Rapport de Stage 4ième année

RIBREAU DAVY

Département:
Informatique, Microélectronique, Automatique

OUTILS D'ANALYSE OPERATIONNELLE

Soutenu par : M. RIBREAU Davy **Tuteur Polytech'Lille :** Mme ROLLAND Nathalie **Lecteur :** M. DEFRANCE Nicolas **Année :** 2012/2013	**Nom de l'entreprise :** Airservices Australia **Responsable :** M. JASON-JONES Andrew

Remerciements

Je tiens à remercier Airservices Australia de m'avoir donné la possibilité d'effectuer mon stage dans ses locaux et de m'initier à l'ingénierie de la gestion aérienne à tous les niveaux. Mes remerciements vont notamment à Claire Marrison, manager en système de sécurité, risque et analyse, qui m'a accueilli au sein de son équipe.

J'exprime toute ma gratitude à Dr. Geoff Aldis, manager de modélisation quantitative, et à tous les personnels de son service qui m'ont entouré de leur chaleureuse attention et de toutes leurs compétences.

Je suis très reconnaissant à Andrew Jason-Jones, ingénieur recherche et analyste au sein de l'équipe, d'avoir été un tuteur de stage attentif et généreux.

Je remercie tout particulièrement Dr. Steve Barry, spécialiste en recherche, de m'avoir apporté une aide précieuse et de m'avoir guidé tout au long de mes apprentissages.

Sommaire

Introduction

J'ai effectué mon stage chez Airservices Australia, au sein du bureau principal situé à Canberra en Australie. Cette entreprise gouvernementale est entièrement auto-financée. L'entreprise est orientée sur tout ce qui concerne l'aérien en Australie, telles que la gestion des surveillances aériennes, l'optimisation du trafic, la mise en place de nouveaux protocoles ainsi que la gestion des infrastructures au sol. Airservices Australia est aujourd'hui connu comme leader mondial dans l'exécution des nouvelles technologies, la communication, la navigation et la surveillance (ADS-B, ADS-C, CPDLC et AIDC).

J'ai intégré l'équipe d'analyse opérationnelle dirigée par Robert Butcher. Ce stage clôturant la 4ième année de ma formation IMA à Polytech'Lille a duré deux mois. Le but de mon stage a été de trouver des procédés fiables et efficaces de traitements de données issues des centres de gestion du trafic aérien australien ainsi qu'étranger (message AIDC) et des appareils (message CPDLC). Mon travail a notamment eu pour objectif de faciliter l'exploitation de ces messages : mise en œuvre dans un projet en développement concernant le contrôle des performances, visualisation des messages les moins réactifs. Il m'a aussi fallu réaliser un logiciel de comparaison des temps de transmission et de statistique des messages CPDLC.

I - L'entreprise

1- Airservices Australia

Airservices Australia est une entreprise gouvernementale avec de nombreuses branches liées au développement des services aéroportuaires, à la gestion des vols, aux communications ainsi qu'à la sécurité.

a) Naissance

Airservices Australia a été créée en 1995 par le gouvernement australien pour séparer les domaines d'application d'Airservices de l'autorité de sûreté de l'aviation civile.

Les responsabilités d'Airservices sont :

- Management de l'espace aérien
- Information aéronautique
- Communication aérienne et surveillance des vols
- Aide à la navigation radio
- Sauvetage aérien et service de lutte contre l'incendie

b) Aujourd'hui

Aujourd'hui la branche principale d'Airservices Australia est situé Alan Woods Building, 25 Constitution Avenue à Canberra en Australie (ACT 2600). Airservices Australia est une entreprise importante en Australie, elle emploie environ 4000 personnes, c'est elle qui assure via ses différents services le bon déroulement des vols dans l'espace aérien australien (gestion du trafic aériens) ainsi que la sécurité au sol (pompier des aéroports).

2- L'équipe

Durant mon stage j'ai fait partie de l'équipe d'analyse opérationnelle. Son but est d'analyser les données des vols et des communications entre les différents centres de relais pour établir des statistiques. Ces statistiques sont ensuite traitées pour établir de nouveaux protocoles et/ou optimiser les protocoles de transfert de données et/ou optimiser les vols. Nous pouvons voir sur la figure suivante ce que doit traiter l'équipe, c'est-à-dire chaque type de données transmises (satellite, GPS, VHF, HF), suivant l'endroit où se trouve le vol, ainsi que les données échangées au niveau du sol.

Schéma reprenant les grandes étapes du traitement des données de l'équipe d'analyse opérationnelle d'Airservices.

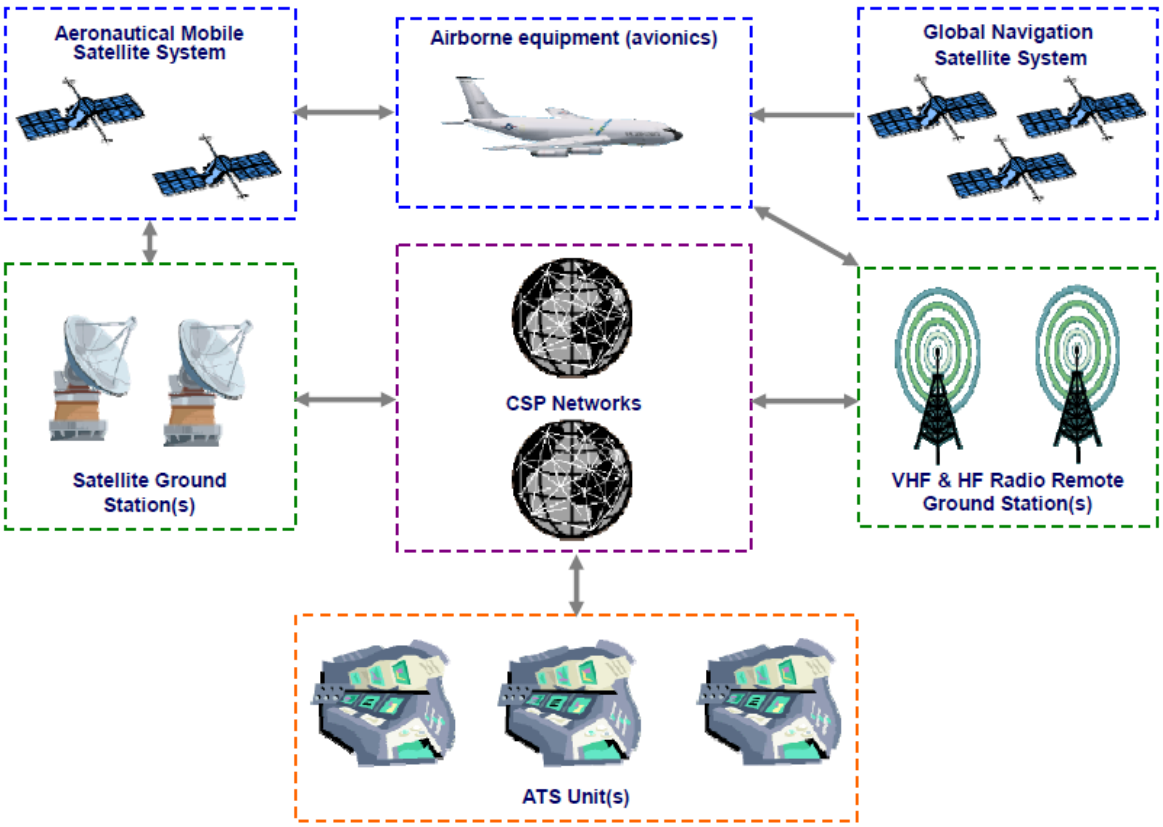


Schéma représentant la structure de communication aérienne
(Source : Global Operational Data Link Document, ICAO 2010)

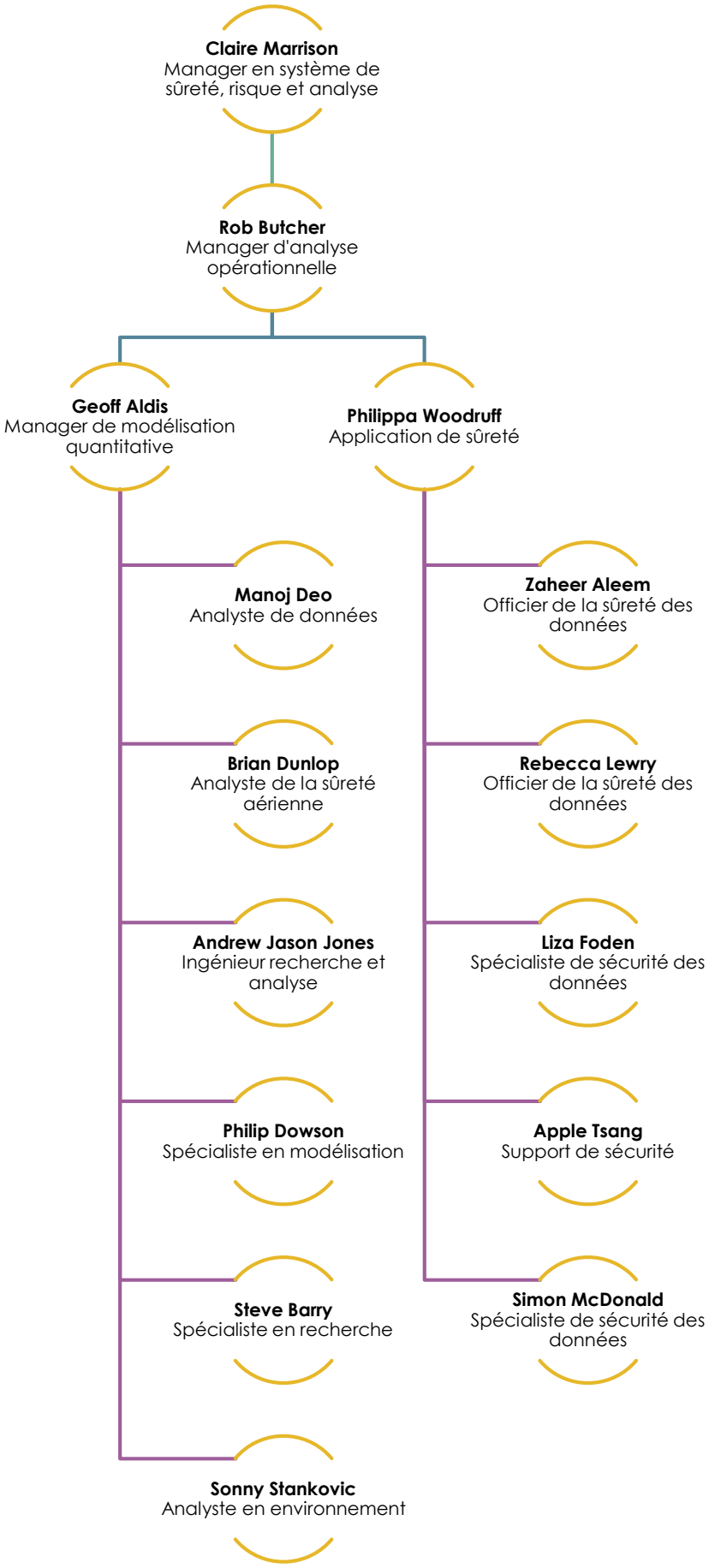


Diagramme de composition de l'équipe

II - Le stage

1- Les attentes, la problématique

Mon maître de stage m'a confié deux principales missions :

- Réaliser un traitement algorithmique sur des fichiers contenant des messages **AIDC** afin de les rendre lisibles, d'en faire ressortir les temps de transmission pour en déterminer leur performance et de classifier les informations contenues pour une utilisation directe et simplifiée par le service. Ce travail fait partie de l'optimisation des transmissions entre les différents centres de gestion du trafic aérien. Il permet de faire ressortir les problèmes et ainsi de les corriger.

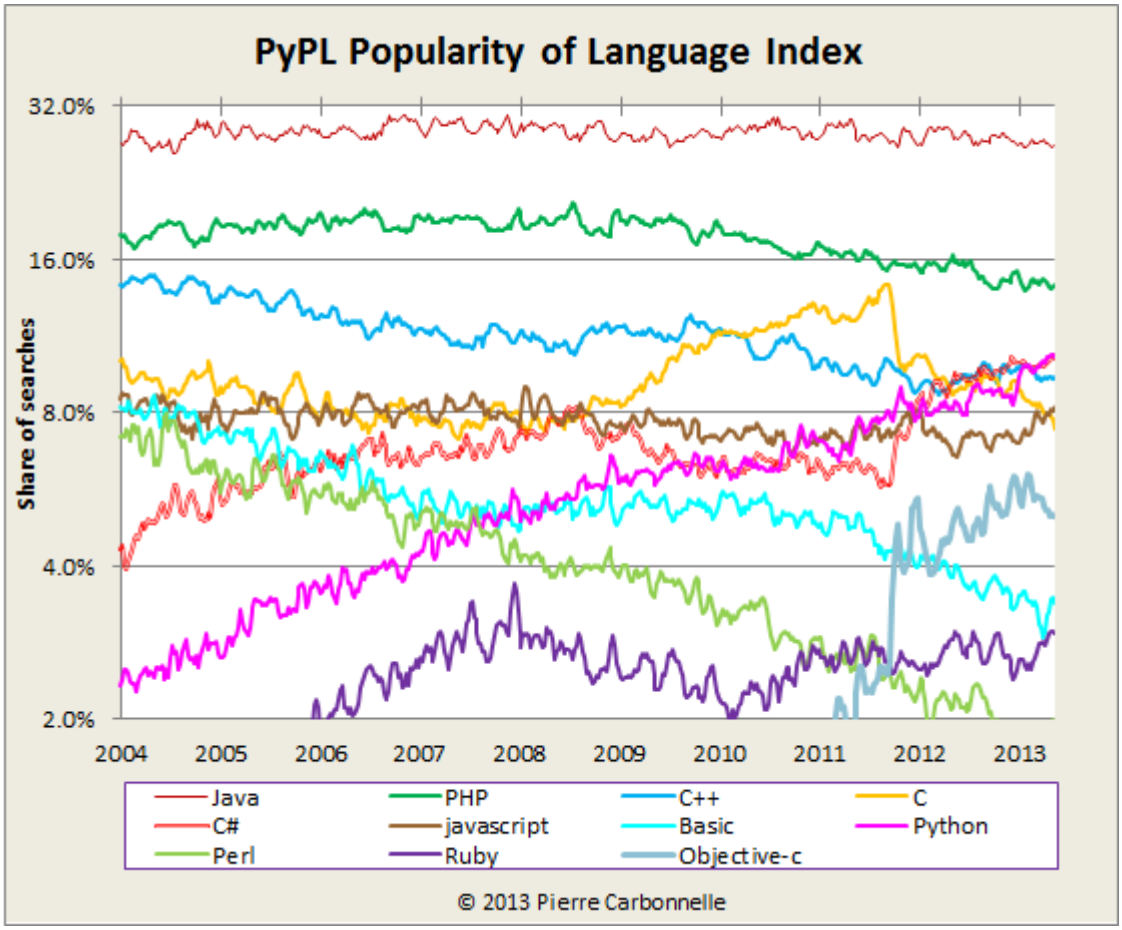
- Réaliser un logiciel de visualisation statistique des messages du type **CPDLC**, mettant en évidence les temps de transmission entre l'envoi et la réception du message, toujours dans le but de détecter des problèmes et ainsi de les corriger.

2- La technique

Le langage de programmation Python est utilisé par l'équipe d'analyse opérationnelle depuis peu de temps (décembre 2012). Les avantages de ce langage sont sa gratuité et sa haute performance au niveau des traitements matriciels. L'équipe souhaite utiliser un seul langage combinant le meilleur des différents langages tels que Perl, R et Matlab. Ce langage de programmation est certes légèrement moins rapide que le C lors de son exécution, mais très rapide à coder ce qui le rend très fonctionnel.

Python est né en Janvier 1994 grâce à Guido van Rossum, depuis ce langage ne cesse d'évoluer pour arriver aujourd'hui à la version 3.3. Python possède une bibliothèque avec de nombreux modules simplifiant la programmation et offrant des possibilités et une puissance croissante. Python est un langage assez jeune comparativement au C apparu en 1972, mais son utilisation augmente d'année en année (voir le graphique ci-dessous). C'est pourquoi nous pouvons dire que Python possède un avenir grandissant dans la programmation moderne, multi plateforme.

Durant le stage, nous avons utilisé la version 2.7 de Python qui offre une meilleure compatibilité avec les modules utilisés par l'entreprise.



Courbe d'évolution des langages de programmation
(Sources: http://sites.google.com/site/pydatalog)

3- La réalisation

Le stage s'est déroulé en trois étapes, les deux premières se sont déroulées sur les deux premières semaines.

a) Apprentissage autonome

Le langage Python étant nouveau pour moi, mon responsable de stage m'a demandé de consacrer deux semaines à son apprentissage. Dans un premier temps un ouvrage sur Python réalisé par Steeve Barry m'a été confié. Celui-ci reprend un certain nombre de fonctionnalités adaptées à l'analyse opérationnelle. Pour tester mes nouvelles compétences, un mini projet m'a été confié : extraire et classifier des données d'une trame de données.

Il m'a fallu extraire des données de cette trame contenant des numéros de vol ainsi que des articles liés aux différents vols. Ces données ont ensuite dû être classées dans des **dictionnaires** (équivalent à des listes en C). Ce qui permet d'utiliser simplement les informations extraites.

Dans un second temps, mon tuteur m'a demandé de réaliser des **classes** et un algorithme de recherche en fonction de mots clés. De cette manière l'on peut trouver rapidement chaque article ou chaque vol contenu dans cette trame, avec seulement quelques mots clés. Ce fut ma première expérience dans le traitement de données à Airservices.

Ce mini projet a demandé une semaine de travail et environ six cents lignes de code. Il m'a permis, entre autre, de prendre conscience à partir d'une situation concrète du caractère primordial inhérent à l'organisation pertinente des données et à la conception de l'application (ici un code informatique).

b) Cours d'entreprise sur Python

J'ai participé dans le cadre de l'entreprise au cours de Steve Barry. Ce cours est dispensé à l'équipe d'analyse opérationnelle pour permettre le passage de Matlab et Perl (langages principaux de l'équipe) à Python. Ce cours m'a permis d'améliorer mes connaissances fraîchement acquise sur Python et de voir des applications directement en rapport avec l'analyse opérationnelle.

Ce cours a permis d'approfondir les parties de Python utile au cycle d'acquisition des données (compression et décompression des données, connexions aux différentes bases de données).

Ce cours m'a non seulement permis de peaufiner mes compétences, mais aussi de voir comment peut se dérouler une formation interne aux entreprises dans le but de garder ses équipes à la pointe des nouvelles technologies et techniques.

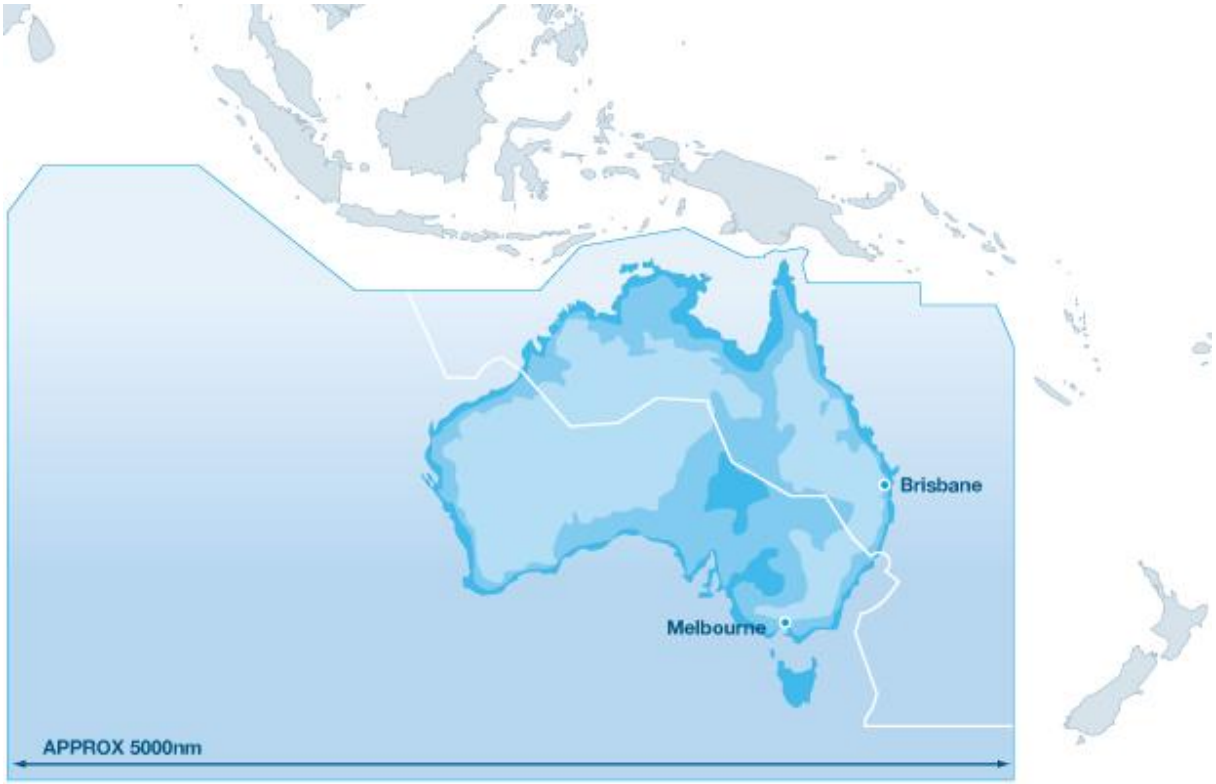
c) Projet concret

a. Traitement AIDC

L'espace aérien est divisé en un certain nombre de FIR (Flight Region Information), chacun étant géré par un fournisseur de services de navigation aérienne. Airservices Australia gérant tous les FIR australiens. Chaque FIR est géré par un centre de contrôle (ATS), qui permet de suivre le déroulement des vols ainsi que la position des avions tout au long de leur trajet. Ces centres établissent l'altitude de vol des avions ainsi que leur vitesse pour permettre des croisements (et des vols) en toute sécurité. Lors de la phase d'approche la tour de contrôle de l'aéroport prend le relais. Si l'aéroport n'en dispose pas, alors le centre ATS guide l'avion jusqu'à l'atterrissage.

En Australie il y a deux centres pour le contrôle en plein vol (ATS), celui de Brisbane (YBBB FIR) ainsi que celui de Melbourne (YMMM FIR). Le premier s'occupe de la moitié Nord du pays et le second de la moitié Sud. Les messages du type AIDC sont des messages de dialogue entre ces différents centres, et plus particulièrement entre Brisbane, Melbourne, l'Indonésie, la Nouvelle Guinée, la Nouvelle Zélande, et Sri Lanka et l'Afrique du Sud. Ils interviennent lors d'une passation de contrôle sur un vol dès lors que celui-ci change de zone de gestion.

Le traitement des messages **AIDC** permet d'effectuer des statistiques sur l'efficacité de ces messages, il permet d'établir de nouveaux protocoles et de nouvelles règles de vols afin d'améliorer les débits aériens ainsi que la fluidité.



Carte montrant les zones d'opération des ATS Australiens
(Source : airservicesaustralia.com)

Pour chaque message AIDC envoyé, le centre receveur envoie un « accusé de réception », un message de type LAM. Sans ce message, aucun ordre ne sera pris en compte. L'importance des scripts à établir est donc de mesurer le délai avant la prise en compte de chaque ordre.

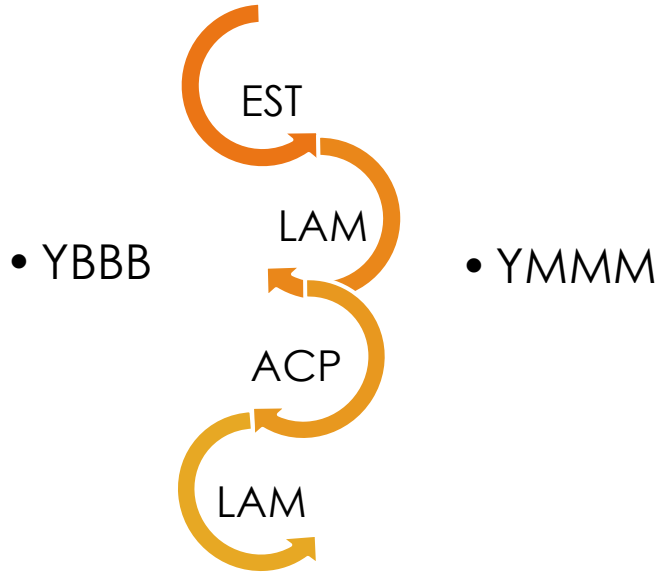


Diagramme du protocole de transmission d'un message EST
(Estimation de coordination)

Le script Python de traitement AIDC que j'ai dû réaliser est basé sur la détection des différents messages via des expressions régulières. Les messages étant tous différents (forme, code, type de contenu) le script ne pouvait se baser uniquement sur la place des mots au sein du message. Il a donc fallu instaurer un algorithme comprenant des expressions régulières afin de rechercher uniquement les 'codes' utiles. Afin que, comme il me l'a été demandé, le programme soit facilement utilisable par l'équipe j'ai décidé d'orienter mon code objet. En effet, j'ai créé une **classe** 'message' qui facilite l'accès aux données et mon programme affecte à chaque message cette classe. L'utilisateur n'a plus qu'à choisir la donnée souhaitée. Cet appel est de la forme :

```
message.Location()
message.Time()
message.Type()
…
```

Les données gérées sont : l'heure d'enregistrement, le code d'erreur, le numéro de message, la région de vol, l'heure d'émission, la clé de vérification d'intégrité des données, la destination du message, le type de message, la source du message, l'identifiant de vol, le corps du message.

```
def Location(self):
     #print self.string
     if 'Record Location' in self.string :
          obj=re.search(r'(.*)n: (.*)\n',self.string)
          tmp = obj.group(2)
          return tmp
     return 'No Location found'
```

Exemple d'une fonction simple d'extraction de localisation

Le fichier de test sur lequel je me suis appuyé était constitué de sept jours de transmission entre Brisbane (YBBB FIR) et Melbourne (YMMM FIR), soit un fichier d'environ un million de lignes, ce qui laisse présager la quantité énorme d'informations à traiter ainsi que l'optimisation des différents codes de traitements. Le but étant de trouver un traitement efficace en terme d'économie de ressources, sans pour autant négliger les informations à extraire, j'estime (à titre personnel) que cette contrainte m'a permis d'améliorer sensiblement ma manière de coder pour la rendre plus performante et plus économique en ressource. Ainsi, compte tenu de la récurrence des analyses et du nombre de messages à traiter, l'optimisation de mon code permet de passer d'un temps de traitement d'une heure à un temps traitement de 60 secondes avec une unité de calcul assez puissante.

```
Record Time    : 2013.02.03-02:28:27
Record Location: 000106740085ECD0
Error Code     : 0 (No error found )

FBA1466 1302030226
FF YMMMZQZF
030226 YBBBZQZF 2.368352-4.130203022649-5.32D4-
(ABI-QLK567D/A1146
-YBBN-YAKKA/0327F240
-YSCB-8/IS-9/DH8D/M-10/SDFGHRZ/P-15/N0350F240 2744S15312E 2746S15310E
 SGT MONDO LOWEP YAKKA CAA SY V169 CB DCT-18/PBN/S1 NAV/GPSRNAV
 DOF/130203 REG/VHQOC PER/C)
```

Exemple d'un message de type ABI (Advance Boundary Information)

La majorité des informations à extraire est à classer dans un **dictionnaire** à double entrée et à clé unique afin d'avoir toutes informations utiles aux analyses statistiques pour le projet ACARS (optimisation générale de donnée régissant les vols). Pour réaliser les différents calculs de performance ainsi que pour établir une base de données des différents messages, j'ai choisi la solution du **dictionnaire**, celui-ci me permettant de placer chaque information obtenue de manière unique dans une 'base de données'.

Pour accéder aux données, il faut utiliser deux clés : la première appelée clé primaire doit être unique afin que chaque message ait son espace mémoire d'attribué ; la seconde doit permettre l'accès aux différentes valeurs. D'un point de vue appliqué aux messages AIDC, il m'a fallu trouver une clé primaire qui permet de procéder à une recherche.

J'ai donc choisi la localisation de l'appareil comme clé primaire, car celle-ci est unique et permet de retrouver un vol uniquement grâce à sa localisation.

L'accès aux données stockées dans le dictionnaire se fait de la manière suivante :

```
NOM_DU_DICTIONNAIRE[clé_primaire][clé_secondaire]
```

---	TYPE	DESTINATION	SOURCES	…	NUM.VOL	TPS.RECEPTION	TPS.ENVOI
LOCALISATION	ABI	YMMM	YBBB		VHQOC	2013.02.03-02:28:27	130203022649
LOCALISATION							
…							
LOCALISATION							
LOCALISATION							

Représentation du dictionnaire

Le travail demandé dans ce projet n'était pas qu'un simple classement d'informations, il a aussi fallu procéder à des analyses (via une programmation universelle) permettant de mettre en évidence les messages ayant un temps de transfert très élevé ainsi que leur vol associé. Pour cela, le calcul basé uniquement sur des temps en seconde a dû comparer chaque temps de transmission des messages de chaque vol afin de déterminer leur efficacité. Le programme renvoie sur la sortie les messages concernant le vol présentant la plus grande durée de transmission toujours à des fins de statistiques. Le code est d'environ 500 lignes.

```
ABI --> 13742
ACP --> 8148
AOC --> 9072
CDN --> 259
CPL --> 286
EST --> 7753
FPL --> 3
LAM --> 48751
LRM --> 150
MAC --> 15
PAC --> 60
THI --> 237
TOC --> 9506

DateTime                Callsign    MsgType    Sources    Delta_time (sec)
2013-02-04 17:29:25------MDG010------ABI---------FIMP----------3025
2013-02-04 17:29:35------MDG010------EST---------FIMP----------2735
2013-02-04 17:35:35------MDG010------TOC---------FIMP----------95
2013-02-04 17:35:45------MDG010------AOC---------YMMM----------96
2013-02-06 15:42:49------MDG010------ABI---------FIMP----------109
2013-02-06 15:47:39------MDG010------EST---------FIMP----------99
2013-02-06 16:37:41------MDG010------TOC---------FIMP----------101
2013-02-06 16:38:01------MDG010------AOC---------YMMM----------99
Number of message: 97982
Nb of message error: 201
Max delta_time: 3025 on 2013-02-04 17:29:25
Min delta_time: 95 on 2013-02-04 17:35:35
```

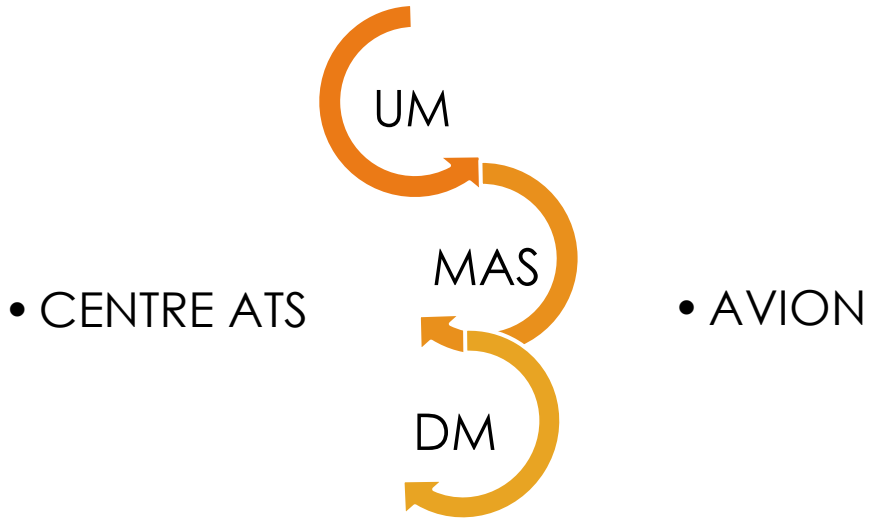
Sortie du script Python de traitement AIDC

La sortie affiche le nombre de messages différents avec leurs types, ainsi que tous les messages du vol ayant le temps de réponse le plus mauvais. La sortie n'affiche pas les accusés de réception (message LAM).

b. Logiciel d'affichage et de statistiques des messages CPDLC

La seconde mission qui m'a été confiée pousse encore plus loin l'utilisation des données reçues ainsi que l'utilisation de Python afin de réaliser un logiciel permettant de visualiser rapidement les différents messages sous forme graphique et indiquant les informations statistiques basiques et utiles des durées de transmission des messages.

Le message **CPDLC** (Controller Pilot Data Link Communication) est l'élément de base du système de messagerie entre les pilotes et les contrôleurs. Ce genre de message est très utile dans des zones difficilement couvertes par les systèmes de communication standard. Ce système utilise la haute fréquence (VHF). Les messages **CPDLC** sont codés pour des raisons de sécurité et de taille. Le protocole de ce type de message est lui aussi très strict et suit un système d'accusé de réception pour s'assurer que le message a été bien transmis.



Protocole de dialogue CPDLC (message Uplink)

Dans le protocole de message **CPDLC**, chaque message provenant du sol possède le code UM (Upload Message) et les messages en provenance de l'avion possèdent quant à eux le code DM (Download Message). Lorsque le centre ATS envoie un ordre à l'avion (par exemple changement d'altitude UM#20 FL380 : passage à 38000 pieds) celui-ci envoie l'UM, dès que l'avion reçoit ce message un accusé de réception automatique est renvoyé au centre ATS, c'est le message de code MAS. Ensuite le message est affiché dans une console dédiée (voir figure suivante), le pilote doit alors répondre WILCO (langage aéronautique signifiant 'Se Conformera'), pour cela il renvoie un message de code DM.

Console d'affichage des messages de type CPDLC dans le cockpit

J'ai travaillé sur les données '**GOLD** appD output' qui regroupent la globalité des données de communication des centres de contrôle aériens entre eux et vers les avions. Sur le schéma cette partie est dénommée F.

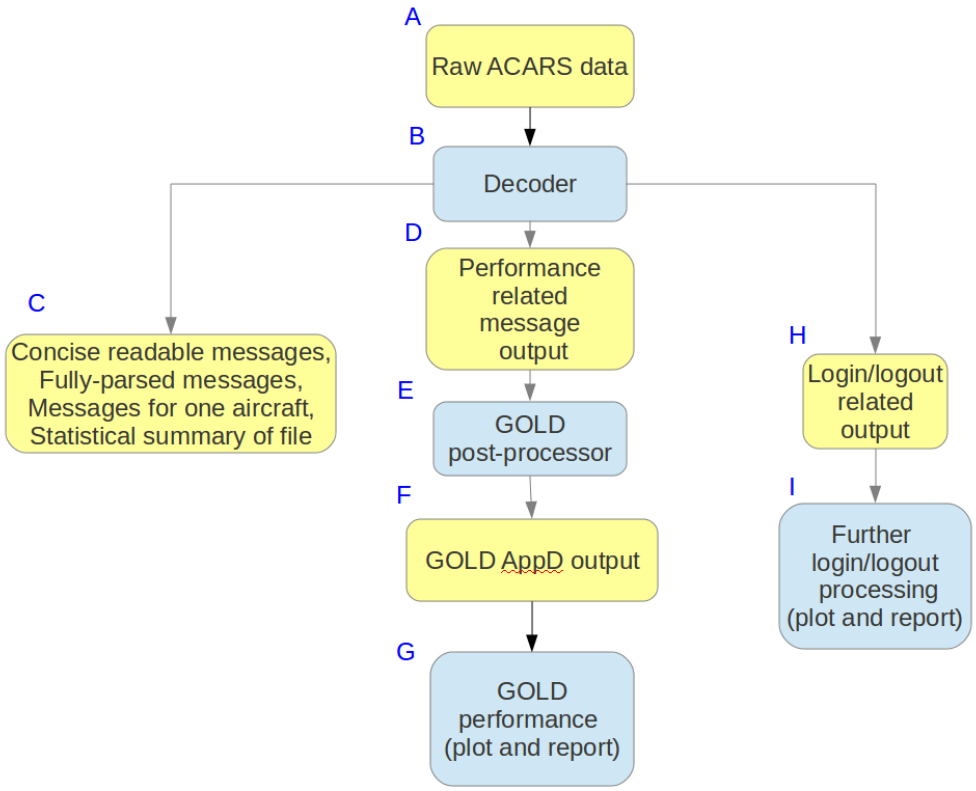


Schéma de l'analyse de données (Bleu = programme, Jaune = données)
(Source : ACARS message processing in safety and assurance
Geoffrey Aldis, Airservices Australia)

Les fichiers que j'ai utilisés dans mon logiciel de traitement ont été traités et sont classés dans un fichier **csv**. Ils permettent d'avoir des fichiers uniformes pour un traitement simplifié. Ainsi chaque phase du traitement est différenciée, ce qui permet des modifications plus simples. Le fichier **csv** à traiter comporte uniquement les messages de type **CPDLC**. Il est constitué de lignes de la forme suivante :

YBBB,55140A,UNK,UNK,20110808,,XXC,01:22:46.863,01:22:59.442,13,01:24:44,01:24:51.587,125,7,**U80 U167**,D0,13.5,**125**,111.5,UNK,I3

Sur cette ligne d'exemple, nous pouvons visualiser les données à traiter. En rouge figurent les codes de message associés à un temps de réponse de l'accusé de réception en bleu. Le but de cette analyse est de déterminer les messages optimaux ainsi que d'établir des statistiques sur chaque message pour déterminer une solution de gestion optimale de l'espace aérien prenant en compte les délais de transmission et d'application.

```
YBBB,33124A,,,20110804,,XXC,00:46:35.230,00:47:05.844,30,00:47:25,00:47:58.693,83,33,U117,D0,48,83,35,UNK,I3
YBBB,37980S,,,20110829,,XXC,07:16:59.291,07:17:08.921,9,07:17:18,07:17:30.776,31,12,U123,D0,16.5,31,14.5,UNK,I3
YBBB,37980S,,,20110829,,XXC,07:25:23.771,07:25:33.775,10,07:25:49,07:26:01.708,38,12,U117,D0,17,38,21,UNK,I3
YBBB,55140A,UNK,UNK,20110808,,XXC,01:18:05.539,01:18:15.845,10,01:18:31,01:18:39.680,34,8,U20,D0,13,34,21,UNK,I3
YBBB,55140A,UNK,UNK,20110808,,XXC,01:22:46.863,01:22:59.442,13,01:24:44,01:24:51.587,125,7,U80 U167,D0,13.5,125,111.5,UNK,I3
YBBB,55140A,UNK,UNK,20110808,,XXC,02:10:26.609,02:10:43.973,17,02:11:54,02:12:01.608,95,7,U117,D0,15.5,95,79.5,UNK,I3
YBBB,55140A,UNK,UNK,20110818,,XXC,11:58:10.894,11:58:27.903,17,11:58:41,11:58:52.403,42,11,U20,D0,19.5,42,22.5,UNK,I3
YBBB,55140A,UNK,UNK,20110818,,XXC,12:18:20.51,12:18:30.621,10,12:18:45,12:18:55.734,35,10,U121,D0,15,35,20,UNK,I3
YBBB,55142A,C17,UNK,20110821,,XXC,22:36:52.159,22:37:01.825,9,22:37:24,22:37:31.843,39,7,U123,D0,11.5,39,27.5,UNK,I3
YBBB,55142A,C17,UNK,20110821,,XXC,22:42:03.702,22:42:16.638,13,22:44:07,22:44:23.603,140,16,U79 U167,D0,22.5,140,117.5,UNK,I3
```

Extrait d'un fichier csv CPDLC à traiter

Le logiciel basé sur Python que j'ai dû réaliser devait être multi plateforme. Ce fut une des premières contraintes. Bien que Python soit un langage multi plateforme, le code doit être adapté. Certains éléments (comme les ouvertures de fenêtres de dialogue) ne fonctionnent pas de la même manière. Il m'a donc fallu réaliser un code gérant chaque caractéristique du système sur lequel il est porté.

Dans un premier temps, pour pouvoir tracer les fréquences d'apparition des messages et procéder aux études statistiques il m'a fallu extraire les données du fichier. Le fichier étant au format **csv** j'utilise une bibliothèque Python nommée **PANDAS,** celle-ci étant spécialisée dans le traitement des fichiers **csv**. Certaines lignes de données contenant plusieurs messages, car ayant les mêmes caractéristiques, et la bibliothèque ne faisant pas la distinction, j'ai dû traiter ce cas grâce à la recherche d'expressions régulières dans les données recueillies par **PANDAS** afin de séparer les différents messages tout en gardant leurs propriétés.

Pour avoir une utilisation optimale des données, chaque message est placé dans un dictionnaire à deux dimensions comprenant comme clé primaire le temps de réponse du message et comme clé secondaire le code du message. Si un message se répète plusieurs fois alors sa valeur correspondante dans le dictionnaire est incrémentée. Voici une représentation graphique du dictionnaire obtenu :

---	U1	U2	U3	…	U117	U118	U119
5S	2	1	0		136	25	112
10S							
…							
125S							
131S							

Représentation du dictionnaire contenant les messages incrémentés

Un **dictionnaire** de statistique a aussi été mis en place, pour n'effectuer les calculs qu'une seule fois et pour pouvoir ensuite les réutiliser à souhait. C'est aussi un tableau à double dimension avec comme clé primaire le code du message et comme clé secondaire les différentes statistiques souhaitées. Une des caractéristiques très importante est le temps de réponse à 95%. C'est-à-dire le temps pour que 95% des messages aient été reçus. Cette information est très utile, car elle permet de voir l'efficacité du message : le but de chacune des statistiques calculées étant d'avoir des informations sur la qualité des messages transmis, ainsi que d'en tirer des comparaisons entre eux.

---	MEDIANE	MOYENNE	VARIANCE	…	MIN	TEMPS 95%	ECART TYPE
U0	27.0	33.955	434.952		12	73	20.856
U2							
…							
U118							
U119							

Visualisation des entrées du dictionnaire de statistiques

Pour éviter une redondance d'appels de fonctions entraînant une dégradation des performances, j'ai mis en place des listes avec les informations principales permettant une recherche rapide et simplifiée sans à avoir à parcourir l'ensemble des **dictionnaires**, par exemple une liste contenant uniquement les noms des messages présents, mais sans redondance de celui-ci.

```
def create_statistic_dico(dico): #dico is the dictonary of every DATA
	global len_x_list

	for x in msg_name_LISTE:
		#tmp_LISTE_sum = [] #give the sum of length
		tmp_LISTE = []
		coeff_LISTE = [] #give the sum of length
		for k, v in sorted(dico.items()):

			for k2, v2 in sorted(dico[k].items()):
				if x == k2 :
					for i in xrange(v2):
						tmp_LISTE.append(k)

					coeff_LISTE.append(v2)
		#print dico
		#print sum(tmp_LISTE_sum)
		#print tmp_LISTE
		statistic_DATA[x]['mediane'] = mediane(tmp_LISTE)
		statistic_DATA[x]['moyenne'] = moyenne(tmp_LISTE)
		statistic_DATA[x]['variance'] = np.var(tmp_LISTE)
		statistic_DATA[x]['max'] = max_in_liste(tmp_LISTE)
		statistic_DATA[x]['min'] = min_in_liste(tmp_LISTE)
		statistic_DATA[x]['std_dev'] = np.std(tmp_LISTE)
		statistic_DATA[x]['nb_message'] = sum(coeff_LISTE)
		statistic_DATA[x]['temps_reponse'] = temp_rep(tmp_LISTE, coeff_LISTE)
```

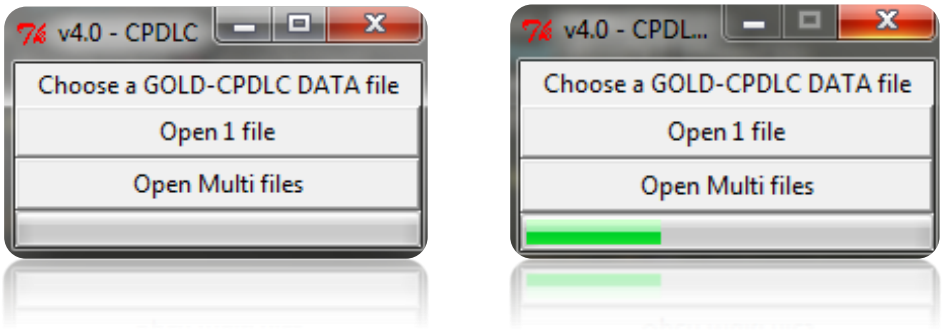
Exemple de la fonction de création du dictionnaire statistique

Pour la réalisation du logiciel de traitement, j'ai opté pour une interface graphique au lieu d'une interface texte classique. Celle-ci permet d'avoir une meilleure ergonomie ainsi qu'une expérience utilisateur accrue. Une fois la charte graphique conçue, il a fallu trouver la solution technique la plus adaptée. Le système devant rester simple, j'ai choisi d'utiliser le système **Tkinter** car intégré par défaut dans chaque installation de Python quel que soit le système hôte (Windows, Linux, Mac OS). **Tkinter** étant multi plateforme, le code nécessite néanmoins quelques optimisations en fonction du système. En effet, chaque système possède ses propres procédures. Prenons l'exemple du traitement des fenêtres (la base d'une interface graphique), les fenêtres Linux sont automatiquement rafraîchies tandis que les fenêtres Windows ne le sont qu'à la fin de chaque action engagée (par exemple à la fin des boucles). La solution dans cet exemple a donc été d'insérer un rafraîchissement obligatoire lors de chaque itération avec le code :

```
root.update_idletasks()
```

root étant l'instance de notre fenêtre définie précédemment

Le logiciel a la possibilité de traiter les fichiers un par un ou d'en assembler afin de constituer une unique base de données, ce qui permet de constituer une analyse sur des durées plus importantes et donc d'améliorer la précision.



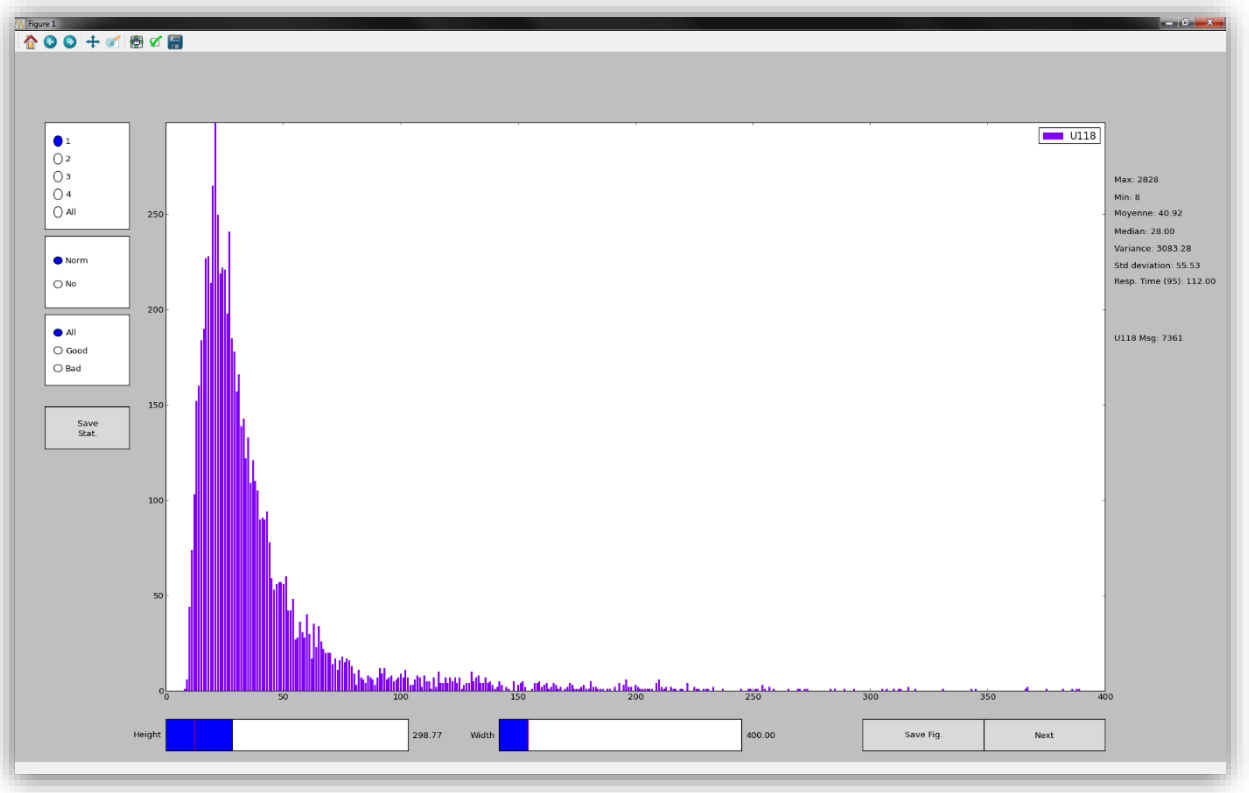
Fenêtre d'accueil (à gauche) et fenêtre en cours de traitement (à droite)

L'affichage graphique fait appel au **dictionnaire** contenant le nombre de messages semblables. Pour l'affichage sous forme de graphe à barres (communément appelé : 'bar graph'), la librairie **matplotlib** est utilisée. Elle permet de tracer toutes formes de courbes. Dans le tracé des courbes, j'ai utilisé des listes pour constituer l'axe des abscisses ainsi que l'axe des ordonnées. L'axe des abscisses représente la durée en milliseconde de réception d'un message. L'axe des ordonnées représente quant à lui le nombre de messages d'un type donné. Le logiciel prend en charge l'affichage de 1-2-3-4 types de message simultanément, ainsi que l'affichage de tous les messages. Le changement de message se faisant en cliquant sur le bouton 'Next'. Les messages sont classés par nom.

Voici les fonctionnalités du logiciel :

- Affichage des messages seuls, ou groupés, ce qui permet une meilleure comparaison des messages entre eux.
- La possibilité de normaliser l'échelle des abscisses, ce qui permet de comparer les messages entre eux. Si la fonctionnalité est désactivée, l'axe des abscisses prend comme maximum le temps de réponse maximum du message étudié. Cette option permet une étude approfondie du message avec une visualisation adaptée.
- Une fonction du programme permet d'afficher uniquement le meilleur message, c'est-à-dire le message qui a le temps de réponse à 95% le plus rapide. Réciproquement une fonction permet d'afficher le message qui possède le temps de réponse le plus long. Généralement cela met en évidence un message d'erreur qui ne se répète pas souvent.
- Curseurs qui permettent d'adapter l'échelle du temps ou de la quantité manuellement pour mettre en évidence ou visualiser l'élément souhaité.
- Possibilité de sauvegarder au format PNG la figure en cours. Cela permet aux utilisateurs d'insérer la visualisation graphique dans des rapports.

- La possibilité de sauvegarder dans un même fichier la totalité des statistiques concernant chaque message. Le fichier de sauvegarde est formaté au format csv, pour un traitement automatique simplifié dans le futur.
- Enregistrement dans un fichier, du détail de chaque message fractionnant l'échelle du temps en 50s, 100s, 150s, 200s, 300s, 2000s, ainsi que le pourcentage lié au nombre de messages reçus dans ces durées.
- Affichage pour chaque message en cours de visualisation des statistiques s'y rapportant.



Vue globale de la fenêtre d'affichage (message U118 affiché)

```
CODE,MEDIANE,MOYENNE,VARIANCE,MAX,MIN,STD-DEV,TPS-REP
U0,27.0,33.9545454545,434.952479339,96,12,20.8555143628,73
U106,20.0,26.6014851485,1138.6010869,506,9,33.743163558,59
U107,20,20.0,0.0,20,20,0.0,20
U108,20,28.4401244168,1250.93070729,505,9,35.368498799,64
U109,21.0,33.0802238806,3349.46557905,1140,10,57.874567636,91
U110,19,19.2,9.36,24,16,3.05941170816,24
U111,10,10.0,0.0,10,10,0.0,10
U114,55,55.0,0.0,55,55,0.0,55
U116,21,28.515681984,2195.79315305,1330,9,46.8592910003,65
U117,23.0,31.410875875,1937.04280813,3023,7,44.0118484971,78
U118,27,40.8105216532,2747.83867631,2828,7,52.419830945,122
U119,27,35.3008130081,1100.09975544,359,10,33.1677517393,77
U120,31.0,40.9194086791,1504.63060566,974,9,38.7895682582,94
U121,28.0,37.5640384231,1024.06158849,586,11,32.0009623057,81
U122,28,33.6956521739,332.559546314,102,17,18.2362152409,57
```

Exemple d'impression des statistiques dans le fichier de sortie statistique

Revenons sur les sorties du logiciel. Celles-ci sont sous la forme de fichiers textes au format csv pour permettre un traitement futur et une réutilisation des données dans différents logiciels sans à avoir à les reformater. Les fichiers de sortie sont nommés : 'STATISTIC' et 'STATISTIC_DETAILS_TIME'. Le premier contient les statistiques pures telles que la médiane, la moyenne, l'écart type (etc.) de chaque message. Le second contient le détail de chaque message suivant un fractionnement dans le temps. Le logiciel génère aussi deux fichiers HTML contenant les mêmes informations, ceci pour donner un aperçu rapide des résultats facilement lisible par l'utilisateur. Les tableaux figurants dans ces fichiers sont triés, le premier est trié en fonction du temps maximal mis par un de ces messages, le second est trié en fonction du nombre de messages contenus dans les différentes intervalles de temps (voir annexe 4-1). Cela permet en un coup d'œil de visualiser les messages non optimisés. Par exemple dans les messages traités dans l'annexe 4-1 nous pouvons voir que le message possédant le code U127 est le plus mauvais car il possède un étalement dans le temps important.

Table of Statistics

CODE	Medianne	Moyenne	Variance	Max	Min	Standard Deviation	Response Time
U111	10.0	10.0	0.0	10	10	0.0	10.0
U107	20.0	20.0	0.0	20	20	0.0	20.0
U36	23.0	23.0	0.0	23	23	0.0	23.0
U110	19.0	19.2	9.36	24	16	3.059	24.0
U42	24.0	20.6	21.44	25	14	4.63	25.0
U161	22.5	22.75	9.188	27	19	3.031	27.0
U32	26.0	25.833	11.806	32	21	3.436	32.0
U6	31.5	31.5	2.25	33	30	1.5	33.0

Vu du fichier HTML des statistiques

d) Intégration à l'entreprise et continuité du stage

Tout au long de mon stage, j'ai assisté à des réunions en tant qu'observateur pour pouvoir mieux intégrer les sujets abordés par l'équipe et développer ainsi les points qui m'étaient demandés dans une parfaite harmonie. Tout au long de ma mission, mon tuteur Andrew JASON-JONES, ainsi que le responsable de l'équipe de modélisation quantitative, Geoff Aldis, et moi-même avons participé à des 'Check-Up' pour suivre le bon déroulement des tâches. A chaque étape, j'ai dû exposer le travail réalisé ainsi que le travail projeté. Chacune de ses réunions m'a permis de garder une ligne directrice correcte vis-à-vis de ce qui était demandé pour fournir un travail pertinent. Lors de ces réunions, différents documents mon été transmis afin de comprendre la chaîne complète de communication.

Il m'a aussi été présenté différents ingénieurs de l'entreprise pour appréhender le travail de chacun et ainsi me rendre compte - en tant qu'assistant ingénieur - des particularités de ce métier.

Pour la suite de mon stage, le logiciel étant presque terminé, les deux semaines restantes vont être utilisées pour débuguer, ainsi qu'optimiser l'interface graphique : par exemple, permettre l'adaptation du logiciel aux différents types d'écrans ainsi qu'aux différentes résolutions. De même ces deux semaines restantes vont être utilisées pour réaliser des tests complets, permettant d'éprouver l'application, ainsi que de rajouter certaines fonctionnalités au besoin.

Conclusion

Durant deux mois j'ai été confronté à la culture de travail Anglo-Saxonne qui m'est apparue assez différente de la nôtre, mais surtout mon stage m'a permis d'en apprendre beaucoup plus sur les moyens de communication et sur le travail d'ingénierie dans le domaine des transports aériens.

Les bénéfices personnels que je tire de ce stage ont trait à ma façon de coder et de collaborer avec les ingénieurs. Compte tenu des quantités de données que j'ai eu à traiter, je pense être devenu beaucoup plus performant sur l'écriture des codes en économisant au mieux les ressources. Ceci me sera probablement utile dans l'élaboration de systèmes autonomes qui eux sont très limités en ressources.

Au final, ma contribution aux recherches d'une équipe d'analyse opérationnelle, leader dans son domaine, m'a procuré une grande satisfaction.

Glossaire

Dictionnaire : Allocation mémoire, fonctionnant comme un tableau à une ou plusieurs entrées permettant le stockage de tout type de variables.

Classe : Permet de regrouper des fonctions pour un même type de données à traiter, ce qui permet de créer des raccourcis de fonctions sur ces données.

AIDC : Message transmis entre les différents centres de contrôle aériens, pour se transmettre la responsabilité d'un vol lors d'un changement de zone de contrôle.
Signification : ATS (Air Traffic Services) Interfacility Data Communications.

CPDLC : Message directement transmis du centre de contrôle aux différents avions afin de leur signifier des changements à faire intervenir, tel que l'altitude ou une déviation.
Signification : Controller Pilot Data Link Communication

GOLD : Ensemble des messages servant à faire transiter les informations des vols en cours.
Signification : Global Operational Data Link Document.

CSV : Fichier formaté avec des délimiteurs choisis. Chaque ligne possède des données délimitées par exemple par des virgules.
Signification : Comma-Separated Values

PANDAS : Librairie Python, consacrée aux traitements des fichiers du type CSV et facilitant la gestion des données contenues, pour un traitement plus simple.

TKINTER : Composant Python permettant la réalisation d'interfaces graphiques multi plateforme.

MATPLOTLIB : Bibliothèque Python permettant de réaliser des graphes plus ou moins sophistiqués. Outils mathématiques de base lors de l'utilisation de Python à des fins de création graphique.

Bibliographie

Rapport: ACARS Message Processing in Safety & Assurance, par Geoff Aldis, Airservices Australia, 2012

Rapport: Global Operational Data Link Document (GOLD), par International Civil Aviation Organization (ICAO), 2010

Document: Asia/Pacific Regional Interface Control Document, par International Civil Aviation Organization (ICAO), 2007

Document (C2): EUROCAT / AIDC Messages ICD, Thales Australia Ltd, 2012

Document: AIDC Performance and Activity Update for Brisbane, Informal South Pacific ATS Co-ordinating Group (ISPACG), 2013

Site Internet: www.airservicesaustralia.com

Site Internet: www.navcanada.ca

Site Internet: www.eurocontrol.int

Site Internet: www.sites.google.com/site/pydatalog

Cours Python: Simple Python, Steve Barry, Airservices Australia, 2013

Annexes

Annexe 1-1 : Liste des messages AIDC

CODE	SIGNIFICATION
ABI	Advance Boundary Information
ACP	Acceptance
ADS	Surveillance ADS-C
AOC	Airline Operational Control; or Assumption of Control
ASM	Application Status Monitor
CDN	Coordination
CPL	Current Flight Plan
EMG	Emergency
EST	Coordination Estimate
FAN	FANS Application Message
FCN	FANS Completion Notification
LAM	Logical Acknowledgement Message
LRM	Logical Rejection Message
MAC	Coordination Cancellation
MIS	Miscellaneous
PAC	Preactivation
REJ	Rejection
TDM	Track Definition Message
TOC	Transfer of Control
TRU	Track Update

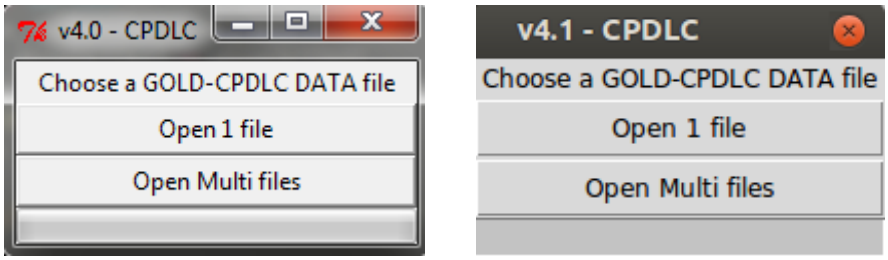
Annexe 2-1 : Trame d'un message du type CPDLC

Id Number	Ref Number	Time Stamp	Lack	Msg Elements
Assigné par le système émetteur et correspondant à une destination	Dois être identique à l'Id-Number fourni lors d'une réponse	Date et heure de l'envoi du message	Indique si un accusé de réception est requis	Message contenu dans la trame

Annexe 2-2 : Exemple de conversation utilisant les messages CPDLC

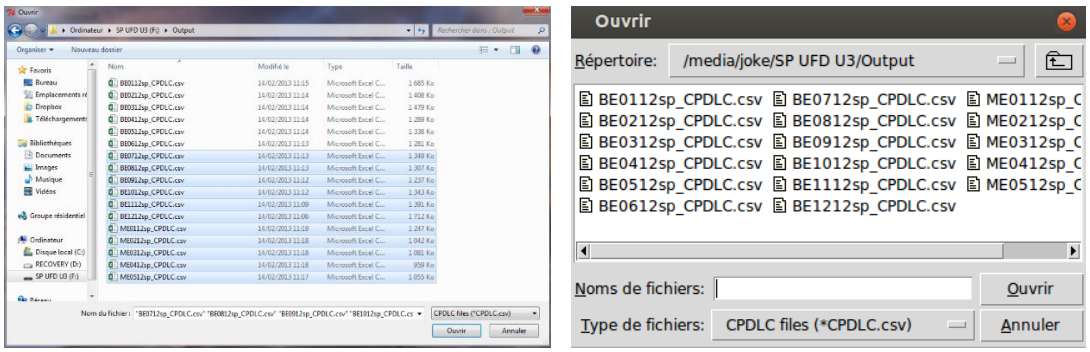
(Source: NavCanada, Customer Newsletter of spring 2013, French Edition)

Annexe 3-1 : Fenêtre d'accueil du logiciel sous Windows et Linux



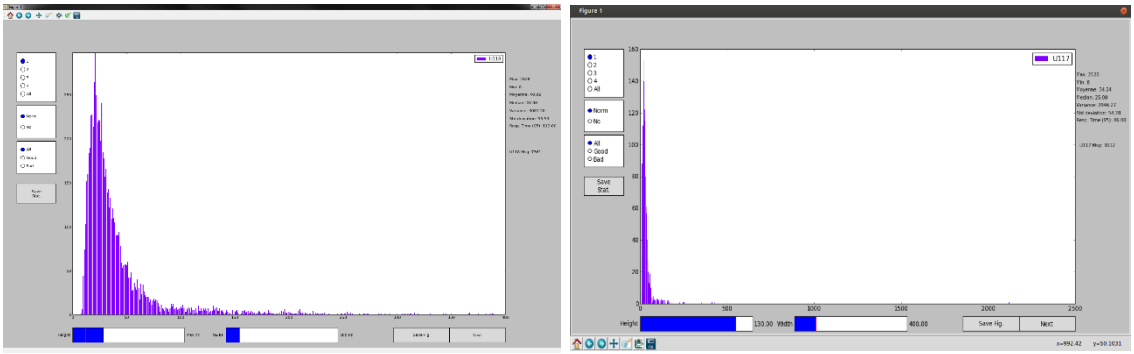
Les interfaces sont les mêmes, la portabilité du logiciel est donc réussie.

Annexe 3-2 : Fenêtre de choix de fichiers sous Windows et Linux



Les fonctionnalités sous Windows et Linux sont exactement les mêmes.

Annexe 3-3 : Fenêtre de visualisation des messages sous Windows et Linux



Annexe 4-1 : Sortie HTML des messages fractionnés dans le temps

Table of Times

CODE	>50ms	>50ms in %	>100ms	>100ms in %	>150ms	>150ms in %	>200ms	>200ms in %	>300ms	>300ms in %	>2000ms	>2000ms in %
U111	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U107	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U36	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U110	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U42	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U161	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U32	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U6	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U46	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U134	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U13	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U39	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U137	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U125	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U64	1	4.348	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U56	1	25.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U114	1	100.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U179	1	20.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U37	1	50.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U148	1	33.333	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U21	2	33.333	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U180	2	13.333	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0
U83	25	62.5	11	27.5	5	12.5	5	12.5	3	7.5	0	0.0
U108	44	6.843	20	3.11	8	1.244	4	0.622	2	0.311	0	0.0
U106	25	6.188	13	3.218	4	0.99	2	0.495	1	0.248	0	0.0
U75	165	9.026	57	3.118	19	1.039	11	0.602	4	0.219	0	0.0
U121	1496	14.969	327	3.272	163	1.631	104	1.041	19	0.19	0	0.0
U30	240	5.033	82	1.719	29	0.608	10	0.21	1	0.021	0	0.0
U120	970	23.128	175	4.173	69	1.645	39	0.93	13	0.31	0	0.0
U19	101	5.752	30	1.708	8	0.456	4	0.228	1	0.057	0	0.0
U109	59	11.007	20	3.731	11	2.052	4	0.746	2	0.373	0	0.0
U166	80	13.445	20	3.361	8	1.345	5	0.84	3	0.504	0	0.0
U27	103	7.607	55	4.062	26	1.92	16	1.182	9	0.665	0	0.0
U116	112	8.169	33	2.407	12	0.875	9	0.656	5	0.365	0	0.0
U123	2666	13.244	631	3.135	263	1.307	163	0.81	32	0.159	0	0.0
U77	494	21.385	114	4.935	46	1.991	22	0.952	5	0.216	0	0.0
U20	1391	6.108	447	1.963	194	0.852	106	0.465	20	0.088	0	0.0
U164	197	9.563	73	3.544	27	1.311	17	0.825	9	0.437	0	0.0
U169	4252	22.757	1113	5.957	459	2.457	226	1.21	85	0.455	1	0.005
U118	5193	17.694	2054	6.999	899	3.063	413	1.407	127	0.433	3	0.01
U74	870	9.263	257	2.736	101	1.075	46	0.49	21	0.224	1	0.011
U117	5699	9.73	1980	3.381	613	1.047	306	0.522	142	0.242	3	0.005
U163	1	100.0	1	100.0	1	100.0	1	100.0	1	100.0	1	100.0
U82	1412	6.744	448	2.14	221	1.056	124	0.592	43	0.205	3	0.014
U127	964	7.038	288	2.102	146	1.066	87	0.635	30	0.219	3	0.022

Résumé

Stage réalisé au sein d'Airservices Australia dans la branche d'analyse opérationnelle. Le but de ce stage était la mise en forme de messages de communication entre différents centres de gestion aériens au sol, ceci pour permettre un accès rapide aux données. De plus j'ai dû réaliser un logiciel de visualisation et de statistique des messages de type CPDLC, message utilisé par les appareils en vol pour récupérer des ordres ou transmettre des demandes aux différents services au sol lorsque le contact radio est impossible. Tous mes traitements ont été réalisés dans le langage Python, langage dont j'ai dû faire l'apprentissage durant mon stage.

Mots clés: Analyse – Logiciel – Python – Graphique - Aviation

Abstract

Training course carried out within Airservices Australia in the branch of operational analysis. The goal of this work experience was based of communication messages between air center on the ground. This to allow a rapid access with the data. Moreover I had to carry out a software of visualization and statistics of CPDLC messages, message used by the devices in flight to recover orders or to transmit requests to the various services on the ground when the radio contact is impossible. All my treatments were carried out in the language Python. Language of which I had to learn first during my training course.

Keywords : Analysis – Software – Python – Graphic - Aviation