AndroIUT

Cahier de maintenance

1. Introduction

AndroIUT est un projet pour technologie mobile destiné aux étudiants et professeurs de l'IUT de Dijon du département informatique.

Il s'articule autour de 4 services qui sont :

- La récupération et l'affichage du planning
- La récupération et l'affichage des emails
- La récupération et l'affichage des notes/absences
- La navigation et l'affichage de fichiers dans l'arborescence FTP

Ce projet est développé suivant 2 axes qui sont le côté client et le côté serveur.

La partie client de l'application s'appuie sur le framework Android qui lui-même est une surcouche du framework Java crée par Oracle.

La partie serveur quant à elle, est développée en utilisant le langage PhP, lequel est particulièrement simple et adapté au développement sur serveur.

En plus de ces deux langages, l'application utilise un certain nombre de protocoles parmi lesquels :

- HTTPS, utilisé pour les requêtes vers le serveur
- LDAP, utilisé pour authentifier un utilisateur sur les serveurs de l'IUT
- FTPS, utilisé pour accéder au serveur FTP de manière sécurisée
- IMAPS, utilisé pour accéder au serveur EMAIL de manière sécurisée

Enfin, la totalité des réponses du serveur étant au format XML, l'application s'appuie sur cette extension pour structurer et dynamiser le contenu en fonction de l'utilisateur.

2. Cahier des charges

Le cahier des charges nous imposait de proposer un niveau de sécurité suffisant tout en offrant une expérience utilisateur simple et confortable.

Le lecteur de flux RSS de l'université de Bourgogne qui devait être intégré à l'application sur la page d'accueil, a finalement été abonné aux vues de son apport à l'application, relativement peu important, à contrario de la surcharge de l'interface utilisateur qu'il engendrait.

L'application c'est vu offrir en revanche la fonctionnalité de pouvoir importer le planning provenant d'ADE directement dans l'agenda Google de l'utilisateur, et ainsi bénéficier de tous les services de rappels et d'informations qu'il propose.

Elle propose aussi la possibilité de télécharger et visionner les fichiers présents sur le FTP si le format dudit fichier est connu du système d'exploitation ou d'une application tierce.

Une autre différence vis-à-vis du cahier des charges est notable au niveau de la sécurité offerte par l'application. Dans le but de restreindre les possibilités de récupérer les identifiants de connexion de l'utilisateur, l'application génère un espace de stockage crypté, en utilisant l'algorithme AES, représenté sous forme d'une map. Toutes les valeurs qui sont stockées dans cette dernière sont alors instantanément cryptées via une clé, générée dynamiquement par le serveur.

3. *Analyse*

L'application respecte le cahier d'analyse dans la majorité des cas sauf pour les services FTP et Emails (service au sens client du terme).

En effet, lors de l'analyse nous n'avions pas pris en compte le fait que les librairies utilisées pour ces deux services ne déchargent pas les traitements (connexions web, traitements, etc…) dans un processus séparé de l'interface.

Android nous imposant de faire toutes les requêtes internet dans un processus différent de celui de l'interface utilisateur, il a fallu penser et mettre en place un système capable de gérer la communication interprocessus dans le cas du FTP, et un système permettant de réutiliser les threads dans le cas du service d'email.

Dans le cas du FTP, l'organisation du service est décrite par le schéma suivant :

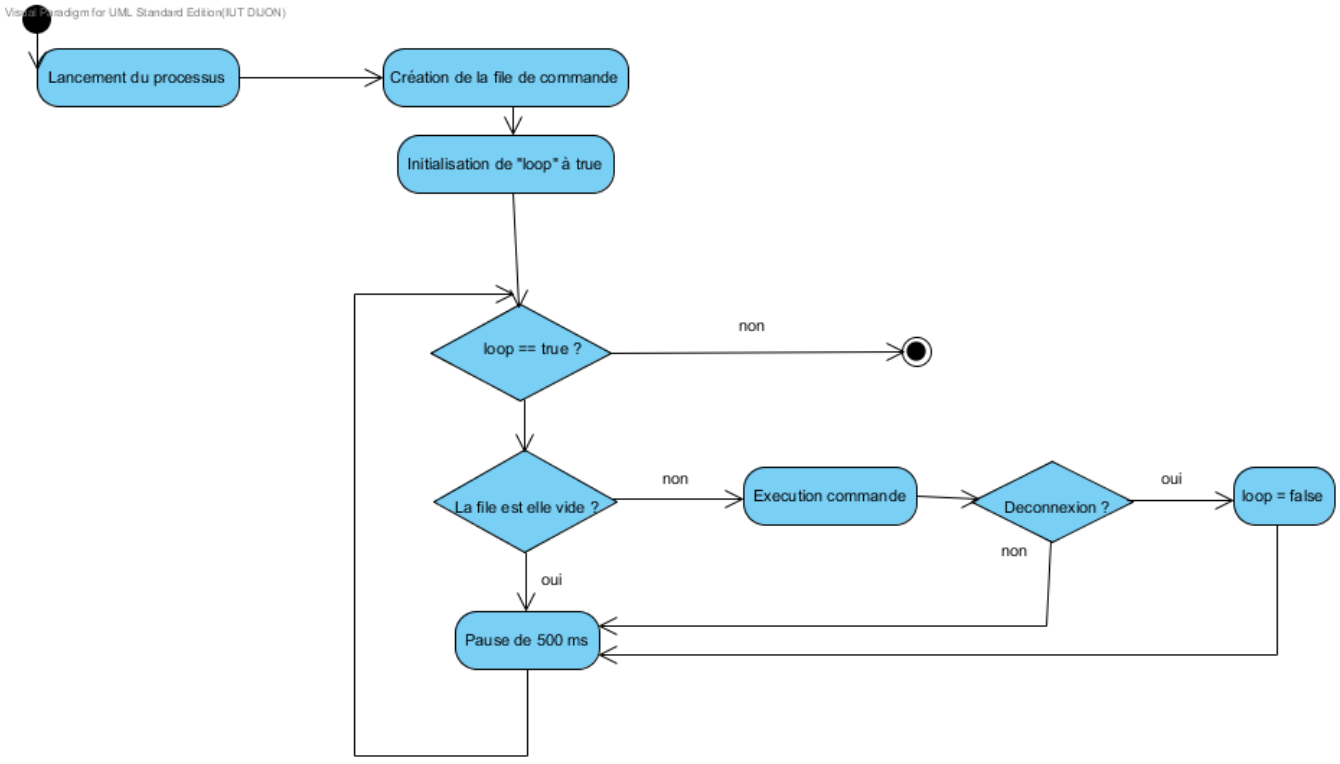
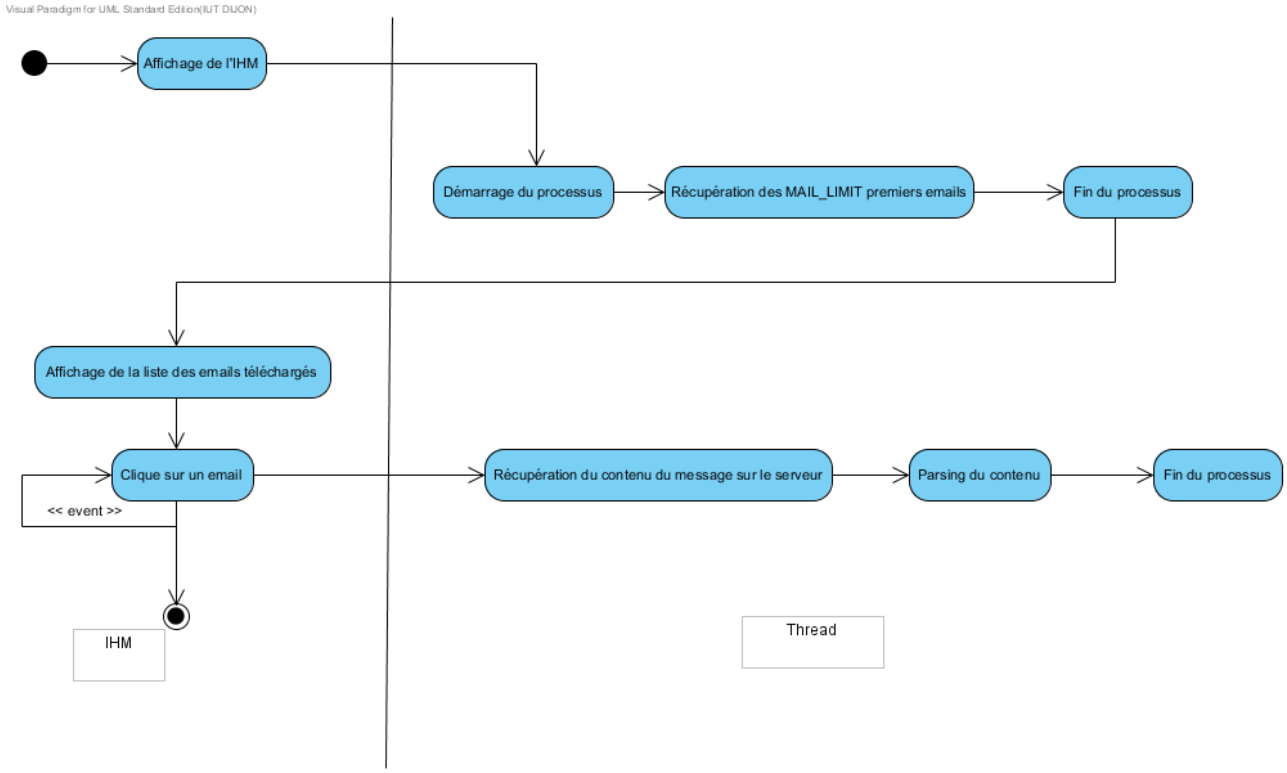
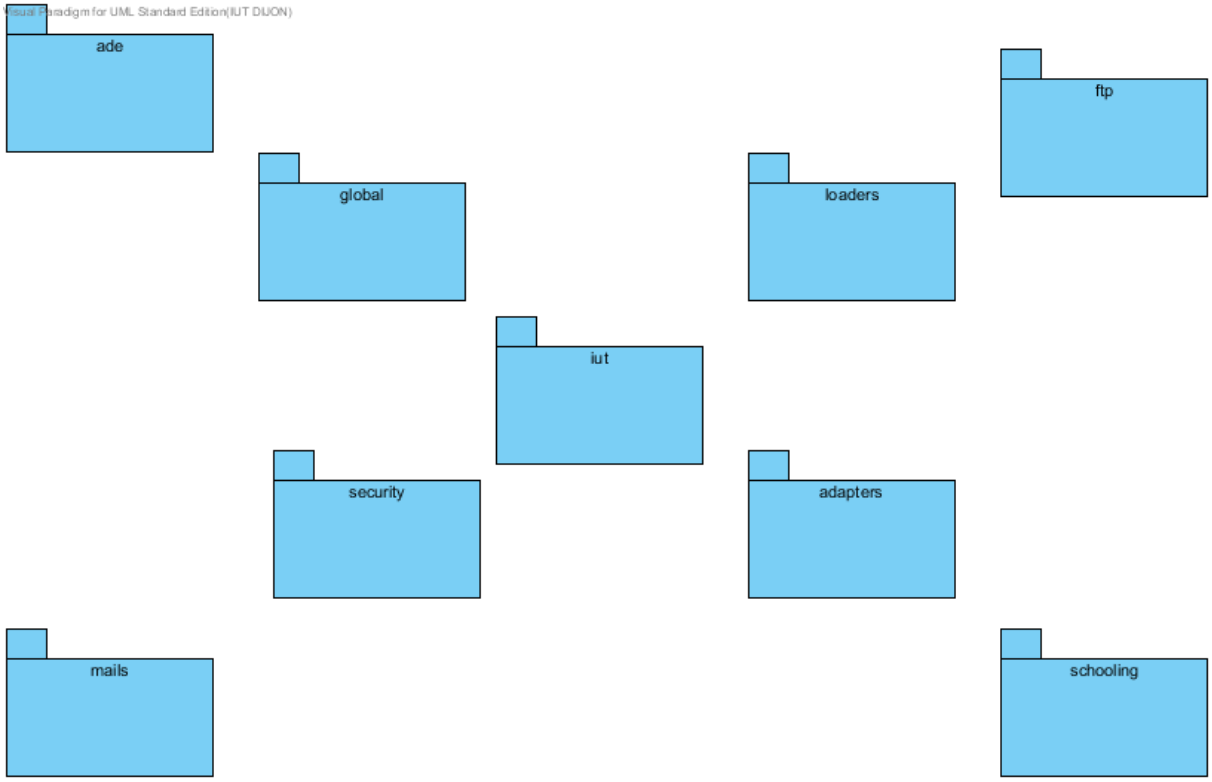


Figure 1 : Description d'un cycle du processus FTP

Pour les emails, l'organisation suit la structure suivante :



4. *Architecture générale du projet*



5. Technologies utilisées

La principale technologie utilisée pour réaliser la partie client de ce projet est Java. Utilisé nativement par Android, il nous semblait donc pour profiter pleinement des fonctionnalités mise à disposition par le SDK Android.

Afin de compléter les fonctionnalités, déjà très avancées, proposé par Android, nous avons utilisé les libraires suivantes :

- Android SDK

Clé de voute, du système d'exploitation, il offre des apis bas niveau pour accéder au matériel installé sur le mobile. Il permet aussi de créer et d'interagir avec l'interface utilisateur.

- JavaMail API

Utilisée dans le but de supporter le protocole IMAP permettant de recevoir les Emails. Cette API a le défaut d'être assez mal documentée par son contributeur principal, à savoir Oracle. L'une de ses principales difficultés est le fait qu'elle ne possède pas de classe permettant de lire le contenu d'un message. La lecture du message doit donc se faire en utilisant des classes personnelles.

- Apache Commons Net API

Cette libraire permet d'utiliser un bon nombre de protocole web, tel que FTP(s). Cette librairie à l'avantage d'être très bien documentée et largement connue, et donc de proposer un nombre impressionnant d'exemples ce qui permet une prise en main simple. Elle offre des fonctionnalités assez vastes et étendues concernant les commandes FTP, et notamment la récupération et l'envoi de fichier sur le serveur FTP.

- [Apache Commons Codec API]

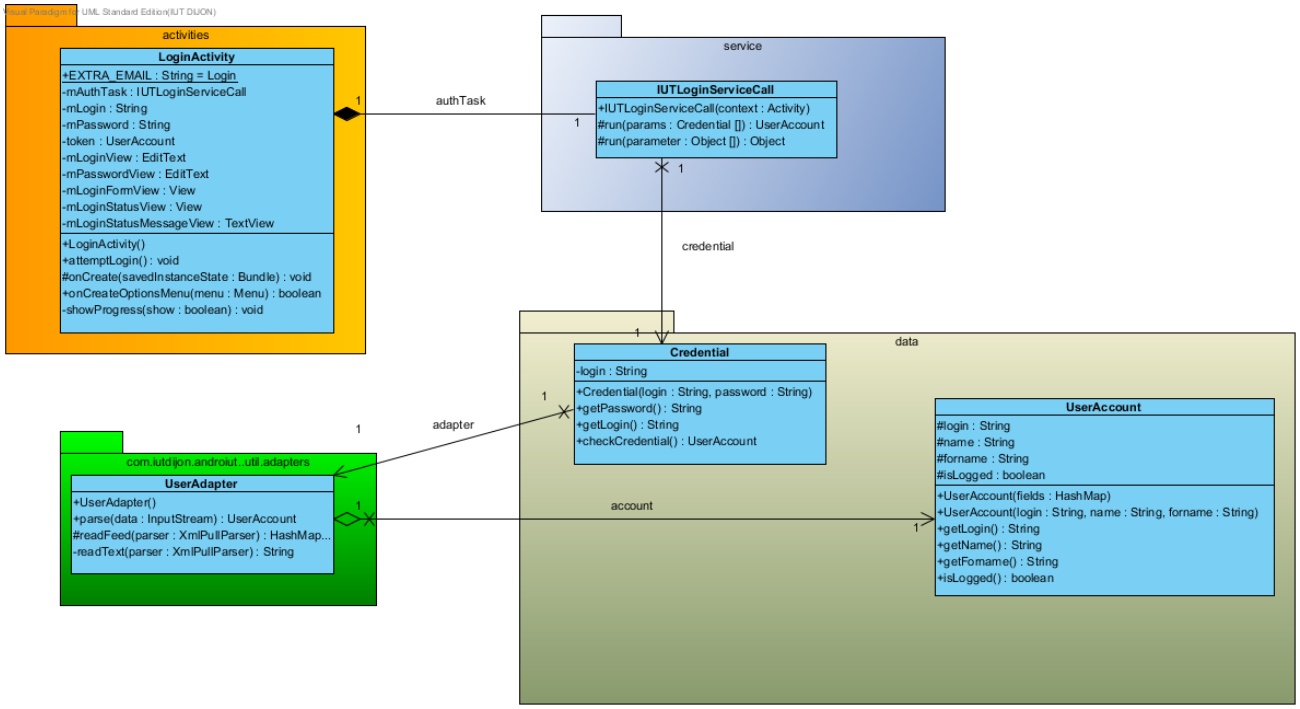
Utilisée principalement dans le but de pouvoir encoder et décoder la Base64, elle fournit aussi des méthodes statiques permettant d'encoder une URL. Elle est utilisée notamment dans le centre de sécurité de l'application qui utilise la Base64 comme vecteur d'initialisation de la clé de cryptage.

- [Apache Commons IO API]

Cette librairie, quasiment indispensable dans un projet Java, permet de s'affranchir de tous les problèmes liés à la lecture / écriture de flux. Elle s'occupe aussi de la gestion de l'encodage d'un flux, en supportant un bon nombre de standards tels qu'UTF-8, ISO 8891 etc. … Enfin, elle permet de faire du monitoring sur l'envoie et la réception de fichier via le réseau. Elle nous permet par exemple d'afficher la progression du téléchargement d'un fichier FTP.

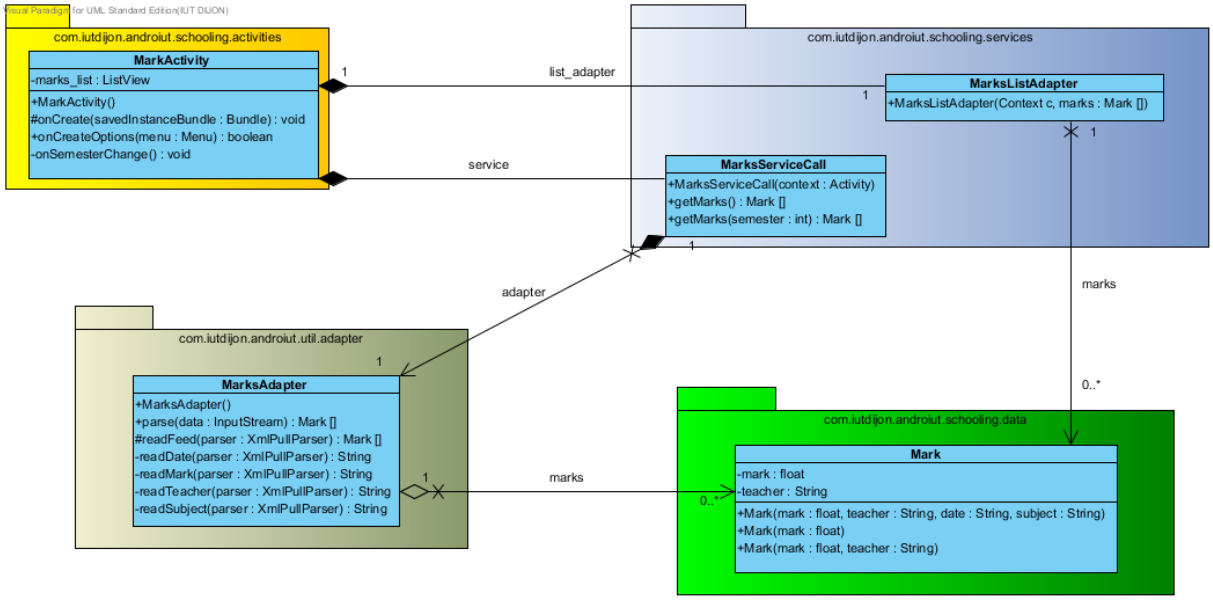
6. *Description précise du projet*

6.1. Connexion



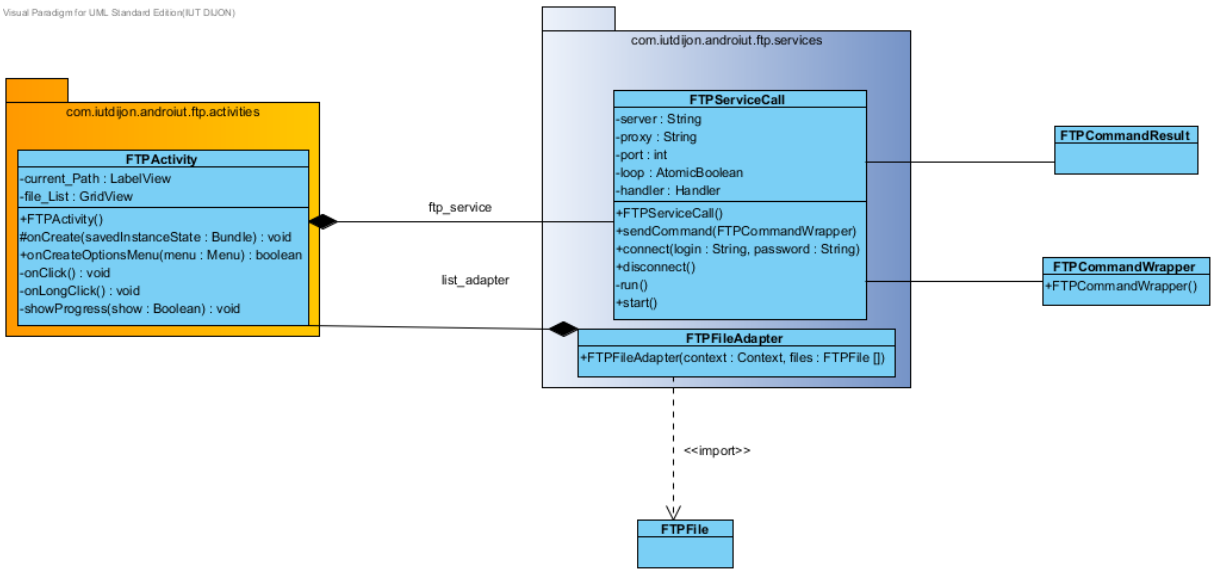
La composition du SecureCenter.java avec l'UserAccount n'est pas spécifiée dans le schéma.

6.2. Notes / Absences

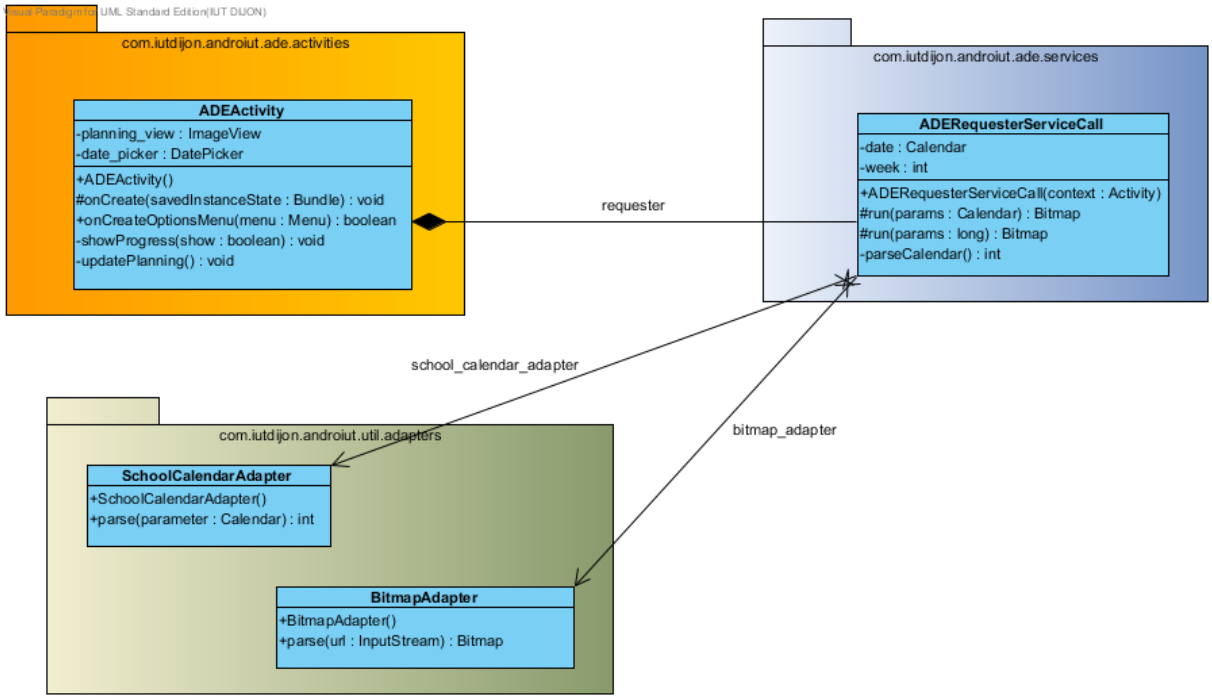


Le service présenté ici est relatif aux notes, il en va exactement de même pour les Absences.

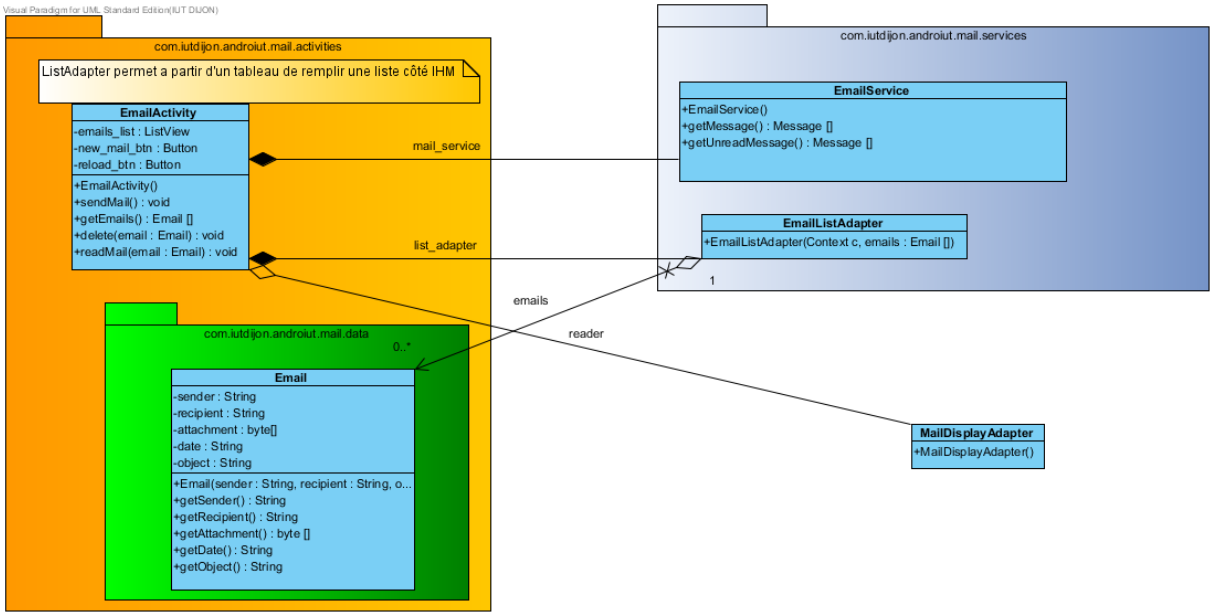
6.3. FTP



6.4. Planning



6.5. Emails



7. Documentation du code source

La Javadoc est fournie dans le dossier « doc » en annexe.

8. *Problèmes non résolus*

Il nous a été impossible de proposer l'envoie de fichier depuis le mobiles vers le serveur FTP. En effet tous les mobiles ne disposant pas d'un gestionnaire de fichiers, il ne nous semblait pas évident de proposer un service fonctionnel et robuste aux utilisateurs.

9. *Développement envisageable*

- Il pourrait-être envisageable d'intégrer la récupération des pièces jointes aux emails.
- La rédaction d'emails directement depuis le mobile pourrait-être intéressante en utilisant par exemple le client mail par défaut.
- La possibilité de pouvoir choisir le semestre dans les notes, pourrait-être un plus. Actuellement, les notes de l'année en cours sont affichées.