

Rapport de stage Elève Ingénieur

Emmanuel KERVIZIC

10 septembre 2010

Table des matières

1	Introduction	3
2	Contexte	4
2.1	Sujet du stage	4
2.2	Présentation de l'environnement	4
2.2.1	DSNA/DTI	4
2.3	Le site de Tahiti	5
2.3.1	Objectif contrôle aérien	5
2.3.2	Les services de la circulation aérienne	6
2.3.3	La zone de contrôle de Tahiti (FIR)	7
2.3.4	Le système de contrôle : TIARE	9
3	Expression du besoin et gestion de projet	11
3.1	Expression du besoin	11
3.1.1	Les besoins initiaux	11
3.1.2	L'évolution des besoins	11
3.1.3	Les risques	12
3.2	Gestion de projet	12
3.2.1	Choix de la méthode de gestion de projet	12
3.2.2	L' Extreme Programming	13
3.2.3	Le cycle en V	14
3.2.4	Les avantages et inconvénients	15
4	Rédaction des spécifications	16
5	Réalisation technique	18
5.1	Le contexte technique opérationnel	18
5.1.1	EUROCATX	18
5.1.2	Le domaine de l'aviation	18
5.2	Base de travail	19

5.2.1	Le langage Python	19
5.2.2	GOOGLE EARTH	20
5.3	Le programme réalisé et ses fonctions	21
5.3.1	Le fonctionnement	21
5.4	Problèmes techniques rencontrés et solution apportées	22
5.4.1	Gestion des erreurs	22
5.4.2	Intersection entre plans de vol et zone ACI	23
5.4.3	Performance du logiciel	25
6	Tests et validation de la réalisation	27
7	Synthèse	28
8	Evolution projet	29
8.1	La mise en place d'une interface graphique	29
8.2	L'automatisation de l'acquisition	29
8.3	La pérennisation des données	30

1 Introduction

Moi, le stage

2 Contexte

2.1 Sujet du stage

Représenter le trafic aérien de la FIR de Tahiti dans le logiciel Google Earth. Le logiciel GoogleEarth permet de représenter un espace tridimensionnel, de placer, à l'intérieur du logiciel, des indicateurs tels que des marqueurs de position, des lignes, des polygones... Le logiciel GoogleEarth utilise des fichiers externes de type XML pour tracer ou représenter ces graphismes. Le format de ces fichiers est ouvert et publié par Google.

Il s'agit, à partir des traces fournies par le système de contrôle de Tahiti, d'afficher le trafic aérien circulant dans la FIR à des fins d'analyse, de vérification de trajectoire, de mesure de distance...

2.2 Présentation de l'environnement

2.2.1 DSNA/DTI

La Direction des Services de la Navigation Aérienne est chargée de rendre le service de navigation aérienne pour l'État français. A ce titre, la DSNA est responsable de rendre les services de circulation aérienne, d'information aéronautique et d'alerte sur le territoire national et ceux d'outre-mer (DOM, TOM, POM). La DSNA s'appuie sur deux directions pour exécuter cette mission :

- La Direction des opérations ou DO,
- la Direction de la Technique et de l'Innovation ou DTI.

La DO est l'acteur opérationnel du contrôle aérien tandis que la DTI est chargée du volet technique. Celui-ci consiste à réaliser ou acquérir les systèmes qui participent à l'exercice du contrôle aérien. Il s'agit de systèmes informatiques permettant d'assister le contrôleur dans ses activités, de chaînes radios pour communiquer avec les aéronefs, de systèmes de traitement de l'information météorologique...

La DTI réalise également de nombreuses études pour traiter les besoins des utilisateurs et les évolutions réglementaires. La DTI réalise le déploiement et le support opérationnel des systèmes qu'elle acquiert ou réalise.

Enfin la DTI fait viser ses systèmes, procédures et formation par l'autorité de surveillance nationale (Direction de la Sécurité de l'Aviation Civile ou DSAC).

La DTI est structurée en domaines qui sont chacun en charge de plusieurs pôles de compétences :

- Recherche & développement, R et D
- Exigences opérationnelles des systèmes, EOS
- Gestion du trafic aérien, ATM
- Communication, navigation, surveillance, CNS
- Déploiement et Support Opérationnel, DSO

Chaque pôle couvre un ensemble de fonctions et d'expertises.

Pôle ATM/VIG :

- Le pôle « Vol et information générale » (VIG) est responsable de la maîtrise d'ouvrage systèmes de traitement des plans de vol et informations générales, à ce titre, le pôle assure le suivi industriel de leur réalisation ou de leur acquisition. Le pôle VIG est également chargé de leur maintien en condition opérationnelles lorsqu'ils sont déployés.
- Le pôle ATM/VIG est notamment responsable de la maîtrise d'ouvrage de systèmes déployés en outre-mer. L'aéroport de Tahiti (Polynésie française) a récemment été modernisé avec un système entièrement acquis auprès d'un industriel, couplé à un radar dans le cadre du projet TIARE, qui s'est terminé en 2009.

2.3 Le site de Tahiti

2.3.1 Objectif contrôle aérien

Le contrôle aérien est un ensemble de services (cf. [2.3.2 page suivante](#)) rendus par les contrôleurs aériens aux aéronefs afin d'aider à l'exécution sûre, rapide et efficace des vols. Les services rendus sont au nombre de trois, appelés « services de la navigation aérienne », dans les buts de :

- prévenir les collisions entre les aéronefs et le sol ou les véhicules d'une part, et les collisions en vol entre aéronefs d'autre part (autrefois appelés « abordages »). Il consiste aussi à accélérer et ordonner la circulation aérienne,
- de fournir les avis et renseignements utiles à l'exécution sûre et efficace du vol : informations météorologiques, information sur l'état des moyens au sol de navigation, information sur le trafic (quand le service de contrôle n'est pas assuré dans cette zone),
- de fournir un service d'alerte pour prévenir les organismes appropriés lorsque les aéronefs ont besoin de l'aide des organismes de secours et de sauvetage, et de prêter à ces organismes le concours nécessaire.

2.3.2 Les services de la circulation aérienne

Comme nous l'avons vu plus haut, le contrôle aérien rend plusieurs services. Nous allons voir ces services plus en détail.

Le service de contrôle

Le service de contrôle est assuré dans les buts suivants :

- Prévenir les collisions entre aéronefs ou entre un aéronef et un obstacle
- Accélérer et ordonner la circulation aérienne

Le plus important reste donc la sécurité des vols. La meilleure description d'un contrôleur est un ange gardien. Il s'assure que rien n'arrivera à l'aéronef pendant son vol par des causes extérieures (autre avion, obstacle), et qu'il arrivera à sa destination le plus vite possible. En outre le contrôleur est responsable de la sécurité des vols sous sa juridiction.

Les moyens qu'utilise le contrôleur pour prévenir les abordages sont la séparation (anciennement l'espacement) et l'information de trafic.

- La séparation consiste à ménager entre deux aéronefs une distance minimale, garantissant la sécurité de ces deux avions.
- L'information de trafic est une information précise sur la position d'un autre aéronef pouvant se rapprocher dangereusement. Le pilote peut ne pas voir qu'un avion se rapproche, l'information de trafic l'aide à voir, afin de permettre au pilote d'éviter l'aéronef conflictuel.

Le service d'information

Le service d'information de vol est assuré sur tout le territoire français. En espace aérien contrôlé, il est assuré par le service de contrôle. Dans les espaces aériens non contrôlés, il est assuré par un organisme UIV (dans les CRNA) et SIV (dans les approches) en vol, ou AFIS sur un aérodrome. Il consiste à délivrer aux aéronefs les renseignements et avis nécessaires à l'exécution sûre et efficace du vol. Ces renseignements peuvent être (liste non exhaustive) :

- Météorologiques : conditions météo sur un terrain, présences d'orages...
- Information sur le trafic (à ne pas confondre avec l'information de trafic) : information sur un trafic connu ou inconnu, en fonction des éléments disponibles, pouvant interférer avec un aéronef.
- État des aides à la navigation
- État des équipements sol d'un terrain

- Amendements de plan de vol
- Information sur la position, aide aux pilotes perdus
- Autres...

Le service d'alerte

Le service d'alerte est aussi vaste que naturel. Il consiste à répondre à tous les besoins des avions qui se disent en détresse, ou dont on peut penser qu'ils sont en détresse. Ce service recouvre des domaines très variés :

- Si un avion a déposé un plan de vol, et que le contrôle à l'arrivée a reçu confirmation qu'il a bien décollé, il doit surveiller que l'avion arrive bien à destination aux alentours de l'heure prévue, et lancer des recherches si ce n'est pas le cas.
- Si un avion ne répond plus à la radio et disparaît du radar, le contrôleur doit vérifier si l'aéronef a eu un problème et s'il s'est écrasé ou posé en urgence. Il déclenche alors les secours pour rechercher l'épave et secourir les occupants.
- Si un aéronef s'écrase sur la piste ou à proximité de l'aérodrome, il déclenche immédiatement les secours et coordonne leur action jusqu'à l'arrivée des renforts.
- Si un pilote signale avoir des problèmes avec son aéronef de nature à entraver le bon déroulement du vol, le contrôleur peut lui donner une priorité absolue à l'atterrissage en écartant tous les autres aéronefs.
- Si le contrôleur sait ou soupçonne qu'un aéronef est détourné, il prévient les autorités compétentes et leur apporte tout le secours nécessaire.

D'une manière générale, ce service est une autorisation légale à porter secours par tous les moyens à un pilote en difficulté. Tout être humain le ferait, mais le service d'alerte donne au contrôleur une justification légale pour retarder ou dérouter certains aéronefs afin de porter secours à un autre.

2.3.3 La zone de contrôle de Tahiti (FIR)

Le transport aérien

L'île de Tahiti est desservie par l'Aéroport International Tahiti Faa'a, situé à 5km au Sud-Ouest de Papeete. Inauguré en 1961, et détenu à 57% par le Territoire de la Polynésie Française¹⁰, c'est le plus important aéroport de la Polynésie française, et le seul aéroport international du territoire. Il s'agit donc de l'unique point d'entrée pour l'immense majorité des visiteurs mais également pour les habitants des autres îles de la Polynésie française.

L'aéroport de Tahiti a joué un rôle primordial dans le désenclavement de l'île (et du

reste du territoire) et le développement du tourisme. L'aéroport assure les liaisons avec une dizaine de destinations internationales : Los Angeles, Paris, Auckland, Sydney, Tokyo, Rarotonga, Santiago, l'Île de Pâques, Noumea et Honolulu¹⁰. Conscient de l'importance des liaisons aériennes internationales dans le développement économique de l'île et du pays, le gouvernement a inauguré en 1998 sa propre compagnie aérienne : Air Tahiti Nui (ATN), qui dessert aujourd'hui 5 destinations à partir de Tahiti : Paris, Los Angeles, Tokyo, Auckland, Sydney.

Concernant le réseau domestique, l'aéroport dessert l'ensemble des archipels de la Polynésie. Air Tahiti est la seule compagnie à desservir régulièrement les îles polynésiennes, assurant la liaison avec une quarantaine d'îles et d'atolls. L'île de Moorea, située à 7 minutes de vol de Tahiti est desservie par Air Moorea, une filiale de la compagnie domestique d'Air Tahiti. L'aéroport de Tahiti est la plaque tournante du trafic aérien, puisque la majorité des destinations sont uniquement desservies par l'aéroport de Tahiti. La centralisation du réseau aérien accentue donc l'attraction et l'influence de Tahiti et de l'agglomération de Papeete sur le reste des îles polynésiennes.

Etendue de la FIR

La région d'information de vol de Tahiti ou « Flight Information Region » (FIR Tahiti) s'étend bien au-delà des eaux territoriales et déborde même sur l'hémisphère nord pour atteindre le parallèle 03 ° 30' Nord, soit près de 3700 km de nord au sud et à peu près autant d'est en ouest, couvrant environ 12,5 millions de km².

Cette FIR constitue le volume au sein duquel la fourniture des services de la circulation aérienne sont assurés sous la responsabilité de l'administration française. Ces services comprennent :

- alerte et sauvetage
- information de vol
- contrôle de la circulation aérienne

La FIR Tahiti est fréquentée par différents types de trafic :

- les vols transpacifiques (entre la côte ouest des Etats-Unis et la Nouvelle-Zélande ou l'Australie)
- la desserte internationale de Tahiti (depuis et vers les Etats-Unis, la Nouvelle-Zélande, l'Australie, le Japon et le Chili)
- les vols intérieurs (desserte domestique des 47 aéroports de Polynésie Française)

Plus de 40 contrôleurs aériens travaillent 24h/24 et 7j/7 dans la tour de contrôle de Tahiti-Faa'a.

Plus de 20 contrôleurs travaillent sur les aérodrômes contrôlés des îles.

En 2006, le centre de contrôle a contrôlé 102 132 mouvements (+2,5%), dont 71477 mouvements IFR¹ et 30655 mouvements VFR².

La zone ACI :

Une fonction de contrôle spécifique, nommée ACI³ ou zone ACI, a été développée dans le système EUROCAT-X pour répondre à des besoins de contrôle. Il s'agit d'une zone particulière limitrophe à la FIR (cf. [2.3.3 page précédente](#)) de Tahiti, dont la limite se situe à 50 miles nautiques de la FIR. La zone ACI encercle la FIR. Il est à noter que cette zone n'est pas sous la responsabilité des contrôleurs aériens français, cependant, les vols pénétrant dans cette région sont visualisés par le système Eurocat-X

Ainsi en visualisant le trafic aérien dans la zone ACI, les contrôleurs peuvent maintenir les séparations entre les aéronefs. C'est-à-dire vérifier que les vols qui sont à l'extérieur et longent la FIR de Tahiti sont séparés des vols évoluant dans cette FIR.

2.3.4 Le système de contrôle : TIARE

TIARE est un système de gestion du trafic aérien pour le centre de contrôle de Tahiti, en remplacement des systèmes vieillissants de visualisation du trafic (VIVO) et de gestion de plans de vol et d'informations générales (SIGMA). Les situations de contrôle auxquelles doivent faire face les contrôleurs sont multiples, il y en a en effet à traiter les spécificités du contrôle océanique, du contrôle d'approche et inter-îles. Le système TIARE est construit à partir de plusieurs « produits sur étagère » :

- EUROCAT-X, système en charge du traitement radar et de la gestion plans de vols.
- ATALIS, système en charge de la préparation des vols, de la gestion des NOTAM, et de la présentation d'informations générales au contrôleur tour et approche.

Les systèmes EUROCAT-X et ATALIS sont connectés au commutateur CAGOU, raccordé aux liaisons externes (RSFTA). ATALIS reçoit également des informations météorologiques en provenance du système local d'acquisition de ces données appelé CAOBS. EUROCAT-X est raccordé au radar secondaire du mont Marau et au réseau ACARS.

PREVOIR SHEMA

-
1. IFR : (soit, en anglais, Instrument flight rules) règles de vols aux instruments
 2. VFR : Visual flight rules, nom anglais de « Vol à vue »
 3. ACI : Area Common Interest, soit une zone d'intérêts commun

La zone Aci (cf. [2.3.3 page précédente](#)), a été développée spécifiquement dans le système EUROCAT-X pour répondre à des besoins de contrôle.

3 Expression du besoin et gestion de projet

3.1 Expression du besoin

3.1.1 Les besoins initiaux

L'objectif initial était de pouvoir réaliser un logiciel banalisé et ergonomique permettant de représenter l'ensemble des données de contrôle (repères, balises, secteurs...) afin de pouvoir visualiser le trafic aérien circulant dans la FIR et la zone ACI. Les bénéfices attendus de cet outil sont :

- l'amélioration de l'analyse et de la compréhension visuelle du trafic aérien de Tahiti,
- la possibilité d'élaborer de statistiques à partir des fonctions de calcul du logiciel,
- une aide dans le travail de définition des points d'entrée dans la zone ACI que réalise le service de contrôle de Tahiti.

3.1.2 L'évolution des besoins

Au début de projet les besoins initiaux ont été définis. Nous verrons par la suite comment ceux-ci ont pu évoluer. Il faut noter que le client est assez dirigiste, il a déjà vu ce produit pour d'autres applications et a donc une vue globale de ce qu'il souhaite en sortie. A savoir :

- Une application étant basée sur le logiciel GOOGLE EARTH.
- Python comme langage de programmation

Par contre le besoins précis de l'utilisation du produit reste indéterminé. C'est pourquoi nous avons orienté notre gestion de projet vers une méthode dite agile (cf. [3.2.2 page 13](#)). Cette méthode nous permettra de redéfinir les besoins tout au long du projet en fonction de ce qui a déjà été réalisé. Et ainsi obtenir un produit correspondant au mieux à ce que le client aurait pu espérer.

Lors du lancement du projet les besoins étaient :

- Représenter le trafic aérien déposé par les plans de vol dans la zone de contrôle de TAHITI dans GOOGLE EARTH.
- Visualiser la configuration de la plate-forme TIARE (zone de contrôle, point caractéristique ...)

Tout au long du projet de nouveaux besoins sont apparus tel que :

- Représenter le trafic aérien en fonction du temps
- Définir approximativement l'heure d'entrée de et sortie des avion dans la FIR (cf. [2.3.3 page 8](#)) en fonction de leur plan de vol déposé.
- Visualiser le vol des avions en temps réel grâce aux données ADS (cf. [?? page ??](#)).
- Visualiser le positionnement des avions estimé par le système TIARE entre deux reports ADS afin de visualiser l'interprétation des données reçue par le système.
- Différencier les type de vol en quatre catégories : Entrant, Sortant, Transit, Interne.

3.1.3 Les risques

Lorsque l'on a comme projet de réaliser une application qui a déjà été réalisé par le passé nous avons une base sur la-quel se référencer (en terme de méthode, de temps, de coûts). Hors sur un projet tel que le nôtre ou même aucun prototype n'a encore été réalisé le risque que cela ne fonctionne pas est très élevé.

C'est pour cela qu'une méthode de gestion de projet dite agile décrite ci-dessous (cf. [3.2.2 page suivante](#)) à été utilisé. Cette méthode nous a permis d'avancer petit a petit afin de susciter des besoins "réalisable". Contrairement à la méthode en V utilisée originellement a la DTI ou les besoins et les spécification sont déterminé avant le début de la réalisation technique.

3.2 Gestion de projet

3.2.1 Choix de la méthode de gestion de projet

Comme nous l'avons vu précédemment, les besoins ne sont pas clairement définis dès le début. Il était donc difficile de pouvoir établir des spécifications claire afin de pouvoir réaliser un cycle en V (cf. [3.2 page 14](#)). Nous avons donc choisis une méthodologie de gestion de projet différente de celle appliquée en temps normal à la DTI.

Cette méthodologie devait nous permettre de débiter un projet sans en connaitre réellement l'aboutissement final tout en gardant de la rigueur et de l'organisation. Nous avons donc décidé d'utiliser une méthode dites agile¹. Après quelque recherche

1. Les méthodes Agiles sont des groupes de pratiques pouvant s'appliquer à divers types de projets, mais se limitant plutôt actuellement aux projets de développement en informatique (conception de logiciel). Les méthodes Agiles se veulent plus pragmatiques que les méthodes traditionnelles. Elles impliquent au maximum le demandeur (client) et permettent une grande réactivité à ses demandes. Elles visent la satisfaction réelle du besoin du client et non les termes d'un contrat de développement.

itérations peuvent devenir plus courtes (par exemple la séparation des plans de vol en catégories tel que : transit, interne ...)

Pour résumé, cette méthode nous amène à réaliser la boucle suivante :

- Analyse du besoin.
- Expression des spécifications
- Réalisation technique
- Test de la réalisation
- Revue logicielle (validations qui permettront de faire évoluer le produit)

3.2.3 Le cycle en V

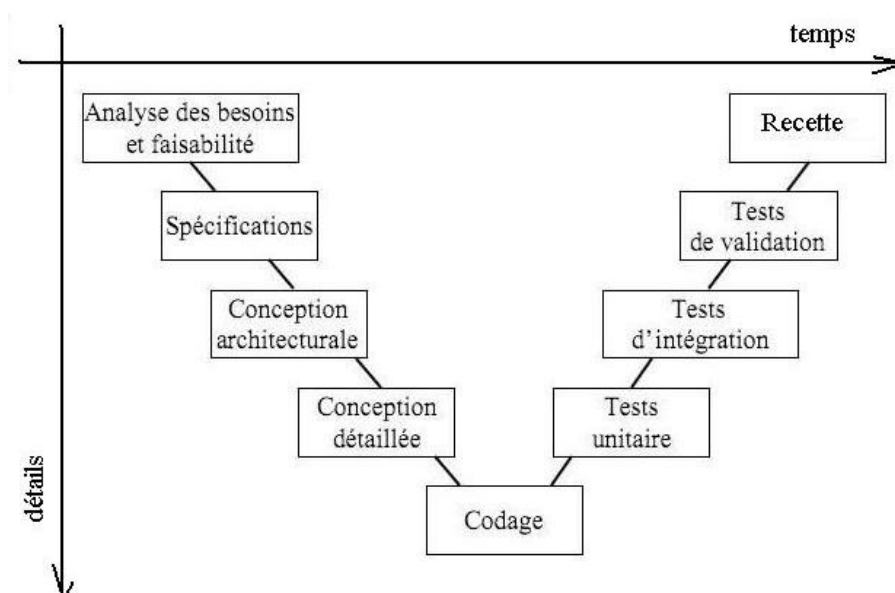


FIGURE 3.2 – Les phases à travers le temps et le niveau de détails.

Le modèle du cycle en V est un modèle conceptuel de gestion de projet imaginé suite au problème de réactivité du modèle en cascade. Il permet, en cas d'anomalie, de limiter un retour aux étapes précédentes. Les phases (cf. Fig. 3.2) de la partie montante doivent renvoyer de l'information sur les phases en vis-à-vis lorsque des défauts sont détectés, afin d'améliorer le logiciel.

Le cycle en V est devenu un standard de l'Industrie logicielle depuis les années 1980 et depuis l'apparition de l'Ingénierie des Systèmes est devenu un standard conceptuel dans tous les domaines de l'Industrie. Le monde du logiciel ayant de fait pris un peu d'avance en termes de maturité, on trouvera dans la bibliographie courante souvent des références au monde du logiciel qui pourront s'appliquer au système.

Les étapes qui constituent cette méthode sont les suivantes :

- Analyse des besoins et faisabilité
- Spécification logicielle
- Conception architecturale
- Conception détaillée
- Codage
- Test unitaire
- Test d'intégration
- Test de validation (Recette Usine, Validation Usine - VAU)
- Recette (Vérification d'Aptitude au Bon Fonctionnement - VABF)

On voit bien que dans notre cas, avec des besoins indéfini, il est impensable d'appliquée une telle méthodologie sans engendrer le risque que le logiciel final ne marche pas ou ne réponde pas aux attentes du client.

3.2.4 Les avantages et inconvénients

Les avantages de cette méthode, dans cette situation, sont :

- Enrichir le produit à chaque itération du cycle. Si le logiciel marche le client peut visualiser immédiatement les besoins qui étaient superficiels, dont il n'avait pas réellement besoin, et au contraire découvrir de nouveaux besoins réellement utiles.
- Rediriger rapidement la conduite du projet. Si le client souhaite rediriger son projet, ceci peut être fait dans les meilleurs délais (changement d'objectifs ou de priorités)
- ... à compléter.

Cette méthode implique tout de même un certain nombre d'inconvénient tel que :

- Le client doit être disponible afin de faire avancer le projet. Chaque validation est vue avec le client et c'est celui-ci qui donne les nouveaux besoins. Ce qui implique que si celui-ci n'est pas disponible, le projet peut vite être bloqué.
- Le projet peut vite dériver. Ce type de méthode requiert des personnes compétentes, aussi bien au niveau Maître d'ouvrage, qu'au niveau maître d'œuvre. Il est facile de s'égarer c'est pourquoi une organisation et une rigueur doivent être entretenues tout au long du projet.
- Lors du début du projet, on manque cruellement de spécifications. On se lance alors dans le développement sans analyse complète.
- ... à compléter.

4 Rédaction des spécifications

Comme vous l'avez compris, utiliser une méthode agile comme l'extreme programming (cf. ?? page ??) implique une perpétuelle réécriture des spécifications. Celle ci sont améliorée, réécrite, ajoutée tout au long du projet. C'est pourquoi dans ce rapport sera cité la dernière version des spécifications.

Les spécifications du logiciel sont les suivantes :

Capture de fichiers de configuration : Les points caractéristiques, route, zone de contrôle et ACI, doivent être récupérés dans les fichiers de configuration du système TIARE afin d'avoir la représentation la plus juste de ce que le système a. Ils doivent être gardé en mémoire pendant toute l'exécution du logiciel afin de pouvoir être utilisé. Les données seront enregistrées dans des objet le temps de l'exécution du programme afin de faciliter leur exploitation. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier le chemin du fichier de configuration.

Capture des données Plan de vol : Les données Plan de vol doivent être récupérées dans les log du système TIARE. Par contre il doit être possible de les récupérer d'un autre fichier contenant des trames FPL au format normalisé par la norme 4444 (cf Bib. [?]). Chaque plan de vol sera enregistré dans un objet ayant un identifiant comprenant : L'identifiant de l'avion, son aéroport de départ ainsi que l'heure et le jour de départ. Cet identifiant a pour but de les différentier et de les référencer dans le temps. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier le chemin du fichier de log.

Capture des données ADS : Les données ADS doivent être récupérées dans les log du système TIARE. Il devra être aussi récupérer dans ces log les points de la position en fonction du temps calculé par le système entre deux report ADS. Les reports ADS et points calculé seront instancié par avion et par vol. L'identifiant de chaque vol sera donc composé de l'identifiant de l'avion ainsi que de la date et l'heure du message de login. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier le chemin du fichier de log.

Les points caractéristiques : Ces points devront être implémentés dans GOOGLE EARTH avec la possibilité de les afficher ou non. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier la possibilité de rééditer ou non le fichier source GOOGLE EARTH. Ces points seront représenté par un triangle de petite taille.

Les zones de Contrôle et ACI : Les zones de contrôle et zones ACI devront être implémenté dans GOOGLE EARTH avec la possibilité de les afficher ou non. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier

la possibilité de rééditer ou non le fichier source GOOGLE EARTH. Ces zones seront représentées par une surface colorée en 2 dimension.

Les routes : Les routes devront être implémentées dans GOOGLE EARTH avec la possibilité de les afficher ou non. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier la possibilité de rééditer ou non le fichier source GOOGLE EARTH. C'est route seront représentée par une ligne de couleur Jaune. Les points définissant cette route ne seront pas illustrés afin de ne pas faire de doublon avec les points caractéristique. Les coordonnées des points de chaque route devront être défini à partir des points caractéristique en mémoire.

Les plan de vol : Les plans de vol devront être implémenté dans GOOGLE EARTH avec la possibilité de les afficher ou non. La configuration du logiciel doit laisser à l'utilisateur la possibilité de spécifier la possibilité de rééditer ou non le fichier source GOOGLE EARTH. Les plans de vol doivent pouvoir être visualisé dans GOOGLE EARTH en fonction du temps. Pour se faire une heure théorique de passage sera calculé par le programme pour chaque point définissant le plan de vol. Toutes les informations concernant chaque plan de vol tel que ça route, les points constituant sa route et sa situation dans le temps devront être regroupé dans un dossier. Le message FPL de l'avion doit être visible dans la description de ce dossier. Les plans de vol seront visible durant toute la durée du vol et représenté par une ligne noire. La visualisation dans le temps sera représentée par un segment de couleur choisie aléatoirement pour chaque vol défini par les deux points les plus proches de l'heure en paramètre dans le logiciel (un point avant et un point après). Ce segment et ses points ne seront visible qu'à partir de l'heure du premier points jusqu'à l'heure du deuxième.

L'intersection du plan de vol avec la zone ACI : L'intersection, si elle a lieu, entre le plan de vol et la zone ACI doit être calculé, définie et représenté dans GOOGLE EARTH par un point rouge accompagné du nom de l'avion et de l'heure d'intersection affiché en rouge également. Ces points devront être contenu dans le dossier du concerné.

Les report ADS : Chaque report ADS sera composé de ces points de report ainsi que des points calculé par le système. Chaque report sera regroupé dans un dossier par vol et aura comme description le message reçu. Chaque point calculé par le système sera attribué et regroupé avec le report précédent. Les vols seront représenté par une ligne blanche retraçant tout les report reçu, ainsi que chaque point affiché dans le temps. L'intérêt étant de visualisé l'écart entre le chemin parcouru par l'avion et le plan de vol déposé ainsi que la différence entre la trajectoire de l'avion et celle calculée par le système TIARE.

5 Réalisation technique

5.1 Le contexte technique opérationnel

5.1.1 EUROCATX

Il faut bien comprendre comment marche le système afin de bien visualiser d'où proviennent les informations. Comme décrit grossièrement dans le schéma (cf. Fig. [5.1 page suivante](#)), EUROCATX récupère les informations sur les plans de vol par l'intermédiaire de CAGOU¹. Il récupère aussi le positionnement émis par l'avion à l'aide de la transmission Satellite, VHF² ou des données radars lors de son approche. Le système EUROCATX donne un accès à la bureautique protégé par un par-feu (Fire-Wall) afin de rendre disponible sur ce réseau un certain nombre d'information. Dans notre cas nous y récupérerons :

- toutes les données de configuration du système tels que les nom et coordonnées des balise référencée, la position des zone de contrôle et des zone ACI ou encore les route utilisée pour décrire les plans de vols.
- Les fichiers de log du Commutateur CAGOU afin de pouvoir exploiter les plans de vol reçus par le réseau RSFTA.
- Tous les reports ADS reçu par satellite et traité par le système.

Le système envoyé les informations récoltées et celle calculées au visues³ situées dans la tour de contrôle au niveau de la Vigie ou de la salle CCR ainsi que de la position déportée à MOREA.

Les données seront donc récupérée dans les fichiers ".asf" pour tous ce qui est de la configuration du système, dans les fichier du FDP pour les plans de vol et dans les fichiers du serveur ADS pour les reports ainsi que pour la position calculée des aéronefs.

5.1.2 Le domaine de l'aviation

Il m'a aussi été nécessaire de prendre connaissance de tous les termes, unité, convention et j'en passe utilisé dans le domaine aéronautique.

1. CAGOU : nom donné au commutateur RSFTA

2. VHF : Very High Frequency, soit une bande radio de très haute fréquence

3. Visue : Nom pour décrire les ordinateur utilisés pour visualiser les données de contrôles

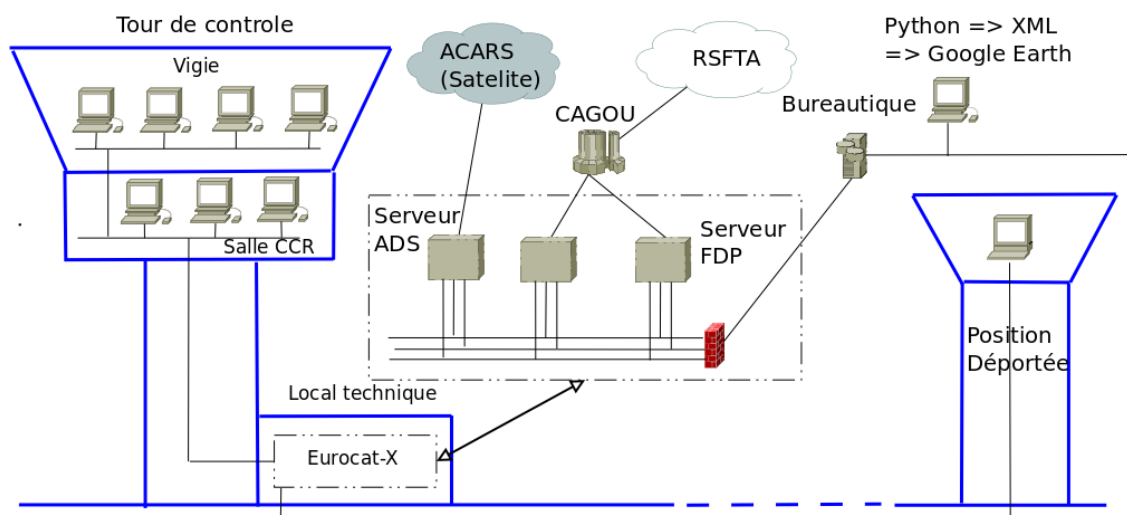


FIGURE 5.1 – Schématisation du système EUROCATX au niveau des tours de contrôle

Les coordonnées et unités :

Tout d'abord est vite venu le problème de conversion de coordonnées, J'ai donc du revoir les conversions de coordonnées sphériques ainsi que les conversions de distances. J'ai également du, comme expliquée ci dessus (cf. 5.4.2 page 23) me remémoré les solutions de calcul du point d'intersection de deux arc de cercle en coordonnées sphériques.

Convention :

Plusieurs conventions on du être acquise comme celle utilisé par le système TIARE pour décrire les report ADS ou encre celle utilisée par les compagnies pour le dépôt de plan de vole. NE PAS OUBLIER DE FAIRE RÉF AU DOCUMENT 4444 ...

5.2 Base de travail

5.2.1 Le langage Python

Bien coder :

Afin de pouvoir apprendre les bonne pratique de la programmation Python j'ai lu un livre intitulé "Programmation Python, conception et Optimisation"[5]. Celui-ci m'a permit de pouvoir d'une part revoir ce qui avait été appliquer lors de mes études et d'autre part avoir une vue global sur le langage et ainsi pouvoir prendre du recule lors du codage.

Celui ci m'a par exemple appris le nouveau style de programmation qui part du principe que chaque nouvel objet défini est basé sur un Objet existant, et que par la même occasion tout en python était Objet (même une simple variable booléenne). Ou encore la manière de vérifier si un objet était faux, égale à 0 ou encore une chaîne vide simplement en demandant si il existait (ex : `"if x!= 0:"` devient `"if not x:"`)

Utiliser les expressions régulières :

L'apprentissage de l'utilisation des expressions régulières⁴, m'a été grandement facilité grâce au site : [http://www.dsimb.inserm.fr/\[2\]](http://www.dsimb.inserm.fr/[2]) et à la documentation en ligne de Python[4]. Il s'est avéré après apprentissage que ces expressions régulières nous ont grandement facilité la faisabilité du projet.

L'optimisation :

Je pourrais citer un passage du livre[5] qui dit :

Fourni dès le départ avec des modules de tests, Python est un langage agile. Le terme agile est originellement issu de la méthodologie de programmation agile (Beck et Al.), très proche de la programmation itérative. Cette méthodologie, qui réduit les risques liés à la conception de logiciels, introduit entre autres des principes de tests continus du code. Vincent LOZANO.

En effet il m'a été rapidement nécessaire de réaliser des tests, aussi bien pour vérifier que mon code était valide que pour vérifier que celui-ci s'exécutait normalement. Il s'est avéré à plusieurs reprises que certaines parties de mon code étaient très gourmandes en processus. L'apprentissage de fonctions de test de code, tel que le module `unittest` décrit plus tard (cf. 5.4.3 page 25), m'a été rapidement nécessaire.

5.2.2 GOOGLE EARTH

GOOGLE EARTH est un logiciel, propriété de la société GOOGLE, permettant une visualisation de la terre en 3 dimensions avec un assemblage de photographies aériennes ou satellites. Ce logiciel donne la possibilité de configurer un environnement,

4. Une expression régulière est en informatique une chaîne de caractères que l'on appelle parfois un motif et qui décrit un ensemble de chaînes de caractères possibles selon une syntaxe précise.

ajouter des lignes, des points ou encore des polygone en 3D en passant par des fichier de configuration au format KML ⁵.

Ce format, qui repose sur le XML ⁶, a l'avantage d'être simple à manipuler. Ça sémantique est définie sur le de google (cf. Bibliographie [3])

5.3 Le programme réalisé et ses fonctions

5.3.1 Le fonctionnement

La configuration : Le programme réalisé ne possède pas encore d'interface (IHM) graphique. Il est donc nécessaire de configurer les option a l'aide d'un fichier de configuration (cf. annexe ?? page ??). Nous pourrons régler par l'intermédiaire de celui-ci :

- Les fichiers Kml à recréer ou non, se qui est utile afin de ne pas avoir à recréer des fichiers statique (tel que la position des point caractéristique ou encore des zones de contrôles) a chaque utilisation tout en laissant a l'utilisateur la possibilité de les mettre a jour simplement.
- Les différant styles et couleurs.
- L'emplacement des fichiers de configuration.
- les description et noms appliqué à chaque catégorie.

L'exécution : Le fichier de configuration renseigné, le programme peut être lancé. Il est possible de le lancer par l'intermédiaire d'un Shell ⁷, par l'intermédiaire de l'interface Python ou encore en direct si les informations pour gérer et lancer les fichiers Python ont été renseignées dans le système d'exploitation.

Le résultat L'exécution du programme réalise une suite d'action :

1. Lire le fichier de configuration afin de déterminer les action à effectuer.
2. Lire les fichiers de configuration du système TIARE afin de récupérer toutes les variable nécessaire sous forme d'objet ⁸ (ex : points caractéristique ...)

5. KML : Keyhole Markup Language, est un format de fichier et de grammaire XML pour la modélisation et le stockage de caractéristiques géographiques comme les points, les lignes, les images, les polygones et les modèles pour l'affichage dans GOOGLE EARTH, dans GOOGLE MAPS et dans d'autres applications.

6. XML : Extensible Markup Language («langage extensible de balisage»), est un langage informatique de balisage générique.

7. Shell : Interface en lignes de commandes

8. Objet : structure de données valuées et cachées qui répond à un ensemble de messages. Cette structure de données définit son état tandis que l'ensemble des messages qu'il comprend décrit son

3. Lire les fichiers de log afin de créer des objets tel que les plans de vol ou encore les reports ADS. Ces objet sont créer non seulement a partir de ses fichiers de log mais aussi a partir des objets créer précédemment (ex : les point des plan de vol désigné par un nom sont convertis en coordonnées à l'aide des points caractéristiques).
4. Créer les fichiers KML désigné dans le fichier de configuration à l'aide des objets instancié.
5. Créer un fichier KMZ a l'aide de tout les fichiers KML afin d'avoir un fichier compact et facile a transporter.

5.4 Problèmes techniques rencontrés et solution apportées

Comme dans tout projet il y a une multitude de problèmes à résoudre. Nous verrons dans cette partie quelques exemples de ces problèmes rencontrés ainsi que la manière dont ils ont été résolus. Cette liste reste bien entendu exhaustive au regard de tous les petits problèmes auxquels nous avons du faire face.

5.4.1 Gestion des erreurs

Problématique : Le premier problème que nous avons rencontré a été celui de la gestion des erreurs. En effet, de la première mise en route du logiciel jusqu'à la fin du stage des erreurs ont du être gérées. Deux types d'erreurs sont revenues :

- Le premier type d'erreur était par exemple une réaction in attendue du logiciel, On pourrait prendre en exemple la conversion de coordonnées reçue en Système sexagésimal⁹ en coordonnées décimales utilisées dans les fichiers KML [5 page précédente](#), qui lors des premiers tests donnait des donnée erronées.
- Le deuxième type était celui du aux erreurs contenues dans les fichiers de log utilisés pour récupérer les informations. Ces erreur faisaient effet boule de neige et venait se répercuter dans le fonctionnement du logiciel.

Résolution : La solution au premier problème a été de mettre en place des test a chaque fonction implémenter ou après avoir réaliser chaque objectif fixé. On appel cette méthode le test continu du code. Grâce à cela nous allons pouvoir déterminer plus rapidement lors d'une erreur futur d'où provient celle-ci. Une méthode simple

comportement

9. (Système sexagésimal : Degrés (°) Minutes (') Secondes ("))

de la mettre en place est de définir un test à réaliser pour valider la fonction ou le code. On détermine donc quel réaction doit avoir un fonction pour un environnement donné et l'on vérifie si le résultat correspond bien avec celui espéré. (Ex : on a la coordonnée 4530N10045E qui correspond à $45^{\circ}30'$ Nord $100^{\circ}45'$ Est. On envoie cette variable dans la fonction de conversion et l'on vérifie que le résultat retourné est bien en décimal : $45,5^{\circ}$ en latitude et $-100,75$ en longitude). Si le résultat est correct la fonction ou le morceau de code est validé, sinon il doit être corrigé. La solution du deuxième problème a été dans un premier temps d'afficher chaque erreur dans la console, mais cela est vite devenu trop compliqué du fait que la console ne retient par défaut qu'un nombre limité de ligne en mémoire et que les ligne trop ancienne sont simplement effacées. On a donc mis en place un système de log permettant, en plus d'avoir accès à l'information la plus ancienne, de pouvoir l'exploiter après avoir fermé la console, effectuer des recherches à l'intérieur et tout avantage que peut apporter un fichier texte. Pour les dernières versions de log, celles-ci sont créées avec des informations relatives au type d'erreur et l'emplacement de l'erreur dans le fichier source, le tout enregistrées dans un fichier comprenant la date et l'heure actuelle dans le nom afin de pouvoir les différencier de chaque exécution du logiciel.

5.4.2 Intersection entre plans de vol et zone ACI

Problématique : Afin de déterminer l'heure d'entrée approximative des avions dans la zone ACI (cf. [2.3.3 page 9](#)) en fonction de leur plan de vol déposé. Il est nécessaire de déterminer le point d'intersection entre leur plan de vol et la zone ACI. En théorie cela paraît simple, il suffit de prendre chaque portion du trajet du plan de vol composé de deux points et formant une droite, et de déterminer si cette droite coupe chaque droite composant la zone ACI. Dans la pratique il s'est avéré que cela était un peu plus compliqué, en effet ces droites sont en réalité des arcs de cercles qui sont composés de deux extrémités définies par des points en coordonnées sphériques (cf. schéma fig. [5.2 page suivante](#)).

Résolution : Étant donné que j'ai effectué un BTS avant d'intégrer l'EIGSI¹⁰, les notions de coordonnées sphériques ne me sont que peu familières. Après avoir vainement cherché sur internet ainsi que dans mon entourage (maître de stage, collègues de travail) je me suis replié sur un forum de mathématiques sur lequel j'ai déposé un sujet explicitant le problème (adresse, cf. bibliographie [1]). Une personne nous a donné une solution qui, après connaissance, semble tellement simple qu'on se demande pourquoi personne n'y a pensé. Cette solution consiste à déterminer les plans définis par les deux points aux extrémités de chaque arc et par le centre de la terre

10. EIGSI : École d'Ingénieurs en Génie des Systèmes Industriels située à La Rochelle

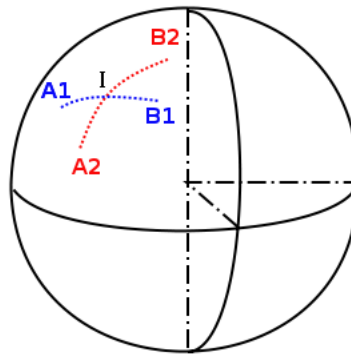


FIGURE 5.2 – Représentation grossière de l'intersection de deux arc de cercle respectivement formé par la trajectoire la plus courte entre deux points situé sur le Globe terrestre

(ainsi nous avons forcément la courbe qu'a suivi l'avion sur ce plan). Il faut ensuite déterminer la normal à chacun des plans pour en déduire la droite d'intersection de ces plans (passant par le centre de la sphère). Une fois cette droite acquise il faut définir son vecteur normal et le convertir en coordonnées sphériques. Ce qui nous donne un des points d'intersection de la droite avec la sphère, l'autre étant situé par définition à l'opposé.

Une démonstration valant amplement un long discours, et à titre informatif, voici ce que cela donne en résolution mathématique. Pour cet exemple nous avons deux arcs représentant 2 trajectoires définies chacune par 2 points A et B (cf. Fig. 5.2). Chaque point sera défini par une latitude et une longitude.

Nous avons donc :

- lat_A la latitude de A
- $long_A$ la longitude de A
- (x_A, y_A, z_A) les coordonnées cartésiennes de A
- I_1 le point d'intersection n° 1
- I_2 le point d'intersection n° 2

Il faut tout d'abord convertir les coordonnées sphériques en vecteur de coordonnées cartésiennes pour A et B :

$$A = \begin{cases} x_A &= \cos(lat) \times \cos(long) \\ y_D &= \cos(lat_A) \times \sin(long_A) \\ z_A &= \sin(lat_A) \end{cases}$$

Il faut ensuite déterminer le plan passant par O, A et B ayant alors pour équation :

$$ax + by + cz = 0$$

où

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix} \wedge \begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix}$$

c'est à dire

$$\begin{cases} a = y_A z_B - z_A y_B \\ b = z_A x_B - x_A z_B \\ c = x_A y_B - y_A x_B \end{cases}$$

L'intersection des deux plans de coordonnées (a, b, c) et (a', b', c') contient le point O, mais aussi le point P de coordonnées (x_P, y_P, z_P) tel que :

$$\begin{pmatrix} x_P \\ y_P \\ z_P \end{pmatrix} = \begin{pmatrix} a \\ b \\ c \end{pmatrix} \wedge \begin{pmatrix} a' \\ b' \\ c' \end{pmatrix}$$

P n'étant pas forcément sur la sphère, il faut trouver un point de la droite (OP) sur cette sphère. Pour cela il suffit de diviser les 3 coordonnées de P par la norme de \overrightarrow{OP} :

$$I_1 = \begin{cases} x_P / \sqrt{x_P^2 + y_P^2 + z_P^2} \\ y_P / \sqrt{x_P^2 + y_P^2 + z_P^2} \\ z_P / \sqrt{x_P^2 + y_P^2 + z_P^2} \end{cases}$$

nous avons donc I_1 et son opposé I_2 , il nous reste donc plus qu'à vérifier si chacun de ces points appartient à un des 2 arcs.

Vous trouverez le code Python correspondant à ces calculs dans la fonction : "verifyIntersection (line, point) :" du module "usualFonction.py" disponible en annexe ?? page ??

5.4.3 Performance du logiciel

Problématique : Les premiers tests du logiciel se sont déroulés sur un nombre limité de fichiers (représenté par un nombre limité d'heure de vol), ce afin de pouvoir les valider rapidement. Lors de l'apparition de fichiers plus volumineux (plus de 300Mo de donnée en entrée, environ 10% en sortie) c'est posé le problème de performance. Avant optimisation l'ordinateur moulinait des heures avant de pouvoir sortir un fichier. Il a donc fallu optimiser le code afin d'alléger le programme en ressources.

Résolution : En cherchant des conseils dans des forums d'informatique ainsi que dans le livre cité précédemment (cf. bibliographie [5], nous avons découvert que Python était un langage orienté par les tests et qu'il disposait donc de bibliothèques

spécialement conçues pour déterminer les point bloquant d'un programme et les fonctions appelées les plus gourmandes.

La fonction retenue pour repérer ce qui est appelé en anglais les Bottleneck¹¹ est la fonction "hotshot" qui à pour but d'analyser un programme dans sa totalité en indiquant notamment les ressources utilisées par chaque fonction appelée. Pour visualiser ce que donne le résultat d'une analyse veuillez vous reporter a la figure ?? page ??.

Les bottlenecks repérés, une réécriture des parties bloquantes à du être effectuée. Cette analyse nous a permis de réduire les ressources et donc le temps d'exécution du logiciel de plus de 80%.

11. Bottleneck : (goulot d'étranglement) point d'un système limitant les performances globales, et pouvant avoir un effet sur les temps de traitement et de réponse.

6 Tests et validation de la réalisation

Tests et validation de la réalisation Démarche pour tester le produit (manque pas des vols) Un fichier même vol, mais fichiers avec des vols supplémentaires Présentation du rendu Améliorations continues : à partir des tests, je repars dans le chapitre précédent (réalisation technique + nouveaux besoins (comparaison FPL//ADSC) ou correction)

7 Synthèse

Synthèse Méthode employée à consommateur de personne à disposition, produit très riche si compétence, adapté et performant, Evolution désordonnée si pas maîtrisé (base de données en plus), changement des spécifications en cours de projet, difficulté de rédaction de spécification produit fini concentre sur le dev et moins sur la doc. Pas de rédaction de manuel d'utilisateur,

8 Evolution projet

Ce projet est loin d'être arrivé a termes. Nous allons donc voir ici ce qui pourrait être fait afin de perfectionner ce logiciel. Les évolution seront axées sur trois points :

- La mise en place d'une interface graphique.
- L'automatisation de l'acquisition.
- La pérennisation des données.

8.1 La mise en place d'une interface graphique

Comme il a été expliqué précédemment (cf. [5.3.1 page 21](#)), la configuration du logiciel est effectuer manuellement par l'intermédiaire de fichiers textes et son exécution est effectuée en ligne de commande. C'est pourquoi une interface graphique faciliterait grandement son utilisation.

Cette interface devrait pouvoir faciliter la configuration et l'exécution du programme, elle pourrait être basée sur des technologie web afin de la rendre portable tout en séparent le traitement des données de l'utilisation du fichier final dans GOOGLE EARTH. En effet le programme pourrait être lancé a distance sur une machine, cela permettrait de sécuriser l'accès au données tout en libérant les ressources du poste de l'utilisateur.

Pour faciliter la configuration un histogramme avec tous les vol figurant entre deux date sélectionnée pourrait être réalisé, cela permettrait de mieux visualiser le trafic et de pouvoir cibler les vols a afficher.

Il pourrait aussi être intéressant d'inclure l'affichage final dans l'interface web, tout en laissant la possibilité a l'utilisateur de télécharger le fichier afin d'exploiter pleinement toutes les fonctionnalité du logiciel GOOGLE EARTH tel que la mesure de distance entre deux points.

8.2 L'automatisation de l'acquisition

Actuellement chaque fichier à traiter est récupéré manuellement. On pourrait concevoir un système qui irait de lui même chercher les fichiers nécessaire dans le système TIARE et les mettre automatiquement à la disposition du programme.

8.3 La pérennisation des données

Dans une optique de pouvoir rejouer simplement des situations passées, on pourrait mettre en place un système de base de données légère tel que SQLite¹. Contrairement aux serveurs de bases de données traditionnels, comme MySQL ou PostgreSQL, sa particularité est de ne pas reproduire le schéma habituel client-serveur mais d'être directement intégrée aux programmes. L'intégralité de la base de données (déclarations, tables, index et données) est stockée dans un fichier indépendant de la plate-forme.

Ce procédé couplé à un traitement automatique permettrait de mettre et garder en mémoire tout les vols disponibles sur le système TIARE. Il permettrait donc de pouvoir rejouer des situations qui n'ont été enregistrées plusieurs mois avant.

1. SQLite est une bibliothèque écrite en C qui propose un moteur de base de données relationnelles accessible par le langage SQL.

Bibliographie

- [1] KERVIZIC Emmanuel and internaute. Titre du thread. <http://maths-forum.com/showthread.php?p=692707#post692707>, june 2010.
- [2] Patrick Fuchs and Pierre Poulain. Expressions régulières et parsing. <http://www.dsimb.inserm.fr/~fuchs/python/python-node13.html>, june 2010.
- [3] Google. Documentation en ligne sur la sémantique des documents kml. <http://code.google.com/apis/kml/documentation/kmlreference.html>, june 2010.
- [4] Python v2.7. Documentation en ligne de python. <http://docs.python.org/>, june 2010.
- [5] Tareck Ziadé. *Programmation Python, Conception et optimisation*. Eyrolles, 2009.