EFREI L2 – ANNEE 2010-2011

PROJET D'ALGORITHMIQUE ET STRUCTURES DE DONNEES

SUJET : CRYPTOLOGIE ET CRYPTANALYSE

ALEXANDRE JOSEPH
GABRIEL VILAINE

TABLE DES MATIERES

INTRODUCTION

La cryptanalyse est une science qui consiste à décrypter un message dont on ne possède pas la clé de chiffrement. Ce processus est appelé une attaque. Il permettra de comprendre le message codé. Cette science remonte à l'antiquité. Le premier document connu est une tablette d'argile qu'un potier avait gravé pour y laissé sa recette secrète. Il avait supprimé des consonnes et avait modifié l'orthographe de certains mots. Au fils des siècles les techniques se sont perfectionnées. Il faut attendre – 200 avant JC pour avoir un vrai système de cryptographie.

Ce projet met en œuvre plusieurs systèmes cryptographiques tels que le code de César, le code de Scytal, le chiffre de Vigenère, La machine d'Enigma.

ANALYSE FONCTIONNELLE GENERALE

TECHNOLOGIES UTILISEES

Notre projet a été développé en C++ tirant ainsi parti de la puissance de la programmation orienté objet avec notamment les notions d'héritage, de surcharge, de polymorphisme ou encore de template.

L'interface graphique du projet a été conçu avec le framework Qt (voir annexe) permettant ainsi de créer une interface portable sur différent système à savoir Windows, GNU/Linux ou encore Mac OS X. Ce framework offre aussi la possibilité de traduire de manière simple le texte de l'interface comme les boutons, les légendes ou encore les messages d'informations.

Les différents algorithmes de cryptographie et cryptanalyse utilisent eux aussi la richesse du framework Qt tel que les hashtables ou les listes chainées. L'utilisation des types complexes fournis par le framework tel que QString plutôt que type String de la librairie standard C++ nous permet de nous abstraire des problèmes d'interopérabilité entre les systèmes.

STRUCTURE GENERALE DE L'APPLICATION

L'architecture du projet a été pensée dès le début pour être modulaire et ainsi facile à maintenir et à faire évoluer.

On distingue ainsi deux parties à la fois distinctes mais étant tout de même en relation qui sont l'interface graphique avec tout ce qui est interaction avec l'utilisateur, réponse aux évènements ou encore contrôle des informations saisies. La deuxième partie est l'algorithmique pure, c'est-à-dire les différentes méthodes de chiffrement, déchiffrement et cryptanalyse.

Bien qu'utilisant le framework Qt, cette deuxième partie est tout à fait utilisable sans interface graphique dans un autre programme par exemple. L'interface elle aussi n'est pas lier à nos algorithmes de cryptographie.

L'intérêt, et pas des moindre, d'une telle architecture est aussi de pouvoir ajouter de manière aisé de nouveau algorithme ou options.

DONNEES TRAITEES

La structure modulaire de notre projet ne nous restreint pas sur le type de donnée que nous pouvons traiter. En effet l'indépendance des différents algorithmes nous permet aussi bien de prendre en compte du texte plat que riche, mais aussi des images, de l'hexadécimal ou encore du binaire.

Les différentes méthodes de cryptographique et cryptanalyse que nous avons développé jusqu'à présent s'applique sur un texte plat ou riche rentré par l'utilisateur.

Nous avions l'idée de permettre à l'utilisateur de sélectionner un fichier (texte ou autre) et de travailler dessus, mais à l'heure actuelle cette fonctionnalité n'a pas encore été implémentée.

METHODES DE CRYPTOGRAPHIE ET CRYPTANALYSE

Pour ce projet nous avons réalisé différentes méthodes de chiffrement et déchiffrement ainsi que de décryptage.

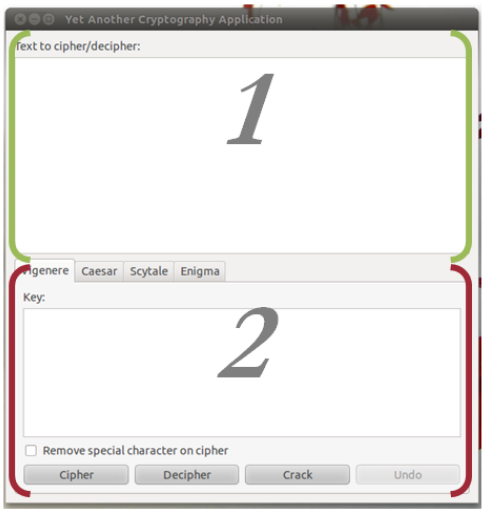
- ***Code César :*** Méthode de (dé)chiffrement mono-alphabétique. Le décryptage est réalisé au moyen d'une analyse fréquentielle. Cette méthode de décryptage aurait pu être couplée à une attaque de type dictionnaire qui donne de bien meilleurs résultats notamment sur les petits textes.

- ***Chiffre de Vigenère :*** Méthode de (dé)chiffrement poly-alphabétique. Pour le décryptage, la longueur de la clé est déterminée grâce à la méthode de cryptanalyse de Babbage. Ce décryptage n'est pas totalement opérationnel (voir analyse détaillée).

- ***La Scytale spartiate :*** Méthode de cryptographie très ancienne. Notre implémentation laisse à l'utilisateur le choix du notre de caractères autour du rayon. La cryptanalyse est réalisé avec une analyse fréquentiel de bi-gramme.

- ***Enigma :*** Simulation de la célèbre machine Allemande. Le choix de la position de départ des rotors est laissé à l'utilisateur. Nous voulions aussi laisser le choix à l'utilisateur le type de rotors utilisé (choix des connexions) ainsi que leurs positions dans la machine. Malheureusement par manque de temps cette algorithme n'est pas totalement fonctionnelle (voir analyse détaillée).

PRESENTATION DE L'INTERFACE



L'interface se découpe en deux parties distinctes comme montré sur la capture d'écran ci-dessus.

La première partie de l'interface, en vert, numéroté 1 sur l'image est destinée à recevoir le texte à chiffrer/déchiffrer. Cette partie est commune aux différents algorithmes. En effet, quelque soit la méthode de chiffrement il est toujours nécessaire que l'utilisateur renseigne l'information sur laquelle travaillé, en l'occurrence ici un texte.

La deuxième partie en rouge numéroté 2 de l'interface est la partie qui regroupe les différents algorithmes avec leurs options et leurs opérations qui leurs sont propre.

Chaque méthode de cryptographie peut donc définir les actions réalisable, que ce soit chiffrer, déchiffrer ou encore casser mais aussi les paramètres à prendre en compte lors de la réalisation de ces méthodes. La capture d'écran nous montre par exemple l'algorithme Vigenère qui nécessite une clé afin de réaliser son (dé)chiffrement poly-alphabétique. D'autres algorithmes comme César demanderons eux le décalage à réaliser ou encore la position des rotors pour Enigma. Il est laisser au concepteur de l'interface le choix de ces options.

ORGANISATION DU CODE SOURCE

La structure générale de l'application nous a conduits à séparer distinctement l'interface graphique des algorithmes de cryptographie.

Le dossier *ui* contient tous les éléments relatifs à l'interface graphique. Le dossier *algo* quant à lui regroupe toutes les méthodes de cryptographie est cryptanalyse.

ANALYSE FONCTIONNELLE DETAILEE

IMPLEMENTATION DES INTERFACES

Comme expliqué dans l'analyse fonctionnelle générale le projet a été découpé en deux parties distinctes mais néanmoins liées, l'interface graphique et les algorithmes de chiffrement.

ÉLEMENTS COMMUNS ET AJOUT D'ALGORITHMES

La classe UI définie la fenêtre générale de l'application. Elle se charge de définir les différents éléments communs aux algorithmes comme la zone de texte principale.

Cette classe définie également les onglets des algorithmes qui seront visible et donc utilisable par l'utilisateur. Pour ajouter un nouvel algorithme il suffit donc d'ajouter le widget de l'algorithme à l'objet *algoTabs* qui est un widget permettant de créer une interface à onglet.

Un onglet représente un algorithme dont sa gestion en termes d'évènement et d'éléments graphiques est laissé au concepteur de celui-ci. L'interface principale n'intercepte aucun événement si ce n'est l'ordre de quitté le programme.

CONCEPTS DE CREATION D'ONGLETS

Un onglet représente un algorithme de cryptographie. Les options fournies par celui-ci déterminent les actions et les0 paramètres offerts à l'utilisateur final. L'interface graphique n'a pas l'obligation de reprendre tous les paramètres des méthodes utilisées derrière.

Typiquement les onglets des algorithmes que nous avons développés proposaient pour la plupart un bouton pour chiffrer, un pour déchiffrer, un autre pour décrypter et un dernier pour annuler la dernière action. Chaque méthode avait ensuite ses propres options pour le chiffre de Vigenère par exemple la clé à utiliser, pour le code de César le décalage à opérer ou encore la position de départ des rotors pour Enigma.

Nos algorithmes avaient aussi tous une option permettant de supprimer les espaces et caractères spéciaux du texte à chiffrer. Ceci permet notamment de réduire la lisibilité du texte ce qui permet de rendre plus difficile une attaque de type dictionnaire se basant sur la longueur des mots.

Un onglet est un widget graphique qui se doit de respecter certaines règles. Le constructeur de la classe prend en paramètre un pointeur vers l'objet représentant la zone de texte commun, permettant ainsi de récupérer les données entrées par l'utilisateur lors d'une action.

Les widget définissant eux même les boutons et autres éléments graphique, il est laissé le soin au concepteur de vérifier que les données fournies par l'utilisateur sont correcte mais aussi l'interception des évènements et leurs traitement en fonction du type de signal.

IMPLEMENTATION DES ALGORITHMES

Les différents algorithmes de cryptographie et cryptanalyse que nous avons implémenté manipule des chaines de caractères. Ces données sont stockées dans des QString afin de s'assurer de la compatibilité entre les systèmes.

La plupart des méthodes de chiffrement étant basé sur un décalage d'une valeur donnée, nous utilisons pour réaliser cette substitution la valeur ASCII des lettres, c'est-à-dire leur valeur numérique dans la table. Ceci permet ainsi de réaliser des additions ou des soustractions de caractères.

Toute les lettres avant traitement sont convertie en majuscule afin de simplifier les démarches mais aussi de rendre le code chiffrée plus difficile à deviner.

Quelque soit l'algorithme de cryptographie celui-ci est toujours muni d'une fonction de chiffrement et de déchiffrement (par fois celles-ci ne font qu'une) ainsi que des fonctions annexes de traitement de données par exemple.

Afin que tous les algorithmes aient une structure commune facilitant la compréhension, nous utilisons l'héritage à la manière des interfaces en Java. Ainsi tous les algorithmes héritent de *BaseAlgo* qui définie les méthodes standard d'un moyen de cryptographie qui seront ensuite surchargé au besoin.

LE CHIFFRE DE VIGENERE

Le chiffre de Vigenère est une méthode de chiffrement poly-alphabétique. Le texte est chiffré au moyen d'une clé. Cette clé doit être au maximum de la longueur du texte à chiffrée. Si la clé est moins longue que le texte celle-ci est utilisée en boucle.

Pour réaliser ce type de chiffrement à la main, on utilise généralement un tableau avec un alphabet en ligne et un autre en colonne appelé carré de Vigenère. Il est possible d'implémenter cet algorithme en reproduisant un carré mais ceci a pour effet de consommer de la mémoire inutilement.

S'il on regarde attentivement la carré de Vigenère on s'aperçoit que pour trouver la lettre chiffré avec la lettre actuelle de la clé il suffit d'ajouter à la position des deux lettres dans l'alphabet. L'application d'un modulo 26 sur cette valeur trouvé permet de retourner à A quand le Z est atteins. Une simple boucle incrémentant la chaine finale avec la lettre nouvellement déterminé permet de calculer la chaine chiffrée.

La méthode de déchiffrement utilise le même principe mais cette fois ci la différence des deux lettres est faite, non plus l'addition.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

Carré de Vigenère

Pour la cryptanalyse du chiffre de Vigenère nous utilisons la méthode de Baddage. Celle-ci consiste à repérer dans le texte chiffré les groupes de lettre de trois caractères ou plus. Ceux-ci sont dus notamment à la répétition de la clé de chiffrement. Il faut ensuite calculer l'écart entre les mêmes groupes de lettre. La longueur de la clé probable est le diviseur commun de ces différents espaces entre les groupes de mots identiques.

La longueur X de la clé étant trouvé, il faut faire découper le texte en X parties, la première lettre de la première partie étant la première du texte, la deuxième la X+1 du texte, la première lettre de la deuxième partie étant la deuxième lettre du texte, la deuxième la X+2 et ainsi de suites.

Il est alors possible d'appliquer sur chaque bloque de texte une analyse fréquentielle afin de déterminer la Xème lettre de la clé. Les différentes lettres déterminées donnent ainsi la clé de chiffrement théorique.

Notre programme implémente la cryptanalyse de Baddage, nous somme ainsi capable de déterminer la longueur théorique de la clé, celle-ci s'affiche dans la console depuis laquelle a été lancé le programme. Nous n'avons pas néanmoins eu le temps de terminer le découpage du texte et les différentes analyses fréquentielles.

LE CODE DE CESAR

Le code de César est une méthode de chiffrement mono-alphabétique. Celle-ci consiste en un décalage de lettre de la valeur donnée (la clé). L'utilisation des valeurs ASCII pour cet algorithme nous montre bien toute sa puissance.

Il aurait été possible d'implémenté une méthode qui décale chaque lettre de la valeur de la clé mais on remarque que cela revient à effectuer un chiffrement avec le chiffre de Vigenère avec une clé d'une lettre. Cette clé composée de la lettre A à laquelle on ajoute la valeur du décalage.

L'algorithme du code de César est donc une classe qui hérite de Vigenère et qui surcharge la méthode permettant de définir la clé.

Pour la cryptanalyse du code de césar, nous avons utilisé la méthode de codage à l'envers. Nous sommes partit du principe qu'il fallait tester toutes les possibilités de décalages des lettres. Ainsi, nous avons effectué tous les cas possibles de décalage entre les lettres, 26 au total. Ensuite nous avons établi une relation entre les fréquences de lettres trouvées dans le message pour chaque décalage et les fréquences de lettre dans la langue française. Une fois cette relation faite, un nombre était attribué à chaque message en fonction des similitudes entre l'analyse fréquentielle du message et celle de la langue française. Ce nombre est égal à la valeur absolue de la différence de la fréquence des lettres dans la langue française et de la lettre dans le message. Ce nombre est multiplié par 1 moins la fréquence de la lettre dans la langue française. Ceci nous donne un nombre que l'on va additionner avec les 25 autres réalisés. Enfin le test qui obtient le résultat le plus élevé des 26 tests est la bonne analyse fréquentiel. Pour avoir le décalage, il suffit de connaitre la place cette analyse dans la liste des 26 analyses et donc d'appliquer ce décalage au message.

LA SCYTALE DE SPARTIATE

La Scytale de Spartiate est un moyen de chiffrer un texte très ancien. Celui-ci consistait à l'origine à écrire son texte sur un ruban enroulé autour d'un bout de bois. Le diamètre de ce bout de bois était la clé. En effet avec un diamètre différent, les lettres ne seraient pas dans le bon ordre et le texte impossible à lire.

La première implémentation qui vient à l'esprit est d'utiliser une matrice avec un nombre de ligne égale au nombre de lettre tenant sur son diamètre. Il convient alors de lire le texte non plus en ligne mais en colonne pour le chiffrer ou déchiffrer. Ce type d'implémentation a le désavantage de nécessiter une matrice qui peut être implémenté par un tableau à deux dimensions, ce qui se révèle consommateur de mémoire pour des textes de tailles importantes.

En analysant des textes en clair et chiffrés rapidement que le chiffrement revient à décalage prédéfinie en fonction du diamètre du bâton. Notre implémentation utilise ce principe et parcours donc une seul fois le texte à chiffrer ou déchiffrer.

Pour implémenté notre solution servant au chiffrement, on commence par définir le nombre de tour du bout de boit qu'il y aurait en prenant la valeur immédiatement supérieur à la longueur du message divisé par le nombre de lettre sur le diamètre du bâton.

On crée ensuite une chaine de caractère que d'espace de la longueur du nombre de lettre sur le diamètre que multiplie la longueur des lignes. Le fait de remplir cette chaine de caractère d'espace permet de pouvoir chiffrer un texte n'ayant une longueur multiple du nombre de lettre sur le diamètre.

On parcourt enfin chaque lettre du texte à chiffrer pour la placer au bon endroit à savoir le numéro de la colonne multiplié par le nombre de caractère auquel on ajoute la position de la lettre dans le texte en clair.

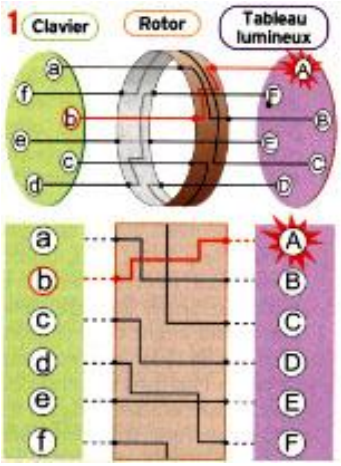
La méthode permettant de déchiffrer la Scytale de Spartiate est encore plus simple, il suffit d'incrémenter la chaine de caractère représentant le texte en clair en prenant la lettre dans le texte chiffré qui se situe à la position du numéro de colonne multiplier par le nombre de lettre sur le diamètre auquel on ajoute le numéro de la lettre actuelle.

La cryptanalyse de la Scytale de Spartiate se fait au moyen d'une analyse fréquentielle de bi-gramme. Pour cela un testes avec les différents nombres de lettres sur le diamètre possible est fait. Nous calculons ensuite un taux en fonction du nombre de bi-gramme trouvé en leur fréquence d'apparition dans la langue française. La solution ayant le taux le plus haut est retenue.

SIMULATION D'ENIGMA

Enigma était une machine utilisée pour chiffrer et déchiffrer des messages pendant la seconde guerre mondiale. Sa force était son nombre de réglage possible qui lui conférait un nombre de combinaisons de chiffrement impressionnants. Le principe de cette machine repose notamment sur des rotors qui tournent à chaque chiffrement de lettre

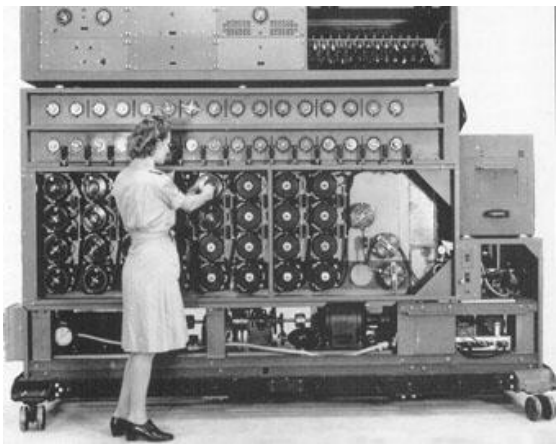
Les rotors sur la machine Enigma forment des connexions électriques, notre implémentation effectue une substitution alphabétique. Nos trois rotors réalise donc trois substitution dans un sens, une quatrième grâce au réflecteur qui ré-entre dans les rotors et ainsi sont encore faites trois autres substitutions. Au total pas moins de sept substitutions sont réalisées.



Avec les mêmes réglages, une lettre sera toujours chiffrée et déchiffré de la même manière. Cela nécessite donc une bijection au niveau des rotors. Notre implémentation semble être très sensible a ce niveau là et ne produit pas toujours le résultat escompté suivant les rotors.

La machine originelle Enigma avait en plus des différents rotors qui pouvait être inter-changé un tableau de connexion qui permettant de brouiller encore plus le message. Cette fonctionnalité n'a pas été intégrée dans l'implémentation actuelle que nous avons faite.

Bien que le nombre de possibilité de chiffrement semble rendre les codes chiffrés par cette machine indéchiffrable, des experts Polonais on réussi à trouver un moyen de cracker ces codes en se basant sur la faiblesse du protocole de transmission des Allemands qui consistait à écrire deux fois un groupe de trois lettre en début de message.



La bombe de Turing

Cette particularité leur permis de décrypter les messages codés par Enigma. La procédure de cassage des codes fut automatise au moyen de machine gigantesque. Alan Turing contribua fortement au développement de ces techniques de décryptage, il donnera d'ailleurs son nom à la machine réalisant cette tache, la bombe de Turing. Cette fonctionnalité n'a pas été développé dans notre projet.

TECHNOLOGIE TIERCES UTILISEES

LOGICIEL DE VERSIONNING GIT

Git est un logiciel permettant de stocker des fichiers en conservant l'historique des modifications qui ont été effectuées dessus. La première version a été publiée en 2005. Son développement s'est poursuivi et a permis quelques améliorations (interfaces graphiques, interfaces Web, scripts évolués). C'est un logiciel libre crée par Linus Torvalds, créateur du noyau linux. C'est un logiciel de gestion de version décentralisé. Ceci facilite grandement le partage de code entre 2 personnes.

Il est capable de faire des liens entre les différentes modifications. Une modification peut être un ajout, une suppression ou bien les deux. Chaque étape d'avancement lié aux modifications est appelé version. Une modification constitue donc l'évolution de deux versions.

Il est possible de travailler indépendamment des autres. Le logiciel se charge de mutualiser un document en le déposant dans un dossier commun. Chaque développeur sera en droit de poser des droits sur certains fichiers pour que ceux-ci ne puissent être modifiés. Cependant, il peut y avoir un problème si deux personnes travaillent sur le même code. En effet, le logiciel de gestion de version est incapable de fusionner deux mêmes codes modifiés indépendamment.

FRAMEWORK C++ QT

Qt est un Framework orienté objet développé par Trolltech en 1991 mais qui appartient aujourd'hui à Nokia. En effet, les systèmes actuels de Nokia sont basés sur Qt. A l'origine Qt était un logiciel payant, mais maintenant, Qt est sous licence LGPL. Dès le début, Il a été utilisé par Kde, un des environnements de Linux.

Qt est un environnement développé en C++ fait pour être utilisé en C++. C'est une bibliothèque multi-plate-forme, ce qui permet la portabilité des applications. Les environnements supportés sont Unix, Windows et Max OS X.

Un Framework est kit de composant logiciel, qui permet d'élaborer l'architecture d'un logiciel. Il est composé de classes mères qui seront dérivés et étendues par héritage selon les besoins.

Il existe différente version de Qt, 4 pour être précis. Nous avons utilisé dans notre projet la version 4.7.

Qt est un logiciel permettant de créer des programmes GUI (Graphical User Interface), appelé interface graphique. Ceci offre beaucoup de possibilité par rapport au mode console.

CONCLUSION

Ce projet nous a permis d'apprendre beaucoup de choses sur la cryptanalyse. De plus il a été intéressant de faire des recherches sur les différents algorithmes a élaboré pour établir nos chiffrement, déchiffrement. La conception de cryptanalyse des différents codes fut un excellent moyen de mettre en avant notre imagination pour casser les algorithmes.

Afin de réaliser ce projet, nous avons utilisé des logiciels tels que « Git » ou « Qt ». Ce sont des logiciels qui demandent un certain temps d'adaptation pour les exploités sans difficultés.

Enfin, nous avons pu acquérir de nouvelles connaissances en C++. Ce projet, par le langage que nous avons choisi d'utiliser, est une introduction à la matière de C++ que nous aurons au second semestre.