Master I Informatique
EPSI Bordeaux

L'Open Graph de Facebook et les services web de Cdiscount

Adrien Cassagne

Sous la direction de M. Gregory Taudin et de M. Fabien Grosyeux

Bordeaux, le 31 août 2012

Remerciements

Je tiens à remercier dans un premier temps, toute l'équipe pédagogique de l'EPSI Bordeaux et les professeurs responsables de la 1ère année de master, pour avoir assuré la partie théorique de ma formation.

Je veux aussi dire ma reconnaissance aux personnes suivantes, pour l'expérience enrichissante et pleine d'intérêt qu'elles m'ont faite vivre durant ces trois mois au sein de l'entreprise Cdiscount :

Monsieur Gregory Taudin, chef de projet et responsable de l'équipe R&D chez Cdiscount, pour son accueil, son aide, et la confiance qu'il m'a accordée dès mon arrivée dans l'entreprise ; sans oublier sa participation au cheminement de ce rapport.

Monsieur Fabien Grosyeux, développeur de la société Cdiscount, pour m'avoir intégré rapidement au sein de l'entreprise et m'avoir accordé toute sa confiance ; pour le temps qu'il m'a consacré tout au long de cette période, sachant répondre à la plupart de mes interrogations.

Monsieur Sylvain Marangon, développeur architecte chez Cdiscount, pour son expertise et son aide précieuse quand j'ai rencontré des difficultés techniques.

Monsieur Yann Treuiller, graphiste chez Cdiscount, pour la réalisation du logo "Les Ambassadeurs de Cdiscount".

Table des matières

I Introduction . . . 1
II Présentation de l'entreprise . . . 3
II.1 Présentation générale . . . 3
II.2 Matériel et méthodes . . . 3
II.3 Organigramme de l'équipe . . . 4
III Le projet . . . 5
III.1 Présentation . . . 5
III.1.1 L'existant . . . 5
III.1.1.1 Shopping Social . . . 5
III.1.1.2 F-connexion . . . 6
III.1.1.3 Widgets sociaux . . . 6
III.1.2 Les objectifs . . . 7
III.1.3 Les outils et les choix techniques . . . 9
III.1.3.1 .NET, C# et Visual Studio 2010 . . . 9
III.1.3.2 Bases de données . . . 10
III.1.3.3 Parallélisme . . . 10
III.1.3.4 Facebook SDK . . . 11
III.1.3.5 Services web . . . 11
III.1.3.6 Surcouche : framework propriétaire Cdiscount . . . 11
III.1.3.7 JQuery Mobile . . . 12
III.1.3.8 ASP.NET MVC 3 . . . 12
III.1.4 Cheminement du projet . . . 13
III.2 Réalisation . . . 14
III.2.1 L'Open Graph . . . 14
III.2.2 Service web : Facebook Collector . . . 17
III.2.3 Service web : Facebook Aggregator . . . 20
III.2.3.1 Création des classements . . . 21
III.2.3.2 Récupération des classements . . . 21
III.2.4 Le site web mobile . . . 23
IV Conclusion . . . 25
V Annexes . . . 26
V.1 Base de données sociale . . . 26
V.2 Architecture du projet . . . 27
Bibliographie . . . 28

Table des figures

1 Logo du Lab social 1
2 Logo Microsoft 3
3 Organigramme partiel de Cdiscount (équipe R&D) 4
4 Capture d'écran du Shopping Social 5
5 Capture d'écran de la F-Connexion 6
6 Capture d'écran des widgets sociaux 6
7 Capture d'écran de la page Cdiscount sur Facebook 7
8 Logo Microsoft .NET 9
9 Logo SQL Server 10
10 Logo IIS 11
11 Logo de JQuery Mobile 12
12 Chronologie du projet en nombre de semaines 13
13 Illustration de l'Open Graph 14
14 Parcours du graphe de Facebook 15
15 Diagramme de classes du service collecteur 17
16 Architecture multi-thread du collecteur 19
17 Diagramme de classes du service agrégateur 20
18 Capture d'écran de l'interface mobile : l'accueil 23
19 Capture d'écran de l'interface mobile : un classement 24
20 Base de données sociale 26
21 Architecture du projet 27

I Introduction

Dans le cadre de la 1ère année de master à l'**EPSI (École Privée des Sciences Informatiques) Bordeaux** [1] j'ai effectué un stage d'une durée de 3 mois chez **Cdiscount** [2], société du groupe Casino située à Bordeaux (rue Fondaudège).

J'ai été recruté en tant que stagiaire pour l'étude des possibilités mobiles et sociales.
Avant mon arrivée, Cdiscount avait fait le choix de créer une petite équipe (3 personnes) de Recherche et Développement (R&D) afin de veiller sur les nouvelles technologies du web et de développer des modules toujours plus innovants. C'est ainsi que j'ai intégré cette équipe.

FIGURE 1 – Logo du Lab social

J'ai rapidement été chargé de l'étude de l'**Open Graph de Facebook** [3] : ce graphe suit une normalisation bien définie et permet l'organisation des données. L'objectif premier était de mettre en place une solution capable de récupérer efficacement des données depuis Facebook vers la base de données interne de Cdiscount.

Au départ, l'utilisation de ces données n'était pas définie. L'équipe de R&D (aussi appelée **Lab social**) souhaitait connaître dans quelle mesure il était possible de récupérer ces données (et quels types de données pouvait-t-on récupérer précisément).

Afin de rendre possible l'exploitation de cette collecte de données, nous avons dû mettre en place une architecture orientée service web s'appuyant sur le **framework** [4] propriétaire de Cdiscount. Cette architecture permet d'appeller le traitement (collecte des données Facebook) à distance et de façon normalisée.

1. EPSI : http://www.epsi.fr
2. Site internet de Cdiscount : http://www.cdiscount.com
3. Open Graph de Facebook : https://developers.facebook.com/docs/opengraph/
4. Framework : un framework est un kit de composants logiciels structurels, qui sert à créer les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture).

Par la suite, nous avons proposé plusieurs applications pour ces données :

- la création d'une application de ludification (gamification) : classification/valorisation des utilisateurs Facebook en fonction de leurs interactions avec Cdiscount,
- la mise en place de statistiques sur le comportement des utilisateurs Facebook,
- la combinaison des données Facebook avec les données de Cdiscount pour améliorer la publicité ciblée.

C'est l'application de gamification qui a été retenue comme prioritaire. Notre travail a alors concerné le développement d'outils nécessaires pour que l'exploitation des données Facebook soit la plus simple possible.

II Présentation de l'entreprise

II.1 Présentation générale

Cdiscount est une société de vente en ligne bordelaise créée en 1998 par les frères Hervé, Christophe et Nicolas Charle. Depuis 2000, Cdiscount est devenu une filiale du groupe Casino. La firme se flatte souvent d'être le premier site de e-commerce Français ! En réalité, Cdiscount est le site de vente en ligne qui réalise le plus gros chiffre d'affaire.

A l'origine spécialisée dans le high-tech, Cdiscount s'est aujourd'hui étendue à de nombreux domaines (loisirs, équipements, etc.).
Depuis 2006, Cdiscount a pris l'initiative d'ouvrir son premier magasin physique au Bouscat. Suivant cette dynamique d'autres magasins s'ouvrent dans les grandes villes de France (Paris actuellement).

Structurellement, la société se découpe en trois parties principales : achat/commerce (300 personnes), la logistique (400 personnes) et le système d'information (250 personnes) (c'est dans cette dernière que j'ai effectué mon stage).

II.2 Matériel et méthodes

Cdiscount entretient un partenariat avec la société Dell [5] pour ses postes de travail et ses serveurs. Les modèles des ordinateurs sont très variables selon les postes occupés par les employés mais c'est la gamme Optiplex [6] qui domine.

La plupart des postes de travail utilisent Windows 7 Entreprise ou Windows XP Professionnel pour les plus anciens (tel fut mon cas).

La partie serveur est assurée majoritairement par des Windows 2008 Serveur R2. C'est sans surprise que les serveurs déploient Team Foundation Server (TFS) comme gestionnaire de version de code source ainsi qu'Internet Information Services (IIS) comme serveur web.
On retrouve aussi un serveur Exchange pour tout ce qui est messagerie, organisation et emplois du temps. Enfin, la société met un serveur SharePoint 2010 à disposition de tous les employés pour communiquer les documents efficacement.

Côté environnement de développement c'est Visual Studio 2010 Premium qui est déployé sur la plupart des postes de développeur. Fidèle à Microsoft, Cdiscount développe son site internet grâce au framework .NET version 4 (langage C#) et s'appuie majoritairement sur SQL Server 2008 comme base de données.

Figure 2 – Logo Microsoft

5. Site internet de Dell : http://www.dell.com
6. Gamme Optiplex de Dell : http://www.dell.com/Optiplex

II.3 Organigramme de l'équipe

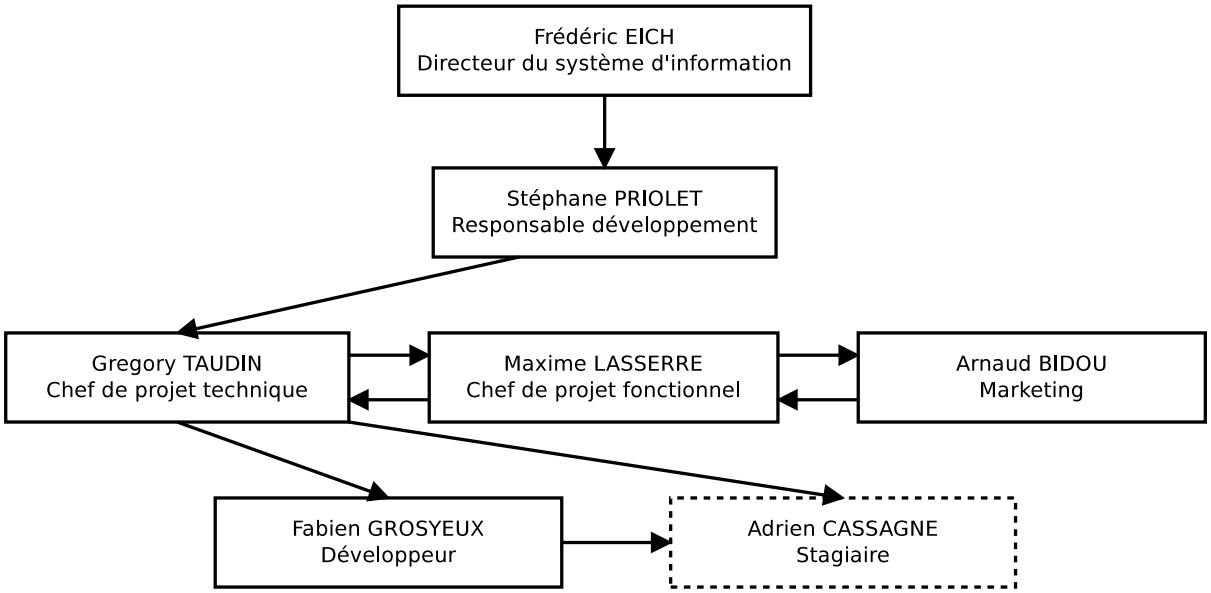

FIGURE 3 – Organigramme partiel de Cdiscount (équipe R&D)

Cette organigramme représente une vison simplifiée de ma hiérarchie et de celle de mes supérieurs. L'équipe R&D, à proprement parler, se compose de M. Gregory TAUDIN, M. Fabien GROSYEUX et M. Maxime LASSERRE.
M. Gregory TAUDIN et M. Fabien GROSYEUX ont été mes tuteurs directs tandis que M. Maxime LASSERRE jouait le médiateur entre M. Grégory TAUDIN, chef de projet technique, et M. Arnaud BIDOU, responsable marketing.

M. Stéphane PRIOLET, superviseur de l'ensemble des développeurs, était un interlocuteur direct de M. Gregory TAUDIN.

III Le projet

III.1 Présentation

III.1.1 L'existant

M. Gregory Taudin et M. Fabien Grosyeux sont les instigateurs de plusieurs **modules sociaux** sur le site Cdiscount. Parmi les plus importants, on retrouve :

- le Shopping Social,
- la F-connexion,
- les widgets sociaux.

III.1.1.1 Shopping Social

Le Shopping Social de Cdiscount est une page dédiée [7]. Elle se compose de plusieurs modules à part entière, s'appuyant chacun sur Facebook :

- les listes d'envies de nos amis,
- les derniers articles de la page Facebook de Cdiscount,
- des classements sur les produits les plus aimés,
- une liste de produits qui pourraient potentiellement nous intéresser,
- la possibilité de partager un panier sur Facebook.

L'objectif de cette page est d'augmenter l'impact de l'e-commerce sur la toile sociale.

Figure 4 – Capture d'écran du Shopping Social

7. Shopping Social sur Cdiscount : http://www.cdiscount.com/shopping-social.html

III.1.1.2 F-connexion

La **F-connexion** [8] permet de s'inscrire ou de se connecter sur Cdiscount par l'intermédiaire de son compte Facebook, cette option facilite l'inscription/connexion au site en dispensant l'utilisateur de certaines informations. Cette fonctionnalité commence à être très répandue sur beaucoup de sites internet. Elle est facilement adoptée par bon nombre d'utilisateurs.

FIGURE 5 – Capture d'écran de la F-Connexion

III.1.1.3 Widgets sociaux

Les widgets sociaux sont la nouveauté pour cette fin d'année. Ils condensent la plupart des fonctionnalités proposées par le shopping social dans de petits widgets (modules) présents sur toutes les pages du site. Les widgets apportent aussi la possibilité de tchatter avec nos amis Facebook alors que l'on est sur le site Cdiscount.

FIGURE 6 – Capture d'écran des widgets sociaux

8. F-connexion sur Cdiscount : https://clients.cdiscount.com

III.1.2 Les objectifs

Dans la continuité des précédents modules, l'équipe du Lab social s'est penchée sur la page Facebook de Cdiscount [9].

Pour rappel, Facebook permet à chacun d'entre nous de constituer un profil à l'image de notre personne et ainsi de communiquer avec des amis qui ont fait de même.
Le géant du social propose aussi d'autres services un peu différents comme la possibilité pour une entreprise de créer une page. Au même titre que le profil représente une personne sur Facebook, une page peut symboliser une entreprise. La page Facebook de Cdiscount propose des articles de façon régulière (approximativement 2 articles par jour).

FIGURE 7 – Capture d'écran de la page Cdiscount sur Facebook

Tous les utilisateurs de Facebook peuvent consulter cette page (et donc ses articles) puisqu'elle est publique. Mais ce n'est pas tout, un utilisateur peut interagir avec la page de différentes manières ; il peut :

– signaler qu'il aime la page Cdiscount,

9. Page Facebook de Cdiscount : https://www.facebook.com/Cdiscount

- aimer des articles,
- commenter des articles,
- ajouter des articles,
- aimer des commentaires d'autres utilisateurs,
- partager des articles de la page Cdiscount avec ses amis.

Commercialement, l'intérêt de cette page est d'exploiter **la "viralité" de Facebook** pour essayer de démarcher de nouveaux clients. Cette viralité utilise les actions de Facebook pour accélérer le processus de communication entre utilisateurs. L'article suivant [1] met en évidence le processus.

Toutes ces actions ne sont pas égales en termes d'efficacité pour la propagation de la marque. La plus pertinente, d'après l'équipe marketing de Cdiscount, est l'action d'aimer la page Facebook de Cdiscount. Elle garantit que l'utilisateur verra toutes les publications de Cdiscount sur son flux. De plus le nombre de **likers** (personnes qui aiment) est un indicateur fort pour mesurer la popularité d'une page.

L'action de partager un article a également un gros impact. Quand on partage un article sur Facebook, tous nos amis voient une publication sur leur flux. De plus cette publication est souvent très visible (elle occupe beaucoup de place sur l'écran).
Les autres actions sont un peu moins importantes en termes de viralité.

Ce potentiel a soulevé plusieurs questions : Serait-il possible de récupérer toutes ces actions ? Si oui, quelles en sont les limites ? Peut-on exploiter ces informations pour devenir plus performant commercialement ?

M. Gregory Taudin, mon tuteur, m'a chargé d'apporter des réponses à ces questions. Ce fut la première partie de mon travail.

Par la suite, mes objectifs ont évolué : il a fallu que j'adapte mes différentes solutions à l'architecture spécifique de l'entreprise. L'enjeu étant de fournir des traitements compatibles avec les procédures d'exécution ordonnancées de Cdiscount.

Dans un premier temps j'ai commencé à étudier la problématique avec les outils de mon choix et sans aucunes contraintes. Mais une fois obtenus quelques résultats convaincants, M. Gregory Taudin a souhaité que j'adopte les normes, les outils et les techniques Cdiscount.

III.1.3 Les outils et les choix techniques

Les parties suivantes sont techniques et décrivent les outils logiciels utilisés et/ou développés par Cdiscount.

Avant de poursuivre, il est bon de noter que Cdiscount a adopté les technologies Microsoft en matière de développement. Ce choix peut être critiquable mais il a l'avantage d'uniformiser le système d'information. De manière générale, les outils proposés par Microsoft sont très attrayants pour les sociétés : ils facilitent le développement en proposant une structure très bien documentée.

Les paragraphes suivants n'ont pas la prétention de décrire la totalité des outils que j'ai utilisés pendant mon stage ; mais, seulement les plus importants à mes yeux.

III.1.3.1 .NET, C# et Visual Studio 2010

Le développement du projet repose sur le framework **.NET** [10] de Microsoft. Ce dernier est sorti en même temps que le **langage C#**. C'est la réponse de Microsoft au langage **Java**. Tout comme Java, C# est un langage interprété ou plutôt compilé "à la volée" (**Just In Time**, JIT). Ce type de langage apporte une grande souplesse de programmation, notamment au niveau de la gestion de la mémoire vive et de la sécurité système. En contrepartie, les performances sont diminuées par rapport à un langage compilé.

Enfin nous avons utilisé Visual Studio 2010, cet environnement de développement mis à disposition par Microsoft est très performant et propose énormément de fonctionnalités. Voici une liste non-exhaustive des plus utilisées :

- un éditeur/analyseur syntaxique de texte,
- un gestionnaire de version de code source,
- un débuggeur très complet,
- un visionneur de projet/solution,
- une console de sortie,
- des options de refactorisation [11] puissantes.

FIGURE 8 – Logo Microsoft .NET

10. Informations sur le framework .NET : http://fr.wikipedia.org/wiki/Framework_.NET
11. Refactorisation : la refactorisation (anglicisme venant de refactoring) est une opération de maintenance du code informatique. Elle consiste à retravailler le code source non pas pour ajouter une fonctionnalité supplémentaire au logiciel mais pour améliorer sa lisibilité, simplifier sa maintenance, ou changer sa généricité (on parle aussi de remaniement).

III.1.3.2 Bases de données

Pour la création des prototypes nous avons utilisé une base de données MySQL. Ce type de base est légère et simple à mettre en place. Elle a permis de faire des tests rapidement. Nous avons pris le temps d'interfacer la base de données avec un mappeur objet-relationnel : NHibernate [12]. Cela nous a permis de nous abstraire de la base de données relationnelle et de nous focaliser sur les traitements.

Cependant, Cdiscount fonctionne majoritairement avec Microsoft SQL Server 2008. Une équipe est même spécialisée dans le contrôle du bon fonctionnement de la base et de son optimisation.

Pour aller plus loin nous avons migré de MySQL vers SQL Server 2008. Cela a impliqué beaucoup de changements puisque nous avons dû abandonner le mappeur objet-relationnel au profit de procédures stockées en Transact-SQL (extension SQL intégrée à la base SQL Server 2008). Cette méthode est plus performante mais demande du travail supplémentaire par rapport à un mappeur objet-relationnel.

FIGURE 9 – Logo SQL Server

III.1.3.3 Parallélisme

Afin d'optimiser nos applications nous avons fait appel à plusieurs bibliothèques C# dédiées à cet effet :

- les threads système (System.Threading),
- Task Parallel Library (TPL).

La première est l'implémentation haut niveau des fils d'exécution (threads). Elle permet la gestion des threads et des sections critiques (partie de code qui peut être exécutée par un seul thread à la fois).

La seconde est une surcouche utilisant les threads et proposant la notion de tâche. La programmation par tâches est beaucoup moins contraignante que la programmation avec des threads classiques. Cependant le contrôle des interactions entre tâches est moins précis que celui que l'on peut obtenir entre les threads. Pour information, cette bibliothèque est relativement récente puisqu'elle est apparu avec le framework 4 de .NET (début 2012).

La combinaison de ces deux bibliothèques nous a permis de limiter la complexité du code.

12. NHibernate : http://nhforge.org

III.1.3.4 Facebook SDK

Facebook ne propose pas directement de **SDK** (Software Development Kit ou kits de développement logiciels) C# pour communiquer avec son graphe (voir la partie III.2.1 consacrée à ce problème pour plus d'informations).
Nous avons eu recours à un SDK [13] open source mis à disposition par plusieurs développeurs volontaires. Il est aujourd'hui le standard C# et propose des fonctionnalités intéressantes :

- préparation des requêtes HTTP,
- exécution des requêtes de manière synchrone,
- exécution des requêtes de manière asynchrone.

Facebook expose un service web (Graph API) via des requêtes HTTP classiques. Cette utilisation est fastidieuse et le SDK permet de la surcharger.
Naturellement, le SDK nous permet d'envoyer ces requêtes à Facebook et de récupérer la réponse selon deux modes différents, synchrone ou asynchrone. Notons que le mode asynchrone s'appuie sur TPL (III.1.3.3).

III.1.3.5 Services web

Le développement des services s'appuie sur **Windows Communication Foundation** [14] (WCF). Ce dernier est un protocole s'appuyant sur SOAP (Simple Object Access Protocol) qui s'appuie lui même sur RPC (Remote Procedure Call). WCF est donc un protocole d'appel de procédure à distance. Il a l'avantage de proposer un standard de développement avec notamment une structure commune aux services web (3 couches).

Pour héberger nos services WCF nous avons utilisé **Internet Information Services**. IIS est très utilisé en environnement Microsoft : il est stable et fonctionnel.

FIGURE 10 – Logo IIS

III.1.3.6 Surcouche : framework propriétaire Cdiscount

Toutes les applications Cdiscount doivent utiliser une surcouche proposée par l'équipe architecture. Elle surcharge un grand nombre de classes pour y intégrer des fonctionnalités utiles à la société. Ce framework permet l'homogénéité des applications et assure qu'elles répondent bien toutes aux mêmes critères.

Voici une liste non exhaustive des fonctionnalités utilisées par les services que nous avons mis en place :

- la surcharge des traces : toutes les erreurs sont automatiquement loggées en base de données et accessibles sur un tableau de contrôle (dashboard),
- la surcharge de la connexion à la base : permet de gérer les chaînes de connexion par fichiers de configuration,

13. Facebook C# SDK : http://csharpsdk.org/
14. Windows Communication Foundation : http://fr.wikipedia.org/wiki/Windows_Communication_Foundation

– la surcharge service : propose des méthodes d'initialisation et de destruction du service (améliore la gestion de la mémoire).

III.1.3.7 JQuery Mobile

JQuery Mobile [15] est la déclinaison pour les mobiles (téléphones, tablettes, etc.) du célèbre framework Javascript : **JQuery** [16]. Pour rappel, JQuery est une surcouche de Javascript qui permet de développer plus simplement et plus efficacement dans ce langage.

La vocation de JQuery Mobile est légèrement différente. Ce nouveau framework est principalement orienté interface utilisateur (UI). L'objectif de JQuery Mobile est de faciliter la conception de sites internet pour les mobiles en terme d'interface mais aussi de performance.

FIGURE 11 – Logo de JQuery Mobile

III.1.3.8 ASP.NET MVC 3

ASP.NET regroupe l'ensemble des technologies Microsoft permettant de développer des sites internet dynamiques. C'est essentiellement le service IIS (III.1.3.5) qui permet l'exploitation d'ASP.NET.

MVC 3 [17] est l'application du patron de conception Modèle-Vue-Contrôleur (MVC). Avec MVC 3, Microsoft propose une interprétation de ce patron bien connu des développeurs. MVC décompose une application en 3 parties bien distinctes : le modèle, la vue et le contrôleur.

Il existe plusieurs implémentations de ce patron mais voici les principales spécifications de Microsoft MVC 3 :

– la vue ne contient pas de traitement, elle se contente d'afficher les données envoyées par le contrôleur,
– le contrôleur fait le lien entre la vue et les modèles et gère **le routage** (aiguillage vers la bonne vue),
– les modèles représentent le métier : traitements et données inhérentes au projet.

15. JQuery Mobile : http://jquerymobile.com
16. JQuery : http://jquery.com
17. Microsoft MVC 3 : http://www.asp.net/mvc/mvc3

III.1.4 Cheminement du projet

Ce chapitre présente la chronologie du projet. Le découpage n'a pas été décidé à l'avance.

Les deux premières semaines ont été consacrées à l'étude du graphe de Facebook. Il était nécessaire de se documenter et de comprendre le fonctionnement précis de cette gigantesque source de données.

J'ai ensuite capturé une partie de ces données pour les stocker en base. Un premier prototype a été réalisé en deux semaines environ. Il s'appuyait sur des outils simples et rapides à mettre en place.

Prometteur mais non règlementaire, il a fallu transformer le prototype en un service web et adopter les normes Cdiscount (intégration du framework propriétaire). Cette étape a été relativement longue (1 mois). En effet, le projet devait répondre aux normes WCF (III.1.3.5) et mettre en place un système de couches et de communication entre ces couches. Travaillant sur un gros volume de données, la communication entre ces couches est coûteuse en performances et en temps. Nous avons dû être très prudent à ce niveau.

Par la suite nous avons trouvé une application à ces données : proposer une page de gamification [18]. La gamification (ou ludification) est un concept qui consiste à appliquer des mécanismes de jeu dans d'autres domaines.
L'idée forte de l'application de gamification Cdiscount était de proposer des classements utilisateurs par rapport à leurs interactions sur la page Facebook de Cdiscount. Par exemple : classer les utilisateurs par nombre de "j'aime" sur les articles.

Cependant les données brutes telles quelles étaient stockées en base ne permettaient pas d'afficher efficacement des classements. Nous avons alors fait le choix de développer un nouveau service spécialisé dans la création de classement et permettant d'historiser ces classements dans la base de données (2 semaines de travail). Ainsi, le temps de création d'un classement était différé de l'affichage.
Ce service a double vocation :

- créer de nouveaux classements,
- récupérer des classements existants.

Enfin, dans l'objectif de présenter un produit fini à l'équipe marketing et de valoriser le travail effectué, nous avons élaboré un site web pour les mobiles permettant d'afficher les classements utilisateurs précédemment calculés (2 semaines environ).

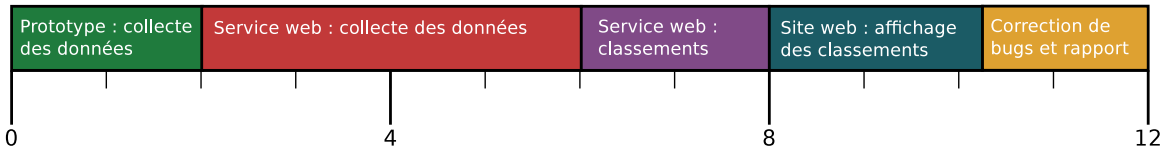

FIGURE 12 – Chronologie du projet en nombre de semaines

18. La gamification : http://fr.wikipedia.org/wiki/Ludification

III.2 Réalisation

III.2.1 L'Open Graph

L'**Open Graph** [19] est le coeur de fonctionnement de Facebook. C'est lui qui assure la cohérence entre les données.

Dans ce graphe non orienté, les utilisateurs, les pages et les applications sont des noeuds.

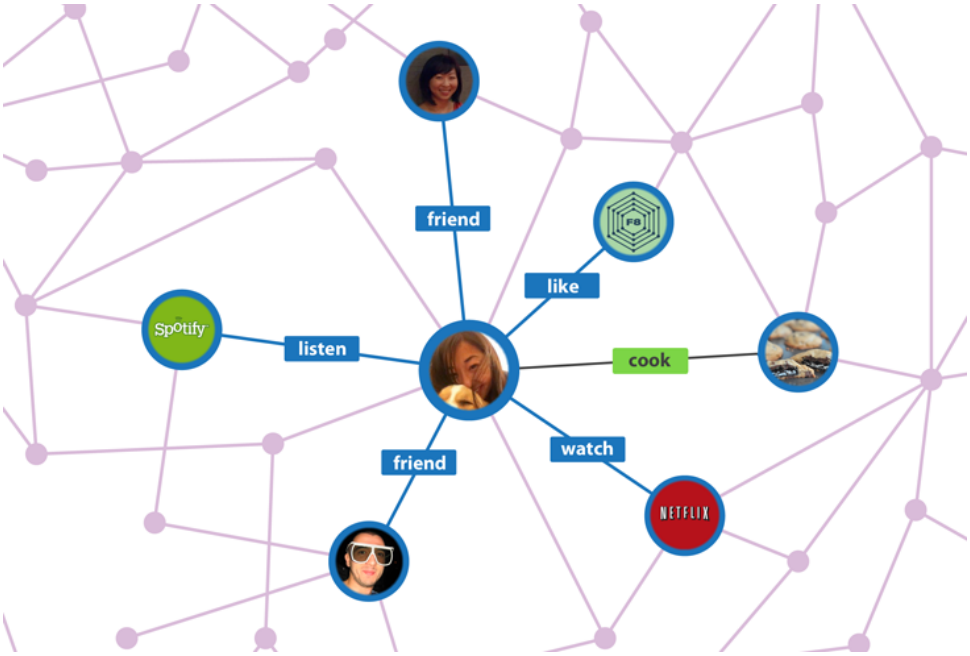

FIGURE 13 – Illustration de l'Open Graph

Les actions (likes, commentaires, être amis avec, etc.) sont les arêtes. Les principes de base sont relativement simples et c'est en partie ce qui fait la force de l'Open Graph de Facebook.
Mais ce qui présente un réel intérêt pour les développeurs c'est que ce graphe est ouvert ("Open" Graph). En effet, il est possible de communiquer directement avec le graphe en récupérant, ajoutant ou modifiant des informations (des objets Facebook).

Dans cette optique, Facebook met la **Graph API** [20] à notre disposition. Cette dernière est un protocole de communication entre une application tierce et le graphe de Facebook.

En réalité, la Graph API et l'Open Graph de Facebook se combinent pour proposer un service web répondant au format de données JSON (format texte alternatif à l'XML, plus concis et moins verbeux, il est moins coûteux en bande passante). Facebook propose aussi une interface d'exploration de son graphe [21], elle permet "d'attaquer" facilement les objets Facebook pour mieux visualiser le fonctionnement général du système.

Enfin, l'Open Graph profite de sécurités assez convaincantes. Les données de Facebook ne sont

19. Open Graph : https://developers.facebook.com/docs/opengraph/
20. Graph API : https://developers.facebook.com/docs/reference/api/
21. Graph API Explorer : https://developers.facebook.com/tools/explorer/

pas toutes publiques et pour accéder aux données privées il faut s'authentifier. Cette authentification s'appuie sur un **jeton d'accès** attribué pour un utilisateur ou une application. Ce jeton possède une durée de vie limité (60 à 120 minutes pour les jetons classiques). Une fois cette durée dépassée il faut renouveler son jeton auprès de Facebook.
Ce système permet d'éviter les gros débordements puisque le temps d'une action négative sur le réseau social est limitée au temps de validité du jeton.

L'objectif étant de récupérer les informations de la page Facebook de Cdiscount, il nous a fallu mettre une stratégie en place pour parcourir le graphe efficacement. Toutes les données ne sont pas toujours accessibles directement et le coût de recherche d'une information peut parfois devenir énorme. De plus le graphe ne permet pas l'accès à toutes les informations et parfois les informations ne sont que partiellement accessibles.

Après analyse du graphe, il s'avère qu'il est impossible de connaître précisément les personnes qui ont partagé un article : il est seulement possible de connaître le nombre de partage d'un article. De même pour les utilisateurs qui aiment (like) la page de Cdiscount, seul le nombre de "j'aime" peut être récupéré.
Enfin, et c'est rassurant, les informations sur les utilisateurs sont très limitées (noms et identifiants seulement).

Le schéma suivant expose la stratégie que nous avons adoptée pour parcourir le graphe. En noir ce sont les objets Facebook (les noeuds du graphe) en blanc ce sont les interactions (les arêtes du graphe). Les mots clefs sont en anglais, à l'identique du graphe Facebook.

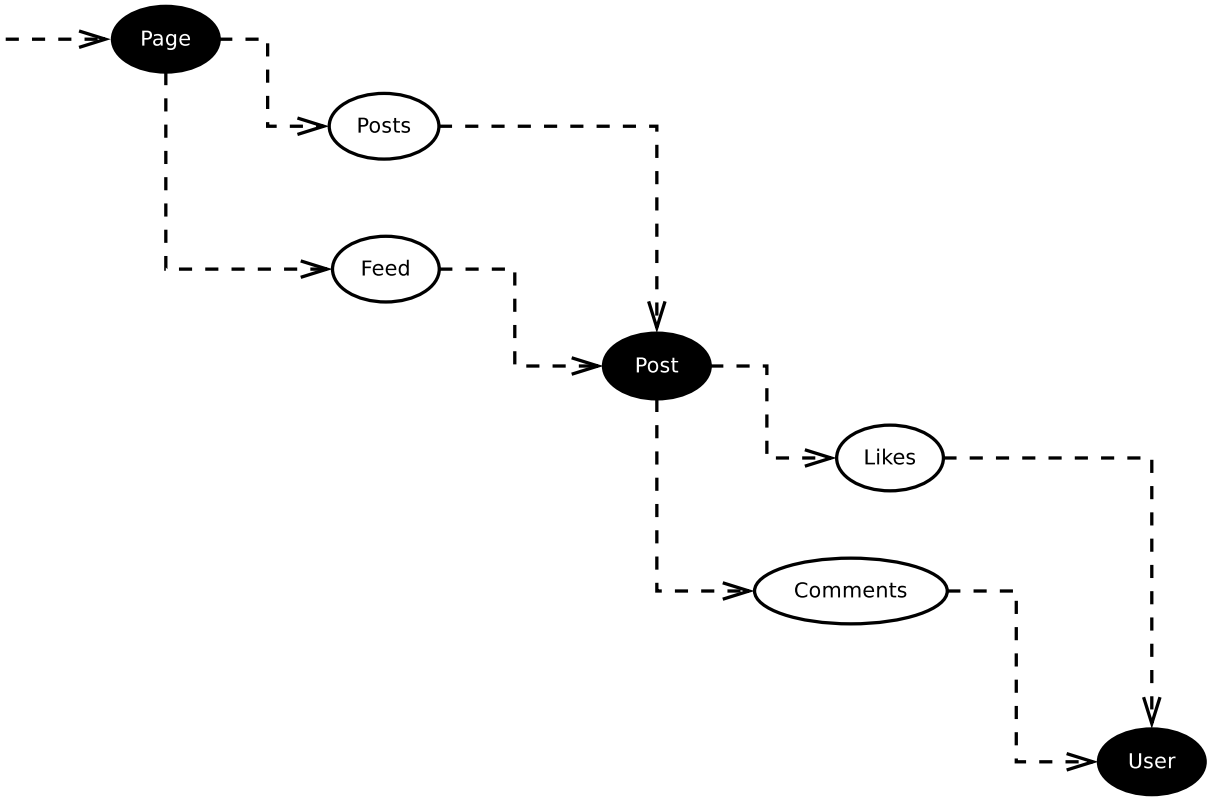

Figure 14 – Parcours du graphe de Facebook

Le point d'entrée de notre parcours est la page de Cdiscount [22] à ce niveau nous avons pu récupérer le nombre de "j'aime" que possède la page ainsi que le nombre de personnes qui parlent de la page.
Les pages proposent l'accès à plusieurs listes mais les deux qui nous intéressent sont "posts" et "feed" :

- la première contient la liste intégrale de tous les articles (post en anglais) proposés par les administrateurs de la page,
- la seconde contient la liste complète de tous les articles soumis par les utilisateurs de la page.

Pour chaque article nous avons pu récupérer les informations suivantes :

- la date de création,
- la date de modification,
- le type (image, texte, vidéo, lien, etc.),
- le nombre de personnes qui l'ont partagé,
- l'utilisateur qui a créé l'article,
- les "likes",
- les commentaires.

Les deux dernières informations (les "likes" et commentaires) sont des arêtes du graphe et sont naturellement reliées aux utilisateurs (c'est la sortie de notre parcours).

Chaque objet Facebook possède un identifiant unique et c'est de cette manière que l'on peut se déplacer dans le graphe.

Enfin, la Graph API impose certaines limites en termes de quantité de données : de manière générale il est possible de récupérer au maximum 500 entrées pour une requête (ce nombre peut varier, Facebook ne donne aucune garantie). Cependant cette limite est actuellement revu à la baisse pour les articles : sans passer par des critères spécifiques (la date), la limite est abaissée aux 200 derniers articles d'un objet (ici l'objet est une page).
Cette information est appréciable car la connaissance de ces limites permet de constituer le nombre optimal de requêtes pour récupérer la totalité des informations.

22. L'objet page Cdiscount du graphe : https://developers.facebook.com/tools/explorer/?method=GET&path=cdiscount

III.2.2 Service web : Facebook Collector

Après l'analyse des possibilités du graphe il a fallu mettre en place une solution capable de capturer ces données pour les sauver dans la base de données Cdiscount.

Dans un premier temps nous avons réfléchi à la décomposition des données en différents objets. Voici le diagramme de classes simplifié que nous avons adopté :

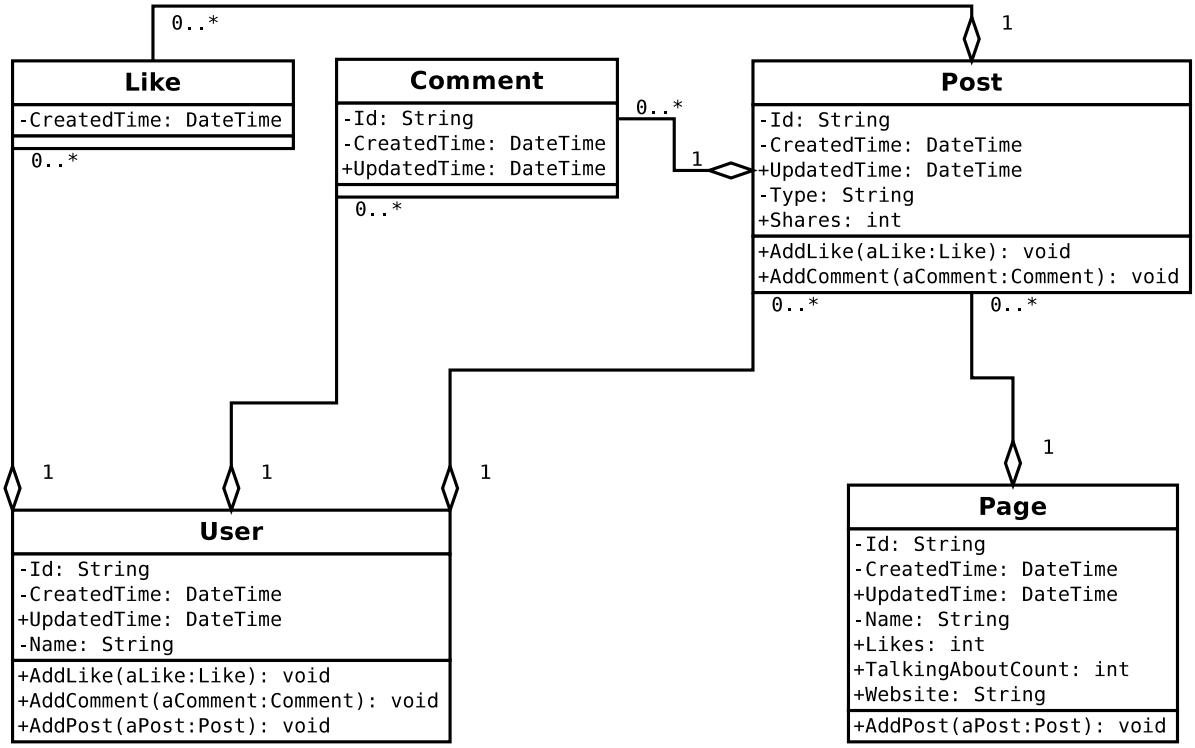

FIGURE 15 – Diagramme de classes du service collecteur

Sans surprise on retrouve les objets de l'Open Graph décrit dans le chapitre précédent (III.2.1). La notation anglaise s'appuie sur celle du graphe de Facebook (par souci de cohérence).

Pour être homogène avec les méthodes de travail habituelles de Cdiscount, nous avons opté pour la création d'un service web et nous l'avons baptisé du doux nom de : **Facebook Collector Service** (le service qui collecte les données depuis Facebook).

Chez Cdiscount l'architecture service repose sur trois couches :
- une couche de données (mapping objet-relationnel à la main),
- une couche de traitement,
- une couche de présentation.

Un utilisateur lambda ne peut interagir qu'avec la couche de présentation alors que d'autres services peuvent, eux, directement communiquer avec la couche de traitement pour être plus efficaces.
Au sein d'un même service, les couches ne peuvent communiquer qu'avec la couche précédente ou suivante (ex : la couche présentation ne peut pas communiquer directement avec les données).

L'intérêt de cette architecture est de séparer les différents traitements pour une meilleure réutilisabilité cependant elle peut parfois complexifier le projet. En effet, la communication entre les couches nécessite un surplus de traitement et amène souvent une redondance des données.

Le service Facebook Collector propose une méthode unique : **collecter les données d'une page Facebook**. Elle prend plusieurs paramètres :

- un identifiant de page Facebook (obligatoire),
- la date à partir de laquelle la collecte débute (facultatif),
- la date jusqu'à laquelle termine la collecte (facultatif).

Cette méthode à pour effet d'envoyer une multitude de requêtes à l'Open Graph via la Graph API. Dans un premier temps nous avons réalisé une version naïve qui envoyait les requêtes les unes après les autres pour ensuite les traiter (envoie d'une requête puis réception de la réponse puis traitement de la réponse puis envoie d'une requête, etc.).
Même si les traitements sont légers, ce type d'architecture est inefficace (très lente) car l'attente d'une réponse est trop longue (plus de 5 secondes sur de grosses requêtes). Il y a deux raisons à ce temps d'attente trop long :

- la latence sur internet est élevée (par rapport à un réseau local),
- Facebook met du temps à traiter certaines requêtes complexes.

Nous avons donc pensé une architecture parallélisée, plus efficiente s'appuyant sur l'envoi de **plusieurs requêtes simultanées** ! Le schéma ci-dessous (16) résume notre architecture multi-thread (en vert ce sont les traitements simultanés).

Nous avons ensuite configuré le service WCF pour qu'il crée une unique instance (mode singleton). De cette manière nous avons géré les appels concurrents : nous avons autorisé une seule exécution de la méthode en simultané.
Ce choix s'explique simplement : l'architecture de la méthode de collecte a été parallélisée de manière à saturer la bande passante et les exécutions côté serveur Facebook. Il serait donc vain d'exécuter plusieurs instances car si elle est bien configurée, une seule instance sature déjà les ressources disponibles.

En termes de performance, une collecte complète de la page de Cdiscount prend environ 300 secondes (5 minutes). Cela représente approximativement 68 000 utilisateurs, 400 articles, 9 000 commentaires, 140 000 "likes" et 1 page soit 217 401 objets. Plus de la moitié de ce temps est dû à la sauvegarde des données dans la base de données de Cdiscount.
La consommation CPU (Central Processing Unit) reste correcte sans jamais dépasser les 50% de la capacité d'un AMD Athlon 64 X2 [23] (2 Ghz par coeur) en pleine charge. Le goulot d'étranglement se situe plutôt au niveau de la bande passante d'internet.
La consomation mémoire est assez élevée avec une occupation moyenne de 500 Mo de la mémoire vive. Cela s'explique par la grande quantité d'objets créés.

23. AMD Athlon 64 X2 : http://fr.wikipedia.org/wiki/Athlon_64_X2

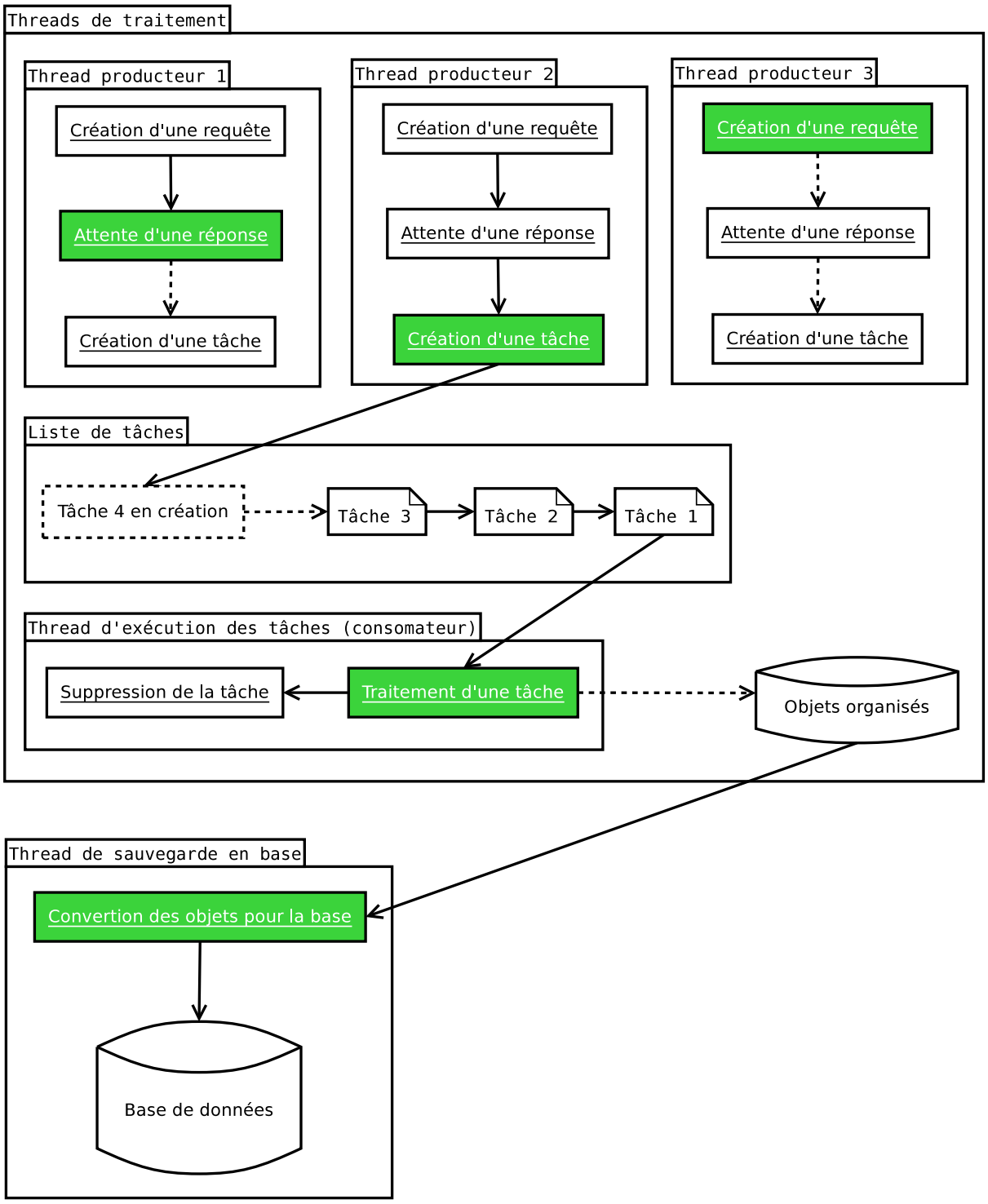

FIGURE 16 – Architecture multi-thread du collecteur

III.2.3 Service web : Facebook Aggregator

Une fois les données brutes collectées nous avons réalisé un nouveau service permettant de calculer des classements utilisateurs. Voici le diagramme de classes sur lequel nous nous sommes appuyés :

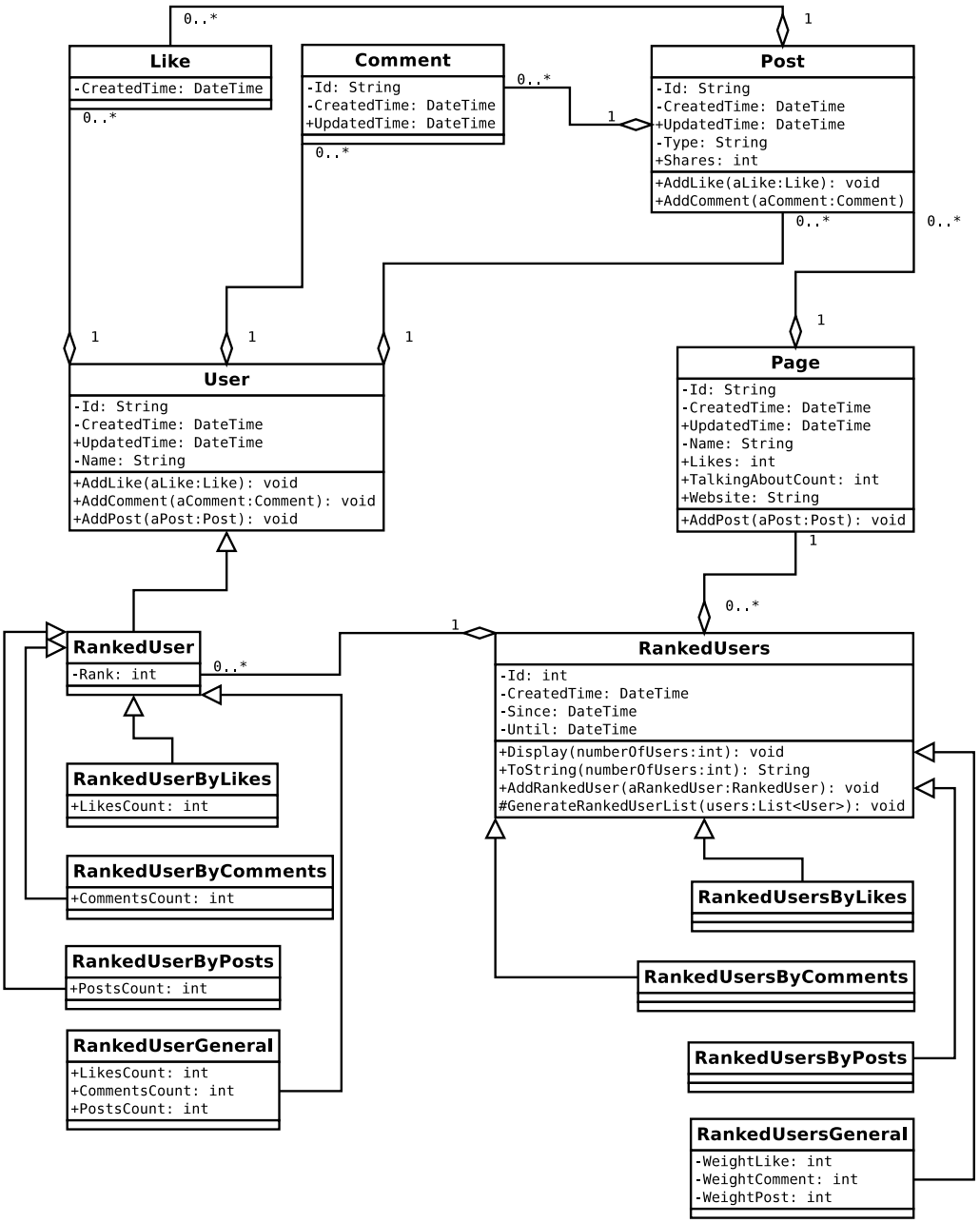

FIGURE 17 – Diagramme de classes du service agrégateur

On remarque l'apparition de deux classes abstraites : ***RankedUser*** (un utilisateur classé) et ***RankedUsers*** (un classement d'utilisateurs).
Un classement d'utilisateurs contient plusieurs utilisateurs classés. Un utilisateur classé hérite d'un utilisateur normal et apporte la notion de rang (rank sur le diagramme). Un classement

possède une date de création et deux dates (since et until) qui déterminent l'intervalle de temps traité par le classement. Cela permet de faire des classements sur les derniers mois (par exemple) au lieu de les faire sur toutes les données disponibles.

Les classes qui héritent de *RankedUser* et *RankedUsers* spécifient le type de classement (par commentaires, par articles postés, par "j'aime", général, etc.).

Pour calculer et récupérer ces classements nous avons adopté la même politique que pour la précédente collecte de données : la création d'un service web (nommé **Facebook Aggregator Service**).
Il n'a pas été nécessaire de paralléliser l'algorithme de classement car les performances étaient suffisantes sans utiliser le matériel de façon plus soutenue. De plus, il est bon de rappeler que ce type de programmation implique une complexification du code source et cela peut devenir un obstacle à l'évolution d'un projet qui n'en a pas forcément besoin.

III.2.3.1 Création des classements

La création des classements s'appuie sur une base de code source générique permettant ainsi de réaliser facilement de nouveaux classements. Un nouveau classement hérite nécessairement de *RankedUsers*, et les nouveaux utilisateurs classés héritent de *RankedUser*.
La fonctionnalité de tri est assurée par tous les *RankedUsers* via la méthode ***GenerateRankedUserList***. Cette dernière prend une liste d'utilisateurs classiques en paramètre et s'appuie sur un comparateur .NET pour effectuer le tri.

En réalité, la classe "liste" (*List*) en .NET propose une méthode de tri (*Sort*). Du point de vue algorithmique elle s'appuie sur Quicksort [24] procédant ainsi par dichotomie. Sort offre différents prototypes mais le plus intéressant reste celui qui prend un objet implémentant ***IComparer*** (le fameux comparateur) en paramètre. Grâce à l'*IComparer* on peut définir précisément l'ordre à appliquer entre deux objets de même types (des objets de type *User* ici).

Du point de vue service, nous avons mis en place un système chaînant les appels aux différents tris de manière séquentielle car l'exécution simultanée de plusieurs tris est très coûteuse en CPU et en mémoire. La solution de chaînage a l'avantage de ne pas surcharger le serveur d'exécution.

III.2.3.2 Récupération des classements

La difficulté a été de construire des requêtes SQL efficaces pour récupérer rapidement les classements. Pour y parvenir nous avons adopté plusieurs stratégies :

- sélectionner uniquement les champs nécessaires,
- diminuer le nombre d'entrées retournées,
- utiliser la fonctionnalité de mise cache proposée par le framework Cdiscount.

Afin de sélectionner un minimum de champs et de diminuer les temps de calcul, chaque utilisateur classé contient un compteur (*LikesCount*, *CommentsCount*, *PostsCount*) correspondant au nombre sur lequel se base le classement. Cela permet de ne pas avoir à récupérer tous les "j'aime", commentaires et/ou articles par exemple. De plus cela permet une meilleure historisation des classements tout en évitant de devoir recompter après leurs récupérations (et donc perdre du temps).

24. Quicksort ou tri rapide : http://fr.wikipedia.org/wiki/Tri_rapide

Ensuite, nous avons mis en place une sélection des n premières entrées uniquement. En effet, on souhaite rarement afficher tous les utilisateurs d'un classement. Le fait de sélectionner uniquement un nombre réduit d'utilisateurs en base améliore considérablement les performances.

Enfin, nous avons utilisé la mise en cache des appels aux méthodes. Cette dernière s'appuie sur le cache ASP.NET et donc sur IIS. Cela permet d'éviter de renvoyer constamment les mêmes requêtes à la base et d'économiser des traitements inutiles de conversion de données. L'utilisation de cette fonctionnalité accélère beaucoup le temps de réponse du service et allège les accès à la base de données.
La durée de mise en cache est configurable. Nous avons arbitrairement choisi de la positionner sur 4 minutes.

III.2.4 Le site web mobile

Pour valoriser le produit, nous avons fait le choix de présenter les classements, précédemment calculés, via un site mobile. En tant que pôle R&D nous avons souhaité proposer une interface moderne. Pour y parvenir nous nous sommes massivement appuyés sur JQuery Mobile (III.1.3.7) pour proposer une interface au look "arrondi" et capable de s'adapter à toutes les résolutions des appareils mobiles.

M. Yann Treuiller, graphiste chez Cdiscount, a apporté sa contribution en nous proposant un logo adapté. Les sites mobiles étant relativement simplistes pour gagner en lisibilité, les éléments graphiques n'ont pas à être aussi nombreux que sur les sites classiques. La seule présence du logo a suffi pour donner un aspect visuel sympathique.

Ci-dessous une capture d'écran de l'accueil du site. Nous avons aussi mis en ligne un prototype statique dédié à la visualisation de l'interface [25].

Figure 18 – Capture d'écran de l'interface mobile : l'accueil

La création du site a fait l'objet de l'expérimentation d'une nouvelle technologie Microsoft : .NET MVC 3 (III.1.3.8). Pour avoir personnellement testé plusieurs frameworks PHP (Joomla Framework et Symfony), j'ai été forcé de constater que la technologie Microsoft propose un standard clair et simple à mettre en place rapidement (quand on connait les patrons de concep-

25. Prototype de l'interface mobile : http://www.potionmagic.eu/topfan/

tion utilisés). De plus, le gain de temps gagné en utilisant cette technologie grandit au fur et à mesure de l'évolution du projet.

En utilisant plusieurs technologies Microsoft, l'avantage est de pouvoir les combiner simplement. C'est le cas du site web qui "consomme" directement le service Facebook Aggregator (III.2.3) pour afficher des classements. La connexion au service WCF s'effectue très naturellement en renseignant l'adresse de ce dernier.

Ci-dessous une capture d'écran du site web présentant le classement des 3 meilleurs likers (personnes qui ont le plus aimé, bouton "j'aime" de Facebook) :

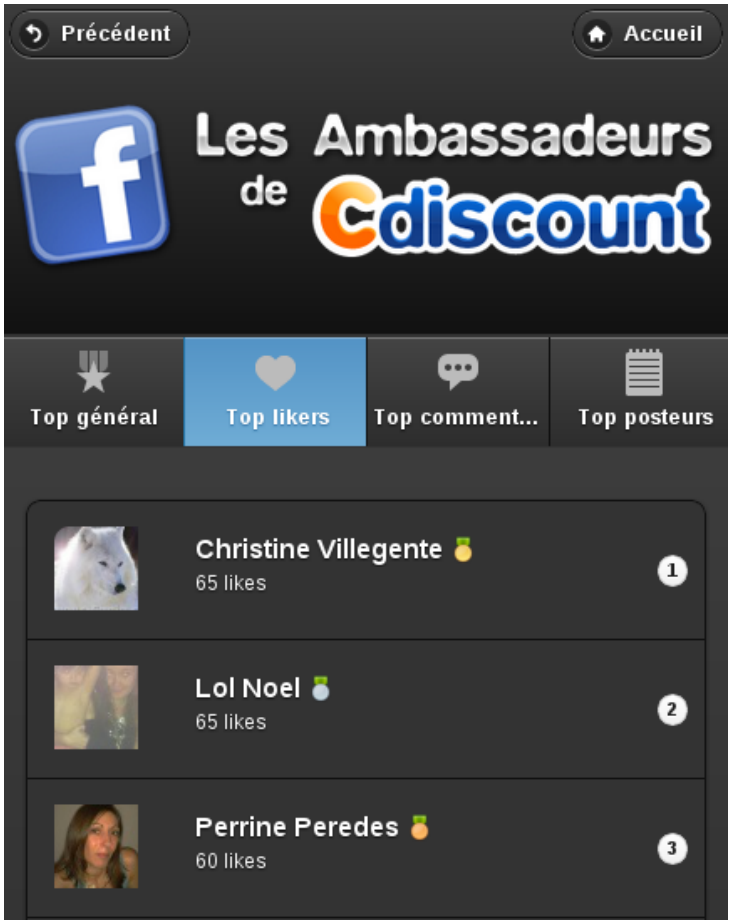

FIGURE 19 – Capture d'écran de l'interface mobile : un classement

Sur cette vue nous avons rajouté un menu permettant de naviguer entre les différents classements ainsi qu'un bouton pour revenir à l'accueil et un bouton pour revenir en arrière.
Cette interface ressemble beaucoup à une application mobile native et offre la possibilité d'être transformée dans ce format pour améliorer la réactivité (il existe des logiciels permettant de le faire automatiquement).

IV Conclusion

Ce stage a été l'occasion, pour moi, de découvrir une société de grande taille. J'ai pu mesurer toute la complexité liée à cette architecture et appréhender une partie des méthodes qui permettent son fonctionnement. On remarque la présence d'une hiérarchie même si la volonté de Cdiscount est d'éviter qu'elle contienne trop de niveaux.
De plus la société est découpée en équipes qui sont spécialisées dans différents domaines. Le travail est réparti par tâches entre les équipes.

De ce point de vue le Lab social est un peu particulier, il est en marge du fonctionnement classique (avec des équipes spécialisées). La vocation du lab n'est pas de produire ou d'être rentable à court terme mais plutôt d'apporter des idées et de nouveaux concepts. Le lab est donc relativement autonome.

Cette façon de travailler m'a particulièrement intéressé puisqu'elle m'a permis de "toucher un peu à tout". En effet, tout en respectant les normes Cdiscount j'ai pu expérimenter mon projet de la base de données en passant par la couche service jusqu'à l'élaboration d'un site internet. Je pense que cela m'a apporté une bonne vision sur l'ensemble des technologies et sur l'architecture informatique de Cdiscount. Au-delà du monde Microsoft, ce sont des choix stratégiques très cohérents.

Cela ne m'a pas dispensé de l'aide de différents spécialistes. Mes tuteurs, M. Gregory Taudin et M. Fabien Grosyeux ont souvent dû me venir en aide pour que je puisse continuer à avancer. J'ai aussi dû faire appel aux compétences de l'équipe architecture, de l'équipe service et de l'équipe graphiste de Cdiscount.

J'ai personnellement constaté un temps d'adaptation plus long que lors de mes précédents stages. Je crois que c'est essentiellement dû à l'apprentissage de technologies que je ne connaissais pas. Les bases théoriques avec notamment la connaissance de certains patrons de conception (MVC, mapping objet-relationnel, notion objets, etc.) m'ont beaucoup aidé à comprendre ce nouveau monde.

Au terme de mon stage je retire une expérience très positive au sein d'une équipe dynamique, agréable et efficace. Je pense avoir mené à bien le projet que l'on m'a confié et j'espère sincèrement qu'il sera utile à l'entreprise.

V Annexes

V.1 Base de données sociale

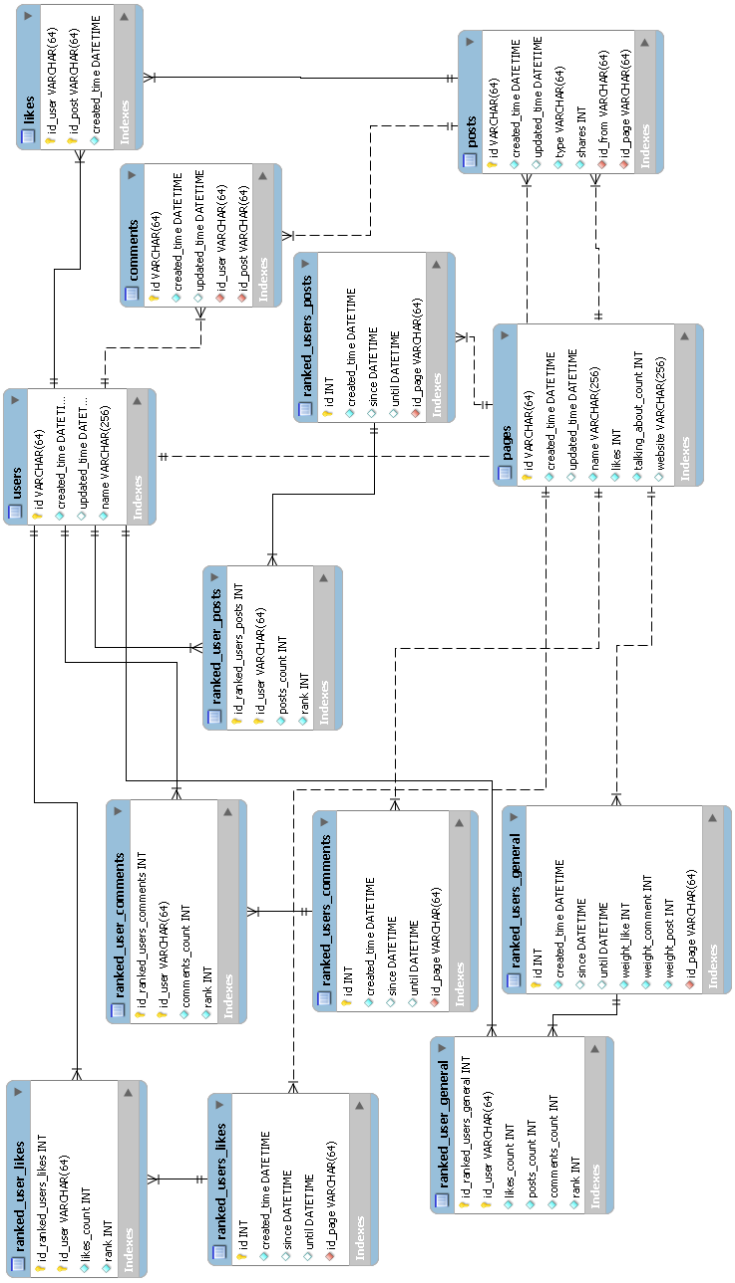

FIGURE 20 – Base de données sociale

V.2 Architecture du projet

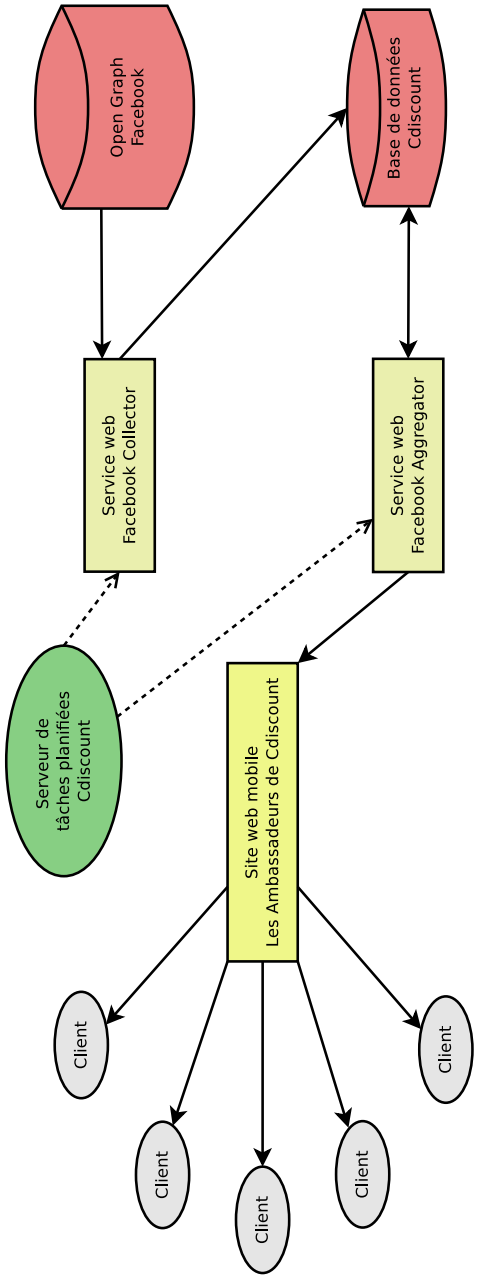

FIGURE 21 – Architecture du projet

Bibliographie

[1] Ezalys. Acquisition de fans facebook – n°7 : la viralité. http://www.ezalys.com/2011/acquisition-de-fans-facebook-la-viralite.html, 2011.