

Méthodes de développement

Guide pour la rédaction d'un document d'architecture réduit

Sommaire

Avertissement.....	2
1 - Objet du guide.....	3
2 - Plan type.....	3
3 - Rédaction du document.....	3
3.1 Introduction	3
3.1.1 Objet	3
3.1.2 Terminologie et sigles utilisés	3
3.1.3 Documents de référence	4
3.2 Présentation générale de l'architecture	4
3.3 Les principaux objets et leurs classes.....	4
3.3.1 Liste des principales classes	4
3.3.2 Relations entre classes.....	4
3.3.3 Rôles et interfaces des principales classes	4
3.4 Scénarios d'interaction entre objets	5
3.5 Les processus et le déploiement	5
3.5.1 Déploiement, programmes, processus et threads	5
3.5.2 Principes de fonctionnement des processus	5
3.5.3 Interactions entre processus ou threads	5
3.6. Interfaces externes du système	5
3.7 Découpage de réalisation (sous-systèmes)	6
3.7.1 Liste des sous-systèmes et leurs relations	6
3.7.2 Fonctions et interfaces externes des sous-systèmes	6
3.8 Dimensionnements et performances	6
4 - Exemple de document d'architecture logicielle	7
Exemple de document d'architecture.....	8

Avertissement

Ce guide est destiné à décrire une architecture de type objet.

Il présente un document réduit adapté à un petit système qui ne présente pas de caractère de sûreté ou de criticité.

Dans le cas contraire un dossier plus étoffé serait probablement demandé.

En particulier pourraient être à rajouter :

- les objectifs de l'architecture et les contraintes,
- le rappel des cas d'utilisation avec les séquences d'événements significatives,
- des diagrammes montrant la réalisation des cas d'utilisation,
- des objectifs de qualité,
- une matrice de traçabilité entre les exigences de la STB et les sous-systèmes et classes.

Pour de gros systèmes, une décomposition sur plusieurs niveaux peut être nécessaire.

Enfin on pourrait y inclure des éléments justificatifs (par exemple en intégrant une partie des remarques préliminaires indiquées plus loin), à moins que la méthode de développement ait prévu un dossier justificatif d'architecture pour expliciter les choix d'architecture.

Dans le cas où l'on aurait à décrire une architecture de type non objet, on pourra s'inspirer du présent guide, mais en l'adaptant au type d'architecture.

1 - Objet du guide

Ce guide fournit un plan type de document d'architecture réduit adapté à de petits systèmes qui ne présentent pas de caractères de sûreté ou de criticité.

Un exemple de document partiellement rédigé basé sur l'exemple présenté dans l'introduction aux guides de rédaction est joint.

2 - Plan type

Le plan type suivant est recommandé pour un document réduit dans le cadre d'une conception objet :

1	Introduction 1.1 Objet 1.2 Terminologie et sigles utilisés 1.3 Documents de référence
2	Présentation générale de l'architecture
3	Les principaux objets et leurs classes 3.1 Liste des principales classes 3.2 Relations entre classes 3.3 Rôles et interfaces des principales classes
4	Scénarios d'interaction entre objets
5	Les processus et le déploiement 5.1 Déploiement, programmes, processus et threads 5.2 Principes de fonctionnement des processus 5.3 Interactions entre processus ou threads
6	Interfaces externes du système
7	Découpage de réalisation (sous-systèmes) 7.1 Liste des sous-systèmes et leurs relations 7.2 Fonctions et interfaces externes des sous-systèmes
8	Dimensionnements et performances

3 - Rédaction du document

3.1 Introduction

3.1.1 Objet

Préciser l'objet du document (description de l'architecture), le nom du système et brièvement sa fonction.

3.1.2 Terminologie et sigles utilisés

Définir les abréviations employées et les termes qui ont un sens particulier.

3.1.3 Documents de référence

Donner les références des documents ayant servi de référence pour l'architecture (la STB principalement) ou pouvant aider à comprendre le problème posé.

3.2 *Présentation générale de l'architecture*

Présenter rapidement l'organisation du logiciel en indiquant les principaux éléments de l'architecture et les principaux choix faits, par exemple :

- découpage en programmes, répartition du logiciel entre les calculateurs,
- processus, et threads éventuels,
- architecture réseau et accès réseau,
- accès aux bases de données (par serveur base de données, base réseau...),
- conception objet,
- mode de programmation (événementiel ou séquentiel...),
- langages et systèmes d'exploitation,
- principaux logiciels réutilisés...

3.3 *Les principaux objets et leurs classes*

3.3.1 Liste des principales classes

Les principales classes pourront être présentées par catégorie sous forme de tableaux indiquant :

- le nom de la classe,
- l'indication du rôle principal de la classe.

3.3.2 Relations entre classes

Les relations entre les principales classes identifiées ci-dessus seront représentées sur des diagrammes de classes montrant les classes et les diverses relations entre classes (généralisation, association...).

Pour que les diagrammes soient lisibles, il faudra répartir les classes sur plusieurs diagrammes. On pourra par exemple répartir les principales classes en packages de classes, et faire pour chaque package de classes un diagramme montrant les classes du package, et les classes des autres packages en relation avec ces classes.

3.3.3 Rôles et interfaces des principales classes

Pour chaque catégorie ou package de classes, un paragraphe indiquera pour chacune des principales classes :

- le rôle et les fonctions assurées,
- les principales données et les principales méthodes,
- le détail de l'interface public.

Le détail de l'interface public pourra être présenté sous forme de tableau, ou en reproduisant le code commenté (C++, Java, C#...) de cet interface.

3.4 Scénarios d'interaction entre objets

Les scénarios d'interaction seront définis à partir des cas d'utilisation. On choisira les séquences d'événements les plus significatives et on construira un diagramme d'interaction entre objets (diagramme de collaboration ou diagramme de séquence de messages) pour chaque séquence retenue.

3.5 Les processus et le déploiement

3.5.1 Déploiement, programmes, processus et threads

Présenter le découpage du logiciel en programmes et les processus attachés aux programmes. Pour les processus multi-threads, indiquez le rôle de chaque thread. Les présentations pourront être faites sous forme de schémas ou de tableaux.

Quelques explications sur le choix des découpages pourront être fournies.

Le déploiement sera représenté sous forme d'un schéma montrant les principaux matériels, les réseaux et les interfaces externes, avec indication des processus exécutés sur chaque calculateur.

3.5.2 Principes de fonctionnement des processus

Présenter schématiquement l'enchaînement des traitements dans chaque processus à partir de son lancement par le système d'exploitation ou de l'arrivée des principaux événements. On pourra par exemple le montrer par des diagrammes montrant l'enchaînement des appels entre objets.

3.5.3 Interactions entre processus ou threads

Les interactions entre processus seront présentées sous forme de diagrammes de séquence de messages, chaque processus étant représenté comme un objet. Ces diagrammes devront présenter les séquences les plus significatives, en particulier la séquence d'initialisation du système.

Dans le cas de processus multi-threads, les interactions entre le processus et les threads seront aussi présentées sous forme de diagrammes de séquence de messages.

3.6. Interfaces externes du système

Les interfaces externes du système varient avec le système. On aura en général un interface opérateur, un interface base de données, des interfaces fichiers, des interfaces de communication, et parfois des interfaces avec des matériels spécifiques, des interfaces avec des outils logiciels...

On fera un paragraphe par type d'interface pour préciser des formats de données, des protocoles d'échanges...

Pour l'interface opérateur, il n'est pas nécessaire de le définir de façon complète, mais on présentera les choix de principe et des maquettes des écrans les plus significatifs.

Lorsque le programme utilise des bases de données, on précisera :

- le contenu des bases de données,
- l'organisation des données (par exemple un schéma conceptuel de données),
- la définition détaillée des tables.

Lorsque le programme utilise des fichiers de données, on fournira la définition de chacun des fichiers.

3.7 Découpage de réalisation (sous-systèmes)

Le découpage en sous-systèmes a pour rôle de répartir les tâches de réalisation entre les sous-groupe de réalisation, chaque sous-groupe se voyant confier la réalisation d'un ou plusieurs sous-systèmes.

Afin de permettre aux sous-groupes de travailler de façon quasi indépendante, on s'efforcera dans cette partie de définir de façon aussi complète et précise que possible :

- le contenu des sous-systèmes,
- les interfaces externes des sous-systèmes

3.7.1 Liste des sous-systèmes et leurs relations

Un tableau présentera les sous-système à réaliser en indiquant :

- le nom et le rôle du sous-système,
- le rôle principal du sous-système,
- les principales classes du sous-système,
- les programmes concernés.

Un diagramme montrera les dépendances entre sous-systèmes (un sous-système dépend d'un autre s'il fait appel à ses services ou hérite de ses classes abstraites).

3.7.2 Fonctions et interfaces externes des sous-systèmes

Pour chaque sous système un paragraphe détaillera :

- ses fonctions, en précisant les principales classes incluses dans le sous-système, et les autres fonctions qui n'apparaissent pas explicitement dans la définition de ces classes,
- les interfaces externes, en précisant les principales classes du sous système qui serviront d'interface pour le sous-système (classes dont les méthodes sont appelées à partir d'autres sous-systèmes), et les autres types d'interface externes tels que des fichiers, des messages réseaux, des messages d'événement Windows, des signaux QT....

3.8 Dimensionnements et performances

On précisera les dimensionnements ou contraintes de performances significatifs pour l'architecture, par exemple :

- nombre maximum d'utilisateurs ou nombre d'utilisateurs simultanés,
- tailles maxima de tables,
- temps de réponse
- délais à respecter...

Pour chacun de ces points, on précisera comment l'architecture les prend en compte.

4 - Exemple de document d'architecture logicielle

La suite de ce document est un document d'architecture logiciel réduit partiellement rédigé. Il répond au démonstrateur de navettes d'aéroport présenté dans l'introduction aux guides de rédaction.

On notera les points suivants :

1 - Comme tout document d'un projet, un dossier d'architecture commence par une page d'entête et une page historique, ces deux pages n'ont pas été reproduites ici.

2 - Le modèle de classe présenté n'a fait l'objet d'aucune validation, il est montré uniquement pour présenter des exemples de rédaction ou de présentation, et pas pour servir de modèle d'architecture

3- Les anglo-saxons ne connaissant pas les accents, tous les identificateurs destinés à la programmation ont été mis sans accents.

4 - Les principales classes résultent :

- du problème physique : Borne, Ligne, Station, Navette
- des structures de données et l'accès à la base de données : DataMngr, DataListe, DataFric, DataDate, DataUser,
- de la gestion des interfaces réseau : Rezo, Formateur, Serveur,
- de la gestion d'autres interfaces : Paiement, Login, Autorisation,
- des bases d'architecture : MainAT, ModelAT, FormAT, GestionAT, MainCC, ModelCC, FormCC, GestionCC.

5 - Pour les diagrammes de classe, il a été décidé de faire 5 diagrammes, chacun contenant un groupe de classes principales (qui apparaît comme un package dans les diagrammes) et les classes des autres groupes en relation avec ces classes.

6 - Choix des sous systèmes

Les deux principaux critères de choix ont été :

- le partage entre spécifique de la borne, spécifique du centre de contrôle, et commun,
- un découpage par technique (réseau, base de données, gestion de l'application).

On arrive ainsi à 5 sous-systèmes.

Exemple de document d'architecture

Sommaire

1 - Introduction.....	9
1.1 Objet	9
1.2 Terminologie et sigles utilisés.....	9
1.3 Documents de référence	9
2 - Présentation générale de l'architecture.....	9
3 - Les principaux objets et leurs classes	10
3.1 Liste des principales classes	10
3.2 Diagrammes de relations entre classes.....	11
3.3 Rôles et interfaces des classes Données partagées (Data).....	12
3.4 Rôles et interfaces des classes Base de données (DB)	12
3.5 Rôles et interfaces des classes réseau (net)	12
3.6 Rôles et interfaces des classes particulières à la borne (AT)	13
3.7 Rôles et interfaces des classes particulières au centre de contrôle (CC).....	13
4 - Scénarios d'interaction entre objets.....	13
5 - Les processus et le déploiement.....	14
5.1 Déploiement, programmes, processus et threads	14
5.2 Principes de fonctionnement des processus	15
5.3 Interactions entre processus	17
6 - Interfaces externes du système.....	17
6.1 Base de données	17
6.2 - Interface opérateur	19
7 - Les sous-systèmes et leurs interfaces externes	20
7.1 Liste des sous-systèmes et leurs relations	20
7.2 Fonctions et interfaces externes des sous-systèmes	20
8 - Dimensionnements et performances	21

1 - Introduction

1.1 Objet

Ce document décrit l'architecture du démonstrateur de navettes d'aéroport, dont le rôle est de fournir des informations sur des bornes à disposition des clients d'autobus navettes pour aéroport.

1.2 Terminologie et sigles utilisés

Ligne

Trajet suivi par un autobus entre l'aéroport et un terminus et inversement, avec des arrêts intermédiaires

Station

Endroits où l'autobus est arrêté ou s'arrête pour faire monter ou descendre des passagers : aéroports, terminus de lignes, arrêts intermédiaire de ligne.

...

1.3 Documents de référence

Spécification technique de besoin du démonstrateur de navettes d'aéroport N° du

2 - Présentation générale de l'architecture

Le logiciel est découpé en deux programmes :

- un programme PAT pour la borne,
- un programme PCC pour les fonctions centre de contrôle.

Sur chaque borne, un processus s'exécute sur le programme PAT, il créera des threads pour les échanges réseau et la commande des périphériques de paiement.

Au centre de contrôle, un processus s'exécute sur le programme PCC et crée deux threads pour les échanges réseau avec chaque borne active, et un thread pour les autorisations carte bancaire.

Les échanges entre les bornes et le centre de contrôle sont faits par l'intermédiaire d'un réseau (fourniture extérieure au système) utilisant les protocoles TCP-IP et ayant une vitesse de transmission limitée pour certains sites d'implantation de bornes.

Afin de limiter les échanges par le réseau, on implantera une base de données sur chaque borne et au centre de contrôle. Les bases de données de chaque borne sont identiques et sont un sous-ensemble de la base de données du centre de contrôle. Les accès aux bases de données seront faits avec ODBC et la programmation sera prévue indépendante du SGBD. Pour le centre de contrôle on pourra utiliser des SGBD comme Oracle ou SQL Server. Pour les bornes on utilisera de préférence un SGBD qui gère la base dans un fichier (Access, SQLite...) afin de simplifier les rechargements de base.

La conception des programmes est de type objet, et le mode de programmation événementiel. Les programmes sont codés en C++ pour Windows XP avec la bibliothèque QT.

3 - Les principaux objets et leurs classes

3.1 Liste des principales classes

Données partagées (Data)

Classe	Rôle principal
Ligne	Informations sur une ligne
Station	Informations sur un arrêt
Navette	Informations sur une circulation de navette
Borne	Informations sur une borne

Base de données (DB)

Classe	Rôle principal
DataMngr	Connexion à la base et exécution des requêtes
DataLigne	Accès aux données lignes, stations, horaires
DataFric	Accès aux tarifs et cumuls de ventes
DataDate	Date, heure et saint du jour
DataUser	Accès aux données opérateur (login, droits, mots de passe)

Classes réseau (Net)

Classe	Rôle principal
Rezo	Accès au réseau (borne et centre de contrôle)
Serveur	Gestion du réseau (centre de contrôle)
Formateur	Formatage et déformatage pour les échanges réseau

Classes particulières à la borne (AT)

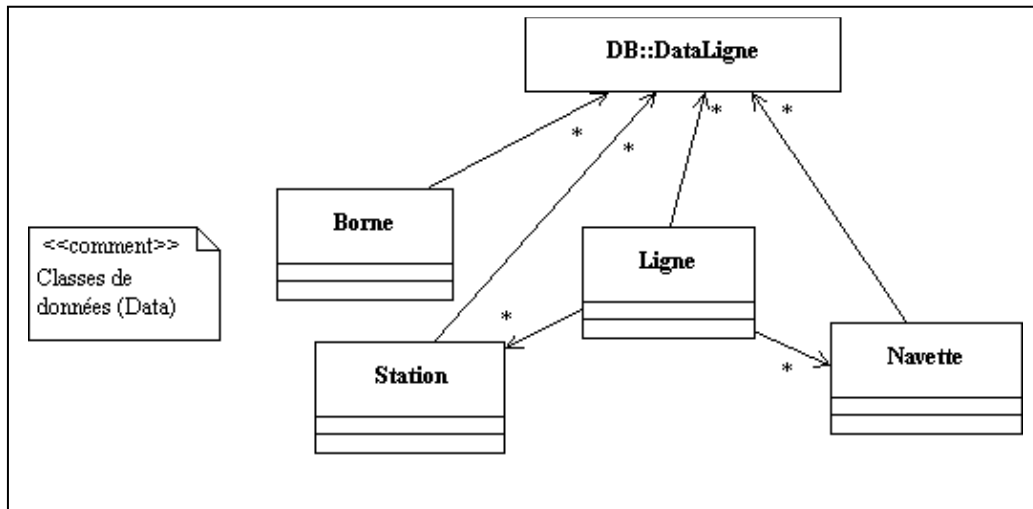
Classe	Rôle principal
MainAT	Gestion de la fenêtre principale et construction de l'application
FormAT	Gestion de la fenêtre Vue principale
GestionAT	Gestion de la borne
ModelAT	Accès aux données partagées
Paielement	paielement des billets

Classes particulières au centre de contrôle (CC)

Classe	Rôle principal
MainCC	Gestion de la fenêtre principale et construction de l'application
FormCC	Gestion de la fenêtre Vue principale
GestionCC	Gestion du centre de contrôle
ModelCC	Accès aux données partagées
Login	Ecran de login
Autorisation	Gestion des autorisations carte bancaire

3.2 Diagrammes de relations entre classes

Classes de données partagées (Data)



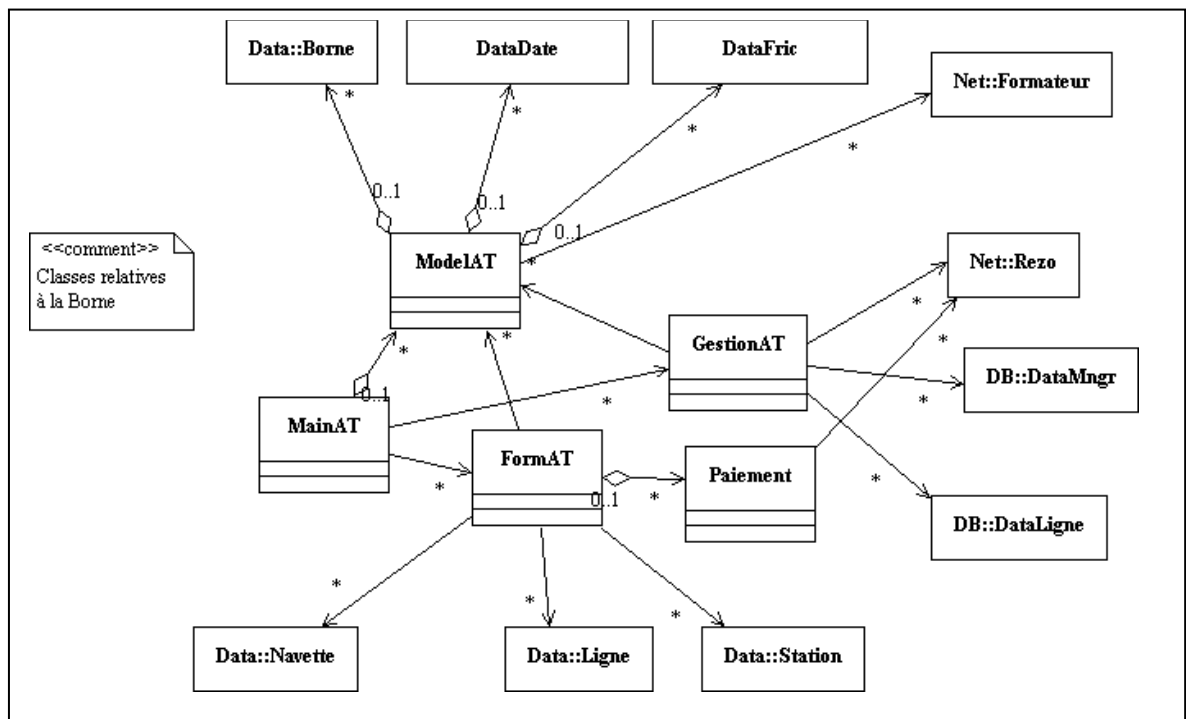
Classes Base de données (DB)

...

Classes réseau (net)

...

Classes relatives à la borne



Classes relatives au centre de contrôle

...

3.3 Rôles et interfaces des classes Données partagées (Data)

3.3.1 Classe Lignes

La classe Ligne permet d'accéder à toutes les informations sur une ligne :

- sens de la ligne (de ou vers l'aéroport), terminus,
- liste des stations de la ligne,
- liste des navettes de la ligne.

Données de la classe (données privées)

```
int lineCode; // line code
bool lineSens; // line way (true from airport)
QString lineTerminus; // line terminus halt ident
```

Interface d'appel public :

```
// construct line object from database with lineCode as line code :
Ligne(int lineCode);
// construct line object from parameters :
Ligne(int lineCode, bool lineSens, QString lineTerminus);
// return line code
int code(void);
// return line way : from airport true, else false
bool sens(void);
// return line terminus ident
QString terminus(void);
// return list of halts identents
QList<QString> halts(void);
// return list of shuttles codes
QList<int> navettes(void);
```

3.3.2 Classe Station

...

3.3.3 Classe Navette

...

3.3.4 Classe Borne

...

3.4 Rôles et interfaces des classes Base de données (DB)

...

3.5 Rôles et interfaces des classes réseau (net)

3.5.1 Classe Rezo

La classe Rezo est l'interface pour tous les échanges entre le centre de contrôle et les bornes à travers le réseau. Elle permet d'initialiser la connexion réseau au niveau d'une borne ou de mettre la fonction serveur en écoute au centre de contrôle, de fermer la ou les connexion réseau, d'envoyer des blocs de données par le réseau, et de prendre en compte les blocs de données reçues par le réseau (qui seront transmises à l'application par un signal).

Les opérations (connexion, envoi de données...) étant réalisées de façon asynchrone, un compte rendu est aussi transmis par signal pour chaque opération.

Interface d'appel public :

```
// AT start connection or CC start listen :
void start(void);
// close all links (CC and AT) and stop listen (CC):
void stop(void);
// send array of bytes of type type from server to AT of ident,
// or from AT to server (ident not used)
// return false if ident not connected or already busy to send :
bool sendData(int type, QByteArray data, QString ident=QString());
// check if ident connected (ident unused on AT) :
bool isConnected(QString ident=QString());
```

Signaux émis :

```
// emit data with type type coming from ident (ident unused on AT)
void postDatas(int type, QByteArray data, QString ident);
// emit operation ending status
void postStatus(int opCode, int status);
```

3.5.2 Classe Serveur

...

3.5.3 Classe Formateur

...

3.6 Rôles et interfaces des classes particulières à la borne (AT)

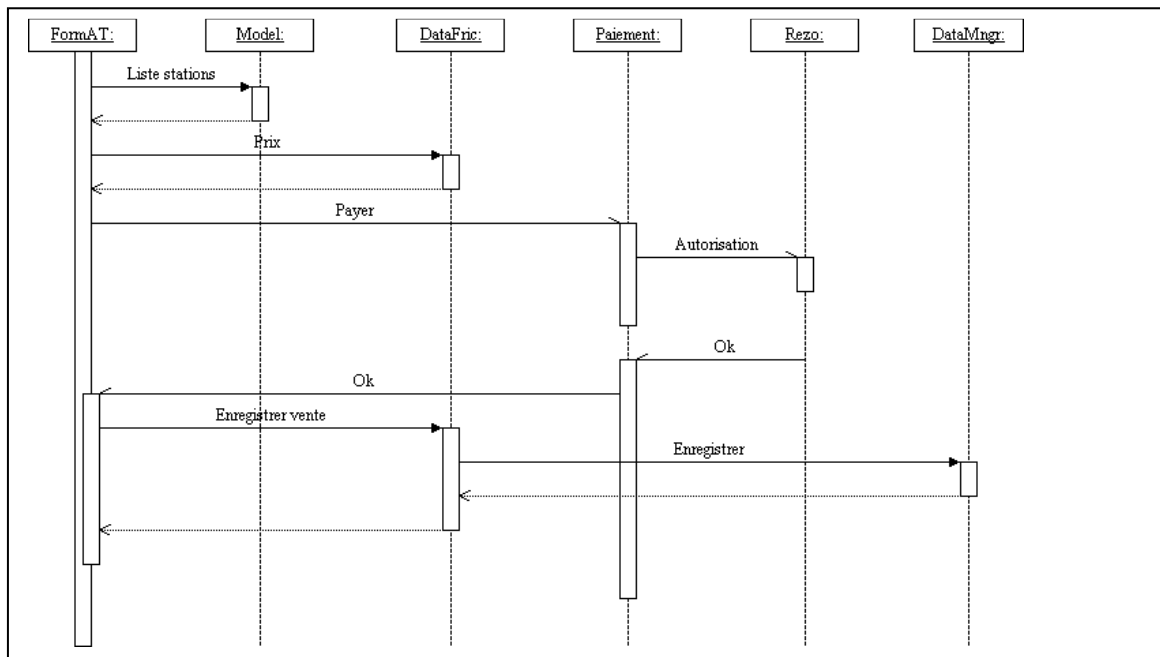
3.7 Rôles et interfaces des classes particulières au centre de contrôle (CC)

4 - Scénarios d'interaction entre objets

Affichage des prochains départs

...

Vente de billets



Réception de message réseau

...

Initialisation de la borne

...

Initialisation du centre de contrôle

...

Surveillance des bornes au centre de contrôle

...

5 - Les processus et le déploiement

5.1 Déploiement, programmes, processus et threads

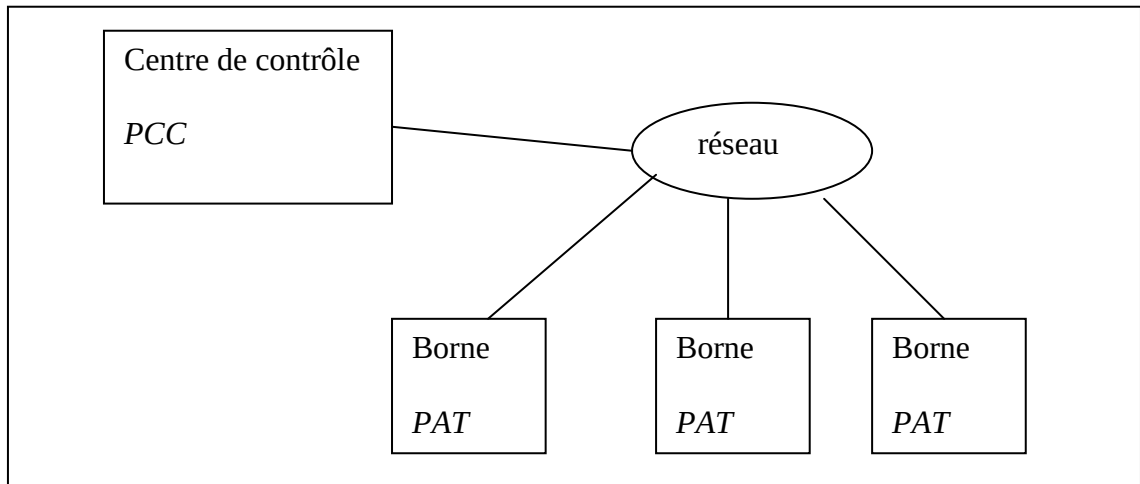
Programmes et processus

Processeur	Programme	Processus	Rôle
Borne	PAT	1 processus PAT	Gestion d'une borne
Centre de contrôle	PCC	1 processus PCC	Opérations du centre de contrôle

Le processus PAT crée deux threads pour les échanges réseau, et un thread pour le paiement des billets.

Le processus PCC crée deux threads pour les échanges réseau avec chaque borne active.

Le déploiement du système peut être représenté par le schéma suivant (les processus sont indiqués en italique) :



Nota

Le réseau est une fourniture extérieure au système.

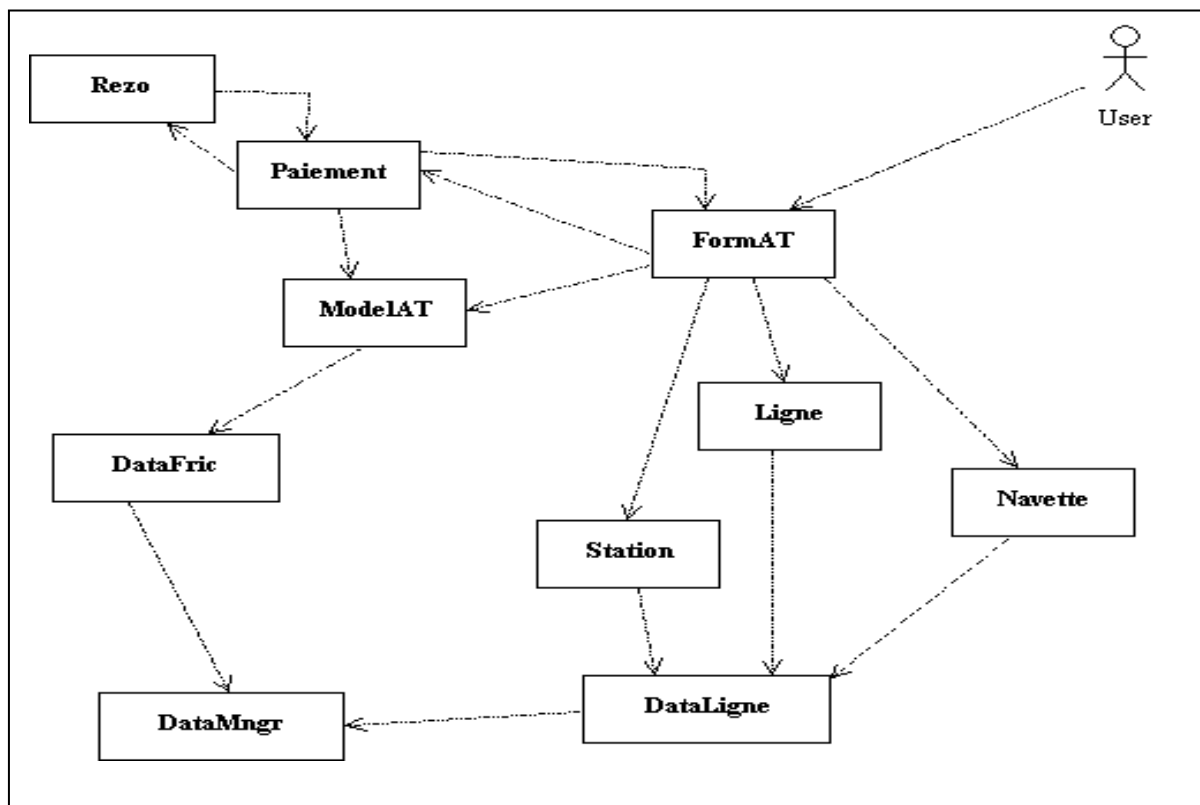
5.2 Principes de fonctionnement des processus

5.2.1 Processus PAT

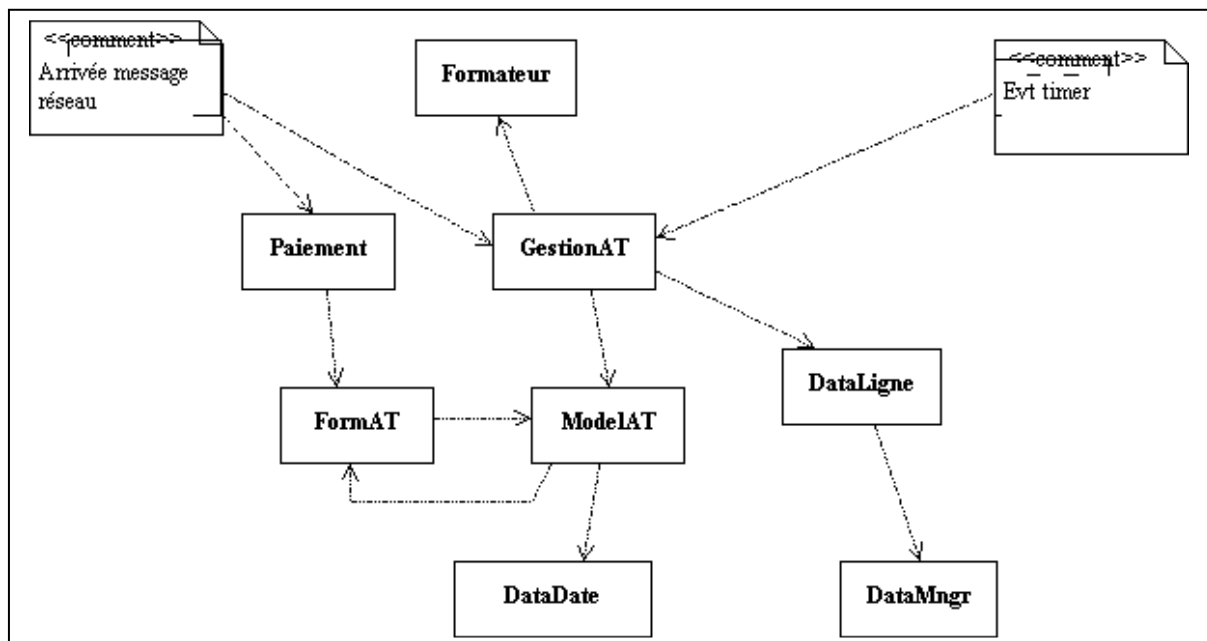
Les diagrammes qui suivent montrent les principaux objets activés et les graphes d'appels entre objets lors de l'arrivée des principaux événements :

- requête usager (informations sur les lignes ou achat de billets)
- déclenchement timer pour rafraîchir les informations générales de l'écran ou message venant du réseau.

Requête usager



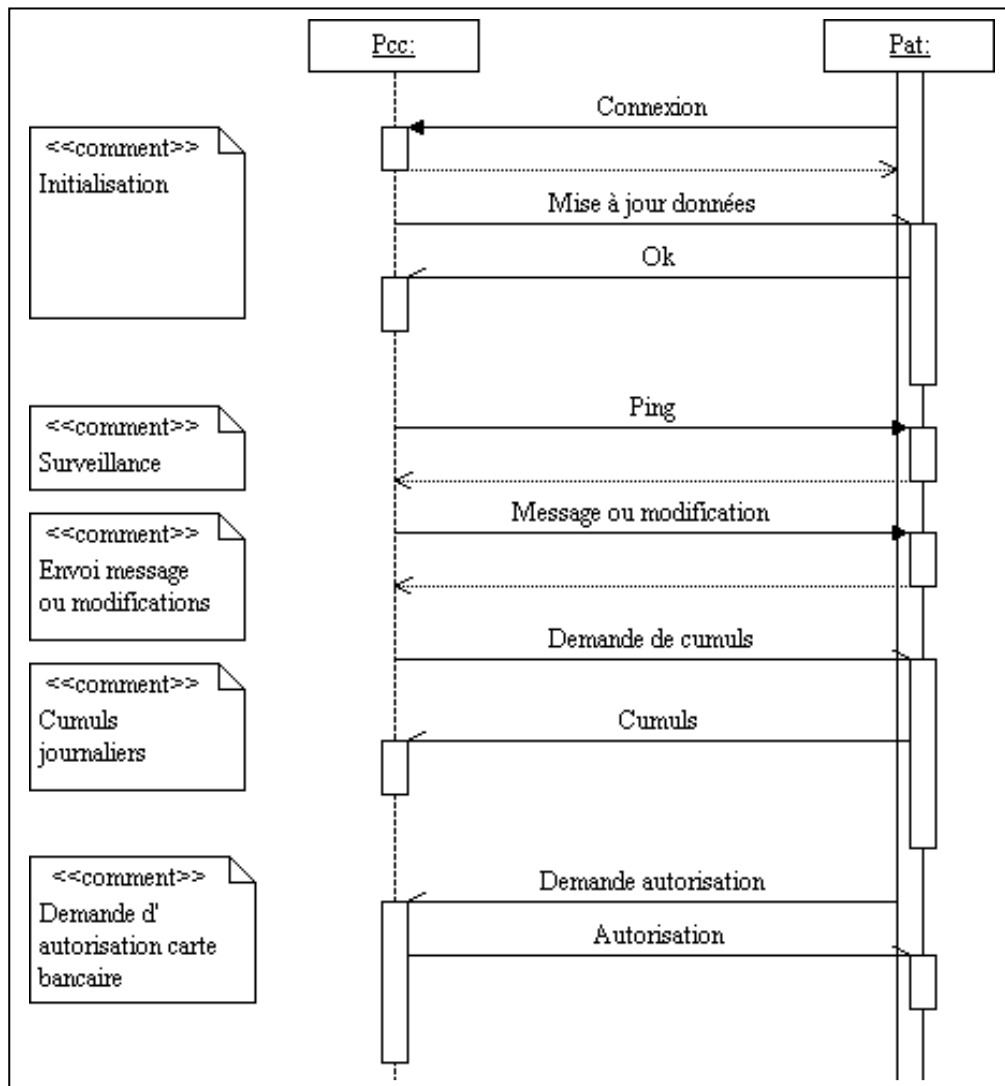
Evènement timer ou arrivée de message



5.2.2 Processus Pcc

...

5.3 Interactions entre processus



6 - Interfaces externes du système

6.1 Base de données

6.1.1 Contenu des bases de données

Les données nécessaires pour le fonctionnement du système sont stockées dans des bases de données au centre de contrôle et dans chaque borne.

La base de données comprend :

- la table des lignes donnant pour chaque ligne le code, le sens et le terminus,
- la table des stations donnant pour chaque station son nom, sa localisation et sa zone tarifaire,
- la table ligne - stations fournissant les associations ligne - station -station suivante,
- la table des navettes fournissant les associations lignes - navettes,
- la table des horaires fournissant les associations navette - station - heure de passage
- la table des tarifs indiquant pour chaque tarif la période de validité et le prix pour chaque zone tarifaire,

- la table des bornes indiquant pour chaque borne le nom, la station d'implantation et son état,
- la table calendrier indiquant le saint du jour pour chaque jour de l'année
- la table des messages textuels, qui enregistre pour chaque message émis, la date et heure d'émission, l'émetteur, les destinataires et le contenu du message,
- la table des cumuls qui enregistre borne par borne pour chaque jour le nombre de billets vendus par zone tarifaire et type de carte de paiement, et les montants correspondants,
- la table des mots de passe qui contient les identificateurs et mots de passe (sous forme codée) des opérateurs et administrateurs du centre de contrôle.

Pour dimensionner ces tables, on retient les valeurs suivantes :

Nombre de lignes possibles	10
Nombre maximum d'arrêts par ligne	10
Nombre maximum de zones tarifaires	10
Nombre maximum de circulations journalières sur une ligne	10 dans chaque sens
Nombre maximum de bornes	50
Nom de station, d'opérateur, d'administrateur	20 caractères maximum
Code navette	4 chiffres
Taille maxima d'un message textuel	40 caractères
Taille des mots de passe	Entre 8 et 16 caractères

Les utilitaires SGDB seront utilisés pour la création de la base, les sauvegardes et restauration, ainsi que pour les opérations de type suppression de lignes, de stations, de messages...

6.1.2 Schéma des bases de données

...

6.1.3 Détail des tables

Table lignes

Champ	Type	Contenu
code	INTEGER	numéro donné à la ligne
isfrom	BOOLEAN	indique si ligne vient de l'aéroport (TRUE) ou y va (FALSE)
terminus	VARCHAR	nom du terminus de la ligne

Table des stations

...

Table ligne - stations

...

Table des navettes

...

Table des horaires

...

Table des tarifs

...

Table des bornes

...

Table calendrier

...

Table des messages textuels

...

Table des cumuls

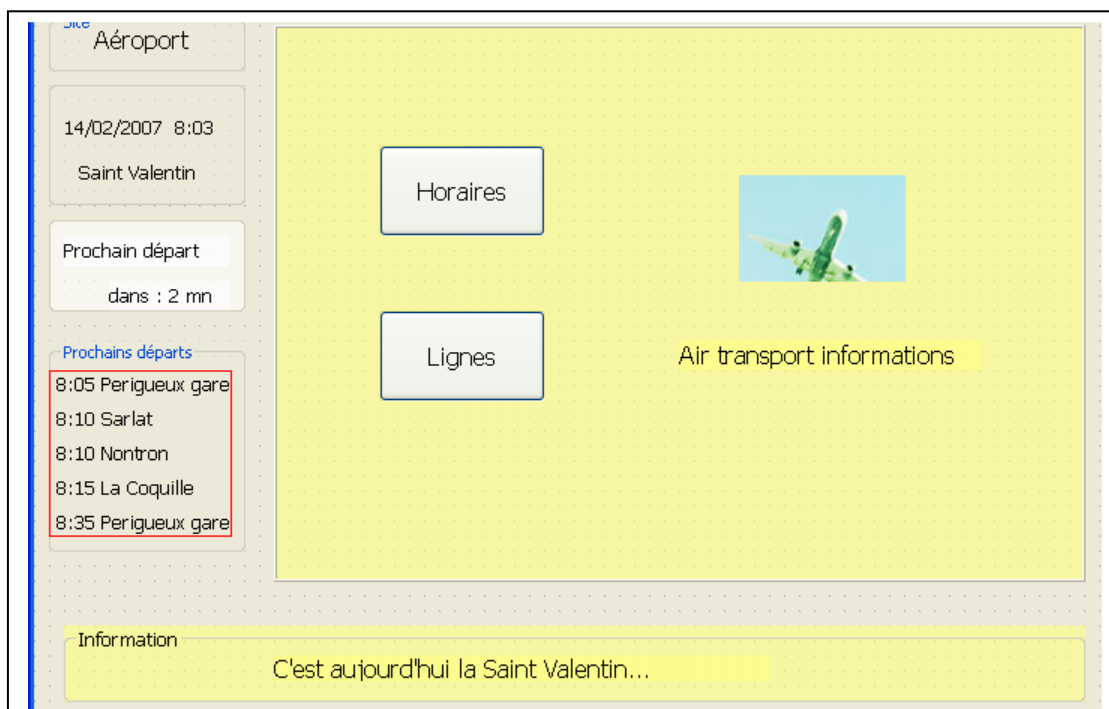
...

Table des mots de passe

...

6.2 - Interface opérateur**6.2.1 Borne**

L'interface opérateur de la borne réelle est un écran tactile, et on prendra en compte les contraintes de cet écran pour le démonstrateur. L'écran sera divisé en deux zones comme présenté dans la maquette ci-dessous. La partie gauche contient les informations permanentes : nom de la station, date et heure et saint du jour, temps restant jusqu'au prochain départ, heures des prochains départ, et en bas le dernier message du centre de contrôle.



La partie droite de l'écran est réservée pour les services à la demande. En l'absence de service demandé elle présente deux pavés de sélection. Lorsqu'un service est demandé elle présente des boutons de sélection et des informations. Les boutons de sélection ont une taille suffisante pour permettre une sélection au doigt.

6.2.2 Centre de contrôle

...

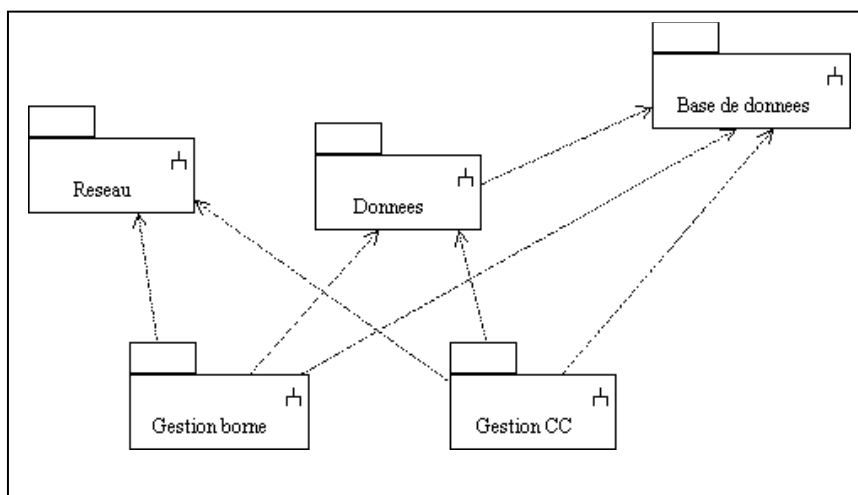
7 - Découpage de réalisation (sous-systèmes)

7.1 Liste des sous-systèmes et leurs relations

Le découpage en sous-systèmes est le suivant :

Nom du sous-système	Principales classes	Rôle	Programmes concernés
RESEAU	Rezo Serveur Formateur	Regroupe les fonctions d'accès réseau	PAT PCC
DONNEES	Ligne Station Navette Borne	Regroupe la définition des données et leur accès	PAT PCC
BASE DE DONNEES	DataMngr DataLigne DataFric DataDate DataUser	Accès à la base de données	PAT PCC
GESTION BORNE	MainAT FormAT GestionAT ModelAT Paielement	Base du programme borne	PAT
GESTION CC	MainCC FormCC GestionCC ModelCC Login Autorisation	Base du programme centre de contrôle	PCC

Les dépendances entre sous-systèmes sont représentées sur le diagramme ci-dessous :



7.2 Fonctions et interfaces externes des sous-systèmes

7.2.1 Sous-système RESEAU

Ce sous-système regroupe les fonctions d'accès réseau.

Il permet de constituer des fonctions d'accès réseau :

- pour une borne avec les classes Rezo et Formateur,
- pour le centre de contrôle avec les classes Rezo, Serveur et Formateur, complétées des classes permettant de gérer les échanges.

La classe Rezo assure l'interface de l'application pour échanger des blocs de données (tableaux d'octets).

La classe Formateur permettra les transformations entre les données à échanger et les tableaux d'octets en entrée ou sortie de la classe Rezo.

L'interface de programmation du sous-système réseau est composé des interfaces des deux classes :

- Rezo,
- Formateur.

Ces deux interfaces sont définis au § 3.

Les échanges avec le réseau seront faits par l'intermédiaire de sockets TCP-IP (classe QTcpSocket), et l'écoute sur le réseau par une socket (classe QTcpServer).

Le nom réseau et le port d'écoute du centre de contrôle sont prédéfinis afin de permettre les demandes de connexion par les bornes.

7.2.2 Sous-système GESTION DONNEES

7.2.3 Sous-système BASE DE DONNEES

....

7.2.4 Sous-système GESTION BORNE

....

7.2.5 Sous-système GESTION CC

....

8 - Dimensionnements et performances

Les paramètres dimensionnants pour les tables de la base de données sont indiqués au § 6.1. Ces paramètres seront pris en compte dans la définition détaillée et la création de la base de données.

Le centre de contrôle doit pouvoir assurer les dialogues avec 5 bornes simulées parmi 50 bornes possibles. Les tables seront dimensionnées pour au moins les 50 bornes possibles, et les écrans seront définis pour représenter les informations d'au moins 5 bornes.

Les temps de réponse à une demande sont limités à 2 secondes. Des essais seront fait sur les premiers prototypes pour vérifier que ce délai est bien tenu pour les requêtes courantes, et voir quelles sont les types de requête qui risquent d'être plus longues et demanderont l'affichage d'un sablier.