



Travail de Bachelor 2011

Filière Informatique de gestion

Smart Mobile Agenda



Etudiant : François Morard

Professeur : Yann Bocchi

Préface

A l'heure actuelle, une majorité de personnes possèdent un smartphone (littéralement « téléphone intelligent »). Mais sont-ils vraiment intelligents ? Ils permettent de saisir ou d'accéder à un grand nombre d'informations. Malheureusement pour l'utilisateur, beaucoup de ces applications sont indépendantes les unes des autres.

Chaque smartphone possède un agenda ou calendrier. Les utilisateurs peuvent saisir différents événements, comme par exemple leurs rendez-vous professionnels, etc. et y associer une alarme qui leur rappelle que le rendez-vous précis va avoir lieu dans un certain temps.

Cependant, une fois les informations saisies par l'utilisateur dans son calendrier, ces dernières n'ont plus aucun lien avec les événements extérieurs. Certes l'utilisateur peut modifier l'alarme, mais il doit le faire manuellement, et il n'en a pas toujours la possibilité.

Par exemple un utilisateur lambda, habitant à Sierre doit se rendre à Crans-Montana pour un rendez-vous le lendemain à 9 heures. Il planifie donc cet événement dans son agenda mobile et y associe une alarme une heure avant. Malheureusement pour lui, durant la nuit une quantité importante de neige est tombée sur la station valaisanne. Il faudra donc à notre utilisateur 30 minutes de plus pour se rendre à Crans-Montana et ce dernier sera probablement en retard.

L'objectif est de réaliser une application utile et utilisable afin de faire interagir le simple calendrier de base disponible sur les smartphones avec différents services qui peuvent être utiles pour l'utilisateur final, tels que la météo, les transports publics, etc.

Dans notre exemple précédent, grâce à l'application, l'utilisateur serait arrivé à l'heure à son rendez-vous. En effet, l'alarme se serait avancée automatiquement dans le temps, grâce à la configuration préalable de l'utilisateur.

Cette application, prénommée « Smart Agenda » a été développée durant les trois mois de travail de bachelor.

D'avance, je vous souhaite beaucoup de plaisir à la lecture de ce rapport.

François Morard

HES-SO Valais, août 2011

Résumé

Introduction

Actuellement, sans compter les applications calendrier natives sur les smartphones, il existe beaucoup d'applications permettant de gérer un agenda.

Malheureusement, très peu, voir aucune de ces applications ne sont réellement intelligentes. En effet, elles ne tiennent pas compte des facteurs externes tels que la météo, les transports publics, l'état des routes, etc.

Le but de l'application Smart Agenda est de rendre les calendriers des smartphones (synchronisés avec Google Calendar) intelligents. C'est-à-dire de pouvoir avancer automatiquement dans le temps l'alarme d'un événement si par exemple il neige ou les transports en commun ont du retard (avec configuration préalable de l'utilisateur).

Serveur Web

Afin que l'utilisateur puisse configurer tout cela, il lui faut une interface web. Cette dernière doit donc être hébergée sur un serveur. Dans ce projet, IIS7 a été installé sur un serveur Windows 2008. Le système de gestion de base de données SQL Server 2008 a également été installé. La plateforme Web ainsi que le service Windows sont développés en ASP.NET (C#) (Langage supporté par l'API de Google Calendar en version 2.0)

Plateforme Web

L'interface web permet à l'utilisateur de s'authentifier sur la plateforme, d'autoriser l'application Smart Agenda à accéder au calendrier Google de l'utilisateur et également de configurer les services auxquels l'utilisateur est intéressé et de configurer différentes règles pour la mise à jour des alarmes. Une petite interface affichant les prochains rendez-vous a également été développée.

Service Windows

Afin de pouvoir mettre à jour les différentes alarmes, il est nécessaire d'avoir un service tournant en arrière-plan. Ce service contrôle les différentes règles et actualise les alarmes.



Motivations personnelles

Lors de la lecture des différentes possibilités de travaux de bachelor, l'idée de Smart Mobile Agenda m'a tout de suite plu. A ma connaissance, il n'existait aucune application qui proposait ces fonctionnalités : « la bonne information au bon moment ». Ce qui me permettrait de relever un défi supplémentaire.

L'idée de rendre un calendrier plus « smart » m'a tout de suite énormément plu.

Remerciements

Je tiens à remercier chaleureusement toutes les personnes m'ayant soutenu durant ce travail de bachelor.

Une attention particulière à :

M. Yann Bocchi, pour m'avoir encadré et suivi durant toute la durée du projet.

Merci également aux personnes ayant participées à la relecture de ce rapport.

Merci enfin à mon amie et à ma famille de m'avoir soutenu durant cette formation.



Déclaration sur l'honneur

Je déclare, par ce document, que j'ai effectué le travail de bachelor ci-annexé seul, sans autre aide que celles dûment signalées dans les références, et que je n'ai utilisé que les sources expressément mentionnées. Je ne donnerai aucune copie de ce rapport à un tiers sans l'autorisation conjointe du RF et du professeur chargé du suivi du travail de bachelor, y compris au partenaire de recherche appliquée avec lequel j'ai collaboré, à l'exception des personnes qui m'ont fourni les principales informations nécessaires à la rédaction de ce travail et que je cite ci-après :

- M. Yann Bocchi

François Morard

Glossaire

Ajax (Asynchronous Javascript and XML)

Combinaison de technologies utilisée en programmation web afin de dynamiser les applications.

Ajax Control Toolkit (ASP.NET)

Projet open-source permettant d'intégrer diverses fonctionnalités Ajax à un site web développé en ASP.NET

API (Application Programming Interface)

Interface de programmation regroupant un ensemble de fonctionnalités mise à disposition des programmeurs par son éditeur

Cookie

Fichier texte enregistré par une application web sur un ordinateur client stockant des informations utiles pour cette dernière

Expression régulière (regex)

Chaîne de caractères décrivant un ensemble de chaînes de caractères correspondants

FTP (File Transfer Protocol)

Protocole permettant d'échanger des fichiers sur internet

GSM (Global System for Mobile Communication)

Norme numérique pour la téléphonie mobile

HTTP (HyperText Transfer Protocol)

Protocole de communication client-serveur



IIS (Internet Information services)

Logiciel de serveur web développé par Microsoft

Parser (Analyseur syntaxique)

Algorithme permettant de décortiquer un texte en fonction de sa structure

RSS

Extension du langage XML permettant à des sites web de diffuser de l'information.

Serveur web

Ordinateur (souvent serveur) sur lequel un logiciel serveur HTTP est installé

SQL (Structured Query Language)

Langage permettant d'interroger ou de modifier une base de données

SQL Server

Système de gestion de base de données développé par Microsoft

Token

Terme anglais (jeton) permettant de désigner un identificateur

UTF-8

Format d'encodage standardisé des caractères

Virtualisation

Technologie permettant de faire fonctionner sur une seule machine physique plusieurs systèmes d'exploitation.

Structure du document

Voici la structure de ce rapport. Un sommaire se trouvant à la page suivante permet également d'avoir une vue d'ensemble des différents chapitres.

Le premier chapitre, « **Présentation du travail** » permet de s'immerger dans le projet et d'avoir un aperçu de la réalisation du cahier des charges.

Le chapitre deux, « **État de l'art** » rassemble différentes applications allant dans le même sens que Smart Agenda.

Le chapitre trois « **Architecture** » explique l'évolution de l'architecture de l'application tout au long du projet.

Le chapitre quatre « **API utilisées** » présente brièvement les différentes API utilisées dans ce travail de bachelor.

Les chapitres cinq « **Serveur Web** », six « **Base de données** », sept « **Plateforme Web** » ainsi que huit « **Service Windows** » traitent de manière détaillée les différentes installations ainsi que le développement des parties distinctes du projet.

Le chapitre neuf « **Problèmes rencontrés** » explique les différents problèmes rencontrés ainsi que les solutions trouvées.

Le chapitre dix « **Améliorations possibles** » résume les différentes possibilités d'améliorations et de développement futur pour Smart Agenda.

Le chapitre onze « **Gestion de projet** » propose un aperçu de la façon dont le projet a été géré durant ces trois mois.

Pour terminer, le chapitre douze « **Conclusion** » donnant un feed-back sur le projet ainsi qu'un bilan personnel.

Les derniers chapitres sont composés de la bibliographie, de la webographie, de la table des illustrations et pour terminer, les différentes annexes.

Sommaire

1	PRÉSENTATION DU TRAVAIL.....	1
1.1	Contexte général.....	1
1.2	Acteurs du projet.....	3
1.3	Définition du cahier des charges.....	3
2	ÉTAT DE L'ART.....	4
2.1	Applications existantes.....	4
2.2	Apport de la création d'une nouvelle application.....	9
3	ARCHITECTURE.....	10
4	API UTILISÉES.....	13
4.1	Google Calendar API.....	13
4.2	Flux RSS des CFF.....	13
4.3	Google Weather API.....	14
5	SERVEUR WEB.....	15
5.1	Installation.....	15
6	BASE DE DONNÉES.....	16
6.1	LINQ.....	16
6.2	Script.....	17
6.3	Accès aux données.....	17
6.4	Écriture des données.....	17
7	PLATEFORME WEB.....	18
7.1	Technologies utilisées.....	18
7.2	Architecture.....	20
7.3	Fonctionnalités.....	21

8	SERVICE WINDOWS	35
8.1	Technologies utilisées	35
8.2	Fonctionnalités	36
8.3	Optimisation	36
9	PROBLÈMES RENCONTRÉS	37
9.1	Application Androïd, application web	37
9.2	Login et autorisation Google	37
9.3	Prise en main des requêtes LINQ	38
9.4	API Google Weather	38
9.5	Création du service Windows	39
9.6	Reminder Google Calendar	39
9.7	Complexité du service	39
10	AMÉLIORATIONS POSSIBLES	40
10.1	Ajout de webservice	40
10.2	Géolocalisation	41
10.3	Temps des reminders dynamiques	41
11	GESTION DE PROJET	42
11.1	Planification	42
11.2	Suivi	44
11.3	Écart de temps entre prévu et réalisé	44
11.4	Nombre d'heures réalisées	44
12	CONCLUSION	45
13	BIBLIOGRAPHIE	46



14	WEBOGRAPHIE	46
14.1	Analyse	46
14.2	Tutoriaux	47
14.3	LINQ.....	47
14.4	Développement	48
15	TABLE DES ILLUSTRATIONS	50
15.1	Code	50
15.2	Illustrations	50
15.3	Schémas	50
15.4	Smart Agenda	51
16	ANNEXES.....	52
16.1	Contenu du CD-ROM	52
16.2	Suivi de projet	53
16.3	Modèle physique de données.....	58
16.4	Schéma du service Windows.....	59
16.5	Procès-verbaux	62

1 Présentation du travail

1.1 Contexte général

Afin de compléter la formation de trois ans en informatique de gestion à la HES-SO Valais, chaque étudiant doit réaliser un travail de diplôme de 360 heures effectives durant environ trois mois en vue de l'obtention du bachelors.

Ce travail a commencé le 16 mai 2011. Vu que le sixième semestre de formation s'est terminé le 1er juillet 2011, la première partie du travail a été effectuée à un taux d'environ 50%. Depuis cette date jusqu'au rendu fixé au 16 août, le taux d'activité a été de 100%.

Le choix du sujet s'est déroulé de la manière suivante : une liste d'environ 30 travaux nous a été présentée et chaque étudiant devait choisir six travaux par ordre de préférence. Mis à part deux exceptions, chaque étudiant a reçu comme travail son premier choix, dans mon cas Smart Mobile Agenda, proposé par M. Yann Bocchi, chef de l'unité Software engineering à l'Institut de recherche de la HES-SO Valais à Sierre.

Voici les différents objectifs du projet :

- Analyse de différentes API de planification pour différentes plateformes mobiles présentes sur le marché
- Design et développement d'une application web d'agenda, basée sur des propositions d'alarmes et de l'interprétation de données intelligente
- Analyse complémentaire de cas d'utilisation liée à la problématique de la planification
- Etablir un rapport final

En quelques mots, l'application permettant aux différents agendas disponibles sur les smartphones de pouvoir interpréter des données externes et ainsi de pouvoir mettre à jour automatiquement les différentes alarmes.

Voici trois scénarios pouvant illustrer l'application :

François Morard

HES-SO Valais, août 2011

1

Scénario n° 1

L'utilisateur, habitant à Sierre en Valais, a planifié un rendez-vous pour une séance à 9h00 à Crans-Montana. Il compte s'y rendre avec le véhicule de son entreprise.

Cet utilisateur a enclenché l'alarme de son Smartphone afin de se réveiller à 7h00.

Malheureusement pour lui, une importante quantité de neige est tombée durant la nuit.

Grâce à l'application Smart Mobile Agenda, l'utilisateur s'est levé une heure plus tôt que prévu. En effet, durant la nuit, l'application a récupéré des informations météo et a ainsi mis à jour l'alarme de l'utilisateur.

Ce dernier a donc pu arriver à l'heure à son rendez-vous, malgré les intempéries.

Scénario n° 2

Un second utilisateur, habitant lui aussi à Sierre, a agendé une séance de travail à Genève pour son entreprise à 13h00. Afin de pouvoir préparer les derniers détails de la séance, il s'y rendra en train.

Il a programmé un rappel à 10h00 afin de ne pas oublier son rendez-vous et arriver à l'heure.

Comble de malchance, tous les trains CFF en direction de Genève circulent avec une heure de retard.

Heureusement que cette personne utilisait l'application Smart Mobile Agenda. En effet, cette dernière a récupéré les informations actuelles des CFF et l'alarme prévue par l'utilisateur a été avancée d'une heure.

Grâce à ces informations, l'utilisateur est ainsi arrivé à l'heure à son rendez-vous malgré les différents problèmes des CFF.

Scénario n° 3

Un troisième et dernier utilisateur, habitant lui aussi à Sierre, a rendez-vous chez le dentiste à Sion à 8h00.

Son réveil sur son Smartphone est planifié à 7h00. Vu que le dentiste n'est pas très loin de son domicile, il compte s'y rendre avec son véhicule privé.

L'utilisateur n'a cependant pas pensé que le jour de son rendez-vous était le même que celui du Slow-up ! La circulation entre Sierre et Sion est fortement ralentie.

Cependant, l'application Smart Mobile Agenda s'est connecté à un Webservice traitant des manifestations et l'alarme a ainsi été avancée d'une heure.

Cet utilisateur a donc pu arriver à l'heure à son rendez-vous.

1.2 Acteurs du projet

Voici les différents acteurs du projet:

- M. Yann Bocchi
Professeur responsable du suivi du travail de bachelor
Head of Software Engineering Unit
Institute of Business Information Systems
yann.bocchi@hevs.ch
- M. François Morard
Etudiant réalisateur du projet
morard.francois@gmail.com

1.3 Définition du cahier des charges

Durant les deux premières semaines de travail, chaque étudiant a du réaliser un cahier des charges afin de définir les différentes tâches à effectuer durant les trois prochains mois.

Le cahier des charges a été accepté par M. Yann Bocchi lors de la séance du 7 juin 2011. Il est disponible sur le CD-ROM ci-joint.

2 État de l'art

Afin de ne pas réinventer de l'existant, voici quelques applications existantes correspondantes à "agenda intelligent":

2.1 Applications existantes

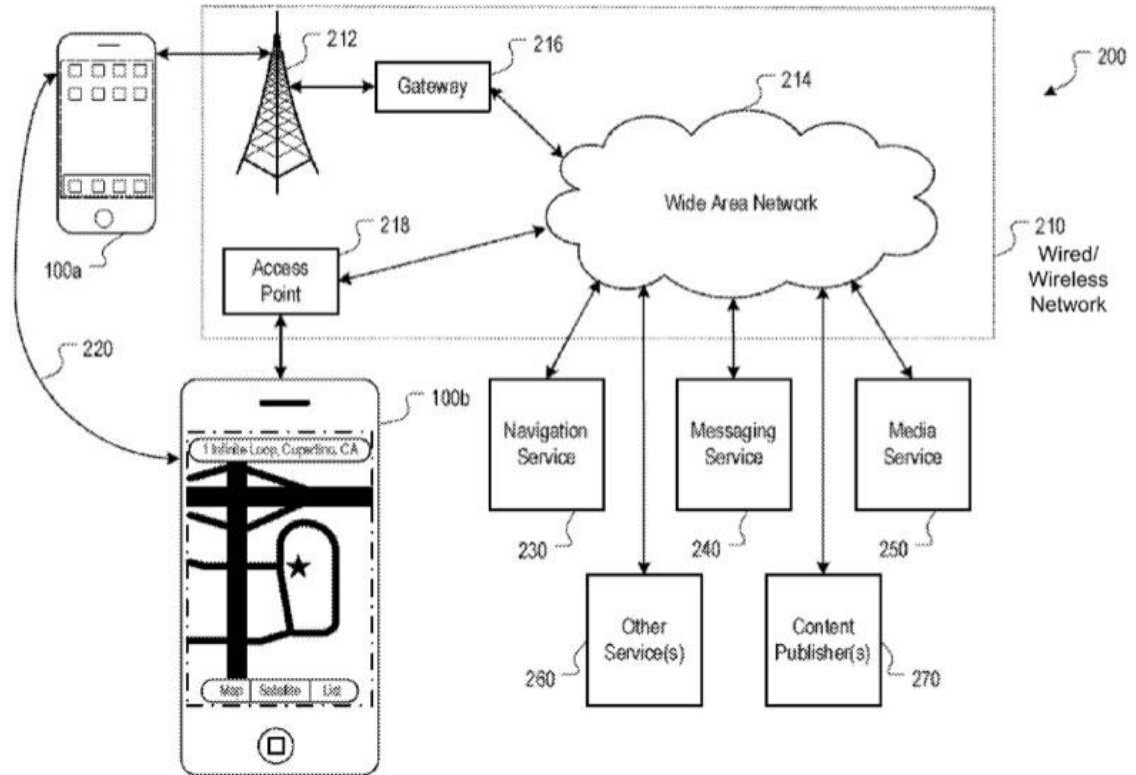
« Apple, une alarme vraiment intelligente »

Apple a déposé le 2 juin 2011 un brevet concernant un système d'alarmes intelligentes géolocalisées. Voici le concept:

Au lieu de déclencher une alarme à un moment donné, cette dernière est déclenchée au passage d'un lieu précis. Par exemple à la place d'avoir un rappel à 17h pour aller chercher du pain, l'alarme s'actionnera lors d'un passage à côté d'une boulangerie.

Il existe déjà quelques autres solutions similaires, telle que Place Clock pour iPhone par exemple. Par contre, ces applications sont moins intelligentes que celle d'Apple.

Illustration 1 : Exemple de schéma



Source : <http://www.igeneration.fr>

Apple Reminders: « A better way to do to-dos »

Cette application, pour iPhone, iPad ou iPod touch, permet d'organiser la vie de l'utilisateur sous forme de « to-do list » avec dates et emplacements. Le calendrier peut fonctionner en mode temporel ou en mode géolocalisation.

Par exemple, si l'utilisateur doit acheter du lait et qu'il passe devant un magasin d'alimentation, son alarme s'activera, puisque Reminders permet de localiser l'utilisateur.

Reminders peut également fonctionner avec iCal, Outlook et iCloud. Ainsi, tous les changements seront synchronisés avec les autres appareils et calendriers.

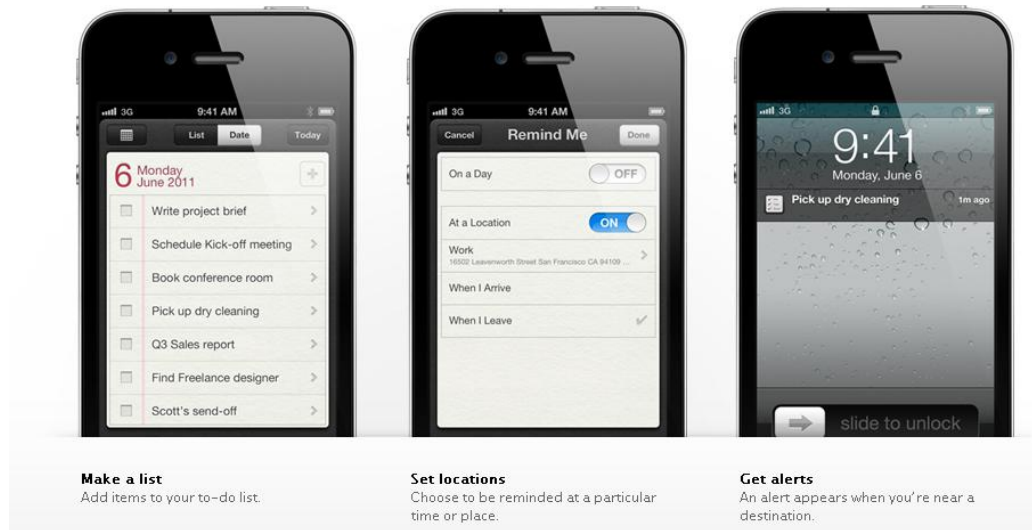
Voici quelques images de son utilisation:

Illustration 2 : Reminders (Vue par liste)



Source : <http://www.apple.com/>

Illustration 3 : Reminders (Vue par emplacement)



Source : <http://www.apple.com/>

Alarm Pills : « Le carnet de santé intelligent sur votre iPhone ! »

Le but principal de cette application est d'avoir, quelles que soient les conditions (pas de réseau, pas de connexion internet, etc.), toujours des alarmes actives.

En effet, une personne malade ne peut en aucun cas louper la prise de ses médicaments.

Alarm Pills est une nouvelle application pour iPhone permettant de programmer un nombre illimité d'alarmes même pour les traitements les plus complexes.

Voici les différentes fonctionnalités:

- Programmation d'un nombre illimité d'alarmes
- Personnalisation des alarmes en fonction de la posologie, des heures de prises, de la durée du traitement
- Possibilité de définir un temps de pause durant un traitement
- Visualisation des alarmes sur le calendrier de l'application (Carnet de santé)
- Synchronisation des alarmes avec le calendrier de l'iPhone

Meeting Agent: « Rendez votre agenda intelligent »

L'application Android Meeting Agent se charge de modifier l'état de la sonnerie de Smartphone en fonction des différents rendez-vous planifiés.

Une fois les différents rendez-vous saisis sur votre agenda, le Smartphone se mettra automatiquement en mode silence lors de vos réunions.

Une fonctionnalité intéressante est aussi disponible: BackSilent.

Il suffit de retourner le Smartphone face contre bureau afin de passer le smartphone en mode silencieux.



2.2 Apport de la création d'une nouvelle application

Après quelques recherches sur internet, nous pouvons constater qu'il n'existe actuellement aucune application correspondant à l'application que nous souhaitons développer.

Beaucoup d'application utilisent les termes "Alarmes intelligentes" ou "Agenda intelligent". Mis à part le brevet déposé par Apple, nous pouvons considérer qu'aucune de ces applications ne sont vraiment "intelligentes" à notre sens. Elles n'ont pas vraiment d'interaction avec le monde extérieur.

Notre application se concentrerait justement sur les interactions entre les différentes alarmes programmées et les événements pouvant se produire tous les jours et nous poser problème en nous mettant en retard pour un rendez-vous.

3 Architecture

Lors de la première semaine de travail, il était décidé de reprendre une application fonctionnant sous Android (politique moins restrictive que sur iPhone) et de l'améliorer en l'interconnectant avec différents services.

Voici un schéma de la première idée d'architecture :

Schéma 1 : Architecture version 1



Après quelques jours de recherches infructueuses quant à la recherche d'applications calendrier existantes, il fut décidé, d'un commun accord avec M. Yann Bocchi de développer une interface web permettant de configurer les différents services voulus par l'utilisateur ainsi que les différentes règles.

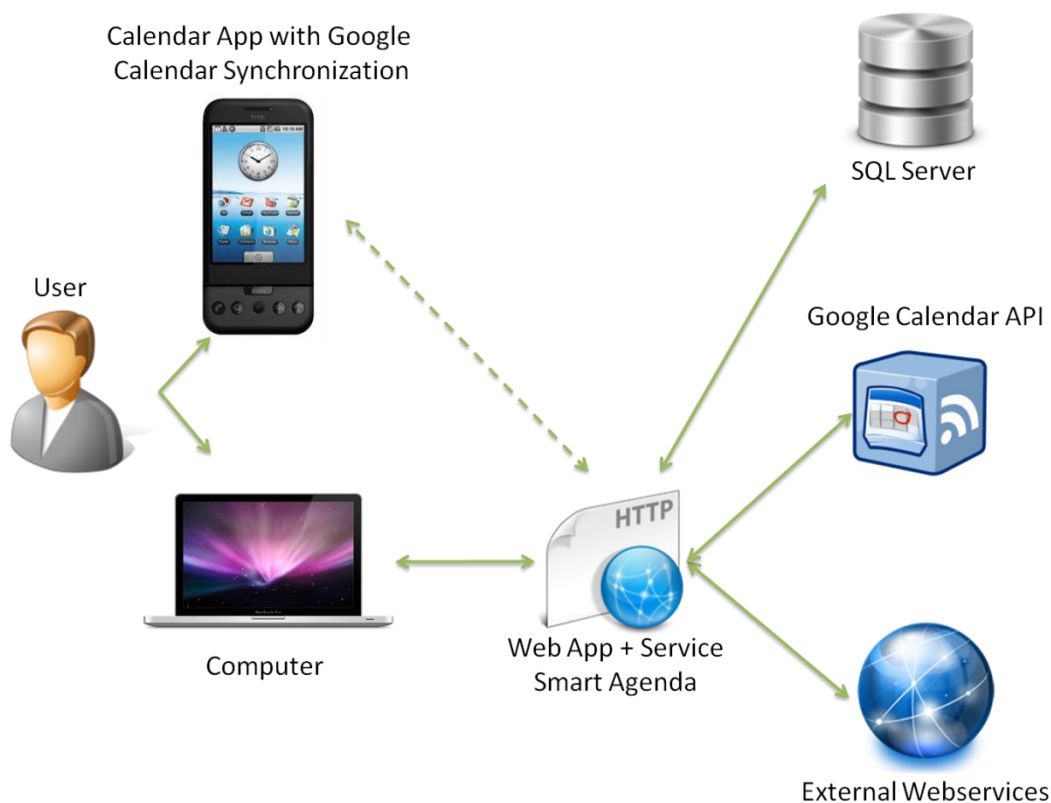
Afin de toucher un plus grand nombre d'utilisateurs potentiels, le choix s'est porté sur l'utilisation du calendrier de Google.

En effet, un utilisateur Apple (iPhone, iPad, etc.) peut tout aussi bien utiliser l'application qu'un utilisateur Android (smartphones HTC, tablette Samsung, etc.). Le calendrier natif de tous ces appareils peut se synchroniser avec Google Calendar.

De plus, aucune installation n'est nécessaire. Les habitudes des utilisateurs restent également inchangées. Ils rentreront toujours de la même manière les différents événements dans leur calendrier, que ce soit directement sur leur smartphone ou sur l'interface de Google Calendar.

Afin d'éclaircir ces différents propos, voici une illustration de l'architecture :

Schéma 2 : Architecture version 2





Voici une description de l'architecture :

L'utilisateur utilise un smartphone (Androïd, iPhone, etc.) possédant un calendrier pouvant être synchronisé avec Google Agenda.

Il désire rendre son agenda « intelligent ». (Pouvant adapter les notifications en fonction d'éléments externes tels que météo, transports public, etc.)

Pour cela, l'utilisateur se connecte à l'application web « Smart Agenda », depuis son smartphone ou son ordinateur. Il peut ensuite configurer les web-services qu'il désire utiliser ainsi que différentes options concernant les alarmes.

Un service tournant en arrière-plan à intervalle régulier communique par la suite avec les différents webservices et adapte l'alarme des événements en fonction des options choisies auparavant.

Le smartphone doit avoir accès à internet avant de pouvoir se synchroniser avec Google Agenda.

4 API utilisées

Afin de pouvoir travailler sur Google Calendar, il a fallu analyser son API.

Pour rendre Smart Agenda intelligent, il a également fallu le faire interagir avec d'autres webservices externes. En voici une brève description.

4.1 Google Calendar API

L'API Google Calendar met à disposition des programmeurs, quasiment toutes les opérations disponibles sur l'interface web de Google Calendar, telles que créer des événements, les modifier ou supprimer. Il est également possible d'accéder à des calendriers privés en utilisant une autorisation au préalable.

Selon les dires de Google, cette API est destinée à des développeurs ayant déjà de l'expérience en programmation.

La version 2 de l'API est disponible pour le framework .NET, Java et Python.

4.2 Flux RSS des CFF

Ne trouvant aucune API à proprement parler pour le service ferroviaire des CFF, une solution a été trouvée en récupérant directement le flux RSS des perturbations du réseau.

En effet, un flux RSS n'est rien d'autre qu'un fichier XML structuré.

Il suffit donc de développer un parser permettant de récupérer les informations nécessaires.

4.3 Google Weather API

Google « Secret » Weather API n'est nullement documentée. Ce service a été créé afin de l'utiliser avec iGoogle. Cette interface a facilement été découverte.

Il suffit d'interroger l'url suivante en y ajoutant la localité désirée :

<http://www.google.com/ig/api?weather=lausanne>

L'API renvoie tout simplement une réponse XML qu'il ne reste plus qu'à parser.

En voici un extrait :

Code 1 : Réponse XML de Google Weather API

```
-<xml_api_reply version="1">
- <weather module_id="0" tab_id="0" mobile_row="0" mobile_zipped="1" row="0" section="0">
- <forecast_information>
  <city data="Lausanne, Vaud"/>
  <postal_code data="lausanne"/>
  <latitude_e6 data=""/>
  <longitude_e6 data=""/>
  <forecast_date data="2011-08-06"/>
  <current_date_time data="2011-08-05 23:50:00 +0000"/>
  <unit_system data="SI"/>
</forecast_information>
- <current_conditions>
  <condition data="Temps clair"/>
  <temp_f data="63"/>
  <temp_c data="17"/>
  <humidity data="Humidité : 94 %"/>
  <icon data="/ig/images/weather/sunny.gif"/>
  <wind_condition data="Vent : N à 2 km/h"/>
</current_conditions>
- <forecast_conditions>
  <day_of_week data="sam."/>
  <low data="16"/>
  <high data="26"/>
  <icon data="/ig/images/weather/chance_of_rain.gif"/>
  <condition data="Risques de pluie"/>
</forecast_conditions>
```

5 Serveur Web

Afin de pouvoir héberger la plateforme web, il est nécessaire d'avoir un serveur web à disposition. Dans le cadre de ce projet, un serveur dédié a été installé.

5.1 Installation

La première étape consistait à récupérer une image virtuelle d'un serveur Windows 2008. Le choix de travailler avec la virtualisation se justifie par le fait que le serveur pourra être déplacé sur n'importe quelle machine physique, ce dans le but de faciliter la suite du développement de l'application et d'en assurer sa pérennité.

Une fois l'image virtuelle récupérée, il fallut installer différents services :

Pour commencer, il a fallu installer un serveur HTTP. Vu que nous travaillons sous Windows, le service « Internet Information Services » (IIS) a été installé dans sa version 7.

Illustration 4 : IIS 7



Source : <http://www.deffeuille-auto.fr/>

Illustration 5 : SQL Server 2008



Source : <http://danstoncloud.com>

Afin de pouvoir stocker les données des utilisateurs de la plateforme web, il faut également installer un système de gestion de base de données (SGBD). Le service SQL Server 2008 de Windows a également été installé.

Finalement, afin de transférer les fichiers de la plateforme web du poste de travail au serveur, un serveur FTP basique a été installé : TYPsoft FTP.

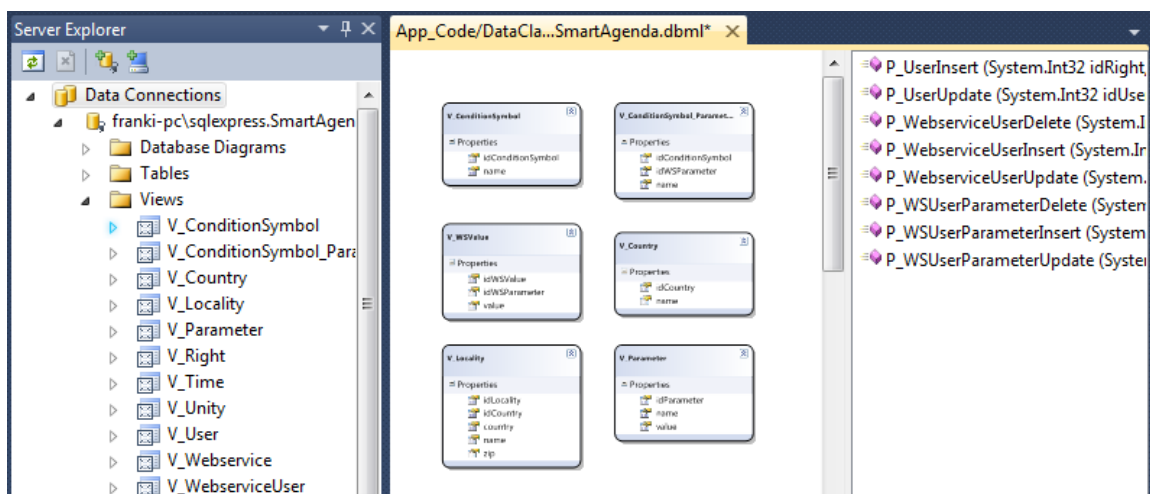
6 Base de données

6.1 LINQ

LINQ (Language Integrated Query) est une nouveauté de Visual Studio 2008 (Extension des langages C# et Visual Basic). Cet ensemble de fonctionnalités permet d'effectuer des requêtes et des mises à jour de données de façon simple. Il permet de mapper (établir une correspondance) une base de données relationnelle en un modèle objet.

Il suffit, depuis l'explorateur de base de données, de glisser les différentes tables, vues ou procédures stockées dans l'interface LINQ. Une fois les modifications effectuées, chaque objet de la base de données est alors accessible tel un objet standard (par exemple String, Datetime, etc.)

Illustration 6 : Fenêtre LINQ (Visual Studio 2010)



Sur ce schéma, nous retrouvons tout à gauche l'explorateur de base de données, au centre les différentes tables et vues ajoutées à LINQ et pour terminer, à droite les procédures stockées également ajoutées à LINQ.

6.2 Script

Le développement de la base de données a été effectué sous forme de script. Voici quelques avantages de cette façon de procéder :

- Permet de conserver une sauvegarde (très légère vu que le script est uniquement un fichier texte)
- Permet en quelques minutes de créer une seconde base de données identique (par exemple pour un environnement de test ou de développement)

6.3 Accès aux données

Tous les accès en lecture de la base de données sont effectués sous la forme de vues. Une vue représente une requête prédéfinie sur une ou plusieurs tables. Elle est un objet à part entière de la base de données.

Une vue permet entre autre de restreindre l'accès à des données sensibles en récupérant uniquement les données appropriées. Elle permet également au développeur de travailler sur des requêtes plus simples.

6.4 Écriture des données

Afin d'empêcher l'accès aux tables et aux vues en écriture, toutes ces opérations passent par des procédures stockées. Cela se justifie d'une part par la diminution du trafic réseau (un seul paquet) et d'autre part d'un point de vue de la sécurité.

7 Plateforme Web

La première partie de programmation de ce travail de bachelor a été la plateforme web. Voici les points les plus importants.

7.1 Technologies utilisées

Framework .NET

Afin d'utiliser la dernière version de l'API de Google Calendar, le Framework .NET a été choisi afin de développer les différentes pages HTML de la plateforme Web.

Il existe deux langages pour développer en .NET :

- C# (C Sharp)
- Visual Basic

Vu que durant notre formation à l'HES-SO nous avons beaucoup travaillé avec Java, la décision de choisir le C# a été retenue, étant donné que ce dernier ressemble beaucoup au Java.

Illustration 7 : Logo Visual Studio 2010



Source : <http://emea.microsoftstore.com>

L'outil Visual Studio 2010 a été installé et le développement s'est fait uniquement avec ce dernier. Microsoft Visual Studio est vraiment très complet et permet de gérer quasiment tous les cas de figures.

CSS

Les feuilles de styles CSS (Cascading Style Sheets) permettent de séparer la structure d'une page HTML de sa couche de présentation. Elles sont utilisées afin d'alléger le code HTML de la page mais également pour permettre au développeur de centraliser toute l'apparence du site dans un seul fichier.

Il en résulte que lors d'un changement de design, seul le fichier CSS est modifié.

En voici un petit extrait pour la balise HTML h1 (Titre 1) :

Nous définissons une marge de 0, une couleur de texte blanche et une police normale de 40px en Arial, Helvetica, sans-serif.

Code 2 : Exemple CSS

```
h1 {  
  margin:0;  
  color:white;  
  font:normal 40px Arial, Helvetica, sans-serif;  
}
```

AnkhSVN

Tout d'abord, qu'est-ce qu'un SVN (Subversion)?

Ce n'est ni plus ni moins qu'un gestionnaire de version. Nous en avons déjà utilisés dans nos précédents projets de groupe lors de notre formation. Cet outil permet à deux ou plusieurs personnes de travailler sur le même fichier d'un projet. Il n'y a donc plus de problème de versioning comme avec l'échange de mails par exemple.

Dans le cadre de ce projet, SVN a surtout été utilisé dans un but de sauvegarde et de récupération de versions antérieures de fichiers.

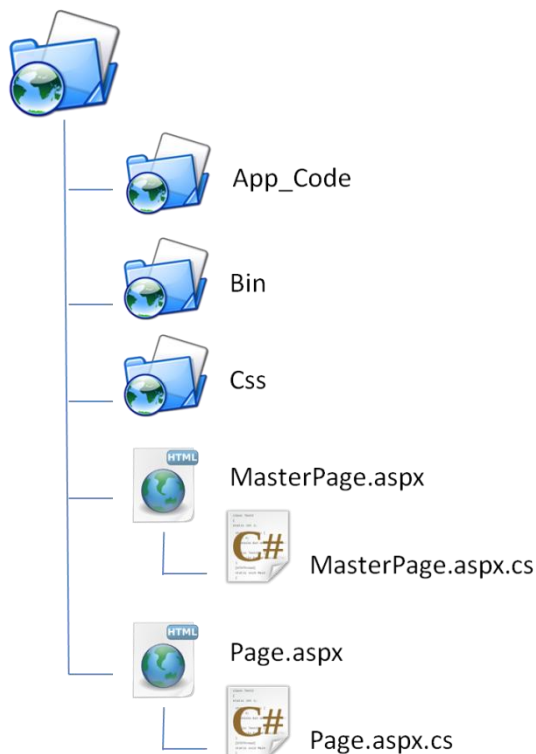
AnkhSVN est simplement un outil permettant d'utiliser Subversion directement dans Visual Studio.

Le serveur SVN utilisé dans ce projet est hébergé par ProjectLocker (<http://www.projectlocker.com/>) gratuitement.

7.2 Architecture

Voici la structure multicouche de la plateforme Web :

Schéma 3 : Architecture Plateforme Web



Le dossier « App_Code » contient différentes classes métier de haut niveau. Il contient également le fichier LINQ de persistance des données.

Le dossier « Bin » contient les différentes bibliothèques nécessaires au fonctionnement de l'application, telles que les extensions GData de Google pour le Calendrier, le contrôle d'accès, le client et divers autres extensions. Il contient également l'ASP.NET Ajax Control Toolkit.

Le dossier « Css » contient les différentes feuilles de styles.

Viennent ensuite les différentes pages de l'application. Là aussi une séparation est effectuée entre la présentation et le contrôleur, situés dans deux fichiers différents.

7.3 Fonctionnalités

À travers ce chapitre, vous pouvez découvrir les différentes fonctionnalités de Smart Agenda dans l'ordre chronologique de leur utilisation.

Création d'un compte Smart Agenda

Afin de pouvoir utiliser Smart Agenda, chaque utilisateur doit se créer un compte personnel.

Voici une illustration du formulaire d'inscription :

Smart Agenda 1 : Formulaire d'inscription

The image shows a 'Sign Up' form with the following fields and values:

Field	Value
Lastname *	François
Firstname *	Morard
Address *	Tsan de la Pîrra 17
Country *	Switzerland
Locality *	Crans-Montana
Email *	morard.francois@gmail.com
Password *	••••
Conf. Password*	••••

Buttons: Sign Up !, Back

Tous les champs sont obligatoires. La liste déroulante de la localité se charge de façon asynchrone grâce à la technologie Ajax en fonction du pays choisi. Plusieurs tests sont effectués : par exemple la validité de l'adresse email (expression régulière), l'unicité de l'adresse email (utilisation comme login), etc. Le bouton Sign up permet de valider l'inscription.

Login

Une fois l'utilisateur inscrit, il peut se connecter avec son adresse email et son login. Afin d'éviter de se connecter à chaque visite, il peut également rester connecté (génération d'un cookie).

Smart Agenda 2 : Menus

Login Menu

Login

Password

Stay connected

> Create an account

Config. Menu

- > Home
-
- > Configuration
-
- > Simulation
-
- > Launch service
-
- > Calendar
-
- > Weather
-
- > CFF
-
- > Sign out
-

Une fois connecté, le menu de configuration apparaît en lieu et place du menu de login.

Autorisation Google Calendar

Lors de la première connexion à Smart Agenda, l'application demandera à l'utilisateur de lui autoriser l'accès au calendrier Google.


Smart Agenda 3 : Autorisation Google Calendar

Google Calendar Authorization

To use the SmartAgenda application, you have to authorize the connection with your calendar.

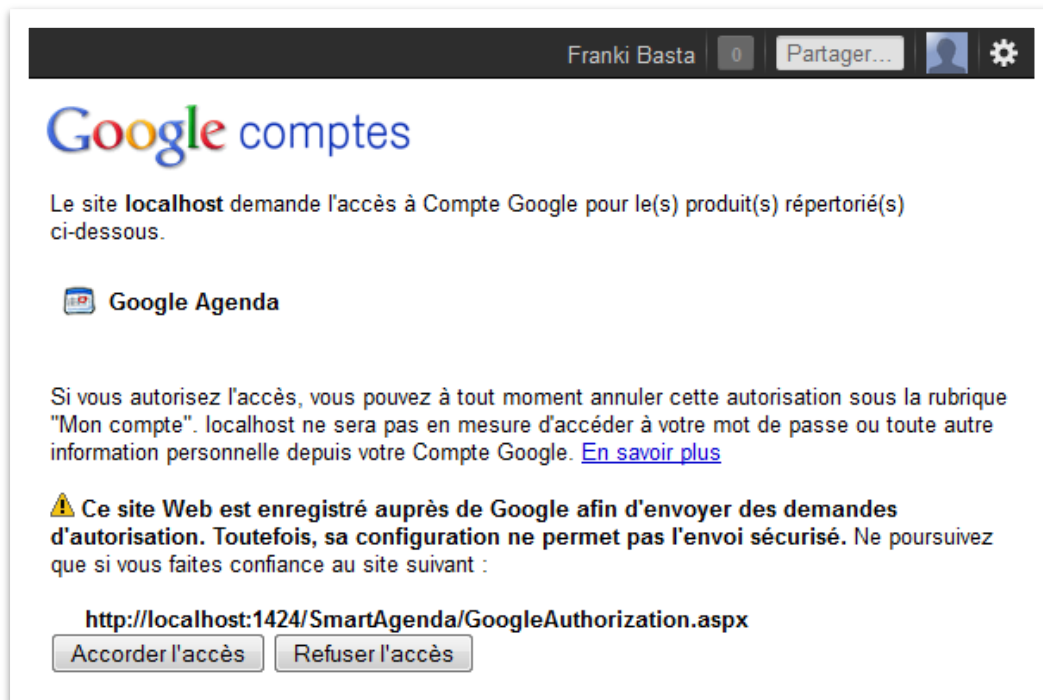
If you allow access, you can always cancel this authorization under "My Account" tab. SmartAgenda will not be able to access your password or other personal information from you Google Account

[Google Login](#)



L'utilisateur se connecte par la suite à son compte Google comme à son habitude. Il doit ensuite accorder l'accès.

Smart Agenda 4 : Autorisation Google Calendar 2



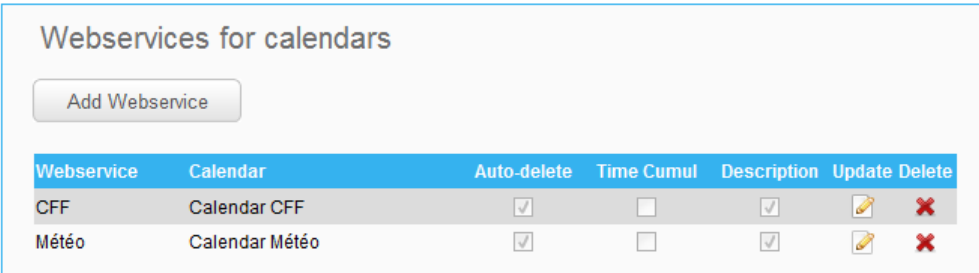
Une fois cette opération effectuée, l'utilisateur a accès à toutes les fonctionnalités de Smart Agenda





Configuration des webservice par calendriers

Il est possible que l'utilisateur, dans Google Calendar, possède plusieurs calendriers.

Smart Agenda tient compte de cette option et offre à l'utilisateur la possibilité de configurer un ou plusieurs services (Météo, CFF) pour chacun de ces calendriers.

Smart Agenda 5 : Configuration d'un webservice pour un calendrier



Webservice	Calendar	Auto-delete	Time Cumul	Description	Update	Delete
CFF	Calendar CFF	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
Météo	Calendar Météo	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

Lors de l'ajout ou de la modification d'un webservice pour un calendrier, l'utilisateur peut choisir trois options supplémentaires.

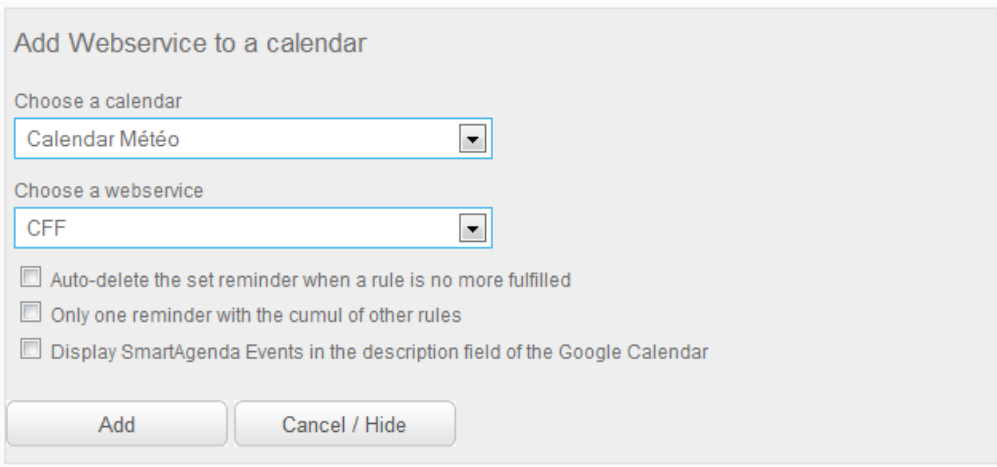
- Suppression automatique d'une alarme ajoutée par l'application Smart Agenda dont la règle n'est plus respectée. Par exemple si un reminder est ajouté à cause de la neige et que plus tard il arrête de neiger, celui-ci sera supprimé automatiquement.
- Cumul des temps des alarmes. Smart Agenda n'ajoutera qu'un seul reminder pour toutes les règles respectées avec le cumul de toutes les durées. Par exemple si un train est en retard, une alarme de dix minutes sera ajoutée à l'événement du calendrier. Si en plus de cela la neige tombe, vingt minutes seront ajoutées au reminder existant.
- Description : Insère dans le champ description de l'événement toutes les actions effectuées par Smart Agenda, telles que l'ajout ou la suppression d'un reminder. Cela permet à l'utilisateur d'avoir un suivi de ce qui s'est passé.

Voici le formulaire d'ajout d'un webservice pour un calendrier.

L'utilisateur doit en premier lieu choisir un de ses calendriers parmi la liste de ses agendas.

Une fois sélectionné, il peut choisir le service désiré. (Seuls les services disponibles et non utilisés par le calendrier sélectionné s'affichent)

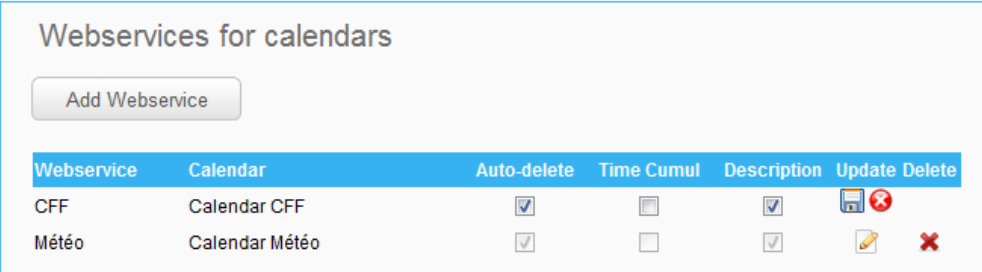
Smart Agenda 6 : Ajout d'un service pour un calendrier







Si l'utilisateur décide par la suite de modifier une des trois options disponibles, il peut le faire à tout moment en cliquant sur l'icône éditer.

Il peut également supprimer un service pour un calendrier s'il le désire.

Smart Agenda 7 : Modification d'un service pour un calendrier



Webservice	Calendar	Auto-delete	Time Cumul	Description	Update	Delete
CFF	Calendar CFF	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		
Météo	Calendar Météo	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>		

Configuration des règles

Dans la seconde partie de la configuration, l'utilisateur peut sélectionner un des webservice qu'il a, au préalable, associé à un ou plusieurs de ses calendriers.







Une fois sélectionné, les différentes règles définies par l'utilisateur s'affichent. Il peut en ajouter de nouvelles, éditer ou supprimer les existantes.

Smart Agenda 8 : Affichage des règles

Webservice reminder configuration

Choose a webservice

Météo ▼

Parameter	Symbol	Value	Reminder	Edit	Delete
Temperature	<	10°	5 minutes		
Humidity	>	99%	10 minutes		
Condition	=	Snow	1 hour		

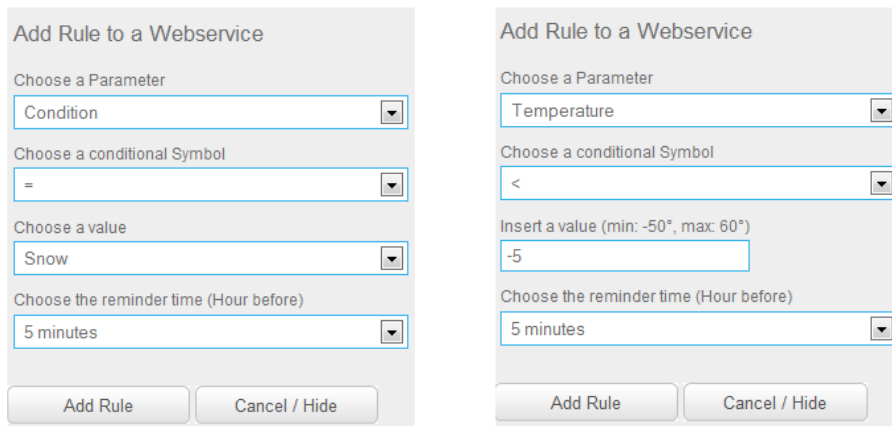
Chaque paramètre a un ou plusieurs symboles de comparaison associés (plus petit que [$<$], égal [=], plus grand que [$>$])

Il existe deux types de paramètres :

- Ceux dont la valeur peut être sélectionnée dans une liste existante, comme par exemple les différentes conditions météo
- Ceux dont la valeur est entrée par l'utilisateur, par exemple la température (comprise entre une valeur minimum et maximum)

En dernier lieu, il faut choisir combien de temps avant l'événement, l'alarme doit être fixée si la règle est respectée.

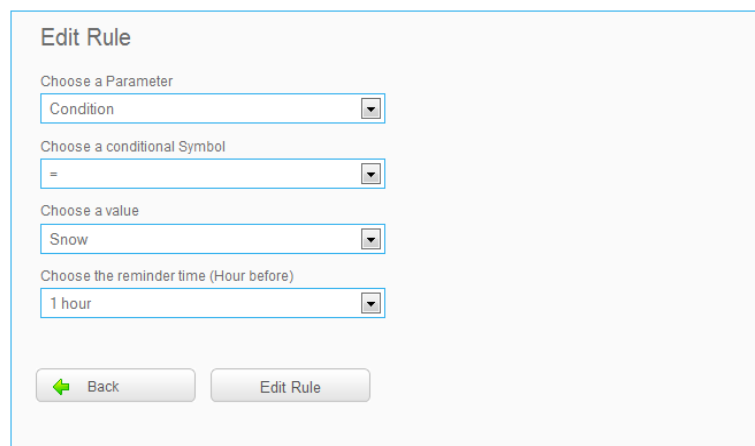
Smart Agenda 9 : Ajout d'une règle



The image shows two side-by-side screenshots of the 'Add Rule to a Webservice' form. Both forms have the same layout with four main sections: 'Choose a Parameter', 'Choose a conditional Symbol', 'Choose a value', and 'Choose the reminder time (Hour before)'. At the bottom of each form are 'Add Rule' and 'Cancel / Hide' buttons.

- Left Screenshot:**
 - Choose a Parameter: Condition
 - Choose a conditional Symbol: =
 - Choose a value: Snow
 - Choose the reminder time (Hour before): 5 minutes
- Right Screenshot:**
 - Choose a Parameter: Temperature
 - Choose a conditional Symbol: <
 - Insert a value (min: -50°, max: 60°): -5
 - Choose the reminder time (Hour before): 5 minutes

Smart Agenda 10 : Modification d'une règle



The image shows a screenshot of the 'Edit Rule' form. It has the same layout as the 'Add Rule' forms, with four main sections and two buttons at the bottom: 'Back' and 'Edit Rule'.

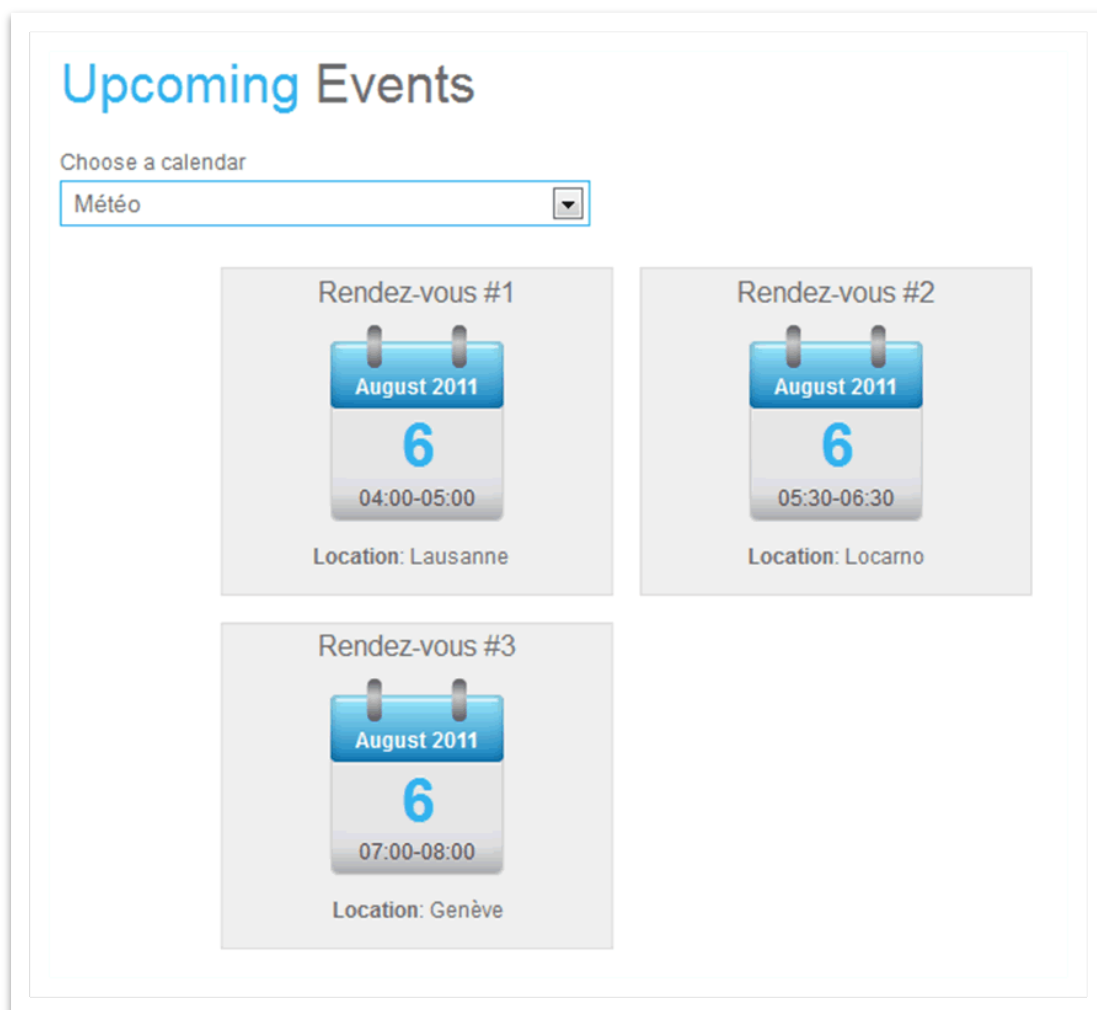
- Choose a Parameter: Condition
- Choose a conditional Symbol: =
- Choose a value: Snow
- Choose the reminder time (Hour before): 1 hour

Prochains événements

En cliquant sur le menu calendrier, l'utilisateur peut consulter ses prochains événements (dans les 60 prochaines heures)

Il sélectionne tout d'abord le calendrier désiré. Les événements s'affichent alors.

Smart Agenda 11 : Prochains événements



L'image du calendrier avec le mois, la date et l'heure a été réalisée grâce aux feuilles de styles CSS.

En cliquant sur l'événement désiré, l'utilisateur accède aux détails de ce dernier.

Il a la possibilité d'éditer tous les champs et d'ajouter ou supprimer des alarmes.

Smart Agenda 12 : Modification d'un événement

Event Update

Title

Location

From...to

 to

Reminders +

Add Reminder

Email Hours

Description

Type	Time	Unity	Delete
alert	10	Minutes	✘

En cliquant sur le champ date, un calendrier, réalisé grâce à l'Ajax Control Toolkit, apparaît et l'utilisateur peut alors sélectionner la date désirée. Le champ est en lecture seule. Cela permet d'éviter les fautes de frappe de l'utilisateur.

Simulation

Afin de pouvoir tester l'application Smart Agenda, une interface de simulation a été développée.

Elle permet aux développeurs de manipuler les résultats des différents web-services grâce à des variables de session.

Cette interface permet de manipuler les trois paramètres de la météo qui sont les conditions actuelles, la température, l'humidité ainsi que les retards pour le service des CFF.

Cette fonctionnalité n'est bien évidemment disponible que sur la version test de l'application.

Smart Agenda 13 : Simulation

Simulation

Weather Simulation

Condition
Enable Snow

Temperature
-12 Disable Temp.

Humidity
Enable Humid.

CFF Simulation

Delay
Enable Delay

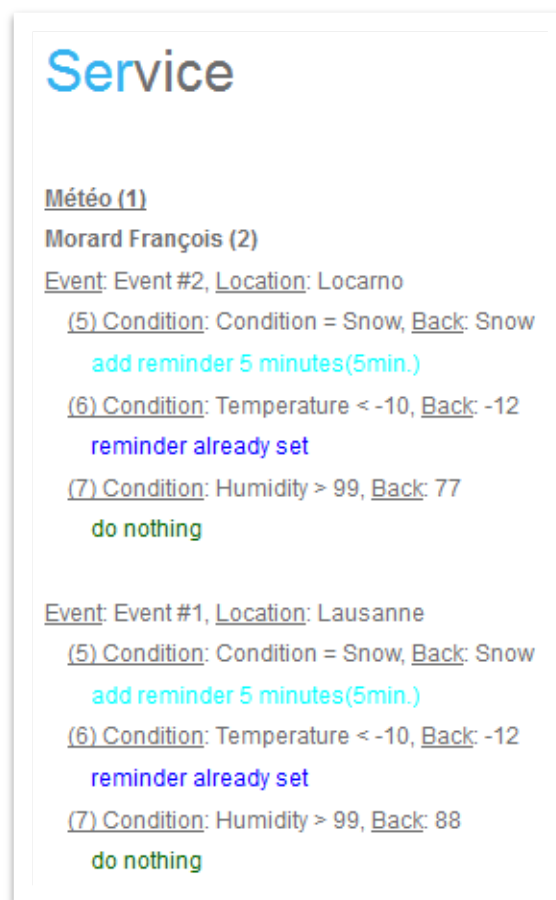
Service

Toujours dans une optique de développement et de test, une interface permettant de lancer manuellement le service a été développée. (Le détail du développement du service se trouve au point 8)

Elle permet d'afficher toutes les actions qui sont effectuées avec une justification.

Voici une partie d'un exemple :

Smart Agenda 14 : Résultat du service



La valeur désignée sous Back correspond à la valeur retour du webservice. Dans ce cas précis, nous pouvons constater que la simulation a été utilisée.

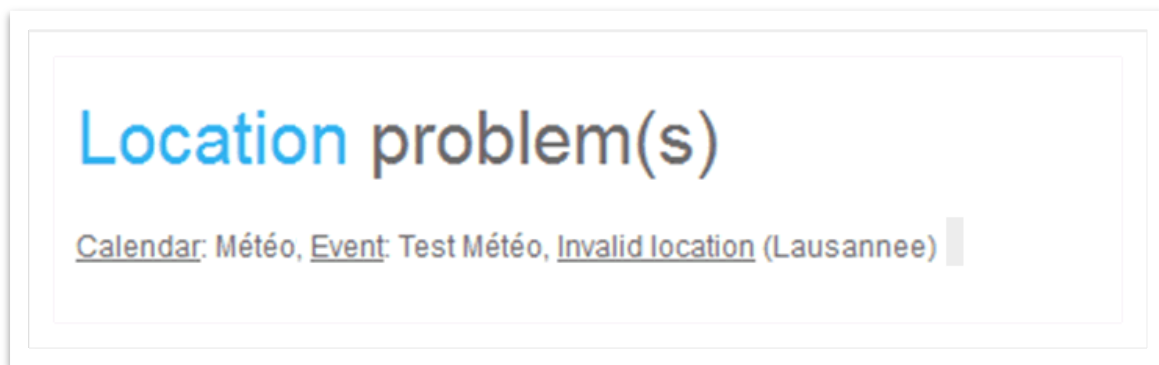
Page d'accueil (Contrôle des localités)

Les requêtes vers certains services se basent sur la localité (par exemple pour la météo).

Il est possible que la localité entrée par l'utilisateur dans le calendrier Google ne soit pas trouvée (petits villages ou faute de frappe).

Afin d'informer l'utilisateur, la page d'accueil de Smart Agenda affiche de telles erreurs.

Smart Agenda 15 : Problème de localité



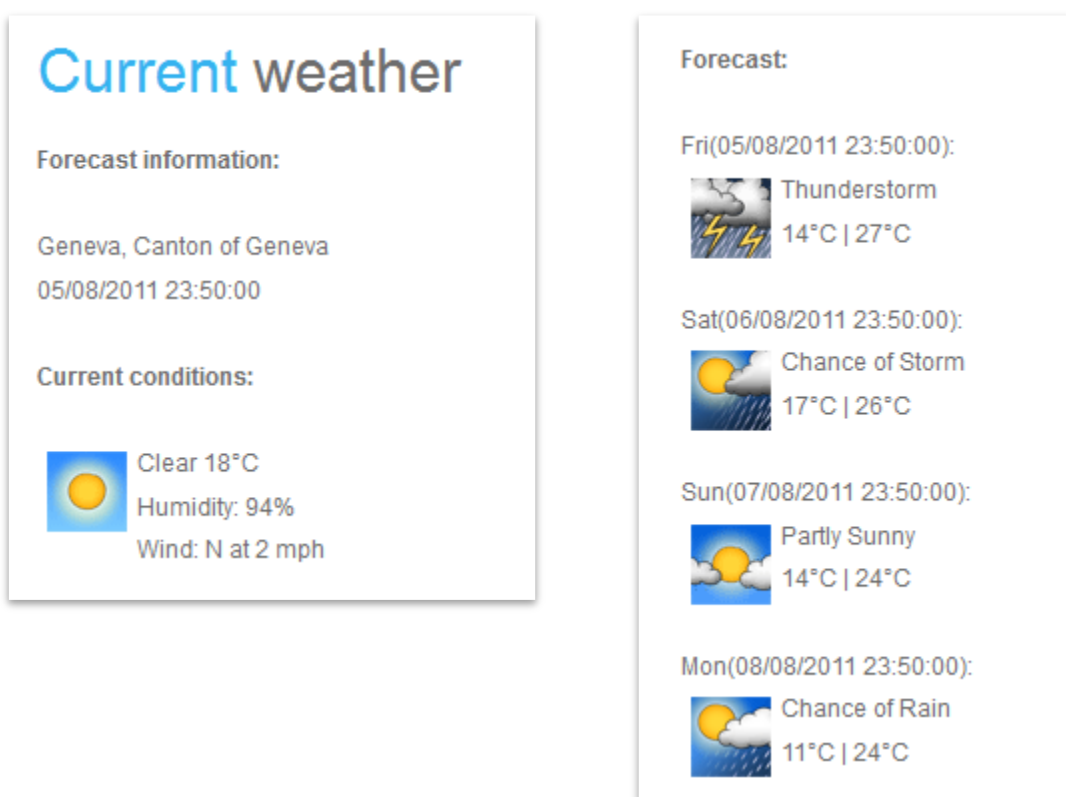
Dans ce cas précis, la localité entrée dans l'événement « Test Météo » qui se trouve dans le calendrier « Météo » est invalide (Faute de frappe)

Chaque fois que l'utilisateur navigue sur l'application Smart Agenda, il est informé des erreurs et peut les corriger via l'interface calendrier disponible ou via Google Calendar.

Météo

Durant la phase initiale du projet, une interface permettant de récupérer la météo actuelle d'une ville ainsi que les prévisions à trois jours a été développée dans le but de tester l'API Google Weather.

Smart Agenda 16 : Météo actuelle et prévisions



Elle a été gardée à des fins de test ou pour une adaptation ultérieure.

CFF

Egalement développée afin de tester le service des CFF, cette interface a été conservée afin d'avoir un aperçu de tous les problèmes actuels sur les lignes ferroviaires suisses et frontalières.

Smart Agenda 17 : Informations sur le trafic ferroviaire

Rail traffic information

SBB CFF FFS News-Feeds

Sat, 06 Aug 2011 00:25:31 +0200

Problems

Construction work: TGV 9274

Fri, 05 Aug 2011 22:12:14 +0200

Construction work: TGV 9274 from Lausanne departure 19:17 via Vallorbe 19:57 - Frasné 20:14 - Mouchard 20:49 - Dole-Ville 21:07 - Dijon Ville 21:32 arrival Paris-Gare-de-Lyon at 23:15 is cancelled.

Please switch to existing train connections.

Duration of construction work 10.11.2011.

8 Service Windows

Une fois la partie web correspondant à la configuration utilisateur de l'application terminée, il fallait alors développer le service permettant de contrôler les règles et ainsi de mettre à jour automatiquement les différentes alarmes de l'agenda.

8.1 Technologies utilisées

Le fonctionnement principal de ce service est en arrière-plan. Il a également une durée d'exécution indéterminée et ne nécessite aucune interface utilisateur.

Vu qu'un serveur Windows à entière disposition est déjà installé, le choix de la technologie s'est porté sur un service Microsoft Windows.

En effet, ce type de technologie convient parfaitement aux pré-requis. De plus, il peut également être démarré automatiquement avec le système d'exploitation. Ceci peut être un avantage si la machine subit un problème et redémarre automatiquement. Le service sera également relancé.

Le développement de ce service peut également se faire à l'aide de Visual Studio 2010 ainsi qu'avec le langage C#.

Le développement d'un service n'est pas comparable à celui d'une application standard. Un service ne peut pas être lancé dans Visual Studio 2010. Il faut tout d'abord compiler le code en un fichier exécutable et ensuite l'installer sur la machine. Le débogage n'est donc pas aisé.

N'ayant jamais développé de tels services, nous avons trouvé intéressant d'en développer un durant ce travail de bachelor.

Illustration 8 : Service Windows



Source : <http://www.astwinds.com>

8.2 Fonctionnalités

Le but premier du service est de contrôler si les règles définies par les utilisateurs sont respectées. Dans ce cas, il faut ajouter à l'événement une alarme, dont le temps est également défini par l'utilisateur lors de la configuration.

Dans un second temps, trois options supplémentaires sont venues se greffer à la configuration (cumul de temps, affichage de la description et suppression automatique des reminders).

L'ajout de ces trois options a passablement compliqué le développement. Ces options ne sont pas singulièrement complexes à programmer, mais elles ont ouvert la porte à une multitude de cas n'ayant pas été imaginés auparavant.

Ce point sera repris plus en détail dans le chapitre suivant « Problèmes rencontrés ».

8.3 Optimisation

La première version du service fonctionnait comme désiré (sans les options supplémentaires de cumul de temps, d'affichage de description et de suppressions automatiques d'alarmes obsolètes).

Illustration 9 : Optimisation



Source : <http://www.problogger.net>

Cependant, l'exécution était plutôt lente alors qu'il n'y avait que très peu de règles et d'événements dans le calendrier.

Par la suite, l'ajout des trois options supplémentaires a permis de tirer la conclusion suivante : Le service n'était pas développé de façon optimale et l'ajout de ces trois options était quasiment impossible.

Le service a donc été redéveloppé et optimisé. Cette opération a été effectuée plusieurs fois pour en arriver à un résultat concluant.

9 Problèmes rencontrés

9.1 Application Android, application web

La première idée de ce travail de bachelor était de développer une application Android.

Après plusieurs recherches et afin d'élargir l'utilisation de Smart Agenda à la majorité des smartphones, il a été décidé, d'un commun accord avec M. Yann Bocchi, de développer une plateforme web.

Il n'est pas toujours facile de s'adapter à ce genre de changement qui est pourtant courant dans le monde de l'informatique.

9.2 Login et autorisation Google

Vu que chaque utilisateur de l'application devrait avoir un compte Google, il a été pensé d'utiliser l'authentification Google afin de se connecter à l'application Smart Agenda.

Malheureusement, cela n'a pas été possible.

Il existe deux moyens d'identifier un utilisateur avec l'API Google Calendar : soit en utilisant le couple nom d'utilisateur et mot de passe, soit en récupérant un token correspondant à l'utilisateur.

En utilisant une identification Google, il n'est pas possible de récupérer la moindre information sur le compte utilisateur. Cette option a donc été écartée.

La solution adoptée est la suivante : Chaque utilisateur créera un compte sur Smart Agenda. Lors de la première connexion, il ne pourra accéder qu'à une seule page - l'autorisation Google - et devra alors accepter l'accès à son calendrier pour l'application Smart Agenda.

Une fois cette opération effectuée, un token unique est enregistré dans la base de données et l'utilisateur peut configurer Smart Agenda à sa guise.

9.3 Prise en main des requêtes LINQ

Lorsqu'il faut exécuter une requête sur la base de données en utilisant LINQ, le langage de cette dernière n'est pas tout à fait équivalent au SQL standard.

Voici un petit exemple comparatif : sélection des utilisateurs dont le nom de famille est Morard :

SQL

Code 3 : Requête SQL

```
SELECT *  
  FROM User  
 WHERE lastname = 'Morard'
```

LINQ

Code 4 : Requête LINQ

```
from u in _dcsa.Users  
where u.lastname.Equals("Morard")  
select u;
```

Le langage n'est pas particulièrement difficile mais il faut s'y habituer.

9.4 API Google Weather

L'API de Google Weather n'est pas officiel. Il n'y a aucune documentation sur cette dernière.

Lors de la première utilisation, tout fonctionne très bien. Par la suite, lors de test, un accent a été inséré dans le nom d'une ville. Plus rien ne fonctionnait. Après quelques heures de recherche, il se trouve que cette API ne fonctionne pas en UTF-8.

Il a donc fallu trouver une variante afin de convertir le résultat en UTF-8 (Conversion en tableau de bytes, puis encodage en UTF-8).

9.5 Création du service Windows

Comme expliqué au chapitre précédent, le développement d'un service Windows n'est pas tout à fait le même que pour une application standard.

Le service ne peut pas être exécuté depuis Visual Studio. Il faut d'abord créer un second projet permettant d'installer le service.

Le débogage n'est pas aisé non plus. Vu que le service ne se lance pas dans Visual Studio, il n'est pas possible de déboguer directement.

Une fois le service installé, il faut attacher un débogueur à son processus.

9.6 Reminder Google Calendar

Smart Agenda ajoute, modifie et supprime des reminders pour les événements du calendrier Google.

Il aurait été utile que les objets de type « Reminder » aient un identifiant unique. Ce qui n'est pas le cas. Ils sont identifiés uniquement par le type d'alarme et le temps.

En effet, il aurait été possible de savoir si un reminder a été ajouté par l'utilisateur ou par l'application.

A cause de ce manque, les propriétés étendues (« Extended Property ») de chaque événement ont dues être utilisées. Elles ne sont pas documentées de façon optimale et ne sont guère faciles à utiliser.

9.7 Complexité du service

Egalement cité au chapitre précédent, le développement du service n'a pas été de tout repos.

Il existe de nombreux cas qui ne sont pas évidents à gérer. Par exemple, si un reminder est ajouté et que l'utilisateur change le temps de la règle ou s'il supprime manuellement un reminder.

Un schéma complet des cas répertoriés se trouve en annexe.

10 Améliorations possibles

L'application actuellement développée est tout à fait utile et utilisable mais ne reste, à l'heure actuelle, qu'un projet pilote.

Afin d'augmenter la probabilité que ce projet devienne réel, il est nécessaire de lui apporter quelques améliorations.

10.1 Ajout de webservice

Lors de la réalisation de ce travail de bachelor, deux webservices ont été implémentés (météo et CFF).

Durant la phase d'analyse de ce projet, d'autres webservices ont été sélectionnés, mais n'ont pas pu être intégrés par manque de temps.

En voici la liste :

- Statuts des avions
https://www.flightstats.com/developers/bin/view/Web+Services/web_services_directory
- Manifestations suisses
<http://www.20min.ch/rss/rss.tpl?type=rubrik&get=526&lang=ro>
- Situation routière
<http://www.truckinfo.ch/>
- Catastrophes naturelles
<http://www.catnat.net/index.php?format=feed&type=rss>

Grâce à la base de données très flexible, l'ajout d'un nouveau webservice peut se faire de façon simple. Il suffit d'intégrer le webservice ainsi que ses paramètres et valeurs dans la base de données et de créer la classe correspondante en C#.

10.2 Géolocalisation

Lorsqu'un utilisateur ajoute un événement dans le calendrier Google, il n'a pas la possibilité de spécifier l'emplacement de départ.

La première solution envisagée serait de faire entrer à l'utilisateur, sous une forme bien précise ce lieu de départ dans le champ description du calendrier. Mais cette solution n'est pas très intuitive pour l'utilisateur final et peut être source d'erreur.

Une seconde solution serait de pouvoir localiser l'utilisateur grâce à la géolocalisation (satellite, GSM, Wifi, etc.). L'utilisateur n'aurait alors aucune configuration supplémentaire à effectuer.

Cette amélioration, plutôt complexe pourrait faire parti d'un projet à elle seule.

10.3 Temps des reminders dynamiques

Afin de rendre l'application le plus autonome possible, il serait judicieux de rechercher automatiquement le temps de l'alarme dans certains cas.

Par exemple, lors de l'utilisation du service CFF, il serait intéressant de pouvoir récupérer l'emplacement de départ (voir géolocalisation au point précédent), afin de pouvoir calculer la distance entre le point de départ et celui d'arrivée.

Grâce à cette distance, il serait possible de calculer le temps du reminder adéquat. Plus cette distance est élevée, plus le risque de retard l'est également. Effectuer le trajet entre Sierre et Sion sous la neige provoquera un plus petit retard que de parcourir Sierre-Zürich.

11 Gestion de projet

Lors des deux premières semaines de travail, un cahier des charges a dû être établi. La version approuvée par M. Yann Bocchi se trouve sur le CD-ROM ci-joint.

11.1 Planification

Lors de la définition du cahier des charges, une planification simple a été réalisée avec l'outil Microsoft Project afin de définir les différentes tâches et leur durée approximative.

En voici un extrait :

Illustration 10 : Planification Microsoft Project

[-] Lancement projet	4 days	Mon 23.05.11	Thu 26.05.11
Préparation poste de travail	1 day	Mon 23.05.11	Mon 23.05.11
Installation et configuration logiciels	1 day	Tue 24.05.11	Tue 24.05.11
Planification initiale	1 day	Wed 25.05.11	Wed 25.05.11
Rédaction "cahier des charges"	1 day	Thu 26.05.11	Thu 26.05.11
[-] Recherche et tests	8 days	Fri 27.05.11	Tue 07.06.11
Recherche et tests d'applications concurrentes	2 days	Fri 27.05.11	Mon 30.05.11
Recherche de calendriers sous Android	2 days	Tue 31.05.11	Wed 01.06.11
Rédaction état de l'art	1 day	Thu 02.06.11	Thu 02.06.11
Recherche de webservices	3 days	Fri 03.06.11	Tue 07.06.11
[-] Apprentissage et installation	7 days	Wed 08.06.11	Thu 16.06.11
Installation d'un serveur web, db ainsi que les outils (ftp, svn, etc.)	2 days	Wed 08.06.11	Thu 09.06.11
Suivre tutoriel sur HTML 5	2 days	Fri 10.06.11	Mon 13.06.11
Suivre tutoriel sur Google Calendar API	3 days	Tue 14.06.11	Thu 16.06.11
[-] Développement web	33 days	Fri 17.06.11	Tue 02.08.11
Design et implémentation de la base de données	2 days	Fri 17.06.11	Mon 20.06.11
Design et implémentation interface	6 days	Tue 21.06.11	Tue 28.06.11
Développement application web	25 days	Wed 29.06.11	Tue 02.08.11
[-] Finalisation	8 days?	Wed 03.08.11	Fri 12.08.11
Tests et corrections des bugs	2 days	Wed 03.08.11	Thu 04.08.11
Mise en production de l'application	1 day	Fri 05.08.11	Fri 05.08.11
Rédaction documentation	2 days	Mon 08.08.11	Tue 09.08.11
Rédaction rapport final	3 days	Wed 10.08.11	Fri 12.08.11

Le projet est divisé en cinq sous-catégories :

Lancement de projet (~7%)

Cette phase a servi à installer le poste de travail, les différents outils nécessaires au développement, une partie d'administration (création des documents, planification, etc.)

Recherche et tests (~13,5%)

La phase de recherche et tests a permis d'établir un état de l'art et d'avoir plus d'informations concernant la suite du projet.

Apprentissage et installation (~11%)

La phase d'apprentissage a permis d'apprendre les connaissances de base afin de pouvoir passer à la phase suivante. L'installation du serveur web était également comprise dans cette phase.

Développement web (~55%)

Durant cette phase, l'application web a été développée (design, développement web et développement du service)

Finalisation (~13,5%)

Cette phase comprend le test de l'application, la mise en production ainsi que la rédaction des différents documents.

11.2 Suivi

Une fois le développement débuté, il a été décidé d'un commun accord avec M. Yann Bocchi, de travailler en utilisant une variante de méthode Agile, sous forme d'itérations.

Il fallait, chaque semaine, présenter un livrable (version fonctionnelle de l'application) ainsi que la planification détaillée de la semaine suivante.

De cette manière, les corrections éventuelles demandées par M. Yann Bocchi pouvaient être facilement intégrées. La planification a également été adaptée en fonction des décisions prises lors de ces séances.

11.3 Écart de temps entre prévu et réalisé

Il n'est vraiment pas évident de planifier le déroulement d'un projet et d'estimer la durée de chaque tâche, surtout sur un projet de longue durée et dont les spécificités et technologies à utiliser ne sont pas clairement définies.

La planification Microsoft Project initiale a été établie de façon trop large. Trop de temps a été affecté pour chaque tâche.

Par contre, le pourcentage de chaque étape du projet a plus ou moins été respecté.

11.4 Nombre d'heures réalisées

Au final, les 360 heures de travail effectif ont largement été effectuées puisque le nombre d'heures totales s'élève à plus de 400 heures (401 heures).

La planification hebdomadaire ainsi que le suivi se trouvent en annexes.

Un journal de travail décrivant les activités quotidiennes se trouve sur le CD-ROM ci-joint.

12 Conclusion

Afin de conclure ce rapport, il est nécessaire de souligner que ce projet, dans son ensemble, s'est très bien déroulé. M. Yann Bocchi m'a laissé une très grande liberté durant toutes les phases de développement.

Ce projet a été pour moi très bénéfique. Il m'a appris dans un premier temps à gérer un projet du début jusqu'à la fin. La planification initiale du projet n'est vraiment pas facile à réaliser mais il est nécessaire d'avoir une ligne de conduite pour ne pas s'égarer et perdre trop de temps.

Le fait de devoir délivrer une version utilisable du projet chaque semaine est également très intéressant. En effet, cela permet au développeur de rester concentré sur une partie précise du projet et il est impératif que celle-ci soit fonctionnelle lors de la présentation.

Une fois la présentation effectuée, les différentes corrections peuvent alors être apportées facilement, contrairement au développement Waterfall (Méthodologie en cascade). Le projet peut alors évoluer durant la phase de développement et le temps d'adaptation est considérablement réduit, puisqu'il n'y a qu'une seule partie à corriger.

D'un point de vue personnel, ce projet a été très instructif. Le développement de service Windows demande une façon différente de programmer, puisque le débogage est difficile et il n'est pas possible de tester l'application très souvent.

L'Ajax Control Toolkit était également une nouveauté pour moi. Beaucoup de fonctionnalités ont été testées mais une seule a été retenue pour ce projet, les autres n'étant pas nécessaires.

De nos jours, gagner du temps est indispensable. Nous ne pouvons donc pas nous permettre d'être en retard. Le développement d'une application permettant de les réduire a été pour moi un réel plaisir.

13 Bibliographie

- [1] DOLLON, Julien, RAVAILLE James, Visual Studio 2010 – Développez pour le web avec C# 4, Framework Entity 4, ASP .NET 4.0, Silverlight 4 et WCS RIA Services, *Eni*, 2010, 512 p.
- [2] GUÉRIN, Brice-Arnaud, ASP.NET 4 avec C# sous Visual Studio 2010, *Eni*, 2011, 531 p.

14 Webographie

14.1 Analyse

- [1] Apple: « Apple – iOS5 – See new features included in iOS 5 ». *Site d'Apple*, [En ligne].
<http://www.apple.com/ios/ios5/features.html#reminders> (Page consultée le 19 mai 2011)
- [2] FURNO, Nicolas : « Brevet : une alarme vraiment intelligente ». *Site igeneration*, [En ligne].
<http://www.igeneration.fr/iphone/brevet-une-alarme-vraiment-intelligente-48322> (Page consultée le 19 mai 2011)
- [3] GurYN : « Meeting Agent : Rendez votre agenda intelligent ». *Forum FrAndroid*, [En ligne].
<http://forum.frandroid.com/topic/7497-meeting-agent-rendez-votre-agenda-intelligent/> (Page consultée le 19 mai 2011)
- [4] « Alarm Pills : Le carnet de santé intelligent sur votre iphone ! ». *Site Hemcel*, [En ligne].
<http://www.hemcel.fr/2010/11/alarm-pills-le-carnet-de-sante.html> (Page consultée le 19 mai 2011)

14.2 Tutoriaux

- [1] Belline : « [Tuto] Serveur FTP avec Typsoft FTP ». *Forum TousPourUn*, [En ligne].
<http://forum.touspourun.org/topic/1548-tuto-serveur-ftp-avec-typsoft-ftp/> (Page consultée le 13 juin 2011)
- [2] CollabNet : « CollabNet User Information Center ». *Centre d'information CollabNet*, [En ligne].
http://help.collab.net/index.jsp?topic=/com.collabnet.doc.anksvn_001/action/ankh_getting_started.html (Page consultée le 6 juin 2011)
- [3] pharr : « Installing IIS 7 on Windows Server 2008 or Windows Server 2008 R2 ». *Site IIS*, [En ligne].
<http://learn.iis.net/page.aspx/29/installing-iis-7-on-windows-server-2008-or-windows-server-2008-r2/> (Page consultée le 13 juin 2011)
- [4] ROMELARD, Fabrice : « Installation de SQL Server 2008 – Version finale ». *Site TechnoS SourceS*, [En ligne].
<http://www.technos-sources.com/tutorial-installation-sql-server-2008-version-finale-97.aspx> (Page consultée le 13 juin 2011)

14.3 LINQ

- [1] MSDN : « Guide de programmation (LINQ to SQL) ». *Librairie MSDN*, [En ligne].
<http://msdn.microsoft.com/fr-fr/library/bb386976.aspx> (Page consultée le 20 juin 2011)
- [2] RaptorXP : « C# 3.0 : Syntaxe d'une requête LINQ ». *Blog de RaptorXP sur le site codes-sources.com*, [En ligne].
<http://blogs.codes-sources.com/raptorxp/archive/2007/11/19/c-3-0-syntaxe-d-une-requ-te-linq.aspx> (Page consultée le 20 juin 2011)

14.4 Développement

- [1] ALEKSEEV, Igor: « Using Google Calendar in ASP.NET Website ». *Site The Code Project*, [En ligne].
<http://www.codeproject.com/KB/aspnet/GoogleCalendarInASPNET.aspx> (Page consultée le 16 juin 2011)
- [2] FISHER, Jeff: « Getting Started with the .NET Client Library ». *Site Google Code*, [En ligne].
http://code.google.com/intl/fr-CH/apis/gdata/articles/dotnet_client_lib.html (Page consultée le 16 juin 2011)
- [3] FISHER, Jeff: « Using AuthSub with the .NET Client Library ». *Site Google Code*, [En ligne].
http://code.google.com/intl/fr-CH/apis/gdata/articles/authsub_dotnet.html (Page consultée le 13 juin 2011)
- [4] FORGOTTEN, S.: « Google Weather Api in ASP.Net ». *Forum stackoverflow*, [En ligne].
<http://stackoverflow.com/questions/5542900/google-weather-api-in-asp-net> (Page consultée le 7 juillet 2011)
- [5] GANAPARTHI, Srinivas: « How to Integrate Google Authentication in Asp.Net Application ». *Blog de Ganaparthi Srinivas*, [En ligne].
<http://srinivasganaparthi.blogspot.com/2011/04/how-to-integrate-google-authentication.html> (Page consultée le 13 juin 2011)
- [6] Google: « Authentication and Authorization in the Google Data Protocol ». *Site Google Code*, [En ligne].
<http://code.google.com/intl/fr-CH/apis/gdata/docs/auth/overview.html> (Page consultée le 13 juin 2011)

- [7] Google: « Data API Developer's Guide: .NET ». *Site Google Code*, [En ligne].
http://code.google.com/intl/fr-CH/apis/calendar/data/2.0/developers_guide_dotnet.html (Page consultée le 24 mai 2011)
- [8] Google: « .NET Client Library Developer's Guide ». *Site Google Code*, [En ligne].
<http://code.google.com/intl/pt-BR/apis/gdata/client-cs.html> (Page consultée le 16 juin 2011)
- [9] JEEVANAND, R.: « Adding and Retiving Extended Property using Google Calendar API ». *Forum stackoverflow*, [En ligne].
<http://stackoverflow.com/questions/4459708/adding-and-retiving-extended-property-using-google-calendar-api> (Page consultée le 13 juillet 2011)
- [10] MSDN: « Introduction aux applications de service Windows ». *Librairie MSDN*, [En ligne].
<http://msdn.microsoft.com/fr-fr/library/d56de412.aspx>(Page consultée le 4 juillet 2011)
- [11] MSDN: « Walkthrough: Creating a Windows Service Application in the Component Designer ». *Librairie MSDN*, [En ligne].
<http://msdn.microsoft.com/en-us/library/zt39148a.aspx#Y570> (Page consultée le 4 juillet 2011)
- [12] Steve: « How to pass in runtime a index of row for List-View? ». *Forum stackoverflow*, [En ligne].
<http://stackoverflow.com/questions/692120/how-to-pass-in-runtime-a-index-of-row-for-listview> (Page consultée le 22 juin 2011)
- [13] TAJAMMAL, Ali : « Google Weather API ». *Site VS ASP.NET*, [En ligne].
<http://www.vsasp.net/apis/google-weather-api/> (Page consultée le 4 juillet 2011)

15 Table des illustrations

15.1 Code

Code 1 : Réponse XML de Google Weather API	14
Code 2 : Exemple CSS	19
Code 3 : Requête SQL	38
Code 4 : Requête LINQ	38

15.2 Illustrations

Illustration 1 : Exemple de schéma	5
Illustration 2 : Reminders (Vue par liste).....	6
Illustration 3 : Reminders (Vue par emplacement).....	7
Illustration 4 : IIS 7.....	15
Illustration 5 : SQL Server 2008	15
Illustration 6 : Fenêtre LINQ (Visual Studio 2010).....	16
Illustration 7 : Logo Visual Studio 2010.....	18
Illustration 8 : Service Windows.....	35
Illustration 9 : Optimisation	36
Illustration 10 : Planification Microsoft Project	42

15.3 Schémas

Schéma 1 : Architecture version 1	10
Schéma 2 : Architecture version 2	11
Schéma 3 : Architecture Plateforme Web.....	20








15.4 Smart Agenda

Smart Agenda 1 : Formulaire d'inscription	21
Smart Agenda 2 : Menus	22
Smart Agenda 3 : Autorisation Google Calendar	22
Smart Agenda 4 : Autorisation Google Calendar 2.....	23
Smart Agenda 5 : Configuration d'un webservice pour un calendrier	24
Smart Agenda 6 : Ajout d'un service pour un calendrier	25
Smart Agenda 7 : Modification d'un service pour un calendrier	25
Smart Agenda 8 : Affichage des règles.....	26
Smart Agenda 9 : Ajout d'une règle	27
Smart Agenda 10 : Modification d'une règle	27
Smart Agenda 11 : Prochains événements	28
Smart Agenda 12 : Modification d'un événement.....	29
Smart Agenda 13 : Simulation.....	30
Smart Agenda 14 : Résultat du service.....	31
Smart Agenda 15 : Problème de localité.....	32
Smart Agenda 16 : Météo actuelle et prévisions	33
Smart Agenda 17 : Informations sur le trafic ferroviaire	34

16 Annexes

16.1 Contenu du CD-ROM

	Database	Contient le modèle physique de données ainsi que le script permettant de créer la base de données
	Documents	Contient les données du travail de bachelor, le cahier des charges ainsi que le rapport
	Project_Management	Contient la planification initiale, les rapports hebdomadaires, les procès-verbaux ainsi que le journal de travail
	Source_Code	Contient la solution de la plateforme web, du service Windows ainsi que le projet d'installation du service
	Read_Me.pdf	Contient les différents accès

16.2 Suivi de projet

Un journal résumant l'activité quotidienne durant les trois mois de travail sur le CD-ROM.

Semaine 1

Tâche effectuée semaine 1	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Installation d'Eclipse	1	1	0	0
Installation Android SDK	1	1	0	0
Installation plugin Android	1	1	0	0
Installation Subclipse SVN	1	1	0	0
Création d'un compte sur ProjectLocker	1	1	0	0
Installation du repository SVN	1	1	0	0
Prise en main d'Android	3	4	0	1
Lecture de tutoriaux Android	3	4	0	1
Recherche d'agenda sous Android	4	4	0	0
TOTAL Heures	16	18	0	2

Semaine 2

Tâche effectuée semaine 2	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Recherche d'informations sur l'API Google Calendar API and Tools	3	4	1	1
Recherche d'informations sur les calendriers Android	3	3	0	0
Lecture de tutoriaux Android	2	2	2	0
Modification du cahier des charges	2	3	0	1
TOTAL Heures	10	12	3	2

Semaine 3

Tâche effectuée semaine 3	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Recherche d'informations sur l'API Google Calendar API and Tools	1	0	1	-1
Lecture de tutoriaux Androïd	2	2	0	0
Recherche d'applications existantes	2	2	0	0
Analyse des applications existantes	2	2	0	0
Définition du cahier des charges	2	2	0	0
Recherche de webservices pour l'application	3	0	3	-3
Analyse des webservices	3	0	3	-3
Ecriture de 2-3 scénarios mettant en valeur l'application (avantages de la solution)	1	1	0	0
Construire état de l'art	3	3	0	0
TOTAL Heures	19	12	7	-7

Semaine 4

Tâche effectuée semaine 4	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Recherche d'informations sur l'API Google Calendar API and Tools	1	1	0	0
Choix du langage	2	2	0	0
Installation des outils	2	2	0	0
Recherche de webservices météo	3	3	0	0
Analyse du webservice météo choisi	3	3	0	0
Apprentissage de l'utilisation de l'API Google Calendar dans le langage choisi	4	4	0	0
Choix et implémentation d'un design	4	4	0	0
Recherche et implémentation d'une Autorisation Google	8	8	0	0
Installation d'un serveur web IIS	4	4	0	0
Installation et conf. des outils (ftp, etc.)	2	2	0	0
TOTAL Heures	33	33	0	0

Semaine 5

Tâche effectuée semaine 5	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Installation d'un SGBD (serveur)	4	4		0
Modélisation de la base de données	8	8	2	0
Création du script de base de données	2	2		0
Création des classes de la DB	4	4		0
Analyse détaillée de l'API Google Weather	4	4		0
Création d'un parser pour utilisation de Google Weather	8	8	4	0
Application web: Développement de la configuration pour les webservice	10	10	15	0
TOTAL Heures	40	40	21	0

Semaine 6

Tâche effectuée semaine 6	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Modélisation de la base de données	2	2	0	0
Création d'un parser pour utilisation de Google Weather	4	4	1	1
Application web: Développement de la configuration pour les webservice (Rules)	15	25		10
Recherche pour insérer gérer les événements dans un calendrier précis	4	4	0	0
Application background (pour éditer les reminders en arrière-plan)	20	0		-20
TOTAL Heures	45	35	1	-9

Semaine 7

Tâche effectuée semaine 7	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Application background (pour éditer les reminders en arrière-plan (webservice météo))	40	40	0	0
TOTAL Heures	40	40	0	0

Semaine 8

Tâche effectuée semaine 8	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Réécriture du service pour modifier les reminders (optimisation du code)	15	15	0	0
Suppression des reminders si la condition n'est plus satisfaite (Extend property)	10	10	0	0
Affichage dans le champ description des différents événements	10	10	0	0
Affichage du calendrier	5	5	5	5
TOTAL Heures	40	40	5	5

Semaine 9

Tâche effectuée semaine 9	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Modification d'une règle (configuration)	5	5	0	0
Ajouter un champ booléen pour savoir si l'utilisateur souhaite qu'un reminder automatiquement ajouté soit supprimé automatiquement dans le cas où la condition n'est plus respectée	15	15	0	0
Ajout d'un champ booléen pour l'affichage de la description ou non	5	5	0	0
Affichage du calendrier	5	15	0	10
Ajout d'une localité de départ pour le service CFF	10	0	10	0
Cumul des temps	15	15	15	15
TOTAL Heures	55	55	25	25

Semaine 10

Tâche effectuée semaine 10	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Réécriture du service	24	32	0	8
Ajout d'une localité de départ pour le service CFF	10	0	10	0
Cumul des temps	15	8	0	-7
TOTAL Heures	49	40	10	1

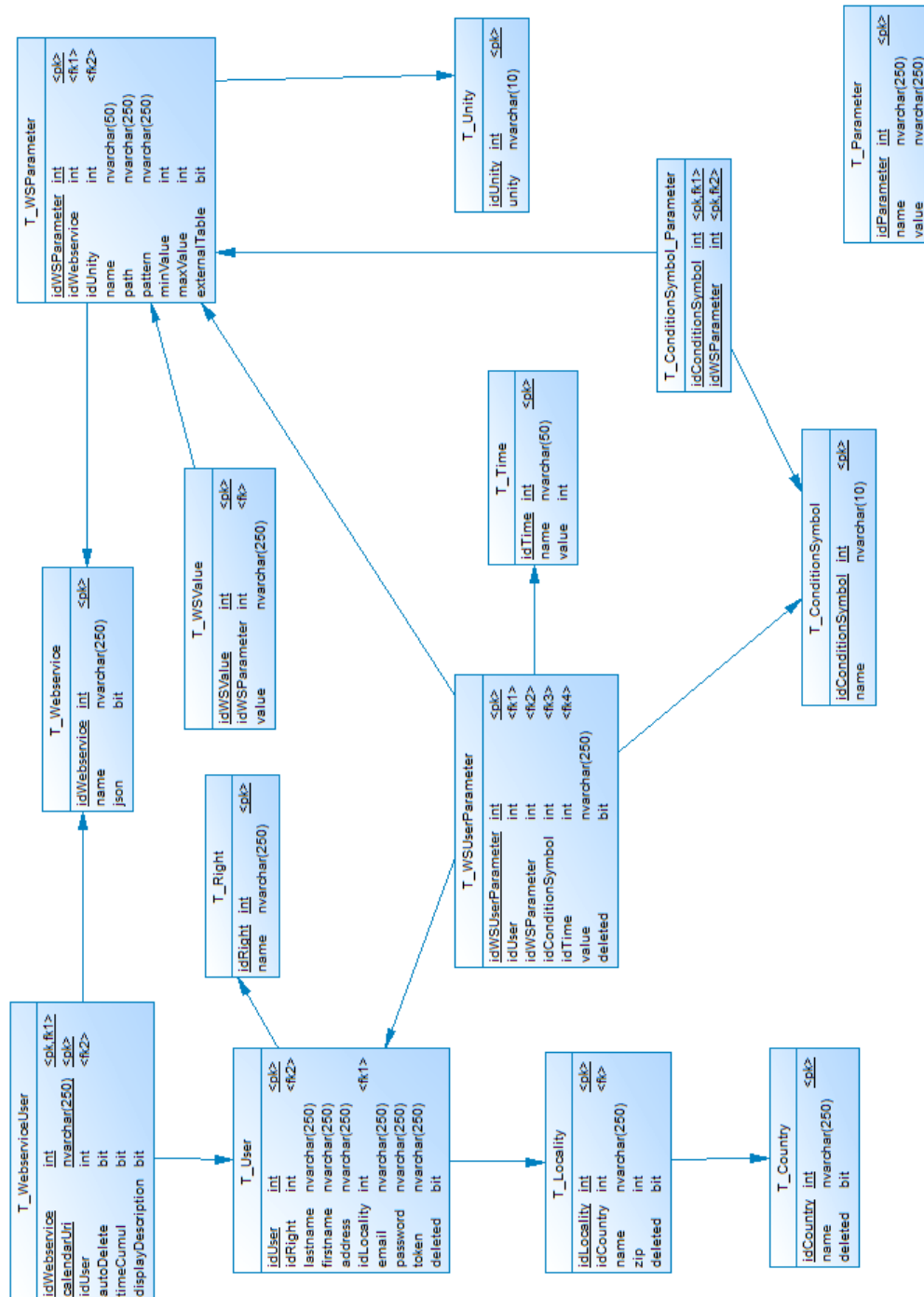
Semaine 11

Tâche effectuée semaine 11	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Tests	8	8	0	0
Correction de bugs	24	16	0	-8
Retouches graphiques	4	4	0	0
Divers retouches	8	8	0	0
TOTAL Heures	44	36	0	-8

Semaine 12

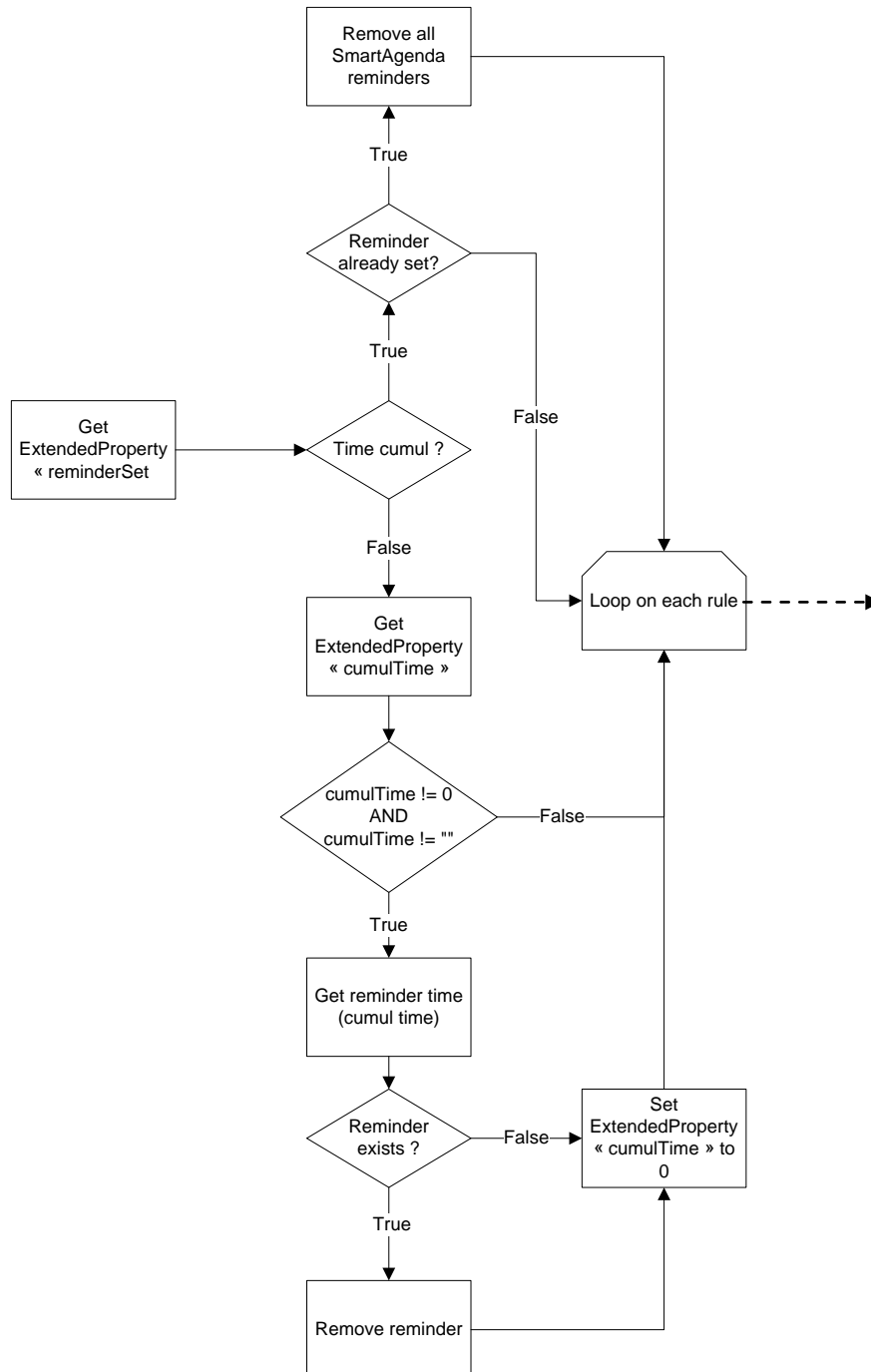
Tâche effectuée semaine 12	Prévu (heures)	Effectué (heures)	Restant (heures)	Différence Prévu / Effectué (heures)
Rédaction du rapport	35	35	0	0
Correction de bug	5	5	0	0
TOTAL Heures	40	40	0	0

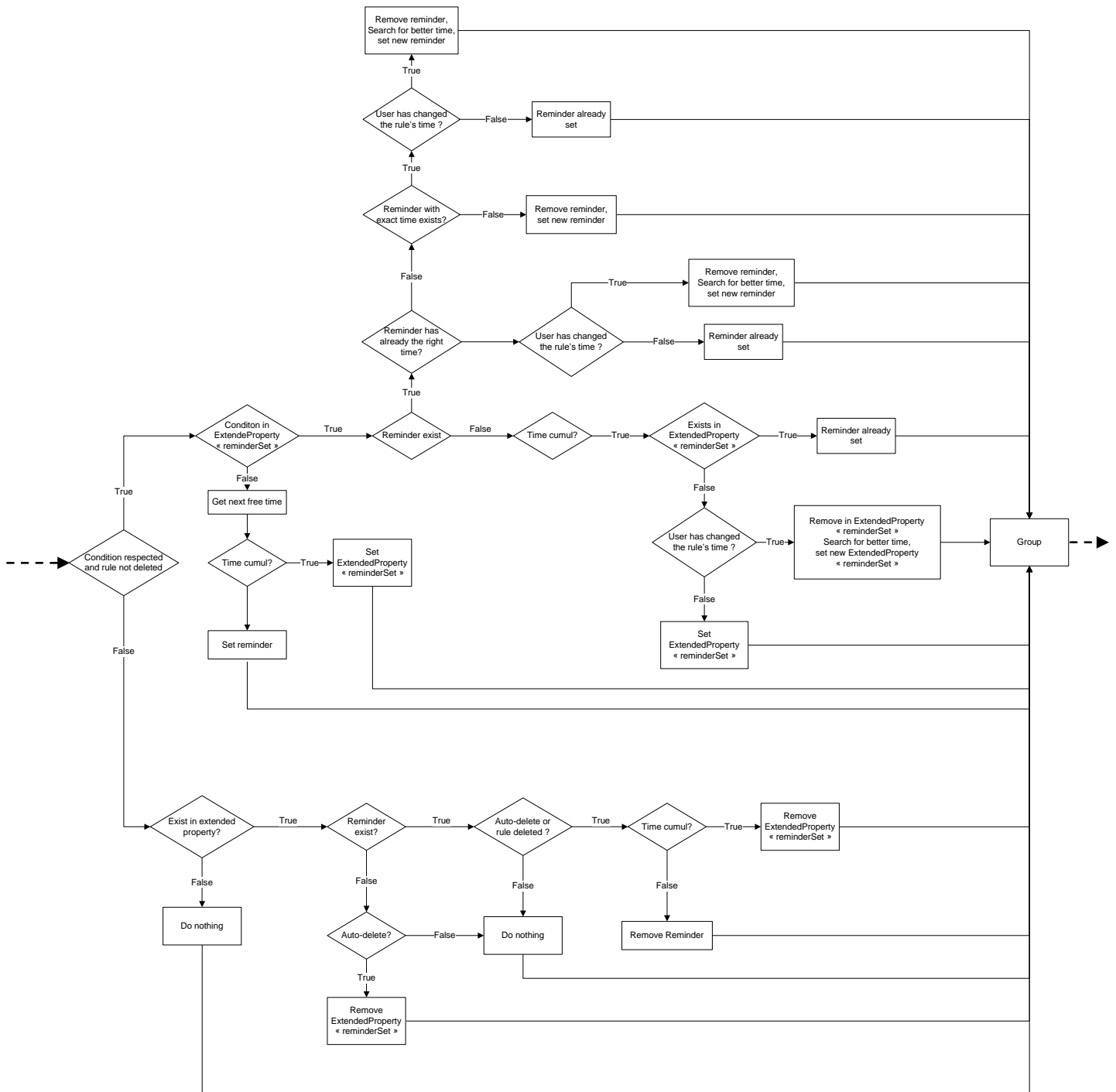
16.3 Modèle physique de données

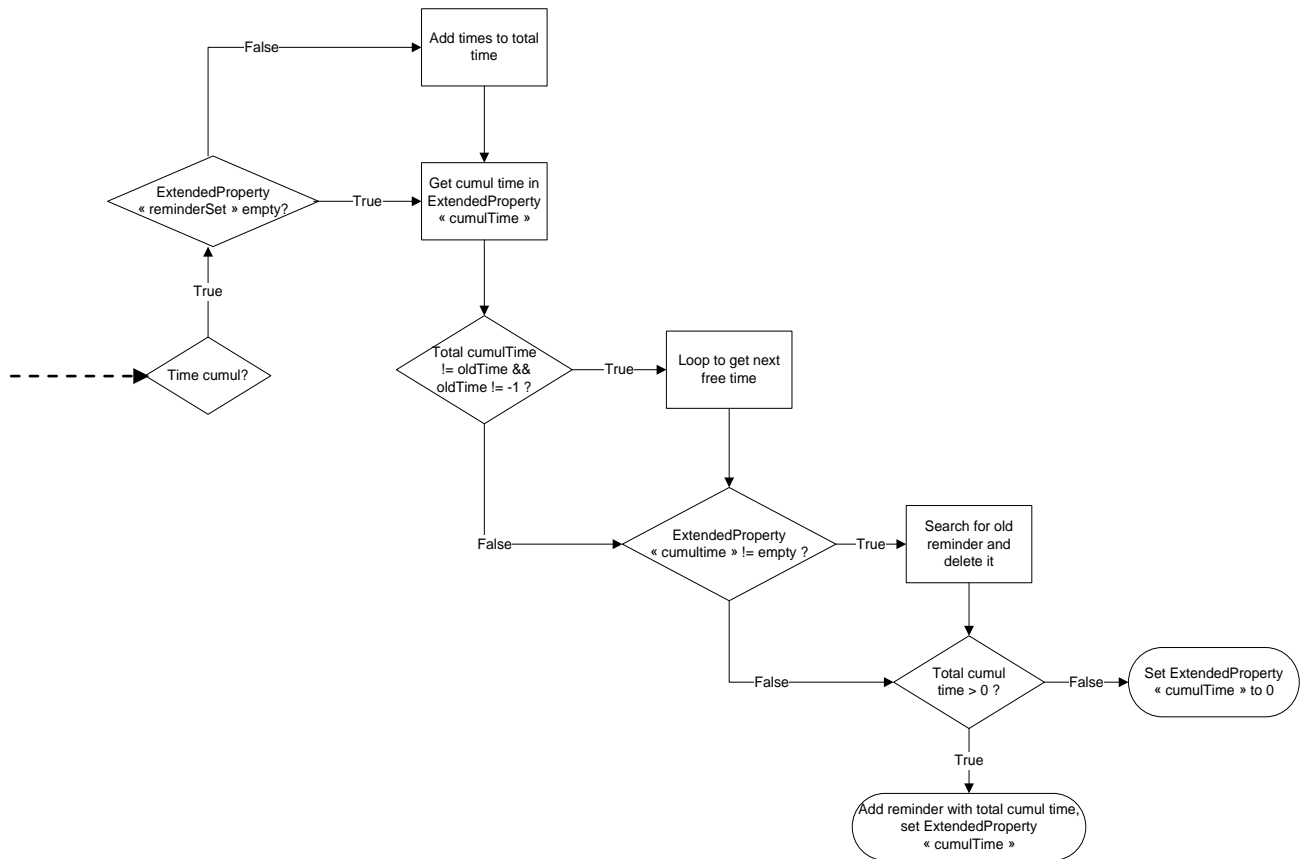


16.4 Schéma du service Windows

Afin de faciliter la lecture, le schéma a été découpé en trois parties.







16.5 Procès-verbaux

Procès-verbal n° 1

Date et heure	19.05.2011 - 14h00-14h25
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Contrôler qu'une API Google Calendar existe et la tester ▪ Création ou recherche d'une application calendrier native (qui fonctionnera avec Google Agenda) ▪ Calendrier simple ▪ Avec 1 seule vue pour commencer (par semaine) ▪ Faire état de l'art ▪ Création d'un service entre l'application Android et l'API Google agenda afin de pouvoir configurer les alarmes (paramétrisation utilisateur) ▪ Emulateur Android, tester l'installation d'applications ▪ J'ai reçu une tablette Samsung Galaxy dont je suis responsable ▪ Ecriture de 2-3 scénarios présentant les avantages de la solution
Prochaine séance	Jeudi 26.05.2011 à 14h00

Procès-verbal n ° 2

Date et heure	26.05.2011 - 14h00-14h30
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<p>Cahier des charges</p> <ul style="list-style-type: none"> ▪ Etoffer avec plus de texte, d'explications, illustrations, scénarios, etc. (voir fichier mail) ▪ Ajouter une partie déroulement (avec les grandes étapes) ▪ Ajouter une partie planning (par semaine) <p>Planification</p> <ul style="list-style-type: none"> ▪ Ajouter les heures prévues pour une tâche ▪ Ajouter les heures effectuées pour une tâche ▪ Ajouter la différence entre le planifié et l'effectué ▪ Ajouter le « reste à faire » (RAF) <p>Architecture</p> <ul style="list-style-type: none"> ▪ Modifier le schéma ▪ Application web ▪ Connexion avec l'API de Google Calendar ▪ L'application sera donc une application web

	<p>Etat de l'art</p> <ul style="list-style-type: none"> ▪ Analyse des solutions existantes ▪ Exemple sur « Mac Generation » (recherche calendrier intelligent, alarme intelligente) ▪ Décrire ce que l'application proposée apporte de plus ▪ But : ne pas réinventer de l'existant <p>Webservice</p> <ul style="list-style-type: none"> ▪ CFF ▪ Transport public ▪ Météo ▪ Info trafic ▪ Routes fermées ▪ Manifestations
Prochaine séance	Mardi 07.06.2011 à 09h00

Procès-verbal n ° 3

Date et heure	07.06.2011 - 09h00-09h20
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Revue du cahier des charges →OK ▪ Modification du lancement de projet (25% trop élevé) ▪ Etat de l'art Compléter avec les dernières informations d'Apple lors de la Worldwide Developers Conference à San Francisco (iCloud en particulier, avec les reminders et la géolocalisation) ▪ Planification feuille Excel → OK ▪ Choix du langage: ASP.NET <p>Fonctionnalités:</p> <ul style="list-style-type: none"> ▪ Création d'un calendrier web pour ajouter, modifier et supprimer des événements → Inutile ▪ Interface permettant de se loguer en utilisant le login et mot de passe Google ▪ Interface de paramétrisation de l'application (reminders et webservice) ▪ Stockage des informations dans une base de données ▪ Service en arrière-plan qui se connectera au webservice et ajoutera les reminders ▪ Interface web (simple et fonctionnelle, style Doodle) ▪ Création d'une VM sur la DMZ pour la production
Prochaine séance	Lundi 20.06.2011 à 09h30

Procès-verbal n° 4

Date et heure	20.06.2011 - 09h30-09h50
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<p>Points effectués:</p> <ul style="list-style-type: none"> ▪ Installation des outils de développement (Visual Studio, SVN, ...) ▪ Installation d'un serveur web IIS, ftp, etc. ▪ Implémentation d'un exemple .net pour le calendrier (récupération d'événements, ajout de reminders en .Net) ▪ Google Weather API ▪ Interface ▪ Login ▪ Autorisation Google (Système de token) <p>A faire :</p> <ul style="list-style-type: none"> ▪ Demander à M. David Russo un accès au serveur web ▪ Formulaire pour créer un compte ▪ Rester connecter avec un compte ▪ Itérations pour la semaine prochaine ▪ Configuration: Possibilité de configurer les webservices par calendrier ▪ Faire une liste avec 8-10 idées de webservices + utilité pour les différents cas ▪ Réfléchir à une astuce pour savoir si un utilisateur va se déplacer en voiture où en transports public.
Prochaine séance	Mardi 28.06.2011 à 09h30

Procès-verbal n ° 5

Date et heure	28.06.2011 - 09h30-10h00
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Installation d'un SGBD (serveur) ▪ Modélisation de la base de données (encore 2 heures de prévues) ▪ Création du script de base de données ▪ Création des classes de la DB <ul style="list-style-type: none"> ○ Connexion avec LINQ (mettre dans le rapport) ▪ Analyse détaillée de l'API Google Weather ▪ Création d'un parser pour utilisation de Google Weather (encore 4h de prévues) ▪ Application web: Développement de la configuration pour les webservices (encore 15h de prévues) ▪ Formulaire de création de compte ▪ Formulaire de login avec la case resté connecté (cookie) <p>Webservices :</p> <ul style="list-style-type: none"> ▪ <u>Météo</u> ▪ <u>CFF</u> ▪ <u>Aéroport (flight status)</u> ▪ <u>Manifestations suisses</u> ▪ <u>Catastrophes naturelles</u> ▪ <u>Situation routière</u> (pas de RSS)
Prochaine séance	Mercredi 6 juillet 2011 à 09h00

Procès-verbal n ° 6

Date et heure	06.07.2011 - 09h00-09h30
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Modifications de la modélisation de la base de données ▪ Création d'un parser pour utilisation de Google Weather et des CFF ▪ Application web: Développement de la configuration pour les webservices (Rules) ▪ Recherche pour insérer et gérer les événements dans un calendrier précis (code) ▪ Application background (service pour éditer les reminders en arrière-plan) ▪ Inverser les 2 dernières colonnes de la planification ▪ Créer un compte gmail pour Yann Bocchi ainsi qu'un compte sur SmartAgenda ▪ Trouver un webservice pour les avions
Prochaine séance	Mardi 12.07.2011 après la séance d'Alexandre à 9h

Procès-verbal n ° 7

Date et heure	12.07.2011 - 09h30-10h10
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Application background (service pour éditer les reminders en arrière-plan, pour le service météo) ▪ Mettre une plage de valeurs pour la dropdownlist Hour before (5min ,10min ,15min ,30min ,45min ,1h ,2h , 3h, 4h) ▪ Prévoir virtualisation du serveur + comptes pour la pérennité de l'application ▪ Prévoir une page de simulation (bouton pour faire la neige, champs texte pour prévoir du retard pour les trains, etc.) ▪ Rechercher les événements jusqu'à 36h à l'avance ▪ Additionner la 3^{ème} colonne dans la planification ▪ Itération avec la simulation et la météo pour la prochaine séance
Prochaine séance	Lundi 18.07.2011 à 10h00 (peut-être mardi)

Procès-verbal n ° 8

Date et heure	18.07.2011 - 10h00-10h15
Lieu	Bureau de Yann Bocchi
Personnes présentes	Yann Bocchi (Professeur) François Morard (Réalisateur de projet)
Personnes absentes	-
Cadre de la séance	<ul style="list-style-type: none"> ▪ Réécriture du service pour modifier les reminders (optimisation du code) ▪ Suppression des reminders si la condition n'est plus satisfaite (Extend property) ▪ Affichage dans le champ description des différents événements ▪ Affichage du calendrier ▪ Possible absence mercredi (20.07.2011) matin ▪ Exemple de rapport ▪ Yann est de retour le 10-11 août ▪ Possibilité de le contacter en cas de problème (Tél: 079 796 11 72) ▪ Ajouter un champ booléen pour savoir si l'utilisateur souhaite qu'un reminder automatiquement ajouté soit supprimé automatiquement dans le cas où la condition n'est plus respectée ▪ Affichage du calendrier (+ modification) ▪ Cumul des temps (Absolu-relatif, colonne dans la config)
Prochaine séance	-