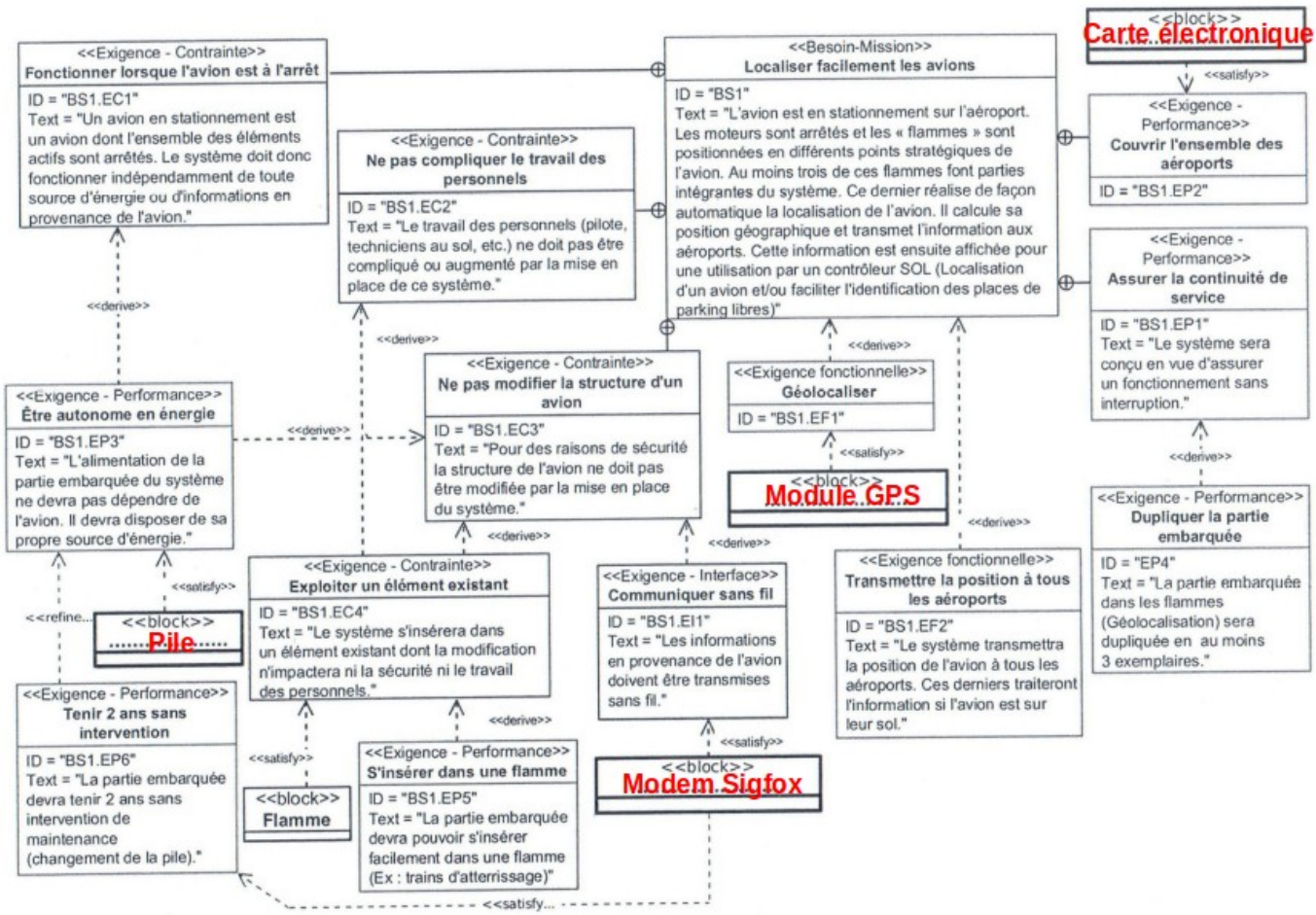
Étude d'un Système Numérique et d'Information
Option A (IR) - Session 2019
Correction tv (version 0.9)

Système de localisation d'avions au sol par flammes connectées

Partie A. Analyse des exigences du système

Q1. Diagramme



Q2. Contrainte : structure de l'avion non modifiable

Partie B. Optimisation de la flamme

Q3.

Trame : $GPGGA,235942.800,,,,,**0**,00,,,M,,M,,*7B

Champ Position Fix Indicator : **0 → Fix not available**
6 virgules

Q4.

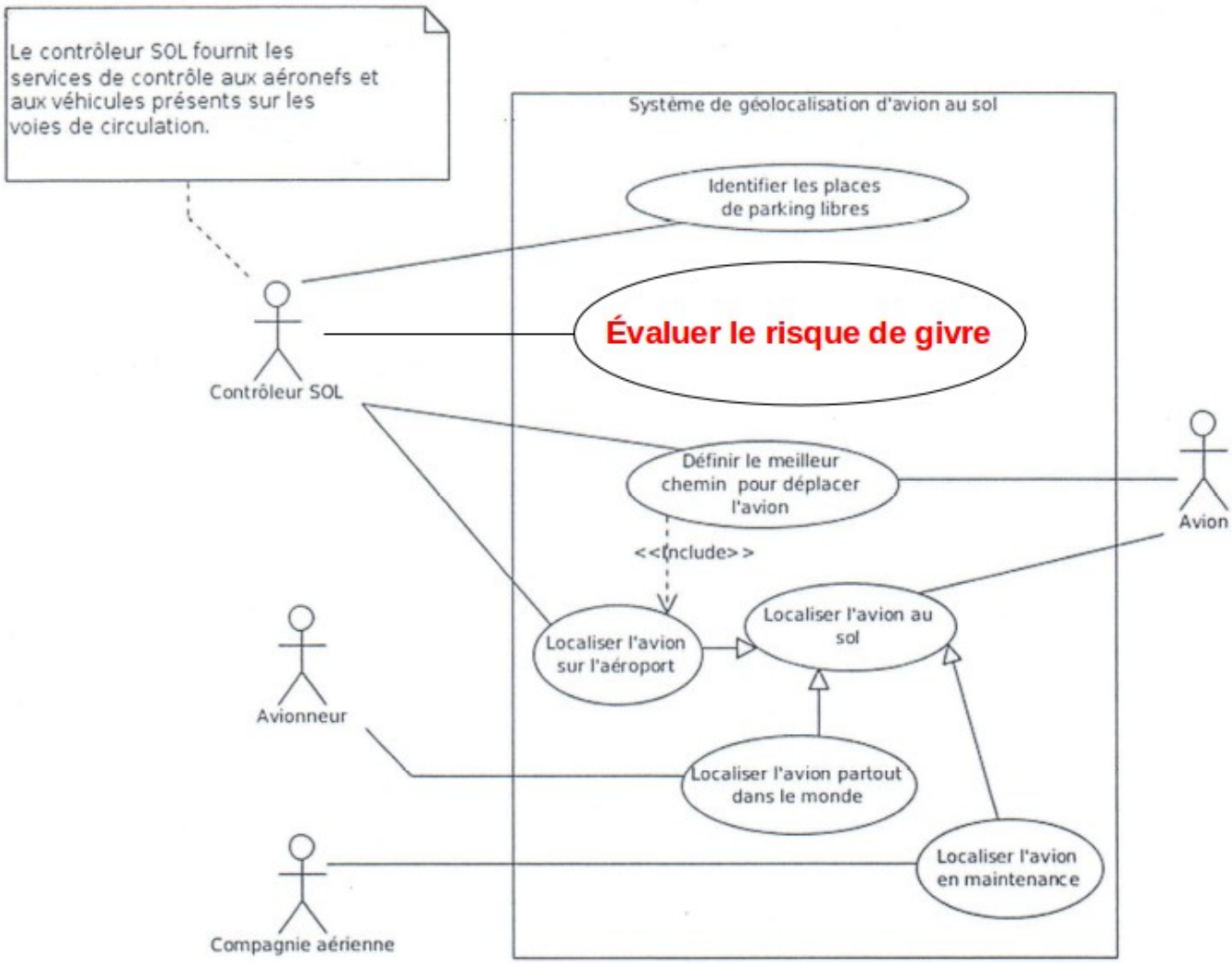
Trame : 38 caractères + <CR> + <LF> = 40 caractères
Format : 1 bit START + 8 bits DATA + 1 STOP = 10 bits
Total = 40 x 10 = 400 bits

Temps transmission = 400 / 9600 = 0,041666667 s → **41,67 ms**

Q5. Trame MTK : **$PMTK314,0,0,0,1,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0*XX<CR><LF>**

Partie C. Amélioration du système

Q6.



Remarque : certains mettraient une association avec l'avion ! Pour moi, l'avion ne devrait pas apparaître à la base sur ce diagramme (il ne tire aucun bénéfice observable de ce système et il est cité dans les cas d'utilisation !).

Q7.

```
class Capteur_T_HR
{
    private:
        I2C *liaisoni2c;
        unsigned char adresse;
        double temperature;
        double humiditeRelative;

    public:
        Capteur_T_HR(I2C *i2c);
        Capteur_T_HR(I2C *i2c, unsigned char adresse);
        double getTemperature();
        double getHumiditeRelative();
        bool acquerirDonnees();
};
```

Q8.

```
Capteur_T_HR(I2C *i2c) : liaisoni2c(i2c), adresse(0x28)
{
}

// ou :
Capteur_T_HR(I2C *i2c)
{
    liaisoni2c = i2c;
    adresse = 0x28;
}

Capteur_T_HR(I2C *i2c, unsigned char adresse) : liaisoni2c(i2c),
adresse(adresse)
{
}

// ou :
Capteur_T_HR(I2C *i2c, unsigned char adresse)
{
    liaisoni2c = i2c;
    this->adresse = adresse;
}
```

Q9. Décodage :

Latitude (en degrés) : 0025 → 37 et 2A2621 → 423833 soit 37,423833
Longitude (en degrés) : FFFB →-5 et 594A10 → 897416 soit -5,897416
Niveau de tension de la batterie (en V) : 0x1F → 31 dixièmes de volt soit 3,1 V

FFFB →

	1 1 1 1 1 1 1 1 1 1 1 1 0 1 1
complément	0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 (4)
+1	0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 (5)
Donc	-5

1	2	3	4	5	6	7	8	9	10	11	12
00	**25**	**2A**	**26**	**21**	**FF**	**FB**	**59**	**4A**	**10**	**1F**	**FF**

→ Terminal Aeropuerto de Sevilla, Calle Aeropuerto Viejo, 41019 Séville, Espagne !

Q10. L'informaticien n'a pas de règle ! TODO : Trouver une règle !

10,3 cm x 250 m = 2,575 km

Q11. À 5 Km près, l'erreur est acceptable

Q12. Sérieusement !

Q13. Partie entière Sigfox : 48° + partie décimale GPS ,000018 = 48,000018 ce qui donne un écart de 1° par rapport à la position réelle de 49° soit une centaine de kilomètres.

Q14.

```
double Position::recomposerCoordonnee(double coordonneeSigFox, double
coordonneeGPS)
{
    unsigned long partieDecSigFox;
    partieDecSigFox = extrairePartieDecimale(coordonneeSigFox);
    unsigned long partieDecSigGPS;
    partieDecSigGPS = extrairePartieDecimale(coordonneeGPS);
    long difference = partieDecSigFox - partieDecGPS;
    int partieEntiere = static_cast<int>(coordonneeSigFox);

    if(abs(difference) > (OFFSET/2))
    {
       if(((coordonneeSigFox > 0.) && (partieDecSigFox < (OFFSET/2))) ||
           ((coordonneeSigFox <= 0.) && (partieDecSigFox > (OFFSET/2))))
       {
           --partieEntiere;
       }
       else
       {
           ++partieEntiere;
       }
    }

    return partieEntiere;
}
```

Partie D. Évolution de la base de données du cloud Airbus

Q15. Clé primaire : **`id`** et Autre champ : **`sigfoxID`**

Q16. **`avionId` clé étrangère** de la table `FlammeConnectee` qui correspond à la **clé primaire `id`** de la table Avion

Q17. SQL :

```
SELECT id FROM FlammeConnectee WHERE sigfoxID = '1D188E';
```

Q18.

```
INSERT INTO DonneesFlamme (latitude, longitude, date, batterie, flammeid)
VALUES ('43.631310', '1.370395', '3.2', '2018/03/20 00:30:00', '2');
```

Q19. FlammeConnectee ???

```
ALTER TABLE DonneesFlamme
  ADD complet boolean default false, //ou bool, binary(1) ou tinyint(1) ???
  ADD temperature float,
  ADD humiditeRelative float
```

Partie E. Circulation de l'information depuis le cloud Airbus

Q20.

Adresse multicast : **`ff1e::e100:0025`**

Unicast	Multicast	Permanente	Temporaire	Portée limitée	Porté globale
	X **`ff00::/8`**		**X** **FLGS = 0001** **T=1**		**X** **SCOP=E**

Q21. Adresse multicast (Id de groupe de diffusion 32 bits) : **`ff1e::e100:0025`** → **`225.0.0.37`**

Q22.

Serveur « filtrage » → Serveur « gestion des emplacements »

	Trame sur DMZ	Trame sur « *backbone* »	Trame sur « services internes aéroport »
MAC Source	**74-D4-35-BE-95-75**	**D4-BE-D9-BE-96-E3**	**00-01-C9-AA-10-1D**
MAC Destination	**D4-BE-D9-BE-96-E2**	**00-01-C9-AA-10-1F**	**74-D4-35-BE-97-8C**
IP Source	**10.31.48.20**	**10.31.48.20**	**10.31.48.20**
IP Destination	**10.31.16.50**	**10.31.16.50**	**10.31.16.50**

Q23. Même chemin : FORWARD eth2 → eth1

`ip6tables -A FORWARD -i eth2 -o eth1 -d ff10::/12 -j ACCEPT`

Q24. eth2 (INTERNET)→ eth1 (DMZ)

Q25. 10.31.48.0/20

30 compagnies → **5 bits** (2^5 = 32 sous-réseaux)
Masque sous-réseaux = 20 + 5 = **/25** (**255.255.255.128**)
host-id = 32 - 25 = 7 bits (2^7 – 2 = 126 hôtes max)

	Adresse sous-réseau	Masque CIDR	Broadcast	Plage	Nb max hôtes
Premier	**10.31.48.0**	**/25**	**10.31.48.127**	**10.31.48.1 à 10.31.48.126**	**2^7 – 2 = 126**
Deuxième	**10.31.48.128**	**/25**	**10.31.48.255**	**10.31.48.129 à 10.31.48.254**	**2^7 – 2 = 126**

Le troisième en 10.31.49.0, puis 10.31.49.128, …

Q26.

	Valeur
Adresse IP source	**10.31.16.5**
Adresse IP destination	**10.31.16.101**
Date	**2018/03/04 03:45:00**
Latitude	**43.631201**
Longitude	**1.371171**

Q27. Src Port : 8080 → Serveur Web

Partie F. Analyse de l'application de filtrage sur le serveur de l'aéroport

Q28. Classe **`RecepteurMcast`** et Méthode **`traitementSurReception()`**

Remarque : il aurait été bien d'indiquer sur le diagramme les classes actives et leurs stéréotypes (<<thread>>).

Q29.

36 : `pthread_mutex_lock(&semMessage) ;` // **tente de verrouiller le mutex afin d'accéder à la ressource critique (buffer)**

40 : `pthread_mutex_unlock(&semMessage) ;` // **déverrouille le mutex afin de libérer l'accès à la ressource critique (buffer)**