);
        }
    }
}
```

```

private void button1_Click(object sender, EventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();
    string nU = T1.Text;
    string mP = T2.Text;
    string mPConf = T3.Text;

    if (mP == mPConf)
    {
        if (T1.Text != "" || T2.Text != "")
        {
            string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesCompte.text";
            StreamWriter fichier1 = new StreamWriter(chemin, false);

            string Enreg;
            Enreg = nU + "|" + mP;
            fichier1.WriteLine(Enreg);
            fichier1.Close();
            MessageBox.Show("compte créer");
            if (Intit==1)
            {
                formu.Opacity = 1;
                this.Close();
            }
        }
    }
    else
    {
        MessageBox.Show("le nouveau mot de passe doit être identique avec celui de
confirmation.");
    }
}

private void T1_TextChanged(object sender, EventArgs e)
{
}
}
}

```

1. Connexion

```

namespace tfc_design
{
    public partial class Form4 : Form
    {
        public Form4()
        {
            InitializeComponent();
        }
        Form3 f3;
        Form2 f2;
        Form5 f5;
        private void button1_Click(object sender, EventArgs e)
        {
            bool Ok = false;
            string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesCompte.text";
            // StreamWriter fichier1 = new StreamWriter(chemin, true);
            if (T1.Text!=" " || T2.Text!=" ")

```



```

{
    if (File.Exists(chemin))
    {
        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] LesChamps = null; ;

        LesChamps =
lESenregistrements[0].Split(separateur);
        if (LesChamps[0]==T1.Text &&
LesChamps[1]==T2.Text)
        {
            Ok = true;
        }
        else
        {
            MessageBox.Show("Informations erronées");
        }
    }

    if (Ok)
    {
        f3 = new Form3();
        int durée = Convert.ToInt16(f3.numericUpDown1.Value);
        f2 = new Form2();
        f5 = new Form5();
        Form1 f1 = new Form1(durée);

        f5.Show();
        f5.Hide();
        f3.Show();
        f2.Show();
        f1.Show();

        this.Hide();
        timer1.Start();
    }
    else
    {
        MessageBox.Show("Remplissez tout les formulaires s'il vous plait!");
    }
}

int compt = 0;
private void timer1_Tick(object sender, EventArgs e)
{
    if (compt<100)
    {
        compt = compt + 10;
    }
    else
    {
        f3.timer1.Start();
        f2.timer1.Start();
        timer1.Stop();
    }
}

private void Form4_Load(object sender, EventArgs e)

```

```

    {
        Gestion_des_forms G = new Gestion_des_forms(this,this);
        G.testPosition();
        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesCompte.text";
        if (!File.Exists(chemin))
        {
            this.Opacity=0;
            Form3 f3 = new Form3(1,this);
            f3.Show();
        }
    }

    private void linkLabel1_LinkClicked(object sender,
LinkLabelLinkClickedEventArgs e)
    {
        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesCompte.text";
        if (File.Exists(chemin))
        {
            this.Opacity = 0;
            Form3 f3 = new Form3(1, this);
            f3.Show();
        }
    }

    private void panel2_Click(object sender, EventArgs e)
    {
        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesCompte.text";
        if (File.Exists(chemin))
        {
            this.Opacity = 0;
            Form3 f3 = new Form3(1, this);
            f3.Show();
        }
    }

    private void T1_TextChanged(object sender, EventArgs e)
    {
        if (T1.Text=="_Nom d'utilisateur")
        {
            T1.Text = "";
        }
    }

    private void T2_TextChanged(object sender, EventArgs e)
    {
        if (T2.Text == "_Mot de passe")
        {
            T2.Text = "";
        }
    }

    private void T1_Click(object sender, EventArgs e)
    {
        if (T1.Text == "_Nom d'utilisateur")
        {
            T1.Text = "";
            T1.ForeColor = Color.Black;
        }
    }
}

```

```

private void T2_Click(object sender, EventArgs e)
{
    if (T2.Text == "_Mot de passe")
    {
        T2.Text = "";
        T2.ForeColor = Color.Black;
    }
}

private void T1_Leave(object sender, EventArgs e)
{
}

private void T2_Leave(object sender, EventArgs e)
{
}

private void pictureBox1_Click(object sender, EventArgs e)
{
    Application.Exit();
}
}
}

```

2. Commande

```

3. using System;
4. using System.Collections.Generic;
5. using System.ComponentModel;
6. using System.Data;
7. using System.Drawing;
8. using System.Linq;
9. using System.Text;
10. using System.Threading.Tasks;
11. using System.Windows.Forms;
12. using System.Threading;
13.
14. namespace tfc_design
15. {
16.     public partial class Form1 : Form
17.     {
18.         public Form1()
19.         {
20.             InitializeComponent();
21.         }
22.         int Duree;
23.         public Form1(int durée)
24.         {
25.             Duree = durée;
26.             InitializeComponent();
27.         }
28.         private void Form1_Load(object sender, EventArgs e)
29.         {
30.             Gestion_des_forms G = new Gestion_des_forms(this, this);
31.             G.testPosition();
32.             //MessageBox.Show(this.Location.X.ToString());
33.         }
34.         private void button1_Click(object sender, EventArgs e)
35.         {
36.             Duree = Gestion_des_forms.duree;

```

```

37.         Thread threadBEEP = new Thread(new ThreadStart(beep));
38.         threadBEEP.Start();
39.     }
40.     public void beep()
41.     {
42.         Console.Beep(38, Duree);
43.     }
44.     public void beep1()
45.     {
46.         Console.Beep(20000, Duree);
47.     }
48.
49.     private void button2_Click(object sender, EventArgs e)
50.     {
51.         Duree = Gestion_des_forms.duree;
52.         Thread threadBEEP = new Thread(new ThreadStart(beep1));
53.         threadBEEP.Start();
54.     }
55.
56.     private void panel1_Click(object sender, EventArgs e)
57.     {
58.         Thread threadBEEP = new Thread(new ThreadStart(beep1));
59.         threadBEEP.Start();
60.         Gestion_des_forms G = new Gestion_des_forms(this);
61.         G.testPosition();
62.     }
63.
64.     private void button3_Click(object sender, EventArgs e)
65.     {
66.         Application.Exit();
67.     }
68. }
69. }

```

3. Block note

```

namespace tfc_design
{
    public partial class Form2 : Form
    {
        public Form2()
        {
            InitializeComponent();
        }

        private void Form2_Load(object sender, EventArgs e)
        {
            actualiser();
        }
        int compt = 0;
        private void timer1_Tick(object sender, EventArgs e)
        {

            if (compt==30)
            {
                timer1.Stop();
            }
            int px = this.Location.X + compt;
            int py = this.Location.Y;
            this.Location = new Point(px, py);

```

```

    compt = compt + 5;

}
private void button1_Click(object sender, EventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();
    NouvNote newNote = new NouvNote(this);
    newNote.ShowDialog();

}

private void LesNotes_Load(object sender, EventArgs e)
{
    actualiser();
}

private void label1_Click(object sender, EventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();
    if (label1.Text != "" || label3.Text != "")
    {

        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifier = new string[lESenregistrements.Length];
        string enregistrement;
        string titre = "", dates = "", note = "";
        int j = 0;

        // StreamWriter fichier1 = new StreamWriter(chemin, true);
        if (File.Exists(chemin))
        {

            for (int i = 0; i < lESenregistrements.Length; i++)
            {
                string[] LesChamps = lESenregistrements[i].Split(separateur);
                // lesItems = new ListViewItem();
                string fusion = LesChamps[0] + " " + LesChamps[1];
                //MessageBox.Show(j.ToString());
                if (fusion == label1.Text && LesChamps[2] == label3.Text)
                {
                    titre = LesChamps[0];
                    dates = LesChamps[1];
                    note = LesChamps[2];
                    enregistrement = "0|0|0";
                    lesEnreModifier[j] = enregistrement;
                }
                else
                {
                    enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
                    lesEnreModifier[j] = enregistrement;
                }
            }

```

```

        j = j + 1;
    }
    File.Delete(chemin);
}

StreamWriter fichier1 = new StreamWriter(chemin, true);

string Enreg;
for (int i = 0; i < lesEnreModifier.Length; i++)
{
    if (lesEnreModifier[i] != "0|0|0")
    {
        Enreg = lesEnreModifier[i];
        fichier1.WriteLine(Enreg);
    }
}

fichier1.Close();
NouvNote noteModifier = new NouvNote(this, titre, dates, note);
noteModifier.button2.Visible = false;
noteModifier.button1.Text = "Modifier";
noteModifier.ShowDialog();
actualiser();
}

private void listView1_SelectedIndexChanged(object sender, EventArgs e)
{
    if (listView1.SelectedItems.Count > 0)
    {
        ListViewItem lesItems = listView1.SelectedItems[0];
        label11.Text = lesItems.Text + " : " + lesItems.SubItems[1].Text;
        label13.Text = lesItems.SubItems[2].Text;
        button1.Visible = false;
        button1.Visible = false;
        //button2.Visible = true;
        // button3.Visible = true;
        button5.Visible = true;
    }
}

private void button3_Click(object sender, EventArgs e)
{
    if (label11.Text != "" || label13.Text != "")
    {
        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifier = new string[lESenregistrements.Length];
        string enregistrement;
        int j = 0;
    }
}

```

```

// StreamWriter fichier1 = new StreamWriter(chemin, true);
if (File.Exists(chemin))
{
    for (int i = 0; i < lESenregistrements.Length; i++)
    {
        string[] LesChamps = lESenregistrements[i].Split(separateur);
        // lesItems = new ListViewItem();
        string fusion = LesChamps[0] + " " + LesChamps[1];
        //MessageBox.Show(j.ToString());
        if (fusion == label1.Text && LesChamps[2] == label3.Text)
        {
            enregistrement = "0|0|0";
            lesEnreModifier[j] = enregistrement;
        }
        else
        {
            enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
            lesEnreModifier[j] = enregistrement;
        }

        j = j + 1;
    }
    File.Delete(chemin);
}

StreamWriter fichier1 = new StreamWriter(chemin, true);

string Enreg;
for (int i = 0; i < lesEnreModifier.Length; i++)
{
    if (lesEnreModifier[i] != "0|0|0")
    {
        Enreg = lesEnreModifier[i];
        fichier1.WriteLine(Enreg);
    }
}

fichier1.Close();
actualiser();
}
}
public void actualiser()
{
    listView1.Items.Clear();
    label1.Text = "";
    label3.Text = "";
    button1.Visible = true;
    button1.Visible = true;
    // button2.Visible = false;
    // button3.Visible = false;
    button5.Visible = false;
    string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";
    // StreamWriter fichier1 = new StreamWriter(chemin, true);
    if (File.Exists(chemin))
    {
        string[] lESenregistrements;

```

```

char separateur = '|';
lESenregistrements = File.ReadAllLines(chemin);
string[] LesChamps = null; ;

for (int i = 0; i < lESenregistrements.Length; i++)
{
    LesChamps = lESenregistrements[i].Split(separateur);
    ListViewItem lesItems = new ListViewItem();
    lesItems.Text = LesChamps[0];
    lesItems.SubItems.Add(LesChamps[1]);
    lesItems.SubItems.Add(LesChamps[2]);
    listView1.Items.Add(lesItems);
}

}

private void button2_Click_1(object sender, EventArgs e)
{
    if (label1.Text != "" || label3.Text != "")
    {

        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifier = new string[lESenregistrements.Length];
        string enregistrement;
        string titre = "", dates = "", note = "";
        int j = 0;

        // StreamWriter fichier1 = new StreamWriter(chemin, true);
        if (File.Exists(chemin))
        {

            for (int i = 0; i < lESenregistrements.Length; i++)
            {
                string[] LesChamps = lESenregistrements[i].Split(separateur);
                // lesItems = new ListViewItem();
                string fusion = LesChamps[0] + " " + LesChamps[1];
                //MessageBox.Show(j.ToString());
                if (fusion == label1.Text && LesChamps[2] == label3.Text)
                {
                    titre = LesChamps[0];
                    dates = LesChamps[1];
                    note = LesChamps[2];
                    enregistrement = "0|0|0";
                    lesEnreModifier[j] = enregistrement;
                }
                else
                {
                    enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
                    lesEnreModifier[j] = enregistrement;
                }

                j = j + 1;
            }
        }
    }
}

```



```

        File.Delete(chemin);
    }

    StreamWriter fichier1 = new StreamWriter(chemin, true);

    string Enreg;
    for (int i = 0; i < lesEnreModifier.Length; i++)
    {
        if (lesEnreModifier[i] != "0|0|0")
        {
            Enreg = lesEnreModifier[i];
            fichier1.WriteLine(Enreg);
        }
    }

    fichier1.Close();
    NouvNote noteModifier = new NouvNote(this, titre, dates, note);
    noteModifier.button2.Visible = false;
    noteModifier.ShowDialog();
    actualiser();
}

private void button4_Click(object sender, EventArgs e)
{
    this.Close();
}

private void button5_Click(object sender, EventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();
    label1.Text = "";
    label3.Text = "";
    button1.Visible = true;
    // button2.Visible = false;
    // button3.Visible = false;
    button5.Visible = false;
}

private void label3_Click(object sender, EventArgs e)
{
    if (label1.Text != "" || label3.Text != "")
    {

        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifier = new string[lESenregistrements.Length];
        string enregistrement;
        string titre = "", dates = "", note = "";
        int j = 0;

        // StreamWriter fichier1 = new StreamWriter(chemin, true);
        if (File.Exists(chemin))

```

```

{
    for (int i = 0; i < lESenregistrements.Length; i++)
    {
        string[] LesChamps = lESenregistrements[i].Split(separateur);
        // lesItems = new ListViewItem();
        string fusion = LesChamps[0] + " " + LesChamps[1];
        //MessageBox.Show(j.ToString());
        if (fusion == label1.Text && LesChamps[2] == label3.Text)
        {
            titre = LesChamps[0];
            dates = LesChamps[1];
            note = LesChamps[2];
            enregistrement = "0|0|0";
            lesEnreModifieur[j] = enregistrement;
        }
        else
        {
            enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
            lesEnreModifieur[j] = enregistrement;
        }

        j = j + 1;
    }
    File.Delete(chemin);
}

StreamWriter fichier1 = new StreamWriter(chemin, true);

string Enreg;
for (int i = 0; i < lesEnreModifieur.Length; i++)
{
    if (lesEnreModifieur[i] != "0|0|0")
    {
        Enreg = lesEnreModifieur[i];
        fichier1.WriteLine(Enreg);
    }
}

fichier1.Close();
NouvNote noteModifieur = new NouvNote(this, titre, dates, note);
noteModifieur.button2.Visible = false;
noteModifieur.button1.Text = "Modifier";
noteModifieur.ShowDialog();
actualiser();
}
}

private void supprimerToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (listView1.SelectedItems.Count > 0)
    {
        ListViewItem lesItems = listView1.SelectedItems[0];
        label1.Text = lesItems.Text + " " + lesItems.SubItems[1].Text;
        label3.Text = lesItems.SubItems[2].Text;
        // button1.Visible = false;
        //button2.Visible = true;
        // button3.Visible = true;
    }
}

```

```

        // button5.Visible = true;
    }
    if (label11.Text != "" || label13.Text != "")
    {

        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifieur = new string[lESenregistrements.Length];
        string enregistrement;
        int j = 0;

        // StreamWriter fichier1 = new StreamWriter(chemin, true);
        if (File.Exists(chemin))
        {

            for (int i = 0; i < lESenregistrements.Length; i++)
            {
                string[] LesChamps = lESenregistrements[i].Split(separateur);
                // lesItems = new ListViewItem();
                string fusion = LesChamps[0] + " " + LesChamps[1];
                //MessageBox.Show(j.ToString());
                if (fusion == label11.Text && LesChamps[2] == label13.Text)
                {
                    enregistrement = "0|0|0";
                    lesEnreModifieur[j] = enregistrement;
                }
                else
                {
                    enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
                    lesEnreModifieur[j] = enregistrement;
                }

                j = j + 1;
            }
            File.Delete(chemin);
        }

        StreamWriter fichier1 = new StreamWriter(chemin, true);

        string Enreg;
        for (int i = 0; i < lesEnreModifieur.Length; i++)
        {

            if (lesEnreModifieur[i] != "0|0|0")
            {
                Enreg = lesEnreModifieur[i];
                fichier1.WriteLine(Enreg);
            }

        }

        fichier1.Close();
        actualiser();
    }
}

```

```

private void modifierToolStripMenuItem_Click(object sender, EventArgs e)
{
    if (label11.Text != "" || label13.Text != "")
    {

        string chemin =
Environment.GetFolderPath(Environment.SpecialFolder.MyDocuments) + "\\MesNotes.text";

        string[] lESenregistrements;
        char separateur = '|';
        lESenregistrements = File.ReadAllLines(chemin);
        string[] lesEnreModifieur = new string[lESenregistrements.Length];
        string enregistrement;
        string titre = "", dates = "", note = "";
        int j = 0;

        // StreamWriter fichier1 = new StreamWriter(chemin, true);
        if (File.Exists(chemin))
        {

            for (int i = 0; i < lESenregistrements.Length; i++)
            {
                string[] LesChamps = lESenregistrements[i].Split(separateur);
                // lesItems = new ListViewItem();
                string fusion = LesChamps[0] + " " + LesChamps[1];
                //MessageBox.Show(j.ToString());
                if (fusion == label11.Text && LesChamps[2] == label13.Text)
                {
                    titre = LesChamps[0];
                    dates = LesChamps[1];
                    note = LesChamps[2];
                    enregistrement = "0|0|0";
                    lesEnreModifieur[j] = enregistrement;
                }
                else
                {
                    enregistrement = LesChamps[0] + "|" + LesChamps[1] + "|" +
LesChamps[2];
                    lesEnreModifieur[j] = enregistrement;
                }

                j = j + 1;
            }
            File.Delete(chemin);
        }

        StreamWriter fichier1 = new StreamWriter(chemin, true);

        string Enreg;
        for (int i = 0; i < lesEnreModifieur.Length; i++)
        {

            if (lesEnreModifieur[i] != "0|0|0")
            {
                Enreg = lesEnreModifieur[i];
                fichier1.WriteLine(Enreg);
            }
        }
    }
}

```

```

        fichier1.Close();
        NouvNote noteModifier = new NouvNote(this, titre, dates, note);
        noteModifier.button2.Visible = false;
        noteModifier.button1.Text = "Modifier";
        noteModifier.ShowDialog();
        actualiser();
    }
}

private void panel1_Click(object sender, EventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();

    // MessageBox.Show(G.Px.ToString() + G.nomDuF() + " " +
G.PosX().ToString()+ " " + G.nomDuF_milieu());
}

private void listView1_ColumnClick(object sender, ColumnClickEventArgs e)
{
    Gestion_des_forms G = new Gestion_des_forms(this);
    G.testPosition();
}

private void panel1_Paint(object sender, PaintEventArgs e)
{
}

}
}

```

4.5 Les solutions matérielles

a. Alimentation

Pour l'alimentation de chaque circuit nous avons préféré une pile chargeable au lithium de 3.7v pour éviter les fluctuations qu'on peut raconter si on le faisait avec une alimentation secteur.

b. Amplificateur

Le principe de l'amplification réside dans le gain de puissance pour un signal. C'est à dire que si j'amplifie un signal, ce dernier gagnera en puissance.

Un amplificateur est un dispositif qui amplifie un signal électrique. Son fonctionnement réside sur le principe de l'amplification, que l'on vient de voir. En électronique, cela ne peut être, par exemple, qu'un circuit intégré, voire même un dispositif simple, tel que le transistor.

En ce qui concerne ce projet les amplificateurs sont des simples amplificateurs à base d'un transistor.

4.6 PRESENTATION DE DIFFERENTS DISPOSITIFS

0. dispositif partie extérieur

C'est dispositif présente trois indicateur équipé de lampe, qui s'allume selon le cas :

- Lampe verte pour signaler l'accès libre (entrer).
- Lampe bleu pour signaler la suspension d'accès (patienter).
- Lampe rouge pour signaler l'indisponibilité ou l'absence.

Nota : la lampe rouge est commander manuellement grâce à un switch sur le côté.

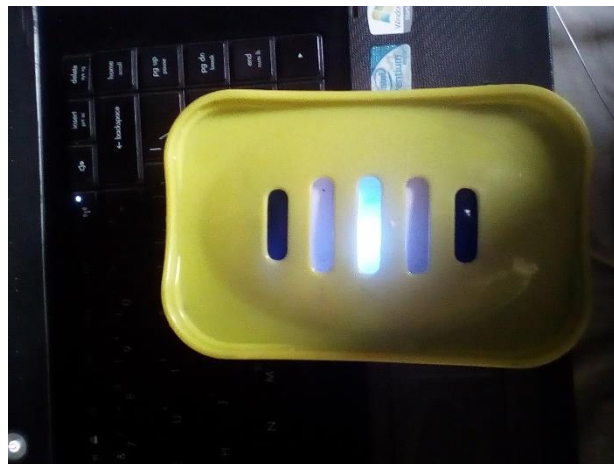


Diffèrent état :

ENTRER :



PATIANTER :



INDISPONIBLE :



2. Dispositif partie intérieur



COCNCLUSION

Tout au long de notre travail nous avons parlé sur la “conception d’un système de supervision des entrées dans un local a accès limité » afin de permettre une bonne gestion des accès par une application informatique et un dispositif électronique mis à disposition, facilitant ainsi les échanges entre l’occupant du local et ceux voulant y accéder.

Nous sommes engagés d’apporter ou d’améliorer des solutions nécessaires aux problèmes cité ci-haut. Pour y arriver nous avons parcouru les différentes phases ou étapes indispensables afin de mieux répondre aux besoins des utilisateurs.

Ceci étant, nous estimons avoir atteint notre objectif, il est temps pour nous de confirmer notre hypothèse de départ selon laquelle l’implantation de ce nouveau système était un moyen incontournable pour mettre ladite supervision des entrées dans un local à l’échelle de la modernisation.

L’informatique étant une science qui évolue quotidiennement, nous ne pensons pas avoir épuisé la problématique de ladite supervision des entrées. Nous sollicitons l’indulgence de nos lecteurs pour des éventuelles imperfections qu’ils auront constatées. Ainsi, vos remarques et conseils sont les bienvenus pour nous permettre d’améliorer notre travail dans le futur.

BIBLIOGRAPHIE

- **OUVRAGE**

- DUPICHOT F, Introduction théorique a l'informatique. Afrique du sud. Africa computing 2013.
- Gérard Leblanc, C# et .net, EYROLLES.

- **DICTIONNAIRES**

- Larousse 2019
- 36 Dictionnaires et recueils

- **NOTES DE COURS**

- Prof. KASORO MULENDA Nathanaël, cours de la programmation orienté objet à l'unikin ;
- Prof. MANDIEDIKA MANDUNDILA Jule, cours d'électronique à l'iti-Gombe.

- **WEBOGRAPHIE**

- www.Commentcamarche.com
- www.sonelec-musique.com
- www.Eyrolles.com
- www.OpenClassrooms.com
- www.elektronique.fr
- www.electronique-3D.fr
- www.wikipedia.org
- www.gotronic.fr
- www.lextronic.com
- www.forums.futura-sciences.com

TABLE DE MATIERE

EPIGRAPHE.....	i
DEDICACE.....	ii
REMERCIEMENTS.....	iii
0. INTRODUCTION.....	1
0.1.PROBLEMATIQUE.....	1
0.2.HYPOTHESE.....	2
0.3.PRESENTATION DU SUJET.....	2
0.4.CHOIX ET INTERER DU SUJET.....	3
1) Choix du sujet.....	4
2) Intérêt du sujet.....	4
0.5.DELIMITATION DU SUJET.....	5
0.6.METHODES ET TECHNIQUES UTILISEES.....	6
1) Méthodes utilisées.....	6
2) Techniques utilisées.....	7
0.7.CANEVAS DU TRAVAIL.....	7
1 CONCEPTS THEORIQUE DE BASE	9
1.1 Système	12
1.1.1 Définition.....	12
1.1.2 Caractéristique d'un système.....	13
1.1.3 Système d'information	13
a) Système de pilotage(SP).....	14
b) Système opérant (SO)	14
c) Système d'information (SI).....	15
1.1.4 Système informatique.....	15
a) Définition d'un système informatique	15
b) Qualité d'un système informatique	15
c) Typologies des systèmes informatiques	16
1.2 Présentation de la démarche UML.....	17
1.2.1 Introduction.....	17
1.2.2 Objectif d'UML.....	17

1.2.3	Caractéristique d'UML	17
1.2.4	Les différents types de diagramme en UML.....	18
a)	Diagramme de cas d'utilisation	18
b)	Diagramme de classe :	20
c)	Diagramme d'objets.....	21
d)	Diagramme de séquence	21
e)	Diagramme d'activités	22
f)	Diagramme de collaboration.....	22
g)	Diagramme de composants	22
h)	Diagrammes d'états-transition	22
i)	Diagramme de déploiement	22
1.2.5	L'approche orientée objet	22
a)	Approche fonctionnelle vers l'Approche objet.....	23
b)	Concepts importants de l'approche objet.....	24
1.3	Définition de quelques notions électroniques.....	25
1.3.1	Notions de base	25
a)	Pourquoi l'électronique a-t-elle été inventée ?.....	25
b)	Qu'est-ce que l'électronique ?	26
c)	Grandeur physique utiliser en électronique.....	26
d)	La notion de conductibilité	29
e)	La masse et notion de référentiel	30
a)	Le référentiel	30
b)	La masse	31
c)	La terre	31
f)	Notions de Schémas électrique.....	32
a)	Lois sur les fils et leurs liaisons.....	32
b)	Lois concernant les dipôles	33
1.3.2	Les composants électroniques.....	34

a) Quelques composant de base :	35
b) Quelques symboles utiliser en électronique :	40
2 CADRAGE DU PROJET	41
2.1 Présentation de la méthode d'ordonnancement	41
2.2 Identification des taches du projet.....	42
2.3 Diagramme de la méthode relative à l'ensemble des opérations	42
2.3.1 Présentation de la méthode PERT.....	42
2.4 Planning d'exécution des taches, estimations des durées.....	43
2.5 Graphe PERT	44
2.5.1 Détermination des niveaux du graphe du projet.....	44
2.5.2 Présentation du graphe PERT.....	46
2.6 Délai de réalisation du projet.....	46
2.6.1 Calcul des dates au plus tôt et dates au plus tard	46
a) Détermination de la date au plus tôt (DTO)	46
b) Détermination de la date au plus tard (DTA).....	47
2.6.2 Calcul des marges libres et marges locales	47
a) Détermination de marge libre.....	47
b) Détermination de marge totale	47
2.6.3 Chemin critique	47
2.7 Calcul de cout du projet :	47
3 MODELISATION DU NOUVEAU SYSTEME D'INFORMATION	48
3.1 Modélisation du logiciel	48
3.1.1 But de la modélisation	48
3.1.2 Cycle de vie d'un logiciel	48
3.1.3 Diagramme de cas d'utilisation (DCU).....	49
3.1.4 Diagramme des séquences	52
3.1.5 Diagramme des classes	52
3.1.6 Diagramme d'activité.....	53

3.2	Modélisation du montage.....	54
3.2.1	Schéma	54
3.2.2	Type des schémas	55
a)	Schéma de principe ou bloc	55
b)	Schéma développé.....	57
c)	Dessin d'implantation	61
d)	Plan de câblage	63
4	REALISATION DU NOUVEAU SYSTEME	65
4.1	Développement technique	65
4.1.1	La stratégie	65
a)	La stratégie prévisionnelle	65
b)	Architecture logicielle	65
c)	Architecture matérielle.....	66
4.2	Les solutions logiciels et technologies utilisé	66
a)	Langage de programmation	66
b)	Environnement de développement	66
c)	Stockage des données.....	67
4.3	Présentation de différentes interfaces.....	68
4.4	Quelques codes de la programmation	71
4.5	Les solutions matérielles.....	86
4.6	PRESENTATION DE DIFFERENTS DISPOSITIFS	87