Mise en place d'un site type Web 2.0 sur un Cloud

Olivier Leclère

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE REGIONAL ASSOCIE DE

Saint-Genis Pouilly

MEMOIRE

présenté en vue d'obtenir

le DIPLOME D'INGENIEUR CNAM

SPECIALITE : INFORMATIQUE

OPTION : SYSTÈMES D'INFORMATION

par

Olivier Leclère

_____

Mise en place d'un site type Web 2.0 sur un Cloud

Soutenu le 24 juin 2010

_____

JURY

PRESIDENT : **Christophe Picouleau**

MEMBRES : **Bertrand David**

Claude Genier

Christine Aïdonidis

Jean-Philippe Trabichet

Résumé

Le « Cloud Computing » devrait d'ici deux à cinq ans devenir une des briques de base des plates-formes informatiques de prochaine génération. Le Laboratoire des Technologies de l'Information (LTI) de la Haute École de Gestion (HEG) de Genève, dont les missions sont de faire de la recherche et de valoriser cette dernière au travers de mandats, a voulu comprendre le concept de « Cloud Computing ».

La première partie de ce mémoire est consacrée à la définition du « Cloud Computing » et présente les cinq concepts fondamentaux de cette nouvelle technologie. Le deuxième chapitre présente les différents types d'applications « Cloud » en les classant en trois catégories (« infrastructure », « platform » et « software »). Les troisième et quatrième parties sont consacrées à la mise en pratique du concept. Le troisième chapitre est consacré à la définition des besoins, des objectifs à atteindre tout en respectant certaines contraintes. Dans cette troisième partie est choisie la plate-forme de « Cloud Computing » pour héberger un projet de « folksonomie ». Ce dernier, qui s'appuie sur les concepts du Web 2.0, consiste à demander à l'internaute de classifier les pages d'un site Web en utilisant ses propres mots-clés. Ce projet facilite la recherche d'information pour les autres internautes. Il leurs permet aussi de donner plus d'importance à certains mots-clés en les évaluant. La mise en œuvre de ce projet est décrite dans le quatrième chapitre.

Cette application Web 2.0 permet de constater les avantages, les inconvénients et les contraintes liées au développement sur un « Cloud » et de vérifier certains des concepts fondamentaux.

Mots-clés : Cloud Computing, pyramide du Cloud, folksonomie, définition, concept, implémentation

Summary

"Cloud Computing" is set to become one of the pillars of next-generation computing. The Laboratory of Information Technologies (LTI) at the University of Applied Sciences Western Switzerland (HEG) in Geneva, whose missions are to conduct research and development, aims to understand "Cloud Computing" concepts and conduct a pilot project using the technology.

This engineering thesis is divided into four chapters. The first chapter defines "Cloud Computing" and presents the technology's five fundamentals concepts. The second chapter presents the different types of "Cloud" solutions and classifies them in three categories ("infrastructure", "platform" and "software"). The third and fourth chapters cover practical aspects. The third chapter defines the requirements and objectives that will allow the LTI team to host a "folksonomy" project on a "Cloud". The project, based on Web 2.0 concepts, consists in asking Internet users to classify the pages of a Web site by using their own keywords. This project facilitates information retrieval for other Internet users. The application, also allows users to rank keywords thus giving more importance to better-ranked words. The implementation of the "folksonomy" project is described in the fourth chapter.

This Web 2.0 application will help determine the advantages, disadvantages and limitations specific to the development of applications on a "Cloud" and will validate the fundamental concepts of "Cloud Computing" defined in the first chapter.

Keywords: Cloud Computing, cloud pyramid, folksonomy, definition, concepts, application

Table des matières

Remerciement

Je tiens à remercier en premier lieu mes collègues du Laboratoire des Technologies de l'Information (LTI) de la Haute École de Gestion (HEG) de Genève et plus particulièrement le Professeur Jean-Philippe Trabichet qui me soutient depuis plus de sept ans dans tous mes projets et qui m'a encouragé à m'inscrire à l'EiCNAM.

Je remercie également Messieurs Gérard Ineichen, Cyril Déchelette et Jérémie Blanchard, mes collègues du centre informatique de la HEG pour le temps qu'ils ont consacré au montage des environnements de développement et de tests qui m'ont été nécessaires.

Je souhaiterai également remercier mes collègues du Centre des Technologies de l'Information (CTI) de l'État de Genève qui m'ont encouragé dans mes recherches en matière de « Cloud Computing ». Je remercie mes collègues Madame Christine Aïdonidis et Monsieur Giorgio Pauletto de l'observatoire technologique et systèmes d'information du CTI pour leurs conseils avisés. Je tiens aussi à remercier Monsieur Julien Conti pour son aide en matière de tests d'accessibilité dans les pages Web.

Merci à Monsieur Claude Genier qui a encadré ce mémoire pour son aide précieuse. Je remercie aussi tous ses collègues du CNAM qui m'ont fait découvrir de nouveaux domaines aussi bien dans l'informatique, que dans les branches de gestion.

Je remercie aussi tous les membres du Comité Interdépartemental des Chargés de Communication (CICC), présidé par Madame Anja Wyden Guelpa, Chancelière d'État, qui ont montré un très grand enthousiasme pour le projet « folksonomie ».

Pour finir, je remercie également mon épouse pour son soutien tout au long de mes études au CNAM et surtout ces douze derniers mois lors de la réalisation de ce mémoire d'ingénieur.

Abréviations

AJAX – Asynchronous JavaScript And XML

AMI ou AMIs – Amazon Machine Images

ASP – Active Server Pages

ASP – Application Service Provider

CPU – Central Processing Unit

DNS – Domain Name System

DTD – Document Type Definition

EC2 – Elastic Compute Cloud (Amazon)

FTP – File transfer protocol

GPL – General Public License

GUI – Graphical User Interface : environnement graphique

HA – « high availability » : Haute disponibilité

HaaS – Hardware as a Service

IaaS – Infrastructure as a Service

IIS – Internet Information Services, anciennement Internet Information Server

JSON – JavaScript Object Notation

JSP – JavaServer Pages

NIST – National Institute of Standards and Technology (États-Unis)

PaaS – Platform as a Service

PHP – Hypertext Preprocessor

PME – Petite et moyenne entreprise

RAM – Random-access memory

RIA – Rich Internet Application

SaaS – Software as a Service

SAN – Storage Area Network

SLA – Service-Level Agreements

SOA – Service Oriented Architecture

SSL – Secure Sockets Layer

VM – Virtual machine : machine virtuelle

VNC – Virtual Network Computing

VPC – Virtual Private Cloud

VPN – Virtual Private Network

WAI – Web Accessibility Initiative

WPF – Windows Presentation Foundation

Glossaire

Cluster : Le « Cluster » ou « Grappe » est un concept architectural en informatique. Il consiste à regrouper des ordinateurs / serveurs pour former une « machine » offrant de meilleures performances, une disponibilité accrue et une facilité de montée en charge.

Cross domain : Principe qui vise à faire communiquer deux domaines (sites Web) ensemble.

Folksonomie : Néologisme désignant un système de classification collaborative décentralisée et spontanée, basé sur une indexation effectuée par des non-spécialistes [WIKI1].

Freemium : Modèle économique qui consiste à offrir un service de base gratuitement et à faire payer les services supplémentaires.

Middleware : Dans un « Cloud » - couche logicielle servant d'intermédiaire entre l'application et les infrastructures techniques.

Twitt : message de 140 caractères posté sur le site de microbloging Twitter.

Web Service : Programme permettant l'échange de données entre des applications.

Introduction

Le « Cloud Computing » a été défini, par Tim O'Reilly, PDG de O'Reilly Media, lors du congrès Web 2.0 en avril 2008[1], comme une des briques de base des plates-formes informatiques de prochaine génération. Le Gartner a confirmé cette annonce durant l'été 2008[2], en qualifiant dans son rapport annuel le « Cloud Computing » comme une technologie émergente.

Ce concept technologique répond aux besoins croissants des entreprises qui veulent réduire leurs coûts liés à l'informatique, tout en ayant accès à une large offre de services qui vont du hardware au software.

Le Laboratoire des Technologies de l'Information (LTI) de la Haute École de Gestion (HEG) de Genève a comme mission de faire de la recherche et de valoriser cette dernière au travers de mandats. Le LTI travaille principalement sur des thématiques liées au Web avec des sujets comme l'Administration en Ligne (AeL), l'infobésité et le Web 2.0. Le « Cloud Computing » devant devenir la plate-forme informatique de prochaine génération, il nous a semblé important de comprendre comment ce nouveau concept technologique fonctionne et ce que nous pouvons en faire.

[1] Joyent, Tim O'Reilly, *What is Cloud Computing*, http://www.youtube.com/watch?v=6PNuQHUiV3Q, publié le 7/05/2008, visionné le 15/10/2009

[2] Gartner, *Gartner Highlights 27 Technologies in the 2008 Hype Cycle for Emerging Technologies*, http://www.gartner.com/it/page.jsp?id=739613, publié le 11/08/2008, consulté le 12/03/2009

Ce mémoire est composé de quatre chapitres principaux. Le premier chapitre définit le concept de « Cloud Computing » en partant de la genèse, puis en détaillant chaque notion qui fait du « Cloud » ce qu'il est. Le deuxième chapitre se focalise sur les solutions de type « Cloud Computing ». Le troisième et le quatrième chapitre présentent une mise en pratique avec la réalisation d'un prototype fonctionnel de site type Web 2.0 hébergé sur un « Cloud ». Nous commençons ce projet en choisissant la solution de « Cloud Computing » la plus adaptée, aux besoins de notre mandant qui est l'Administration cantonale genevoise. Puis nous détaillerons toute la partie technique du projet. Nous finirions ce dernier par des recommandations pouvant l'améliorer.

1 Définition du « Cloud Computing »

Dans ce premier chapitre, nous allons tout d'abord présenter les concepts antérieurs au « Cloud Computing » qui ont permis l'émergence de ce nouveau type de plate-forme informatique. Nous présenterons ensuite l'historique du « Cloud Comptuing » et l'origine de ce terme. Puis nous développerons les cinq concepts de base qui permettent de qualifier ce qu'est le « Cloud Computing ». Nous conclurons ce premier chapitre en donnant notre propre définition du « Cloud Computing ». Cette définition sera basée sur une analyse des définitions proposées par le monde académique et sur la réflexion que nous aurons conduite dans ce chapitre.

1.1 Genèse du « Cloud Computing »

Le « Cloud Computing » est l'évolution de concepts informatiques étudiés et développés depuis les années septante. Ces concepts ont évolué avec la technologie, les besoins des utilisateurs et un besoin constant de réduire les coûts liés à l'informatique. Dans cette section, nous commencerons notre voyage dans le temps avec l'« Utility Computing », puis le « Grid Computing » et nous la finirons avec les « Server farms » ainsi que les « Web farms ».

1.1.1 « Utility Computing »

L'« Utility Computing » ou « Calcul à la Demande » est un concept qui revient régulièrement dans l'histoire informatique.

En 1974, George J. Feeney, dans sa synthèse d'une table ronde « Utility computing – A superior alternative » [F_74], introduit le concept d'« Utility Computing ». Dans cet article il annonce que les 60'000 sociétés qui avaient un mainframe dans les années septante, devraient suivre le même chemin que l'industrie. Au début du XXème siècle, les usines produisaient leur électricité pour leurs propres besoins. En moins d'un demi-siècle, ces dernières ont externalisé leur production d'énergie afin de rationaliser les coûts. Pour George J. Feeney, les sociétés modernes devraient à terme aussi externaliser leur informatique principalement pour des raisons économiques.

Plusieurs facteurs ont conduit à ce changement dans la consommation d'électricité par l'industrie. Il s'agit des économies d'échelle, du fait que les coûts peuvent être variables et que la technologie de production de cette énergie évolue rapidement. George J. Feeney rapporte aussi que grâce à l'« Utility Computing » les sociétés et les universités pourront utiliser des machines sophistiquées en ne payant que ce qu'elles consomment.

En 2002[3], les sociétés IBM, Sun et HP investissent dans le concept d'« Utility Computing » et proposent toutes les trois leur propre solution. L'argumentaire[4] [5] [6] de ses prestataires de services est une diminution des coûts, une simplification de la gestion des infrastructures et une possibilité d'augmenter ou de diminuer la puissance de calcul en fonction des besoins. Depuis, d'autres acteurs, comme Microsoft, Google ou Amazon ont lancé leur propre solution.

Le concept d'« Utility Computing » est rémanent et toujours à la base des projets d'externalisation des solutions informatiques. Nous retrouvons les mêmes arguments économiques pour le « Grid Computing », les « Web Farms » et le « Cloud Computing » ; seuls les concepts techniques évoluent.

1.1.2 « Grid Computing »

Le concept de « Grid Computing » ou « Grille Informatique » est né en 1997 lors d'un séminaire intitulé « Building a Computational Grid » tenu au Laboratoire national d'Argonne[7]. En 1998, Ian Foster et Carl Kesselman, les deux organisateurs de ce séminaire, ont publié un livre intitulé « The Grid : Blueprint for a New Computing Infrastructure »[8]. Cet ouvrage est souvent considéré comme la bible du « Grid ».

---

[3] John G. Spooner et Sandeep Junnarkar, *IBM talks up 'computing on demand'*, http://news.cnet.com/IBM-talks-up-computing-on-demand/2100-1008_3-963807.html, publié le 30/10/2002, consulté le 06/08/2009

[4] IBM, *Application on Demand™*, http://www-935.ibm.com/services/us/index.wss/itservice/aod/a1011244, consulté le 06/08/2009

[5] Sun, *Java Technology - Sun N1*, http://www.sun.com/software/learnabout/n1/, consulté le 06/08/2009

[6] HP, *Animation Company "Fast Forwards" Production with HP Utility Data Center*, http://www.hp.com/hpinfo/newsroom/press/2003/030731a.html, consulté le 06/08/2009

[7] http://www.anl.gov/

[8] Ian Foster, Carl Kesselman. *The Grid : Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1ère edition 1998

Le « Grid » ou « Grille » est une analogie au réseau électrique, « power grid » en anglais, où l'utilisateur se branche au réseau et paie ce qu'il consomme sans se soucier d'où vient le courant. Le « Grid » a été initialement mis au point pour permettre le partage des ressources au sein de la communauté scientifique. Les ressources partagées [CERN1] peuvent être de natures diverses comme des données de tests, des logiciels, du matériel informatique et même des télescopes ou des microscopes.

Le terme « Grid » est rapidement devenu un élément vendeur et il a donc été utilisé pour vendre de nombreux projets comme par exemple : « Data Grids » ou « Cluster Grids », etc. Ian Foster a donc précisé sa définition en 2002 en décrivant en trois points le « Grid Computing » [F_02] :

1. Un « Grid » coordonne des ressources partagées qui ne sont pas gérées de manière centralisée. C'est-à-dire qu'il permet de coordonner différents systèmes indépendants au sein d'une même entreprise ou au sein d'entreprises tierces tout en abordant les questions de sécurité, de droits d'accès et de paiement.

2. Un « Grid » utilise des interfaces, ainsi que des protocoles ouverts et standards afin de permettre l'authentification, la découverte des services et leur utilisation.

3. Un « Grid » doit fournir un niveau de service de qualité, malgré l'utilisation de ressources de natures différentes. Ce niveau de service doit être adapté en fonction des besoins comme le temps de réponse, le débit, la disponibilité, la sécurité ou la colocation des ressources.

Ces trois points laissent le débat ouvert sur ce que sont une gestion centralisée des ressources, des interfaces et protocoles ouverts et un service de qualité. Toutefois, Ian Foster précise qu'un « Cluster » informatique n'est pas un « Grid ».

Techniquement un « Grid Computing » s'architecture en quatre couches [CERN1]. La première couche est la « couche réseau » qui permet d'interconnecter les différents éléments entre eux. La deuxième couche est constituée des ressources effectives faisant partie de la grille, telles que des ordinateurs, des systèmes de stockage ou même des capteurs tels que des télescopes ou autres instruments qui peuvent être connectés directement au réseau. La troisième couche est le « middleware ». Elle permet de gérer les interactions entre les différentes ressources. Cette couche gère l'organisation et la distribution des tâches aux différentes ressources. Pour faire cela, le « middleware » s'appuie sur des agents capables d'échanger des métadonnées et des « brokers » chargés de négocier l'authentification et l'autorisation d'utilisation. Le « broker » se charge aussi de conclure la transaction pour l'utilisation et le paiement de la ressource. Le « Globus toolkit » est un « middleware » très répandu qui permet entre autres de gérer l'affectation des ressources, de garantir la sécurité, de surveiller et découvrir les services et de répliquer les données. La dernière couche, qui est la seule à être visible par les utilisateurs, est la couche applicative. C'est elle qui contient tous les types d'applications développées pour le « Grid ». Ces applications peuvent être de natures diverses comme des applications scientifiques ou financières, devant réaliser de gros calculs sans contraintes de temps. Pour fonctionner sur un « Grid », une application doit être « gridifiée » pour interagir avec le système et pouvoir soumettre des demandes à la « Grille ».

Pour conclure cette sous-section, nous pouvons dire que malgré la souplesse offerte par un « Grid » tant au niveau du type de ressources qu'à la quantité des ressources utilisables à un temps T, un « Grid Computing » n'a pas été conçu pour héberger des sites Web et Web 2.0. Ceci est dû à sa nature de brokering de tâches en fonction des besoins et des disponibilités.

1.1.3 « Server farm » et « Web farm »

Les fermes de serveurs et plus particulièrement les « Web farms » se sont développées au début des années 2000 [CT_01, WY_01] pour répondre aux besoins de la bulle internet. Ces « fermes » hébergent jusqu'à plusieurs centaines de serveurs montés en « Cluster ». Le nom « farm » a été choisi par analogie avec les grandes granges américaines qui servent au stockage des céréales. Tout comme pour l'agriculture, ces fermes de serveurs sont installées en campagne ce qui permet aux exploitants de diminuer les coûts de stockage et de garantir un approvisionnement facilité en eau (nécessaire pour le refroidissement des machines) et en électricité.

Ces « fermes » de serveurs ont permis de répondre au besoin de « haute disponibilité » (HA – « high availability » en anglais) nécessaire à une grande majorité de services accessibles au travers de l'internet. La HA permet de garantir l'accomplissement des tâches même en cas de défaillance d'un des nœuds (« nodes » en anglais) du « Cluster ». Ce principe de « haute disponibilité » est réalisé grâce à la redondance des infrastructures physiques et logiques ainsi que la mise en place de mécanismes de reprise sur panne (« failover » en anglais) et de répartition de charge (« load balancing » en anglais). Malgré la HA l'utilisateur peut ressentir des baisses de performance, mais ne perdra pas complètement l'accès au service qu'il est en train de consommer.

Pour répondre à ce besoin de « haute disponibilité » dans les « Web Farms » deux concepts sont primordiaux : celui de la « scalabilité » et celui de la « répartition de charge ».

- La « scalabilité » permet de facilement augmenter la taille du « Cluster » en ajoutant des machines. Cette propriété permet de faciliter la montée en charge grâce à l'augmentation de la puissance de calcul et de la mémoire offerte dans la ferme. Ce concept permet aussi de garantir la « haute disponibilité » en permettant le remplacement d'une machine sans interruption de service.

- La répartition de charge en anglais « load balancing » est un énorme défit pour les ingénieurs et fait l'objet de nombreuses publications [BDH_03, CT_01, WY_01]. L'objectif de toutes les solutions proposées est de minimiser le temps de réponse moyen et d'éviter de surcharger une machine particulière. Pour atteindre ce but, les paramètres à prendre en compte sont divers et varient en fonction des projets.

  La solution la plus simple, pour le « load balancing », est basée sur la répartition de charge au niveau du serveur DNS (Domain Name System). Cette première solution ne fait que convertir un nom de domaine en une adresse IP. Cette solution est externe au « Cluster » et elle ne permet pas d'éviter de diriger l'utilisateur sur une machine surchargée ou qui ne répond plus.

  La seconde solution consiste à utiliser un serveur frontal, appelé communément « dispatcheur », pour répartir la charge. Le « dispatcheur », pour faire son travail, prend en compte plusieurs paramètres tels que la nature du site (statique ou dynamique), l'URL demandée, les cookies ou la popularité momentanée du site. En fonction de la nature du site statique ou dynamique, le « dispatcheur » mettra en œuvre un algorithme différent.

Pour des sites statiques, hébergés sur un « Cluster » de serveurs homogènes, les algorithmes « Round-Robin » ou « Random » sont idéaux. L'algorithme « Round-Robin » répartit la charge de manière séquentielle entre tous les serveurs. L'algorithme « Random » se sert d'une fonction aléatoire pour répartir la charge. Ce dernier peut donc momentanément surcharger un serveur.

Pour des sites dynamiques ou des sites hébergés sur des « Clusters » de serveurs hétérogènes, il est nécessaire d'utiliser des algorithmes adaptés et capables d'estimer la charge des serveurs. L'algorithme « Weighted Round-Robin » est très utilisé pour répartir la charge en fonction de la puissance de la machine. « Perceptive » est quant à lui un algorithme qui se base sur une analyse des demandes anciennes et actuelles. L'algorithme « Least Connections » surveille en permanence le nombre de connexions actives sur chaque nœud du « Cluster » et il dirige la demande sur celui qui a le moins de connexions. Finalement, « Fastest Response Time » est un algorithme qui calcule le temps de réponse de chaque serveur et affecte le travail au serveur qui répond le plus rapidement.

D'autres architectures de répartition de charge ainsi que d'autres algorithmes peuvent être développés pour répondre à des besoins spécifiques. Les ingénieurs de Google, dans l'article « Web Search for a Planet: The Google Cluster Architecture » [BDH_03], décrivent la solution mise en œuvre par ce leader des moteurs de recherche, pour répondre aux requêtes faites sur son propre moteur de recherche. La solution consiste premièrement en une répartition de la charge au niveau du serveur DNS. Cette première étape dirige l'utilisateur sur le « Cluster » le plus proche de lui géographiquement. Dans une seconde phase, un serveur exécute la demande en interrogeant un serveur d'« index » et complète le contenu de la page de résultats en interrogeant un serveur dit de « documents » avant de la retourner à l'internaute.

Dans leur article « Load Balancing for Clustered Web Farms » [WY_01] Joel L. Wolf et Philip S. Yu, ingénieurs chez IBM, proposent d'héberger plusieurs sites Web indépendants sur un nœud du « Cluster » et chacun de ces sites Web est lui-même répliqué sur plusieurs nœuds du « Cluster ». Ce chevauchement de sites sur plusieurs nœuds permet d'optimiser l'utilisation des ressources du serveur, car un site peut être temporairement fortement sollicité tandis que les autres sites hébergés sur le même

serveur ne le sont pas. Cette solution, de partage des ressources physiques pause toutefois un problème pour la confidentialité des données, car ces dernières sont toutes hébergées sur le même serveur. Un concurrent pourrait donc accéder à des données qui ne lui appartiennent pas.

Pour conclure cette section, nous pouvons dire que les « Web Farms » ont permis et permettent toujours d'héberger des services accessibles sur l'internet nécessitant une très haute disponibilité. La principale difficulté, pour le développement des services hébergés dans ces fermes, est liée au fait que chaque machine reste indépendante, malgré qu'elles fassent toutes parties du même « Cluster ». Il faut donc, tout comme pour le « Grid Computing » développer spécifiquement les applications, pour qu'elles fonctionnent sur un « Cluster ».

1.2 Apparition du « Cloud Computing »

Le « Cloud Computing » est l'évolution des notions que nous avons décrites dans les sections précédentes. Le terme « Cloud » (nuage en français) a été cité pour la première fois en 2001. Cette année là, Microsoft[9], en lançant sa nouvelle plate-forme de développement « .Net » présente le « Cloud » comme l'endroit où seront hébergés les « Web Services ». Ces services seront accessibles sur l'internet et l'utilisateur pourra les consommer sans se soucier d'où ils sont hébergés et avec quelle technologie ils sont développés. Le terme « Cloud » est utilisé par analogie au symbole communément utilisé pour représenter l'internet dans les schémas des ingénieurs réseau. C'est en août 2006, lors du congrès Search Engine Strategies qu'Eric Schmidt, PDG de Google. présente un nouveau modèle architectural informatique où les données et les

[9] Microsoft, Sanjay Parthasarathy, *WinHEC Conference 2001*, http://www.microsoft.com/presspass/exec/sanjay/03-26winhecsanjayp.mspx, publié le 26/03/2001, consulté le 11/09/2009

services seront hébergés quelque part sur le « Cloud »[10]. Le 24 août de la même année Amazon annonce son nouveau service « Elastic Compute Cloud » (EC2). Ce service Web, qui est le premier du genre, offre aux développeurs le premier ordinateur redimensionnable sur le « Cloud »[11]. Grâce à cette solution, chaque développeur peut virtuellement créer le serveur dont il a besoin pour son projet et le dimensionner pour ses besoins.

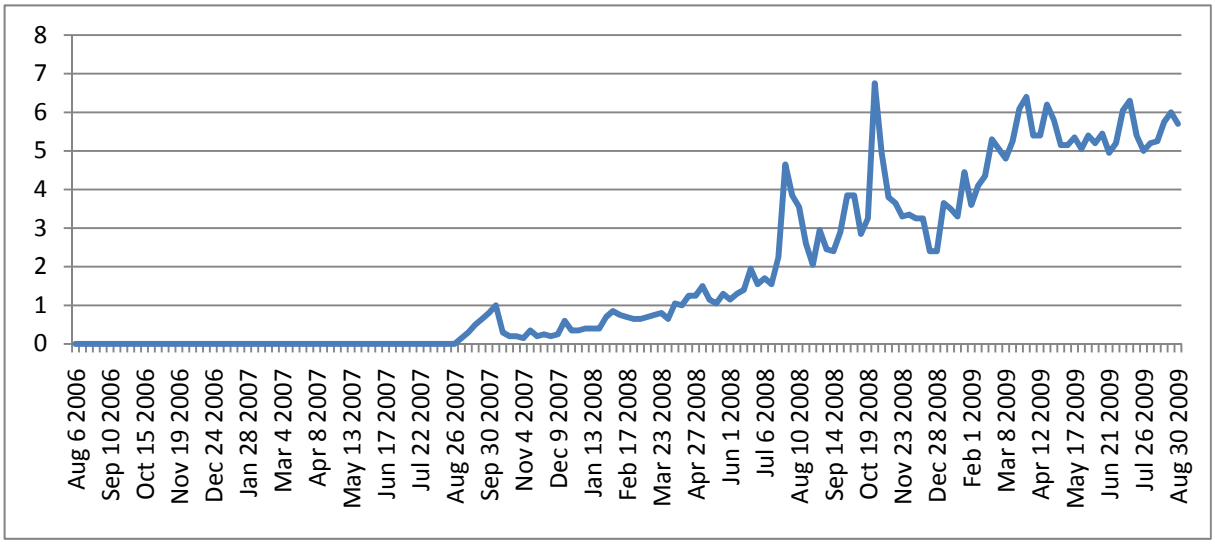

Figure 1 : Trafic moyen pour le terme « Cloud Computing »[12]

C'est au second semestre 2008 que le terme « Cloud Computing » se répand et devient un concept phare en informatique, comme le montre la figure (1) ci-dessus. Ce graphique se base sur les statistiques de recherche sur Google pour le terme « Cloud Computing » entre le 1er janvier 2004 et le 30 août 2009. Sur ce graphique, nous constatons un pique de recherches en

[10] Eric Schmidt, *Search Engine Strategies Conference, Conversation with Eric Schmidt hosted by Danny Sullivan*, http://www.google.com/press/podium/ses2006.html, publié le 09/08/2006, consulté le 05/07/2009

[11] Amazon, *Announcing Amazon Elastic Compute Cloud (Amazon EC2) - beta*, http://aws.amazon.com/about-aws/whats-new/2006/08/24/announcing-amazon-elastic-compute-cloud-amazon-ec2---beta/, publié le 26/08/2006, consulté le 11/09/2009

[12] google.com/trends, Trafic moyen pour le terme « Cloud Computing », généré le 12/09/2009

octobre 2008. C'est à ce moment précis qu'Amazon a lancé la version commerciale d'EC2[13]. Depuis, Amazon a aussi ouvert son premier « data center » européen en Irlande, afin de diminuer les temps de latence réseau liés aux connexions transatlantiques. Cette ouverture a aussi permis à Amazon d'offrir une solution conforme aux standards européens en matière de stockage de données[14][15]. Entre 2008 et le premier semestre 2009, tous les autres grands acteurs du marché informatique, comme Google[16], Microsoft[17], IBM[18], HP[19], Dell[20], Sun[21], Cisco[22], VMware[23], Xen[24] et Ubuntu[25] ont rejoint le marché du « Cloud Computing » ou ils ont lancé des projets liés à ces concepts.

[13] Amazon, *Amazon EC2 Exits Beta and Now Offers a Service Level Agreement*, http://aws.amazon.com/about-aws/whats-new/2008/10/23/amazon-ec2-exits-beta-and-now-offers-a-service-level-agreement/, publié le 23/10/2008, consulté le 11/09/2009

[14] Amazon, *Amazon EC2 Crosses the Atlantic*, http://aws.amazon.com/about-aws/whats-new/2008/12/10/amazon-ec2-crosses-the-atlantic/, publié le 10/12/2008, consulté le 11/09/2009

[15] Le Monde Informatique, Emmanuelle Delsol, *Le cloud d'Amazon EC2 arrive en Europe*, http://www.lemondeinformatique.fr/actualites/lire-le-cloud-d-amazon-ec2-arrive-en-europe-27631.html, publié le 11/12/2008, consulté le 17/12/2008

[16] http://code.google.com/appengine/, consulté le 11/09/2009

[17] http://www.microsoft.com/azure/, consulté le 11/09/2009

[18] http://www.ibm.com/grid/, consulté le 11/09/2009

[19] http://h71028.www7.hp.com/enterprise/us/en/technologies/cloud-computing.html, consulté le 11/09/2009

[20] http://www.dell.com/cloudcomputing, consulté le 11/09/2009

[21] http://www.sun.com/solutions/cloudcomputing/, consulté le 11/09/2009

[22] http://www.cisco.com/en/US/netsol/ns976/index.html, cosnulté le 11/09/2009

[23] http://www.vmware.com/technology/cloud-os/, consulté le 11/09/2009

[24] http://www.xen.org/products/cloudxen.html, consulté le 11/09/2009

[25] http://www.ubuntu.com/cloud, consulté le 11/09/2009

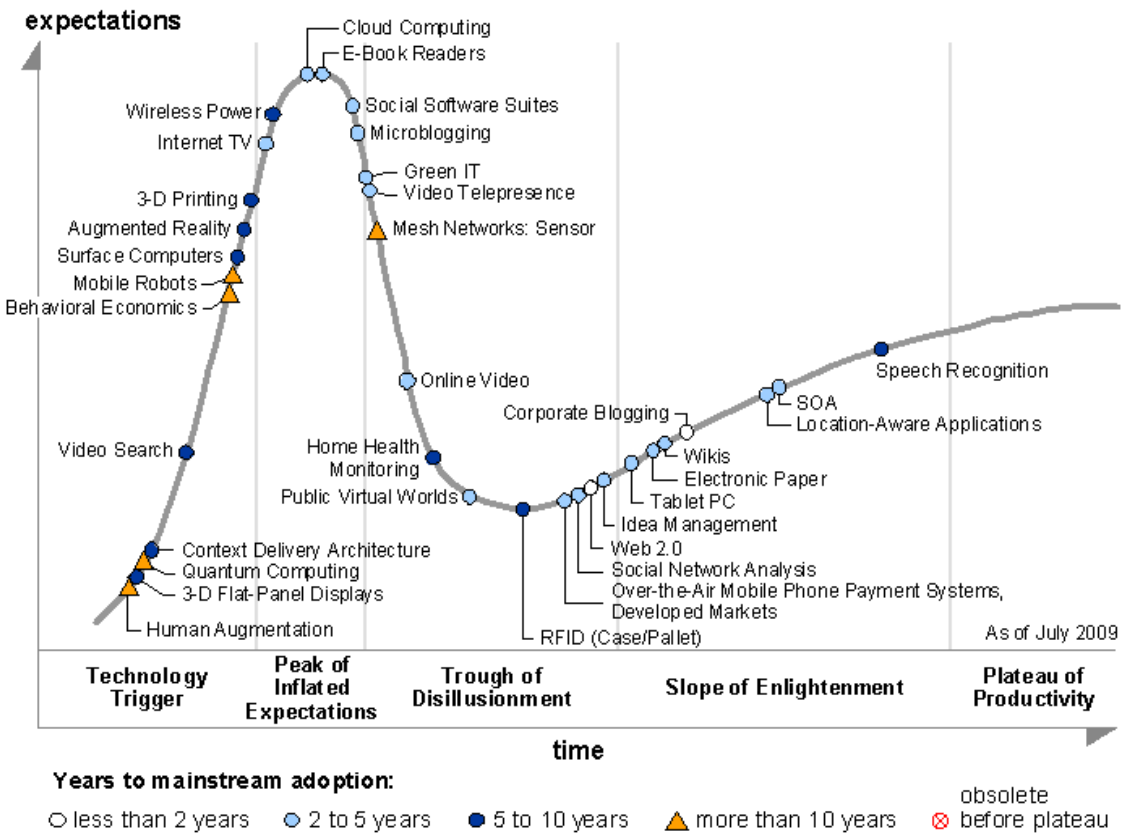

Figure 2 : courbe de « hype » 2009 pour les technologies émergentes[26]

Pour analyser les évolutions technologiques, le Gartner publie chaque année sa courbe de « hype » (popularité) pour les technologies émergentes[27] (figure 2 ci-dessus). Le « Cloud Computing » a été annoncé dans ce rapport pour la première fois en 2008 et il a atteint en 2009 le sommet du « pic des attentes exagérées ». Ce phénomène est dû au fait que le concept de « Cloud Computing » est nouveau. Comme avec toute nouveauté, pour dominer le marché, les concepteurs se doivent de promettre l'impensable. Dans son article « Twenty-One Experts Define Cloud Computing » [G_09], Jeremy Geelan relève déjà 21 définitions reprises et complétées dans la publication scientifique « A Break in the Clouds: Towards a Cloud

[26] Source Gartner, http://www.gartner.com/it/page.jsp?id=1124212, publié le 11/08/2009, consulté le 12/09/2009

[27] Gartner, *Gartner's 2009 Hype Cycle Special Report Evaluates Maturity of 1,650 Technologies*, http://www.gartner.com/it/page.jsp?id=1124212, publié le 11/08/2009, consulté le 12/09/2009

Definition » [VRCL_09]. Durant nos recherches, nous avons trouvé d'autres définitions comme celle de l'université de Berkeley [AFG_09], celle de l'université de Karlsruhe [WTK_09] ou la quinzième version de la définition du National Institute of Standards and Technology (NIST) américain [MG_09]. Toutes ces définitions présentent des aspects novateurs et prometteurs pour l'informatique, mais comme le rappellent les auteurs de la définition du NIST le « Cloud Computing » est encore un paradigme en pleine évolution. Toutefois, comme pour les autres technologies émergentes, le concept de « Cloud Computing » devrait, selon le Gartner, rapidement tomber dans le « fossé des désillusions », car les acteurs du marché ne pourront tenir toutes leurs promesses. Un des premiers exemples est Amazon, qui pour éviter les problèmes de goulets d'étranglement réseau et afin de permettre à ses clients d'envoyer des gigas octets de données rapidement, a du proposer l'envoi par courrier (Fedex) de disques durs[28] [29].

Finalement, le concept de « Cloud Computing » devrait atteindre, selon les analyses du Gartner, le plateau de la productivité d'ici 2 à 5 ans. Toujours selon le Gartner, ce concept devrait à terme aussi modifier le monde des technologies en offrant l'informatique comme un service consommable et non plus comme un investissement couteux. Dans les sections suivantes, nous allons donc présenter les concepts qui nous semblent déterminants pour donner une définition du « Cloud Computing ».

[28] GigaOM, Stacey Higginbotham, *Amazon's New Service Goes Postal Over Slow Broadband*, http://gigaom.com/2009/05/21/amazons-new-service-goes-postal-over-slow-broadband/, publié le 21/05/2009, consulté le 13/06/2009

[29] AWS Import/Export, http://aws.amazon.com/importexport/, consulté le 12/09/2009

1.3 Les concepts déterminants pour la définition du « Cloud Computing »

Les définitions du « Cloud Computing » sont nombreuses [AFG_09, G_09, McK_09, MG_09, P_09, TCW_09, VRCL_09, WTK_09]. Chaque définition introduit un ou plusieurs concepts qui selon leur auteur qualifie ce qu'est le « Cloud Computing ».

La définition du National Institute of Standards and Technology américain [MG_09] est souvent considéré comme étant une des références en la matière et elle présente les cinq caractéristiques essentielles du « Cloud Computing ». Ces caractéristiques ont été reprises par le panel d'intervenants à la conférence « Cloud Computing » lors du forum de l'UIT Telecom World 2009 à Genève [TCW_09]. Ces cinq concepts : service, « pay as you go » (paiement du consommé), élasticité et scalabilité, virtualisation et accès par l'internet de manière publique ou privée vont être détaillés dans les sous-sections ci-dessous. A la fin de l'étude de ces cinq notions, nous ferons une synthèse ce qui nous permettra de donner notre propre définition du « Cloud Computing ».

1.3.1 « X as a Service »

Le concept de service[30] en informatique est récurrent. En présentant le concept d'« Utility Computing » en 1974, George J. Feeney [F_74] basait son analyse sur le fait qu'au début du XXème siècle les usines produisaient elles-mêmes leur électricité et que cinquante ans plus tard elles sont devenues consommatrices d'un service « l'électricité ». En cinquante ans, l'électricité est devenue un service, public suivant le pays, de base, disponible et accessible pour tout le monde. Pour George J. Feeney l'informatique au sens large suit le même processus et deviendra à terme une collection de services de base à la disposition de tous.

[30] Larousse.fr, service n.m: usage que l'on peut faire de quelque chose, consulté le 17/10/2009

En informatique, un « service » peut être défini comme une tâche qui a été préparée de manière à ce qu'elle puisse être automatisée et livrée aux clients à leur demande tout en garantissant sa cohérence.

Durant les années nonante, le Web s'est développé. Dès le début des années deux milles, les startups ont commencé à proposer aux entreprises de louer des applications métiers en ligne au travers d'interfaces Web simples ou d'une interface type client-serveur [P_09]. Ce nouveau modèle d'affaires a été appelé « ASP » (Application Service Provider). Pour les clients, l'« ASP » permettait de se débarrasser de la problématique d'exploitation, souvent lourde et coûteuse, de leurs applications. Ces startups ont aussi permis l'accès aux PME à des solutions comme Oracle General Ledger ou toutes autres solutions coûteuses et difficiles à installer, pour un prix moindre grâce au partage du coût des licences et des ressources informatiques entre plusieurs clients.

Avec le développement du concept de « client riche » durant la première décade des années deux milles, les solutions type « ASP » ont été remplacées par des solutions appelées « SaaS » (Software as a Service). Le « client riche » est une évolution des « clients légers » (pages Web non dynamiques lues dans un browser) qui intègre les fonctionnalités des « clients lourds » ou d'applications de type client-serveur. Le « client riche » est utilisé dans le browser et il met en œuvre des technologies telles qu'AJAX, Flex, WPF, etc. pour afficher les pages Web et reproduire le comportement des applications de type « client lourd ». Le terme « RIA » (Rich Internet Application) est aussi communément utilisé pour désigner les applications de type « client riche ». Nous présenterons de manière plus détaillée les solutions de type « Saas » dans notre section « Cloud Application » (2.1.3).

Le concept de « SaaS » a ensuite été adapté pour l'utilisation à distance de hardware « HaaS » (Hardware as a Service) ou « IaaS » (Infrastructure as a Service) grâce au développement de la virtualisation et des RIA. Ce concept permet d'administrer et d'exploiter un « data center » virtuel hébergé sur le « Cloud ». Avec ces services, l'entreprise cliente ne paie que ce qu'elle utilise. Ce type de service évite aussi à l'entreprise les désagréments liés à l'exploitation d'un « data center », tels que la redondance des infrastructures, la sécurité au sens large, les piquets, etc. Ce modèle économique permet aussi de diminuer les coûts en matière d'infrastructure et de ressources humaines pour l'entreprise cliente. Nous présenterons de manière plus détaillée les solutions de type « Iaas » ou « HaaS » dans notre section « Cloud Infrastructure » (2.1.1).

Afin de répondre aux besoins des développeurs informatiques, qui ne se soucient guère des problèmes de hardware et d'infrastructure, le concept de « X as a Service » a été adapté selon les mêmes principes que pour le « SaaS » et le « IaaS » dès 2008[31][32], pour leur offrir une plate-forme d'hébergement d'applications en ligne. Ce type de solution est communément appelé « PaaS » (Platform as a Service). L'architecture technique des « PaaS » repose en général sur une couche appelée « middleware » qui permet aux développeurs d'applications de faire abstraction du système sur lequel leur application tournera. Comme pour le « SaaS » et le « IaaS » le développeur accède à sa plate-forme au travers d'internet. Nous présenterons de manière plus détaillée les solutions de type « PaaS » dans notre section « Cloud Platform » (2.1.2).

Pour résumer cette sous-section, nous pouvons dire que le concept de « X as a Service » consiste à consommer un service qui peut être matérialisé sous la forme d'un software, un middleware ou même d'un hardware en ligne sur le « Cloud », au travers d'une interface Web de type « client riche », de la même manière que nous consommons de l'électricité pour éclairer une pièce.

[31] Google, *Introducing Google App Engine*, http://googleappengine.blogspot.com/2008/04/introducing-google-app-engine-our-new.html, publié le 07/04/2008, consulté le 15/06/2009

[32] itbusiness.ca, *Windows in the cloud and a services offering from Microsoft*, http://www.itbusiness.ca/IT/client/en/CDN/News.asp?id=50556, publié le 30/10/2008, consulté le 15/06/2009

1.3.2 « Pay as you go »

Le concept de « pay as you go » permet à l'utilisateur de ne payer que ce qu'il consomme réellement, sans forfait minimum. Le prix est donc calculé à l'aide de ratios tels que processeurs par heure, giga-octets de disques par mois, etc.

Cette notion permet de réduire radicalement les coûts liés aux infrastructures informatiques. Joel L. Wolf et Philip S. Yu, dans leur article « Load Balancing for Clustered Web Farms » [WY_01], mentionnent que la sous-utilisation des serveurs est du gaspillage. Cette sous-utilisation est courante dans les entreprises, mais elle est nécessaire pour répondre à des pics de demandes. Selon Sun et AMD [GS_08], seulement 10 à 15% de la puissance des machines est utilisée à l'heure actuelle. Même sous-utilisées, ces machines consomment de l'énergie et elles occupent de la place inutilement.

Les graphiques ci-dessous illustrent les différents cas de sous-utilisation et de surutilisation des ressources informatiques. Le premier graphique (figure 3 ci-dessous) présente le cas d'une société qui a les ressources informatiques pour absorber tous les pics. Les zones grisées montrent la part de budget perdue dû à une sous utilisation des ressources.

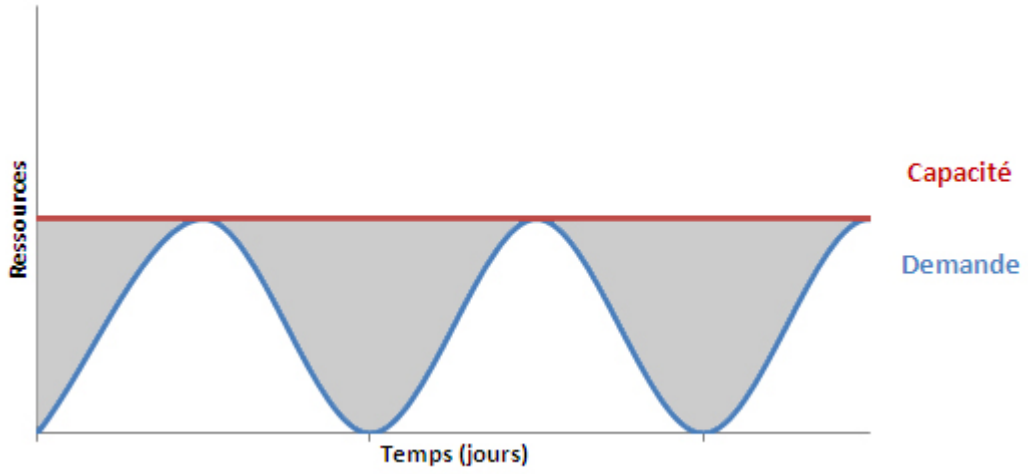

Figure 3 : Besoins en ressources informatiques, surestimation

Ce deuxième graphique (figure 4 ci-dessous), présente le cas d'une société qui ne peut pas absorber tous les pics. Cette compagnie risque de perdre des clients, car ces derniers ne seront pas satisfaits de la prestation offerte. Les zones grisées montrent la part de temps où les capacités informatiques ont été sous-estimées. Cette zone est généralement difficile à estimer d'un point de vue technique, mais encore plus d'un point de vue budgétaire.

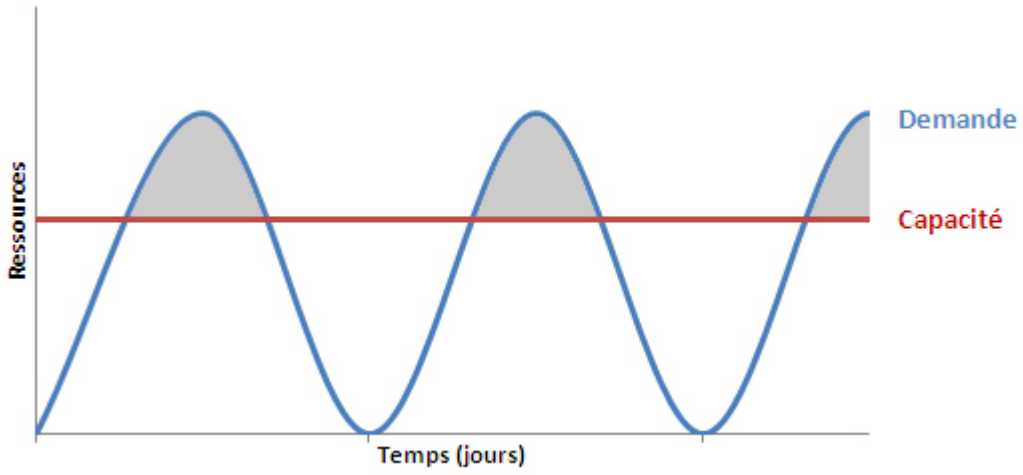

Figure 4 : Besoins en ressources informatiques, sous-estimation

Finalement, ce dernier graphique (figure 5 ci-dessous) montre le cas idéal. La société ne paie que pour ce qu'elle consomme grâce aux mécanismes de scalabilté et d'élasticité (sous-section 1.3.3) offerts par le « Cloud Computing ».

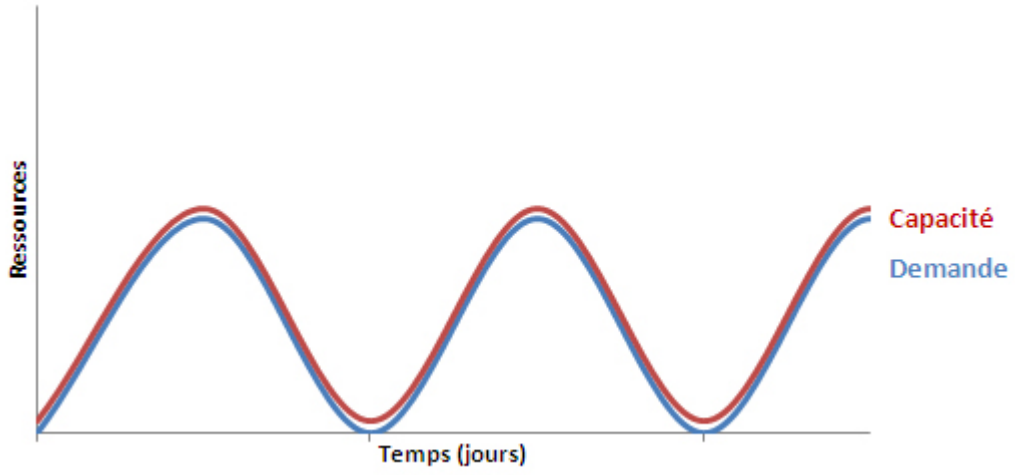

Figure 5 : Besoins en ressources informatiques, la capacité correspond à la demande

En conclusion, nous pouvons dire que le « Cloud Computing » avec son modèle économique « pay as you go » permet en général d'éviter le gaspillage des ressources informatiques et donc de diminuer les coûts d'exploitation liés à l'informatique.

1.3.3 Scalabilité et élasticité

Les concepts de scalabilité et d'élasticité sont les pendants techniques du concept décrit ci-dessus « pay as you go » (1.3.2). Ils offrent à l'utilisateur l'impression d'avoir en permanence des ressources de calcul et/ou de stockage illimitées [MG_09]. Ces ressources peuvent être facilement et rapidement ajoutées ou retirées, de manière automatique parfois, afin de répondre aux besoins de l'utilisateur.

La scalabilité est la capacité d'un système de s'adapter aux dimensions du problème qu'il a à traiter[33]. C'est-à-dire que la puissance de calcul, la mémoire ou le stockage utilisé par le système peuvent facilement être augmentés ou diminués en fonction des besoins.

L'élasticité est l'aptitude d'un corps à reprendre, après sollicitations, la forme et les dimensions qu'il avait avant d'être soumis à ces sollicitations[34]. Dans notre cas le « corps » est le système informatique et les « sollicitations » sont les besoins en matière de puissance, de mémoire ou de stockage.

L'élasticité permet donc d'automatiser le mécanisme de scalabilité des ressources informatiques mises à disposition sur le « Cloud Computing ».

1.3.4 Virtualisation

Le concept de virtualisation offre une vue logique plutôt que physique, de la puissance de calcul, de la capacité de stockage, et des autres ressources informatiques. Elle permet de faire tourner sur une même machine physique une ou plusieurs machines logiques. Ces machines logiques peuvent fonctionner avec des systèmes d'exploitation différents. Nous pouvons donc faire tourner virtuellement un Linux sur une machine Windows et vice versa.

La virtualisation permet aussi de faire fonctionner plusieurs machines logiques sur une machine physique et par conséquent de diminuer le gaspillage des ressources tel que nous l'avons décrit dans la sous-section « pay as you go » (1.3.2).

La virtualisation découple le système d'exploitation et les applications du système physique sur lequel ils fonctionnent. Ce principe permet soit de les porter sur de nouveaux environnements

[33] CNAM, *Cours CMSL - Icssea 2001*, http://deptinfo.cnam.fr/CMSL/icssea/icssea2001/cours.html, publié en décembre 2001, consulté le 17/10/2009

[34] Larousse.fr, consulté le 17/10/2009

physiques, soit de les migrer d'une machine à une autre. La virtualisation offre aussi des mécanismes capables de capturer l'état de la mémoire vive, ce qui accélère et simplifie le déplacement ou la duplication d'une instance de machine virtuelle.

En virtualisant les différents éléments hardware d'une machine physique, il devient alors très facile d'augmenter ou de diminuer la taille de chacun de ces éléments en fonction des besoins du moment.

Nous en concluons que la virtualisation est le concept qui permet au « Cloud Computing » d'offrir à un moindre coût tous les types de machines : petite ou puissante ; sur Linux, Windows ou Unix ; etc.

1.3.5 Public et privé

Il existe trois modèles types de « Cloud Computing » : publics, privés et hybrides [GL_09, MG_09, SUN_09]. Ils sont tous accessibles par un seul type de médium – l'internet.

Les « Clouds » publics sont généralement exploités par des sociétés tierces comme Amazon, Google, Microsoft, GoGrid[35], etc. (nous étudierons plus précisément ces acteurs dans les chapitres suivants (2 et 3.3)). Ce sont des infrastructures mutualisées et partagées par un grand nombre d'utilisateurs, accessible par l'internet. Ces « data centers » sont construit un peu partout sur la planète, généralement dans des lieux où l'électricité est bon marché et proche d'une source d'eau (rivière, fleuve) pour refroidir les machines. Nous pouvons citer l'exemple de Google qui a ouvert en 2006 un « data center » à The Dalles dans l'Oregon aux États-Unis (environ 370 km de Seattle). Ce « data center » se trouve au bord de la rivière Columbia et proche d'une usine hydroélectrique.

Les « Clouds » publics permettent de réduire les prix et les risques grâce aux économies d'échelle par la multiplication des infrastructures. Ce modèle permet aussi d'assurer les capacités de scalabilité et d'élasticité que doit offrir le « Cloud Computing ». Le « Cloud » public

[35] http://www.gogrid.com/

étant hébergé dans un environnement partagé et exploité par une société tierce, son exploitation peut poser un certain nombre de problèmes comme :

- La confidentialité des données peut-être mises en danger par les autres utilisateurs.
- En fonction d'où sont hébergées les machines, certaines lois peuvent porter un préjudice à l'utilisateur [AFG_09].
- Les performances ne sont pas constantes, car dépendantes de l'activité des autres utilisateurs.

Les « Clouds » privés sont des solutions dédiées à un client. Ils sont hébergés soit au sein de l'entreprise soit chez un prestataire de services. Le client a par conséquent un contrôle total sur ses données, il maîtrise la sécurité et bénéficie d'un niveau de service garanti. Par contre, les « Clouds » privés sont souvent limités dans leurs capacités techniques.

Les « Clouds » hybrides combinent les deux types de « Cloud ». Ce modèle permet de garantir la sécurité et la confidentialité des données tout en offrant la possibilité d'absorber une plus grande charge de manière temporaire. Ils sont généralement utilisés pour absorber les pics de charge. Le problème majeur de ce modèle est le transfert des données entre les « Cloud privés » hébergés généralement au sein de l'entreprise et le « Cloud public » qui se trouve quelque part sur la planète.

Pour résumer cette sous-section, nous pouvons dire que les termes « public » et « privé » ne définissent pas forcément où se trouve physiquement le « Cloud ». Souvent le terme « public » connote que la solution de « Cloud Computing » est hébergée quelque part sur l'internet et le terme « privé » que cette dernière est hébergée au sein de l'entreprise, et ceci, quelle que soit la nature de son contenu.

1.4 Synthèse des définitions du « Cloud Computing »

Dans les sections précédentes nous avons défini les cinq concepts de base du « Cloud Computing ». Ces notions sont répandues et ont atteint une certaine maturité depuis plusieurs années dans le monde informatique, mais elles sont généralement utilisées de façon individuelle. Le concept de « Cloud Computing » ne fait donc qu'associer des concepts architecturaux, techniques et commerciaux afin de répondre à un besoin émis il y a plus de trente ans [F_74].

Ces notions sont reprises dans les nombreuses définitions relevées par les auteurs des articles [G_09, VRCL_09], ainsi que dans les publications qui nous ont permis de mieux comprendre ce qu'est le « Cloud Computing » [AFG_09, McK_09, MG_09, P_09, TCW_09, WTK_09]. En lisant ces définitions, nous constatons que les notions de « service », de « scalabilité » et de « virtualisation » sont des concepts primordiaux, mentionnés dans plus de la moitié des définitions étudiées. Deux autres notions sont moins souvent mentionnées, mais elles ne nous semblent pas anodines.

Pour la première, il s'agit de l'idée de « centralisation » relevée dans la définition du « Cloud Computing » d'Aaron Ricadela [G_09]. Cette notion nous rappelle l'architecture terminal/serveur des années soixante. Cette architecture centralisée a disparu au début des années nonante avec l'augmentation exponentielle des ordinateurs personnels dans les entreprises. La centralisation des systèmes informatiques a commencé son retour au début des années deux mille avec l'apparition des « ASP » (1.3.1) et s'accroit avec un des concepts de base du « Cloud Computing » qu'est la notion de service.

La seconde notion est encore trop souvent mise de côté, mais elle est la clé pour choisir un prestataire de « service » de type « Cloud Computing ». Il s'agit de la notion de « service-level agreements » (SLA). Le SLA est une notion qui devient importante lors de la première panne. Ces pannes sont rares, mais peuvent bloquer des millions d'utilisateurs, faire perdre beaucoup d'argent à une entreprise, voir faire perdre des

données précieuses aux utilisateurs[36]. Ces problèmes font donc généralement l'objet de milliers d'articles, et de millions de Twitt montrant le mécontentement des clients. Afin de diminuer le mécontentement du consommateur, cette notion devra être développée par les prestataires de « service » et les pénalités en cas de problème doivent être clairement définies.

Les deux derniers concepts, « pay as you go » et « public et privé », que nous avons étudiés dans les sous-sections (1.3.2 et 1.3.5) ci-dessus, sont des notions reprises dans un quart des définitions seulement. Le concept de « pay as you go » est principalement un argument économique. Certaines publications scientifiques prouvent que ce modèle économique n'est pas forcément moins cher pour le client qu'un autre [AFG_09]. Le concept de « pay as you go » permet de réduire les coûts pour des organismes qui ont des besoins très variables en ressource informatique, ou pour les petites structures qui n'ont pas les moyens d'avoir leur propre service informatique. Ce point est à étudier avec attention avant de migrer son informatique sur « Cloud ». Le concept « public et privé » a été développé pour rassurer les utilisateurs de solutions de « Cloud Computing ». Selon Neal Leavitt, au moment de la publication de son article, en janvier 2009, dans « Computer » du IEEE [L_09], 75 pourcent des responsables informatiques craignaient pour la sécurité de leurs données sur un « Cloud Computing ». Les sociétés qui offrent des solutions de type « Cloud » ont donc développé les modèles « public, privé et hybride » pour garantir la sécurité des données et rassurer leurs clients.

[36] TechCrunch, Jason Kincaid, *T-Mobile Sidekick Disaster: Danger's Servers Crashed, And They Don't Have A Backup*, http://www.techcrunch.com/2009/10/10/t-mobile-sidekick-disaster-microsofts-servers-crashed-and-they-dont-have-a-backup/, publié le 10/10/2009, consulté le 10/10/2009

1.5 Définition personnelle du « Cloud Computing »

En nous basant sur la lecture des définitions [AFG_09, G_09, McK_09, MG_09, P_09, VRCL_09, WTK_09], ainsi que sur les réflexions conduites ci-dessus en étudiant les cinq concepts de base, nous pouvons procéder à une définition du « Cloud Computing » :

> *Le « Cloud Computing » est un méga ordinateur « scalable et élastique », pour lequel l'utilisateur ne paie que ce qu'il consomme. Cet « ordinateur virtuel » héberge des services (software, hardware ou middleware) accessibles depuis l'internet de manière publique ou privée.*

Le « Cloud Computing » est un concept jeune qui doit encore se stabiliser et trouver un nom plus orienté business[37]. Par exemple, Sam Palmisano PDG d'IBM préfère parler d'« infrastructure hautement virtualisée » pour parler de « Cloud Computing ». Tous les acteurs du marché avancent leurs arguments : facilités de développement, solution accessible depuis n'importe où, économique, scalable, dynamique, etc. L'avenir nous montrera ce que nous garderons de ce concept qui devrait révolutionner le monde informatique selon une analyse du Gartner en mai 2008[38].

[37] InformationWeek, Bob Evans, *HP's Hurd, IBM's Palmisano Agree: Cloud Is Lousy Name*, http://www.informationweek.com/blog/main/archives/2009/11/hps_hurd_ibms_p.html, publié le 5/11/2009, consulté le 5/11/2009

[38] Gartner, *Gartner Identifies Top Ten Disruptive Technologies for 2008 to 2012*, http://www.gartner.com/it/page.jsp?id=681107, publié le 28/05/2008, consulté le 13/12/2009

2 Présentation des solutions du marché

Dans le chapitre précédent nous avons donné notre définition du « Cloud Computing ». Ce concept est généraliste et il englobe aussi bien des solutions de location de hardware sur internet, que des solutions applicatives accessibles en ligne. Nous allons, dans les sections ci-dessous, classifier les différentes solutions (en les regroupant en trois catégories hardware, middleware et software) et ce, en utilisant la pyramide du « Cloud Computing ».

2.1 La pyramide du « Cloud Computing »

Dans notre section « Définition personnelle du « Cloud Computing » » (1.5), nous définissons le « Cloud Computing » comme *un méga ordinateur … qui héberge des services (software, hardware ou middleware) accessibles depuis l'internet*. Cette définition est générale et permet d'englober tous les types de services : software (SaaS), hardware (IaaS) ou middleware (PaaS) offerts par les acteurs du marché du « Cloud Computing ». Afin de faciliter la comparaison entre les différentes solutions du « Cloud Computing », certains spécialistes [E_08, L_08, S_08] les ont hiérarchisées en trois catégories. Michael Sheehan [S_08] les présente sous forme d'une pyramide (figure 6). Selon lui, cette pyramide pourrait être inversée, si elle était basée sur le nombre de services offerts par catégorie. Elle est ainsi présentée afin de montrer une hiérarchie de services (hardware, middleware et software) qui pourraient être interdépendants. À l'heure actuelle, une application de bureautique « SaaS » comme celle offerte par ZoHo[39] n'est pas dépendante d'une solution de type « IaaS » comme EC2[40] d'Amazon. À terme cela pourrait changer, ainsi certaines applications « SaaS » seront hébergées sur des « IaaS » ou des « PaaS ».

Dans les sous-sections suivantes nous allons définir les trois couches : « Cloud Infrastructure », « Cloud Platform » et « Cloud Application » de la pyramide du « Cloud Computing » et présenter

[39] http://writer.zoho.com/

[40] http://aws.amazon.com/ec2/

des solutions du marché qui pourront être utilisées pour la « mise en place d'un site type Web 2.0 sur un Cloud ». Nous ne ferons que lister les autres solutions qui ne correspondent pas, selon nous, à nos besoins pour ce projet.

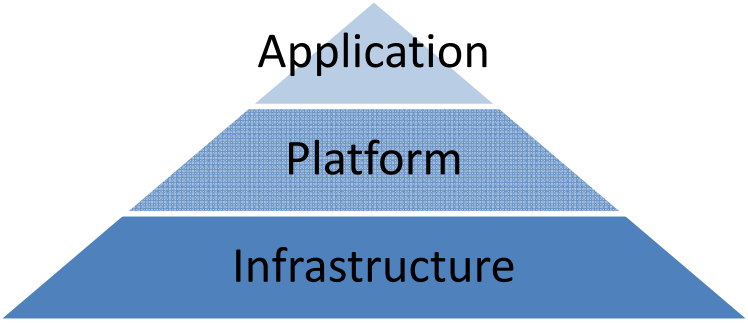

Figure 6 : Pyramide du « Cloud Computing »

2.1.1 « Cloud Infrastructure »

Cette catégorie, à la base de la pyramide du « Cloud Computing » regroupe les solutions qui permettent aux développeurs et aux ingénieurs système d'administrer et d'exploiter un « data center » virtuel hébergé sur le « Cloud ». Les solutions de type « Cloud Infrastructure » laissent aux spécialistes une grande liberté pour contrôler et paramétrer ces infrastructures informatiques (système) selon leurs besoins. Les solutions offertes dans cette couche sont souvent aussi appelées solutions « IaaS ». Dans cette catégorie sont regroupées aussi bien des solutions de stockage de données comme Amazon S3[41], que les solutions d'hébergement de serveurs (virtuels). Dans cette seconde catégorie nous pouvons citer les solutions que nous avons étudiées comme Amazon EC2[42] [Annexe A] et ElasticHosts[43] [Annexe B], ainsi que d'autres

[41] http://aws.amazon.com/s3/

[42] http://aws.amazon.com/ec2/

[43] http://www.elastichosts.com/

solutions du marché qui sont similaires comme GoGrid[44], Rackspace Cloud[45], RightScale[46], Skytap[47] et Orange Business[48].

L'avantage principal, offert par les solutions de type « Cloud Infrastructure », est un contrôle complet des infrastructures utilisées. Les inconvénients sont plus nombreux pour le moment. Ces solutions nécessitent plus de travail et d'expertise pour leur mise en œuvre, comparées aux solutions des couches supérieures de la pyramide du « Cloud Computing ». Ces solutions nécessitent souvent un programme tiers pour mettre en œuvre les mécanismes de « scalabilité ». De plus, les solutions de type « Cloud Infrastructure » fonctionnent généralement avec un mécanisme de multiplication d'instances (clonage de machines virtuelles), ce qui nécessite la mise en place d'un répartiteur de charge et impose certaines contraintes architecturales pour le développement d'application. Le dernier inconvénient est lié aux coûts cachés. Le prix des licences des systèmes d'exploitation et des programmes exécutés sur ces machines virtuelles peut être élevé, car les licences sont souvent facturées au processeur et non selon le modèle économique « pay as you go » processeur / temps (1.3.2).

2.1.2 « Cloud Platform »

Cette catégorie de la pyramide du « Cloud Computing » permet aux développeurs de déployer leurs applications sans se soucier des contraintes système, ainsi que de s'affranchir des problèmes de scalabilté et d'élasticité. Souvent les solutions offertes dans cette couche sont appelées solutions « PaaS ». Cette couche de la pyramide regroupe deux types de solutions. Les premières sont accessibles en ligne comme Microsoft Windows Azure[49] [Annexe C], Engine

[44] http://www.gogrid.com/

[45] http://www.rackspacecloud.com/

[46] http://www.rightscale.com/

[47] http://www.skytap.com/

[48] http://www.orange-business.com/fr/entreprise/thematiques/solutions-IT/cloud-computing/index.jsp

[49] http://www.microsoft.com/windowsazure/

Yard[50], Force.com[51], GigaSpaces[52], Google App Engine[53] [Annexe D] ou Heroku[54]. Les secondes sont des solutions de type middleware à installer en interne ou sur des solutions de la couche inférieure « Cloud Infrastructure » comme Appistry[55] [Annexe E], AppScale[56] [Annexe F], VMware vSphere 4[57] [Annexe G] ou Xen Cloud Platform[58] [Annexe H]. Ces middlewares sont développés pour héberger tous types d'applications, pour lesquelles le développeur ne veut pas se soucier des problèmes de scalabilté et d'élasticité.

Les avantages principaux des solutions de type « Cloud Platform » sont une simplification des développements pour les applications de type « Cloud » et un affranchissement des contraintes système, de scalabilté et d'élasticité. Les inconvénients sont liés au choix de la plate-forme qui est souvent une solution fermée (propriétaire) avec ses propres contraintes. Le choix du ou des langages de programmation est limité. De plus, les prestataires de services qui offrent des solutions de « PaaS » imposent souvent des règles contraignantes pour le développeur, comme l'interdiction d'utiliser des mécanismes de session.

[50] http://www.engineyard.com/

[51] http://www.salesforce.com/platform/

[52] http://www.gigaspaces.com/

[53] http://appengine.google.com/

[54] http://heroku.com/

[55] http://www.appistry.com/

[56] http://appscale.cs.ucsb.edu/

[57] http://www.vmware.com/products/vsphere/

[58] http://www.xen.org/products/cloudxen.html (en cours de développement le 20/01/2010)

2.1.3 « Cloud Application »

Cette catégorie au sommet de la pyramide du « Cloud Computing » regroupe toutes les applications (webmail, suite de bureautique en ligne, CRM, ERP, etc.) hébergées sur l'internet, capables de gérer un grand nombre d'utilisateurs simultanément et pour lesquelles le client ne se soucie pas d'où elles sont hébergées et comment elles fonctionnent. Ces solutions sont développées pour répondre aux besoins de l'utilisateur final et elles n'offrent que peu de possibilités de paramétrisation. Les solutions offertes dans cette couche sont souvent aussi appelées solutions « SaaS ». Les solutions regroupées dans la catégorie « Cloud Application » permettent de répondre à des besoins spécifiques du client. Il peut s'agir de la gestion de ses propres clients avec par exemple Salesforce.com[59], de la gestion de sa messagerie avec une solution comme Proofpoint[60] ou celle de la bureautique avec les solutions Google Apps[61], Google Docs[62], iWork.com[63], ZoHo[64] ou 24SevenOffice[65].

Les avantages principaux des solutions de type « Cloud Application » sont les suivants : un accès direct à l'application depuis n'importe où dans le monde, une simplicité d'utilisation, une mise en œuvre de solutions éprouvées. Un autre avantage des solutions de type « SaaS » est qu'elles sont parfois gratuites et mettent en œuvre le modèle économique freemium. L'inconvénient majeur est lié aux fonctionnalités souvent limitées de l'application. Ces solutions ne permettent généralement pas d'être adaptées pour répondre à des besoins spécifiques.

[59] http://www.salesforce.com/

[60] http://www.proofpoint.com/

[61] http://www.google.com/a/

[62] http://docs.google.com/

[63] http://www.iwork.com/

[64] http://www.zoho.com/

[65] http://www.24sevenoffice.com/

3 Mise en pratique du concept de « Cloud Computing »

Nous commençons ce chapitre par un rappel du contexte énoncé dans le document « proposition de sujet de mémoire » du 17 juin 2009.

> *Les missions du Laboratoire des Technologies de l'Information (LTI) sont de deux types. La première mission est la réalisation de mandats, tels que le développement d'applications, la réalisation de sites Web, la création de graphismes et la maintenance de sites Web. La seconde mission est la recherche, principalement dans le domaine du Web, sur des sujets comme l'Administration en Ligne (AeL), l'infobésité ou le Web 2.0.*
>
> *En 2005, Tim O'Reilly a défini le Web 2.0 comme étant le Web communautaire et interactif.*
> *En 2008, Amazon a été la première société à offrir une nouvelle solution d'hébergement basée sur les concepts dits de « Cloud Computing ». D'autres acteurs du marché des Technologies de l'information et de la communication (TIC), comme Google et Microsoft ont rapidement suivi. Depuis le début de l'année 2009, les fabricants de hardware ont rejoint le marché en offrant des machines spécialement conçues pour ce type d'architecture.*
>
> *Le Web 2.0 nécessitant de grosses ressources informatiques, il nous semble important d'étudier comment nous pourrions héberger un site Web 2.0 sur une plate-forme de type « Cloud ».*

Dans les sections ci-dessous, nous allons premièrement définir les besoins liés à la mise en œuvre de ce projet. Nous rappellerons aussi succinctement ce qu'est le Web 2.0 et les études que nous avons menées sur ce sujet au sein du Laboratoire des Technologies de l'Information. Nous décrirons ensuite les objectifs à atteindre pour valider la partie technique ainsi que la réalisation de ce projet. Nous terminerons ce chapitre en choisissant, à l'aide d'outils de gestion,

la plate-forme de « Cloud Computing » qui sera utilisée pour la réalisation et la validation de ce projet.

Pour réaliser ce projet, je suis encadré par le Professeur Jean-Philippe Trabichet responsable du Laboratoire des Technologies de l'Information. En tant que chef de projet, je suis soutenu par tous mes collègues du LTI, des ingénieurs système de la Haute École de Gestion de Genève (HEG) et par mes collègues du Centre des Technologies de l'Information (CTI) de l'État de Genève.

Les besoins et les objectifs ont été validés par mon supérieur et les membres du Comité Interdépartemental des Chargés de Communication (CICC) de l'État de Genève. J'ai ainsi commencé ma mission par une analyse des différentes plates-formes de « Cloud Computing », avant de choisir une plate-forme pour le projet. Ensuite, j'ai travaillé sur la partie modélisation du projet. Après avoir développé et déployé l'application, avec l'aide de mes collègues, j'ai conduit plusieurs campagnes de tests.

3.1 Définition des besoins

Dans cette section, nous allons préciser les deux besoins identifiés pour ce projet. Le premier besoin est un besoin métier. Il consiste à mettre en place une application de type Web 2.0 qui permet aux utilisateurs du site de l'État de Genève d'étiqueter chaque page Web avec des mots-clés qui leurs sont propres, afin de faciliter la rcherche. Le second besoin est une exigence technique. Il consiste à choisir une plate-forme de « Cloud Computing ». Ce second besoin permet d'étudier les contraintes et les opportunités liées à l'hébergement d'applications Web sur un « Cloud ».

3.1.1 Le Web 2.0

Notre premier besoin est lié au Web 2.0. Dans cette sous-section, nous allons commencer par définir ce concept. Puis nous présenterons les études et les projets mettant en œuvre le Web 2.0 que nous avons déjà conduits. Nous terminerons cette sous-section par une spécification du besoin.

Les définitions du Web 2.0 sont multiples, une idée maîtresse permet de les résumer. Les sites Web 1.0 fournissent des informations. Les sites Web 2.0 placent l'internaute comme un acteur ayant un rôle actif, collaboratif et donc productif sur le site Web visité.

Dès le début du Web 2.0 en 2005, les responsables de la communication de l'Administration cantonale genevoise ont régulièrement sollicité les services du Laboratoire des Technologies de l'Information afin d'étudier les possibilités d'intégration de nouveaux services de type Web 2.0 dans le site www.ge.ch. L'un de ces services est l'offre de flux RSS. Depuis début 2006, l'État de Genève l'offre à ses citoyens. Les flux RSS sont un des composants de base pour l'échange d'information dans les sites Web 2.0.

Dans le cadre de ces études, nous avons travaillé sur la thématique des blogs qui sont des sites Web d'information faciles à créer et à mettre à jour par tout un chacun sans formation de Webmaster. De plus, ils permettent l'interaction avec les citoyens qui peuvent publier leurs commentaires. Les blogs n'ont jamais été mis en service sur le site officiel de l'État car les discussions peuvent rapidement prendre une tournure politique contraire aux principes d'éthique et de déontologie des Administrations.

En 2008, nous nous sommes intéressés aux sites http://del.icio.us et twine.com qui permettent de partager des favoris et de les qualifier. Il s'agit d'applications alliant les concepts du Web 2.0 et du Web sémantique. Ce dernier concept permet de fournir une intelligence au système, basée sur une meilleure connaissance du contenu. Cette tendance à collaborer pour étiqueter de l'information se retrouve sur des sites de partage de photos comme Flickr.com, de liens comme del.icio.us et twine.com, ainsi que sur des sites de commerce en ligne comme Amazon.com. Ce concept est à considérer par les éditeurs de site Web et aussi par les rédacteurs des sites administratifs.

Au second semestre 2009, un audit du moteur de recherche du site de l'État de Genève a été mené par une spécialiste en information documentaire[66]. Un des points qui a été soulevé par cette étude est lié au vocabulaire utilisé. Le vocabulaire du site www.ge.ch peut être abscons pour les personnes externes à l'État et les termes spécifiques au métier sont généralement inadaptés pour le grand public.

Suite à cet audit du moteur de recherche et fort de nos études en matière de Web 2.0, nous avons proposé au Comité Interdépartemental des Chargés de Communication, présidé par la Chancelière Madame Madame Anja Wyden Guelpa, de développer un prototype fonctionnel d'application Web 2.0, mettant en œuvre des mécanismes d'étiquetage collaboratif ou « folksonomie ». Cette application permet à tout internaute d'étiqueter une page Web et de rechercher les pages au travers d'un nuage de tag.

En conclusion de cette sous-section, nous pouvons résumer notre besoin en matière de Web 2.0 en un mot : collaboration. En mettant en place une application qui permet au citoyen de collaborer avec l'Administration, cela nous permettra de mettre en pratique les réflexions que nous avons eues depuis 2005. Cela nous donnera également la possibilité de développer notre infrastructure applicative pour conforter la communication bidirectionnelle.

[66] Madame Karine Pasquier, assistante HES à la HEG

3.1.2 Le « Cloud Computing »

Notre second besoin est d'étudier le concept de « Cloud Computing ». Nous commencerons cette sous-section par rappeler l'importance que va prendre le « Cloud Computing » dans les prochaines années. Nous la terminerons en spécifiant nos besoins en la matière.

Tim O'Reilly, PDG de O'Reilly Media, a défini, lors du congrès Web 2.0 Expo en avril 2008[67], le « Cloud Computing » comme une des briques de base des plates-formes informatiques de prochaine génération. Le Gartner a annoncé en août 2009[68] que le « Cloud Computing » sera un des concepts technologiques majeurs d'ici deux à cinq ans.

Après avoir défini ce qu'est pour nous le « Cloud Computing » dans le premier chapitre, nous allons choisir et tester une plate-forme dans les sections ci-dessous. Le résultat de cette étude nous permettra, au travers d'une réalisation pratique, de constater les contraintes techniques liées à l'hébergement d'une application Web sur un « Cloud ».

Cette étude m'a permis de présenter personnellement, en janvier 2010, le concept de « Cloud Computing » aux collaborateurs du Centre des Technologies de l'information de l'État de Genève. Lors de cette présentation, j'ai dévoilé ma perception de ce que pourrait être la future plate-forme informatique de l'État de Genève. En mars 2010, j'ai aussi introduit la thématique du Technology Day d'eb-Qual[69] sur le « Cloud Computing ». Cette présentation a permis à toutes les sociétés présentes lors de cette journée et commercialisant des solutions de type « Cloud », de répondre aux interrogations que j'ai soulevées en matière de sécurité, de niveau de service et de domiciliation de serveurs. J'ai finalement eu l'opportunité de présenter cette étude aux

[67] Joyent, Tim O'Reilly, *What is Cloud Computing*, http://www.youtube.com/watch?v=6PNuQHUiV3Q, publié le 07/05/2008, visionné le 15/10/2009

[68] Gartner, *Gartner's 2009 Hype Cycle Special Report Evaluates Maturity of 1,650 Technologies*, http://www.gartner.com/it/page.jsp?id=1124212, publié le 11/08/2009, consulté le 12/09/2009

[69] http://www.eb-qual.ch/

étudiants du MSSI[70] de la HEG, de leur exposer les concepts liés au « Cloud Computing » ainsi que quelques solutions de type « IaaS », « PaaS » et « SaaS ».

En conclusion de cette sous-section, nous pouvons dire que nos besoins en matière de « Cloud Computing » sont principalement tournés vers l'étude de ce nouveau concept, afin d'évangéliser nos clients, nos partenaires ou nos étudiants et pouvoir répondre à leurs interrogations en la matière ; mais aussi de nous préparer pour la nouvelle plate-forme informatique que va devenir le « Cloud ».

3.2 Objectifs et contraintes du prototype fonctionnel d'un site Web 2.0

Afin d'améliorer et faciliter la recherche d'informations dans le site Web de l'État de Genève, nous avons développé un prototype fonctionnel, mettant en œuvre le concept d'étiquetage collaboratif ou « folksonomie », basé sur le concept de collaboration du Web 2.0. La « folksonomie » est un *néologisme désignant un système de classification collaborative décentralisée spontanée, basé sur une indexation effectuée par des non-spécialistes* [WIKI1].

Dans le cadre de ce projet, nous avons développé un premier prototype permettant aux citoyens de qualifier, à l'aide de mots-clés qui leurs sont propres, les informations publiées sur les pages des sites Web de www.ge.ch. Cette fonctionnalité améliore le classement des pages Web et facilite la recherche d'informations.

[70] http://www.hesge.ch/heg/mba_mssi/

Ce projet de « folksonomie » met en place un Widget pour étiqueter chaque page Web avec un ou plusieurs mots-clés. Un second module permet à l'internaute de retrouver l'information qu'il recherche à l'aide des mots clés qui lui sont propres, au travers de ce nouveau plan de classement.

3.2.1 Objectifs à atteindre pour ce projet :

1. L'étiquetage des pages Web : ce développement permet d'ajouter des mots clés dans les pages Web. Il permet aussi à l'internaute de confirmer ou d'infirmer l'importance d'un mot clé dans une page.
2. La recherche par mots-clés : ce second développement permet aux internautes de lister les liens associés à un mot-clé. La liste des liens présentée au citoyen peut être affinée en permettant de choisir plusieurs mots-clés.
3. Le « Cloud Computing » : ce projet doit permettre de choisir, tester et valider une solution de type « Cloud Computing » qui nous affranchit des problèmes liés à la répartition de charge et de la haute disponibilité.

3.2.2 Contraintes du projet :

1. Ce développement ne doit pas impacter le code déjà mis en œuvre dans la centaine de sites hébergés sur les serveurs de www.ge.ch.
2. Il doit s'intégrer facilement dans des sites développés dans les langages suivants : ASP, PHP et Java.
3. Ce développement doit respecter la charte éditoriale Web de l'État de Genève.
4. Le niveau de service pour ce projet doit être d'au moins 99.95% (moins de 5 heures d'interruptions par an).

3.3 Choix d'une plate-forme de « Cloud Computing »

Dans cette section, nous allons sélectionner à l'aide d'outils de gestion, la plate-forme de « Cloud Computing » qui hébergera notre prototype fonctionnel (POC). Pour faire ce choix, nous utiliserons trois outils :

- Le « comparatif des coûts » est un tableau qui permet de récapituler tous les coûts pour une solution déterminée. Dans cette matrice nous présenterons les coûts liés à l'infrastructure technique. Nous présenterons aussi les coûts des logiciels nécessaires au fonctionnement de l'application hébergée. Finalement, nous estimerons les coûts des ressources humaines nécessaires à l'exploitation d'une plate-forme de « Cloud Computing ».

- La « matrice de préférence » est une méthode qui permet de définir un ordre de priorité et d'importance entre les critères retenus pour notre projet. Dans cette sous-section (3.3.2) nous commencerons par définir et qualifier chacun de ces critères. Nous comparerons ensuite chacun d'entre eux avec tous les autres, afin de déterminer l'importance de chaque critère. Cet ordre d'importance nous permettra de pondérer le résultat obtenu à l'aide de notre troisième outil l'« analyse multicritères ».

- L'« analyse multicritères » est une méthode qui permet de comparer plusieurs solutions entre elles, à l'aide de notes pondérées pour chaque critère d'évaluation. Les notes, pour cette évaluation vont de zéro à dix. « Zéro » signifiant que le critère ne peut pas être appliqué pour la solution et « dix » signifiant que le critère est totalement pris en compte dans la solution.

Dans ces analyses, certains critères, impératifs (par exemple : data center en Suisse ou standards ouverts) pour notre mandant principal – l'État de Genève, ont volontairement été rendus facultatifs. Ces critères impératifs, de par leur nature bloquante lors d'un projet, limiteraient notre choix à la solution « vSphere 4 de VMware » hébergée au sein de notre data center. Ce changement de priorité nous permet d'étudier de nouvelles solutions et ainsi proposer à l'avenir de nouvelles solutions à nos clients.

Pour cette étude et la mise en place de notre prototype fonctionnel d'un site type Web 2.0 sur un Cloud, nous avons retenu cinq solutions de « Cloud Computing » :

- Amazon EC2, avec sa solution de type « IaaS », ne révolutionne pas l'offre en matière d'hébergement Web dynamique. Les hébergeurs dans ce domaine sont nombreux et souvent meilleur marché qu'Amazon. Nous retenons toutefois la solution EC2 car Amazon est un des pionniers en matière de « Cloud Computing ». Il a aussi su améliorer son offre durant sa première année d'exploitation commerciale. De plus, Amazon est un des rares prestataires de service à offrir de l'hébergement aussi bien aux États-Unis qu'en Europe.

- AppScale est une solution open source de type « Cloud Platform » qui, pour le moment, permet principalement de tester des applications destinées à fonctionner sur Google App Engine. En ouvrant sa plate-forme à d'autres solutions de stockage de données que la solution propriétaire de Google, AppScale nous semble une solution d'avenir. AppScale se déploie sur la solution Amazon EC2.

- Azure est la solution de « PaaS » commercialisée par Microsoft. C'est une des seules solutions du marché qui permet d'exécuter un nombre important de langages de programmation. De plus, Microsoft dans sa solution de « Cloud Computing » offre un service de base de données (Sql Server) que nous utilisons déjà pour certains projets de l'État de Genève. Azure permet aussi de se connecter depuis son « Cloud » à des applications existantes au sein de l'entreprise.

- Google App Engine est une des seules solutions de « PaaS » à promettre une plate-forme scalable et élastique de manière transparente pour le client. Google avec ses nombreux data centers répartis autour du monde offre aussi une qualité de service indéniable, malgré qu'elle ne soit pas garantie.

- « VMware vSphere 4 – Dell » est une des solutions les plus abouties pour le déploiement de « Cloud Platform » au sein d'une entreprise. Ayant la possibilité de facilement tester cette solution au sein de notre laboratoire[71], nous nous devons de prendre en compte cette solution dans notre étude.

Quatre sur cinq des solutions retenues pour ce projet et énumérées ci-dessus sont de type « Cloud Platform ». Comme nous l'avons défini dans notre section (2.1.2) ces solutions permettent aux développeurs de déployer leurs applications sans se soucier des contraintes système, ainsi que de s'affranchir des problèmes de scalabilté et d'élasticité. Notre objectif premier étant de mettre en place un site Web 2.0 sur un « Cloud », nous ne prenons pas en considération des problèmes d'infrastructure. Nous ne voulons pas non plus tester une solution

[71] Le Laboratoire des Technologies de l'Information, bénéficie des infrastructures de la HEG pour ses projets de recherche. L'État de Genève exploite aussi bien des solutions de chez : Dell, Sun (Oracle), IBM, etc.

type « Cloud Application », car nous ne pourrions pas nous rendre compte des problèmes techniques liés à la mise en place d'une application scalable et élastique.

Nous conclurons cette section (3.3) par une étude des résultats obtenus. Cette analyse nous permettra de retenir la solution qui nous semble la plus adaptée à notre projet de mise en place d'un site type Web 2.0 sur un « Cloud ».

3.3.1 Comparatif des coûts

Le « comparatif des coûts » se présente sous la forme d'un tableau qui récapitule les coûts annuels de chaque solution de « Cloud Computing » que nous avons retenue pour l'hébergement de notre projet « folksonomie ». Les quatre premières solutions : Amazon EC2, AppScale, Azure et Google App Engine sont des solutions de type « Cloud public » et donc hébergées en externe. Pour comparer ces solutions, nous avons calculé le coût de la location de deux instances serveur, avec 10 Go de stockage en base de données. Ce prix comprend aussi un trafic de données de 25 Go en entrée et 250 Go en sortie. La dernière solution, « VMware vSphere 4 – Dell », est une solution de type « Cloud privé » hébergée en interne. Le prix des infrastructures techniques étant amorti sur 3 ans minimum, nous avons divisé ce prix par trois. Nous devons aussi noter que cette dernière solution permet d'héberger au minimum 8 machines virtuelles, une par core de processeur. VMware garantissant un nombre supérieur de machines virtuelles par core, le coût annuel de l'infrastructure technique pourrait être encore diminué.

Pour chacune des solutions analysées, nous avons prévu un budget de deux jours et demi pour des maintenances liées aux infrastructures techniques. Nous ne prévoyons pas de budget ni pour la maintenance applicative, ni pour le développement. Ces coûts pouvant fortement varier en fonction du projet.

Solution / Centre de coût	**Amazon EC2**	**AppScale**	**Azure**	**Google App Engine**	**VMware vSphere 4 – Dell**
Infrastructure technique	13'800.-	13'800.-	14'000.-	12'300.-	(38'000.-) (an) 12'670.-
Logiciel	0.-	0.-	0.-	0.-	18'200.-
Ressources humaines	2'500.-	5'000.-	2'500.-	2'500.-	27'500*.-
Total annuel	**16'300.-**	**18'800.-**	**16'500.-**	**14'800.-**	**(58'370.-) 19'200.-**

Tableau I : comparatif des coûts annuels en francs Suisse (état au 21 janvier 2010)

** 25 jours par an sont prévus pour les mises à jour du système et le monitoring des serveurs. Les éventuels coûts de maintenance hardware sont couverts par le service gold vendu par Dell.*

Dans le tableau (I) ci-dessus, la solution la plus chère (58'370.- CHF) est celle de type « Cloud privé » mise en place sur une solution « VMware vSphere 4 – Dell ». Ce prix peut être divisé par quatre pour obtenir le coût d'hébergement d'une VM aux capacités similaires à celle des autres solutions comparées dans cette matrice. Le prix annuel d'une telle machine serait alors de 16'500.- CHF. Toutefois, ce prix ne prend pas en compte toutes les charges (loyer, électricité, refroidissement, réseau). Nous estimons le coût total de la solution « VMware vSphere 4 – Dell », charges comprises, à 19'200.- CHF par an. Ce prix reste plus élevé que celui proposé par les hébergeurs de type « Cloud public », mais il montre qu'une solution de type « Cloud privé » ne fait pas exploser le budget d'une société (PME). Ce prix montre aussi qu'une PME peut bénéficier aussi de petites économies d'échelle en rationalisant son informatique. De plus, si la société garde son infrastructure technique une ou deux années supplémentaires, cette dernière ne lui coutera plus que des coûts d'exploitation et de licence VMware, le hardware étant déjà amorti.

La solution la moins chère est celle de Google App Engine (14'800.- CHF). Google offre une plate-forme de qualité, mais le service n'est pas garanti par un SLA.

Amazon et Microsoft avec Azure offrent pour un prix similaire, tous les deux une plate-forme de bonne qualité avec un SLA quasi équivalent. L'avantage, pour les développeurs, de la plate-forme Azure est qu'ils n'ont pas à se soucier des problèmes liés à l'installation du système d'exploitation et du serveur applicatif.

La dernière solution AppScale est hébergée dans le cadre de ce chiffrage chez Amazon. Les coûts liés à l'infrastructure technique sont donc identiques. Par contre, cette solution nécessite un peu plus de travail de maintenance et donc un prix légèrement plus élevé.

3.3.2 Matrice de préférence

Dans cette sous-section nous allons définir et qualifier les critères qui nous permettront d'évaluer et comparer les solutions de « Cloud Computing ». Ensuite, nous les comparerons entre eux afin de définir leur ordre d'importance et ainsi pouvoir pondérer les notes obtenues par chacune des solutions retenues dans ce projet, pour chacun de ces critères.

3.3.2.1 Critères d'analyse

Afin de choisir la solution de « Cloud Computing » pour notre projet avec l'État de Genève, nous avons défini treize critères importants pour évaluer et comparer les solutions entre elles. Ces critères couvrent les domaines suivants : la qualité, la sécurité, les aspects techniques et la situation géographique.

Afin de garantir une compréhension par tous de ces treize critères nous les avons définis comme suit :

1. « Cloud privé » signifie que les applications et les données sont hébergées soit au sein de l'entreprise, soit dans une structure dédiée et sécurisée chez un prestataire de service. En cas d'hébergement externe les données sont accédées au travers d'un VPN.

2. « Cloud public » signifie que les applications et les données sont entièrement hébergées sur des infrastructures mutualisées.

3. « Communauté / documentation / support » englobent toutes les solutions d'aide mises à disposition des utilisateurs de solution de « Cloud Computing » pour développer, déployer et exploiter leur solution.

4. « Data center en Europe » signifie que les applications et les données sont hébergées dans un des pays membres de l'Union Européenne ou de l'Association européenne de libre-échange (AELE).

5. « Data center en Suisse » signifie que les applications et les données ne sont hébergées qu'en Suisse.

6. « Data center n'importe où » signifie que les applications et les données sont hébergées n'importe où dans le monde.

7. « Elasticité » signifie que la solution de « Cloud Computing » est capable de grandir et de reprendre sa taille initiale, entièrement automatiquement.

8. « Facilité de déploiement » signifie que l'utilisateur de la solution de « Cloud Computing » peut déployer son application de manière simple soit à l'aide d'un programme type FTP, soit au travers d'une interface Web.

9. « Langages de programmation » signifie que les langages autorisés pour le développement sont des langages de programmation couramment utilisés sur le marché comme : Java, .Net, PHP, Python ou Ruby.

10. « Scalabilté » signifie que la solution est capable d'absorber la montée en charge.
11. « Sécurité des données » signifie que la solution garantit que les tiers non autorisés ne peuvent accéder aux données. Cela signifie aussi que les données sont régulièrement backupées et qu'elles peuvent facilement être récupérées lors de migration.
12. « SLA » signifie que la solution offre un certain niveau de service garanti et des pénalités en cas de non-respect. Le SLA définit des valeurs quantifiables pour chaque point important (réseau, serveur, temps de réponse, etc.) de son service.
13. « Standards ouverts » signifie que la solution peut facilement être transférée d'une solution à une autre.

3.3.2.2 Analyse des critères

Dans cette sous-section nous avons comparé chacun des critères (tableau II ci-dessous), définis dans la sous-section ci-dessus (3.3.2.1) entre eux afin de faire ressortir leur niveau d'importance. À l'intersection des deux critères comparés, un nombre est indiqué. C'est le numéro du critère le plus important du couple comparé. Par exemple : en comparant le couple « 1. Cloud privé » et « 2. Cloud public », nous avons estimé que dans le cadre de ce projet, il est plus important de tester une solution de type « Cloud privé » qu'une solution de type « Cloud public ». La valeur retenue à l'intersection du couple comparé est « 1 ».

	1. Cloud privé	2. Cloud public	3. Communauté / documentation / support	4. Data center en Europe	5. Data center en Suisse	6. Data center n'importe où	7. Elasticité	8. Facilité de déploiement	9. Langages de programmation	10. Scalabilté	11. Sécurité des données	12. SLA	13. Standards ouverts
1. Cloud privé	X	1	3	1	5	1	7	1	1	10	11	12	1
2. Cloud public		X	3	4	5	2	7	8	9	10	11	12	13
3. Communauté / documentation / support			X	3	3	3	3	3	3	3	11	12	13
4. Data center en Europe				X	5	4	7	8	9	10	11	12	13
5. Data center en Suisse					X	5	7	8	5	10	11	12	5
6. Data center n'importe où						X	7	8	9	10	11	12	13
7. Elasticité							X	7	7	7	11	7	7
8. Facilité de déploiement								X	8	10	11	12	8
9. Langages de programmation									X	10	11	12	13
10. Scalabilté										X	11	12	10
11. Sécurité des données											X	11	11
12. SLA												X	12
13. Standards ouverts													X

Tableau II : Matrice de préférence

En comptant, dans le tableau (II) ci-dessus, le nombre de fois qu'un critère a été défini plus important qu'un autre, nous avons établi le tableau (III) ci-dessous.

Critères	Points	Pourcentage
11. Sécurité des données	12	15
7. Elasticité	10	13
12. SLA	10	13
3. Communauté / documentation / support	9	12
10. Scalabilté	8	10
1. Cloud privé	6	8
5. Data center en Suisse	6	8
8. Facilité de déploiement	6	8
13. Standards ouverts	5	6
9. Langages de programmation	3	4
4. Data center en Europe	2	3
2. Cloud public	1	1
6. Data center n'importe où	0	0

Tableau III : Synthèse de la matrice de préférence

Le tableau (III) ci-dessus fait ressortir que le critère le plus important pour nous, afin de choisir une solution de « Cloud Computing », est la « sécurité des données » (11). Ce critère est un frein au développement du « Cloud Computing » public de manière générale, car les sociétés gèrent les données confidentielles de leurs clients et/ou des données qui pourraient être intéressantes pour des concurrents. Le niveau de sécurité des données peut aussi varier en fonction de la domiciliation du serveur physique.

Dans les cinq premiers critères, nous retrouvons l'« élasticité » (7) en deuxième place et la « Scalabilité » (10) en cinquième place. Ces deux concepts sont pour nous à la base du nouveau concept qu'est le « Cloud Computing ». Ils sont une des valeurs ajoutées du « Cloud » par rapport à la méthode actuelle de gestion d'un « data center ». Ces deux concepts permettent d'initier des mécanismes qui permettront rapidement de réduire les coûts d'exploitation.

La qualité du service liée au « SLA » (12) et la notion de « communauté / documentation / support » (3) sont aussi très importantes pour nos projets. Ces deux critères sont dans le quinté de tête. Le critère numéro trois a souvent été un facteur d'échec lors de projets menés par nos clients. En effet, une communauté sans le soutien de l'industrie ne vit pas longtemps. Pour garantir sa pérennité, cette communauté doit donc être soutenue par un des grands acteurs du marché comme Google, IBM, Microsoft, etc. Le SLA est l'« assurance vie » du client, sans un SLA clairement détaillé et précisant toutes les conditions d'exploitation, le client ne pourra pas se retourner contre son prestataire de service en cas de problème.

Les trois critères suivants « Cloud Privé » (1), « data center en Suisse » (5) et « facilité de déploiement » (8) montrent encore le grand intérêt de notre mandant, l'Administration cantonale, à garder la maitrise sur ses informations et sur le système d'information en général.

Les cinq derniers critères sont d'importance moindre. Le fait que nous n'ayons attribué aucun point au dernier critère « data center n'importe où » (6) traduit le fait que l'État ne veut prendre aucun risque avec ses données, même si elles ne sont pas critiques.

3.3.3 Analyse multicritères

L'analyse multicritères nous permet de comparer entre elles les solutions retenues à la section (3.3). Dans le tableau (IV) ci-dessous, nous attribuons une note pour chaque critère défini dans les sous-sections ci-dessus (3.3.2.1). Cette note est pondérée avec le pourcentage obtenu par chaque critère dans la sous-section « analyse des critères » (3.3.2.2). La somme de tous les points obtenus permet de comparer les solutions entre elles. Dans le tableau (V) le coût annuel de chaque solution est divisé par le nombre point, ce qui permet d'établir le « prix par point ». Ce score permet de définir quelle solution offre le meilleur rapport qualité / prix.

	Solution	Amazon EC2		AppScale		Azure		Google App Engine		VMware vSphere 4 – Dell	
Critères	**Pourcentage**	**Notes**	**Points**	**Notes**	**Points**	**Notes**	**Points**	**Notes**	**Points**	**Notes**	**Points**
11. Sécurité des données	15	4	62	4	62	6	92	5	77	8	123
7. Elasticité	13	0	0	4	51	4	51	9	115	5	64
12. SLA	13	9	115	7	90	8	103	0	0	5	64
3. Communauté / documentation / support	12	6	69	5	58	7	81	7	81	9	104
10. Scalabilté	10	10	103	10	103	10	103	10	103	10	103
1. Cloud privé	8	5	38	5	38	0	0	0	0	10	77
5. Data center en Suisse	8	0	0	0	0	0	0	0	0	10	77
8. Facilité de déploiement	8	7	54	6	46	9	69	9	69	8	62
13. Standards ouverts	6	8	51	10	64	8	51	5	32	5	32
9. Langages de programmation	4	9	35	5	19	7	27	5	19	9	35
4. Data center en Europe	3	10	26	10	26	5	13	0	0	0	0
2. Cloud public	1	10	13	10	13	10	13	10	13	0	0
6. Data center n'importe où	0	10	0	10	0	10	0	10	0	0	0
Total Point		**88**	**565**	**86**	**569**	**84**	**603**	**70**	**509**	**79**	**740**

Tableau IV : analyse multicritère

Solution	Amazon EC2		AppScale		Azure		Google App Engine		VMware vSphere 4 - Dell	
	Notes	**Points**	**Notes**	**Points**	**Notes**	**Points**	**Notes**	**Points**	**Notes**	**Points**
Total	**88**	**565**	**86**	**569**	**84**	**603**	**70**	**509**	**79**	**740**
Charge financière		16'300		18'800		16'500		14'800		19'200
Prix par point		**29**		**33**		**27**		**29**		**26**

Tableau V : analyse multicritère et détail du coût par points

3.3.4 Analyse des résultats

Dans les tableaux présentés dans la sous-section ci-dessus (3.3.3), nous constatons que la solution de type « Cloud Privé » « VMware vSphere 4 – Dell » est la solution la plus intéressante. Ceci aussi bien au niveau du nombre de points obtenus – 740, qu'au niveau du coût par point (prix de la solution divisé par le nombre de points) qui est égal à 26.

Ce score est obtenu principalement grâce à l'excellent niveau de sécurité offert par une telle plate-forme. En effet, en ayant le contrôle total de la solution nous pouvons protéger totalement les données derrière des mécanismes de firewall et n'ouvrir que le ou les ports nécessaires pour l'application. De plus, nous savons qui peut accéder aux machines aussi bien d'un point de vue logique que physique. Les autres solutions de type « Cloud Public » recommandent de crypter toutes les données afin de garantir la confidentialité de ces dernières, ce qui peut générer des surcoûts importants.

Le deuxième point qui fait pencher la balance est « Communauté / documentation / support » (3). Ce point obtient une note plus élevée dans la solution « VMware vSphere 4 – Dell », car il est inclus dans le prix des contrats de support aussi bien du côté de Dell que du côté de VMware. De plus, ces sociétés ont des représentants techniques et commerciaux basés en Suisse ce que n'offrent pas forcément les autres sociétés.

Même si nous n'avons pas défini le critère « Data center en Suisse » (5) comme impératif, ce point obtient la note maximum de dix, car il reste important pour notre mandant – l'Administration cantonale. Le fait que la solution « VMware vSphere 4 – Dell » permet de faire du « Cloud privé » (1) est aussi un facteur qui influence fortement le résultat final. Ce critère permet de garantir un meilleur contrôle des données aussi bien du point de vue de la confidentialité que de celui de la sécurité de ces informations.

Finalement, la solution « VMware vSphere 4 – Dell », contrairement à certaines autres solutions comme « Amazon EC2 » qui nécessitent l'achat d'un service supplémentaire « Elastic Load Balancing », nous permet de répondre à nos objectifs (3.2.1) : *tester et valider une solution de type « Cloud Computing » qui nous affranchisse des problèmes liés à la répartition de charge et de la haute disponibilité*. De plus, avec les services inclus dans la solution « VMware vSphere 4 – Dell » nous pouvons respecter notre contrainte (3.2.2) : *le niveau de service pour ce projet doit être d'au moins*

99.5%. Cet objectif ne peut être atteint avec la solution de Google qui n'offre aucune garantie de service, ou avec les solutions de Microsoft et Amazon qui nécessitent l'utilisation d'instances de machines virtuelles basées dans au moins deux zones géographiques, ce qui double les coûts.

Pour l'hébergement de notre prototype nous retenons donc la plate-forme « VMware vSphere 4 – Dell », hébergée au sein de nos salles machines.

4 Réalisation technique

Après avoir choisi la plate-forme qui hébergera notre projet d'« étiquetage collaboratif » ou « folksonomie », nous allons maintenant décrire tous les éléments qui ont permis la réalisation de ce projet. Nous commencerons ce chapitre par une description de l'architecture complète de la solution. Elle consiste en une description des infrastructures techniques et applicatives. Nous exposerons subséquemment les résultats des tests conduits sur cette application. Nous conclurons ce chapitre par une série d'améliorations à conduire pour une industrialisation de cette application.

4.1 Modélisation architecturale

Dans cette section nous allons détailler les trois couches mises en œuvre dans ce projet. La première, qui est la plus basse, décrit les infrastructures physiques. La deuxième expose l'architecture du serveur applicatif. Les dernières sections (4.2 et 4.3) sont consacrées à l'application en elle-même et elles détaillent les cas d'utilisation, le diagramme de déploiement, le modèle physique de données et le code de l'application.

4.1.1 Architecture serveur physique

Suite à notre analyse multicritères (3.3.3), nous avons retenu pour ce projet la solution « VMware vSphere 4 – Dell ». Cette solution garantit un excellent niveau de sécurité pour les données, même si elles ne sont pas critiques dans le cadre de ce projet. De plus, cela nous permet aussi d'héberger une première application sur un cluster de machines répondant à notre définition du « Cloud Computing ».

Ce choix nous permet donc de garder un contrôle total sur les serveurs et de rassurer la maîtrise d'ouvrage quant à la sécurité des données. Cela nous donne également la possibilité de prendre en main une architecture de cluster de serveurs que tous les fabricants de hardware commencent à commercialiser et ce, en limitant les risques.

Ces serveurs sont hébergés dans nos salles machines et offre un niveau de service (SLA) tendant vers le cent pourcent. De brèves interruptions de service, inférieures à cinq heures par an, seront nécessaires pour effectuer les maintenances nécessaires sur le SAN (Storage Area Network). Ce dernier élément est le seul non redondant de cette nouvelle architecture, mais il pourra facilement être doublé dans le futur.

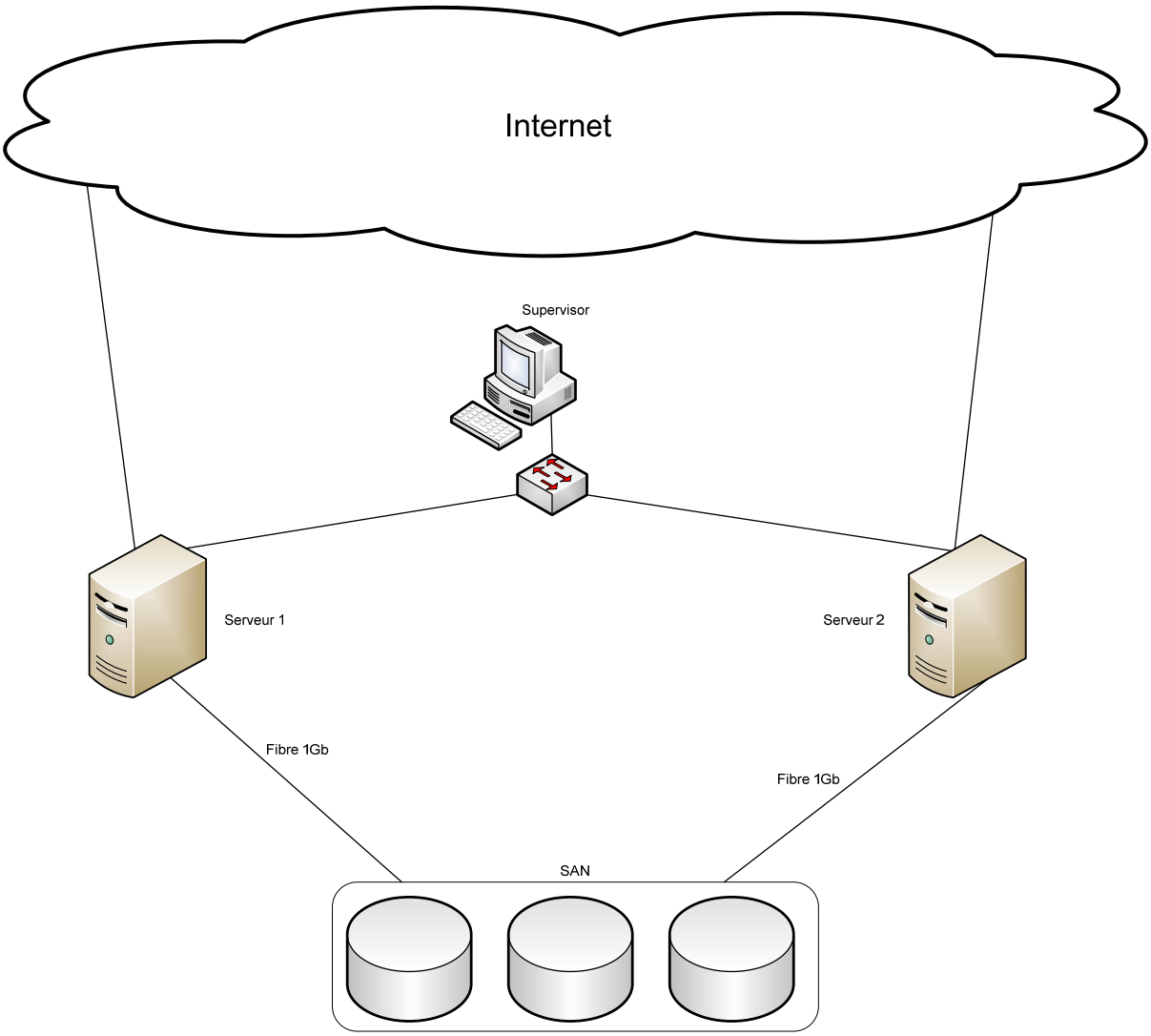

Figure 7 : Schéma d'architecture serveur

Le schéma ci-dessus présente l'architecture serveur qui héberge notre application de « folksonomie ». Ce cluster « VMware vSphere 4 – Dell » est composé de deux serveurs Dell. Chaque serveur a deux processeurs Xeon Quad-Core à 2.4 GHz et 48 Go de RAM. Ces serveurs sont connectés à un SAN (Storage Area Network), par deux liens fibre à 1 Go chacun. Ce SAN est composé de quinze disques de 300 Go chacun. Les serveurs sont aussi branchés par un lien réseau à 100 Mo à un simple ordinateur qui sert de « supervisor ». Le « supervisor » monitore les deux serveurs en permanence

pour gérer la haute disponibilité (HA). Finalement, en faisant abstraction de tous les mécanismes de routage et de sécurité, chaque serveur a un lien direct sur « Internet ».

Sur chaque machine est installée la suite complète « vSphere 4 » de VMware. Cette solution est composée de deux couches. Une couche communément appelée « hypervisor » qui virtualise tous les éléments du hardware et une couche composée de services. Cette seconde couche permet à VMware d'offrir une solution scalable, robuste et sécurisée pour sa propre solution de « Cloud Computing ». Nous décrirons ci-après les services les plus importants mis en œuvre pour notre projet :

- « VMware VMotion » et « VMware Storage VMotion » permettent de déplacer une machine virtuelle d'une machine physique à une autre sans interruption de service. Ce service permet d'effectuer une maintenance aussi bien hardware, que logicielle sur une des machines physiques sans arrêter les serveurs virtuels, ni perdre les sessions des utilisateurs.

- « VMware Fault Tolerance » est le service qui permet de garantir une disponibilité continue de la machine virtuelle même en cas de panne de la machine physique. Ce service est primordial pour offrir ce que nous appelons de la « haute disponibilité » (HA).

  C'est grâce à une combinaison des deux services « VMware VMotion » et « VMware Fault Tolerance » qu'un SLA supérieur à 99.95% peut être offert pour ce projet de « folksonomie ». Ce niveau de service concerne les parties hardware et la plate-forme d'hébergement des machines virtuelles. Ce taux peut être porté à 99.999% (interruptions inférieures à dix minutes par an), si toutes les infrastructures physiques sont doublées et réparties entre au moins deux salles machines. Pour ce projet le « SAN » n'a pas été doublé.

- « VMware High Availability » permet de redémarrer une machine virtuelle dans la minute en cas de panne de hardware ou de problème avec le système d'exploitation.

  Dans le cadre de projet nécessitant de la HA, il est important de mener des campagnes de stress du système complet. Lors de l'exploitation, ceci permet de diminuer les problèmes de surcharge du système d'exploitation liés à l'application.

- « VMware DRS » est le service qui gère la scalabilté et l'élasticité. Il permet d'allouer des ressources (processeur, mémoire) supplémentaires à une machine virtuelle dynamiquement et sans interruption de service. Lors de la création d'une machine virtuelle, une certaine

puissance processeur et une certaine quantité de mémoire vive sont réservées. Une limite maximum de ressource disponible peut-être fixée. Cette limite maximum a pour avantage de limiter le risque de voir les performances des autres machines virtuelles se dégrader à chaque ajout d'une nouvelle machine virtuelle. Par contre, cette limite maximum ne permettra pas à la machine virtuelle d'utiliser des ressources non allouées en cas de fort pic de charge.

VMware offre aussi des services pour garantir la sécurité de tout le système et assurer les backups. Dans le cadre de ce projet, ce type de service est directement géré par les ingénieurs système en charge des machines et il n'est pas forcément celui vendu par VMware.

4.1.2 Architecture serveur applicatif

La solution « VMware vSphere 4 – Dell », telle que décrite à la sous-section (4.1.1) ci-dessus, peut héberger une quarantaine de machines virtuelles. VMware garantit le fonctionnement de cinq machines virtuelles par « core » de processeur.

Pour notre projet d'application de type Web 2.0, nous utilisons une machine virtuelle avec un processeur de 2.4 GHz et 2 Go RAM. En fonction des besoins et grâce aux mécanismes « VMware DRS », la machine peut consommer plus de ressources hardware à condition que ces dernières ne soient pas réservées ou déjà allouées aux autres machines virtuelles du cluster.

Sur cette instance de serveur applicatif, notre équipe d'ingénieurs système a installé la version 8.04 d'Ubuntu server comme système d'exploitation. Sur ce système d'exploitation, nous avons installé un serveur Apache 2.2.8 avec PHP 5.2.4. Un serveur de base de données MySQL 5.0.51a administré au travers de PhpMyAdmin 2.11.3 a aussi été installé pour le stockage des données.

Cette architecture de serveur applicatif, de type monolithique, nous permet de répondre à certains de nos objectifs (3.2.1) : *s'affranchir des problèmes liés à la répartition de charge et de la haute disponibilité*. En ayant une seule machine disponible en permanence grâce aux mécanismes « VMware VMotion » et « VMware Fault Tolerance », nous nous affranchissons des mécanismes de clustering de serveurs (décrit dans notre sous-section 1.1.3) et de répartition de charge.

Cette solution nécessite moins de ressources humaines pour la gestion du serveur virtuel. Elle permet aussi de facilement migrer tout le projet d'un environnement de développement, vers un environnement de recette et finalement vers un cluster de machines de production.

Cette architecture permet également de simplifier la gestion de la sécurité. En effet, une architecture monolithique évite la mise en place d'une solution de reverse-proxy ou de tunnel SSL entre le serveur frontal, le serveur applicatif et le serveur de base de données. Ce dernier point fait l'objet de projets constants, au sein de nos équipes, afin de toujours améliorer la sécurité.

Par contre, cette solution monolithique ne permet pas de mettre en place des solutions MySQL, telles que la réplication (master – slave) ou le clustering, pour faciliter la montée en charge de la base de données.

Étant donné la nature prototype fonctionnel (POC) de ce projet d'application Web 2.0, ces considérations d'améliorations seront abordées dans la section (4.4).

4.1.3 Architecture applicative

L'application met en œuvre quatre use-cases principaux que nous décrirons formellement ci-dessous (4.2.1) pour l'application « folksonomie ». Les deux premiers permettent à tous les internautes de soumettre des demandes à l'application. Le troisième permet de valider ou supprimer les données soumises par les internautes. Le dernier use-case permet à l'internaute de consulter l'ensemble des données au travers d'un Web service.

Pour rappel, le but de ce projet est de réaliser un prototype fonctionnel. Lors de l'industrialisation de l'application, il faudra étudier et développer plusieurs use-cases supplémentaires comme la gestion

des utilisateurs et des rôles ou la prise en compte de la gestion de plusieurs sites Web dans une seule instance de l'application. Ces cas d'utilisations ne sont pas primordiaux pour le fonctionnement de l'application, mais ils en facilitent l'administration.

L'application « folksonomie » est divisée en quatre composants. Deux d'entre eux englobent toute la partie soumission de données. Un autre composant s'occupe de toute la partie recherche de données au travers d'un Web service. Le dernier composant sert à administrer le contenu soumis par les internautes. Tous ces composants seront détaillés dans la sous-section consacrée au diagramme de déploiement (4.2.2) ci-dessous.

D'un point de vue technique et dans un souci de performance, chaque fonctionnalité de l'application fait l'objet d'un script procédural indépendant. Seuls les paramètres et certaines fonctions comme la connexion à la base de données et la gestion des erreurs de l'application ont été regroupés dans un fichier commun. Ceci afin d'éviter de les réécrire dans tous les scripts. Nous décrivons le code de manière plus précise dans la sous-section (4.2.4).

4.2 Développement du site Web 2.0 « Folksonomie »

Dans cette section nous allons détailler toute la partie fonctionnelle et technique de notre projet de « Folksonomie ». Nous commencerons cette section par la description des quatre use-cases développés dans le cadre de ce prototype fonctionnel. Nous décrivons ensuite les packages applicatifs. Puis nous présenterons le modèle physique de données. Nous finirons cette section avec une description des éléments principaux du code PHP.

4.2.1 Use-cases

Pour ce prototype fonctionnel qui permet à chaque internaute d'étiqueter (ajouter des mots-clés) une page Web de façon collaborative, nous avons identifié quatre use-cases (figure 8). Le premier permet à l'internaute d'ajouter un mot-clé. Le deuxième cas d'utilisation modélise le processus d'évaluation d'un mot-clé. Le troisième diagramme décrira la partie de gestion des mots-clés par un administrateur. Finalement, nous décrivons la partie recherche de mots-clés et URLs au travers d'un Web Service.

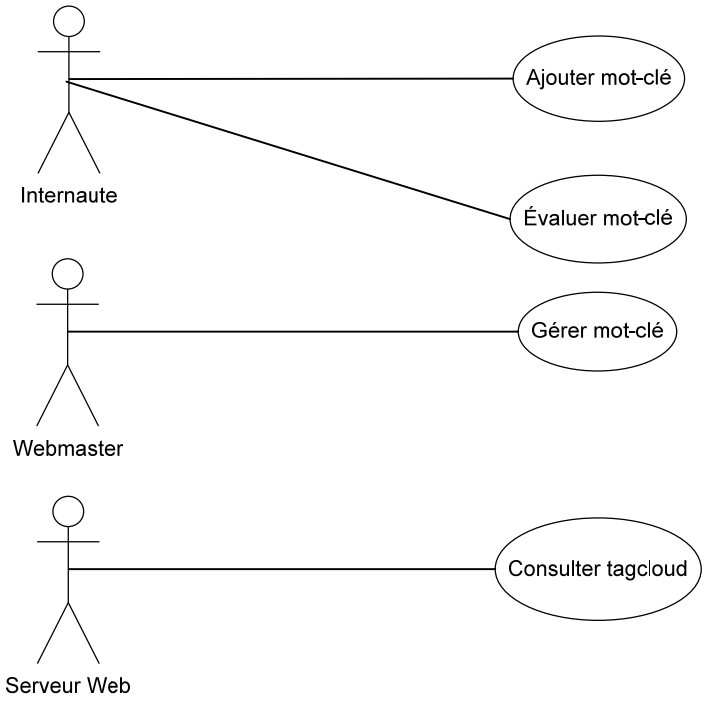

Figure 8 : diagramme de use-cases

4.2.1.1 Ajouter mot-clé

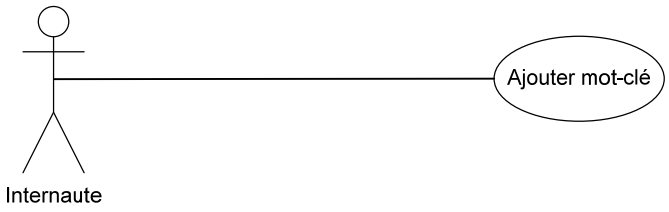

Figure 9 : use-case « ajouter mot-clé »

- **Nom :** Ajouter mot-clé
- **Acteur primaire :** Internaute
- **Pré-requis :** L'application est chargée dans la page Web
- **Déclencheur :** L'internaute clique dans le champ de saisie
- **Flot de base :**
  1. L'utilisateur saisit du texte
  2. Le système propose une liste de mots-clés commençant par « la saisie »
  3. L'utilisateur valide le mot-clé
  4. Le système enregistre le mot-clé
  5. Le système met à jour la liste des mots-clés dans la page web
- **Flots alternatifs :**

  2.1. L'utilisateur sélectionne un mot-clé de la liste

  2.1.1. Le système complète le champ de saisie

  2.1.2. Le système met le focus sur le bouton de validation

2.2. L'utilisateur ignore les propositions et continue sa saisie. Continue au point 2

4.1. Le système connait le mot-clé

4.1.1. Le mot-clé est interdit

4.1.1.1. Le système avertit l'internaute

4.1.1.2. Fin du use-case

4.1.2. Une relation mot-clé – page existe

4.1.2.1. Le système incrémente la note du mot clé

4.1.3. Le système ajoute une relation mot-clé – page

4.2. Le système ne connait pas le mot-clé

4.2.1. Le système ajoute le mot-clé

4.2.2. Le système ajoute une relation mot-clé – page

4.2.1.2 Évaluer mot-clé

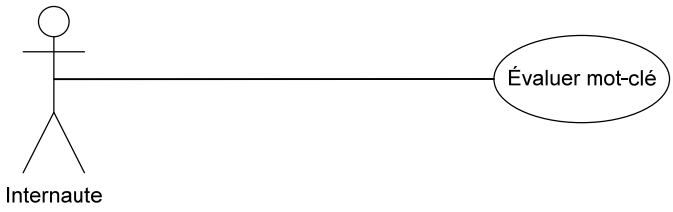

Figure 10 : use-case « évaluer mot-clé »

- **Nom :** Évaluer mot-clé
- **Acteur primaire :** Internaute
- **Pré-srequis :** L'application est chargée dans la page Web, un mot-clé est affiché
- **Déclencheur :** L'internaute clique sur un bouton « thumb up » ou « thumb down »
- **Remarque :** Ce use-case est similaire pour les deux types de boutons, seule l'opération en base de données est différente. « Thumb up » incrémente d'un la valeur de l'enregistrement et « thumb down » décrémente d'un la valeur de l'enregistrement
- **Flot de base :**
  1. L'utilisateur clique sur le bouton
  2. Le système met à jour la valeur en base de données
  3. Le système pose un cookie sur le client
  4. Le système désactive et masque les boutons du mot-clé associé
- **Flots alternatifs :** aucun

4.2.1.3 Gérer mot-clé

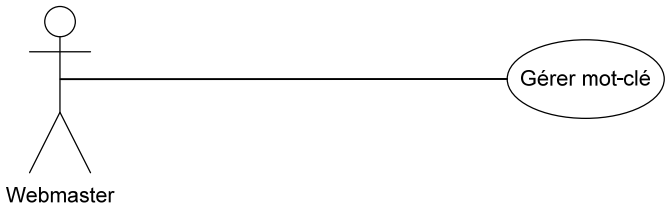

Figure 11 : use-case « gérer mot-clé »

- **Nom :** Gérer mot-clé
- **Acteur primaire :** Webmaster
- **Pré-requis :** Le webmaster est authentifié
- **Déclencheur :** Le webmaster charge la page Web de gestion des mots-clés
- **Flot de base :**
  1. L'utilisateur clique sur un des boutons d'actions (éditer, valider, bloquer)
  2. Le système valide les données
  3. Le système exécute l'action
  4. Le système réaffiche la page Web de gestion des mots-clés
- **Flots alternatifs :**

  2.1. Le webmaster « valide » un mot-clé

  2.1.1.Le système change le statut du mot-clé

  2.2. Le webmaster « bloque » un mot-clé

  2.2.1.Le système change le statut du mot-clé

  2.2.2.Le système supprime les relations mot-clé – page

2.3. Le webmaster « édite » un mot-clé

2.3.1. Le système vérifie si le mot-clé existe

2.3.1.1. Le mot-clé existe

2.3.1.1.1. Le système met à jour les relations mot-clé – page

2.3.1.1.2. Le système bloque l'ancien mot-clé

2.3.1.2. Le mot-clé n'existe pas

2.3.1.2.1. Le système ajoute le mot-clé

2.3.1.2.2. Le système met à jour les relations mot-clé – page

2.3.1.2.3. Le système bloque l'ancien mot-clé

4.2.1.4 Consulter tagcloud

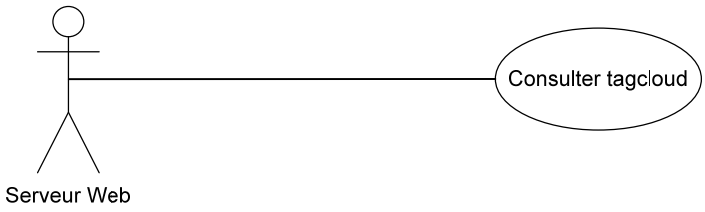

Figure 12 : use-case « consulter tagcloud »

- **Nom :** Consulter tagcloud
- **Acteur primaire :** Serveur Web
- **Déclencheur :** Un internaute appelle la page Web qui consomme le Web service
- **Flot de base :**
  1. Le serveur Web interroge le Web service
  2. Le Web service génère l'XML
  3. Le serveur Web parse le flux XML
  4. Le serveur Web affiche le résultat
- **Flots alternatifs :**

  2.1.1.Aucun paramètre n'a été passé

  2.1.1.1. Le système retourne la liste de tous les mots-clés

  2.1.2.Une ou plusieurs clés sont passées

  2.1.2.1. Le système fournit la liste du ou des mots-clés sélectionnés, la liste des mots-clés associés et la liste des URLs correspondant à la demande

Les quatre use-cases décrits ci-dessus sont des cas d'utilisations simples et qui peuvent être implémentés de manière indépendante. D'autres use-cases, comme une gestion des utilisateurs « webmaster » ou une gestion des sites Web, devront être ajoutés lors d'une phase d'industrialisation.

4.2.2 Diagramme de déploiement

Notre application de « folksonomie » se compose de quatre composants (« Application », « Key », « TagCloud » et « GUI JS »), répartis sur deux nœuds (« x : User Machine » et « x : Linux Server »). Un troisième nœud (« www.ge.ch »), qui héberge un composant tiers (« Parser XML ») est nécessaire pour gérer l'affichage (conversion de XML en HTML) des données retournées par le Web service du composant « TagCloud ».

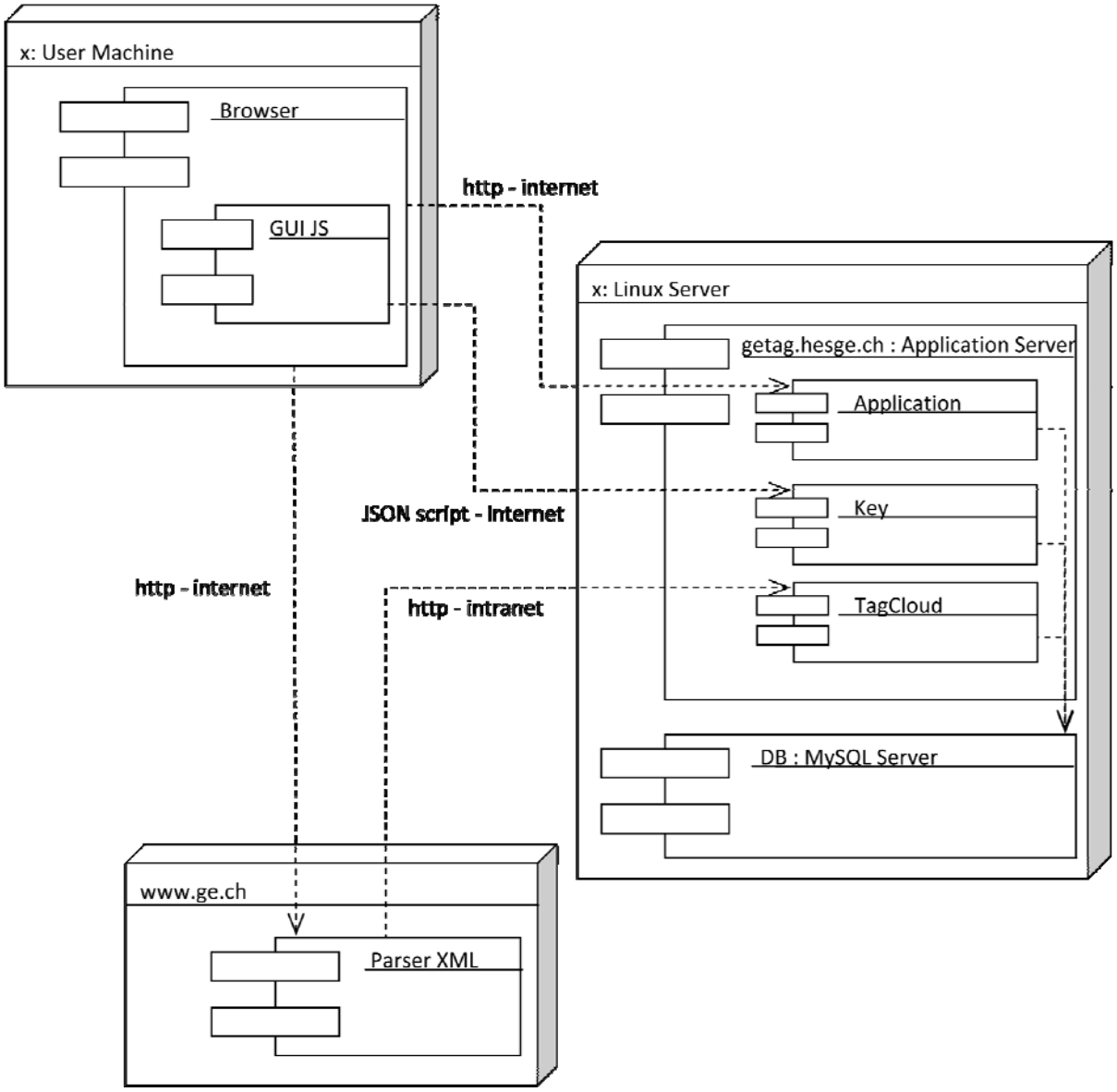

Figure 13 : diagramme de déploiement

Dans le nœud « user machine », nous avons le composant « GUI JS ». Ce composant, instancié depuis une page Web, gère la connexion client – serveur en utilisant la librairie « JSONscriptRequest ». Cette librairie a pour avantage, par rapport aux solutions AJAX avec « XMLHttpRequest », d'offrir une connexion « cross-domain ». Ainsi, le composant « GUI JS » peut être instancié dans une page Web hébergée sur un site Web différent de celui qui héberge l'application « folksonomie ». Sans ce type

de connexion, nous ne pourrions pas héberger notre service sur un « Cloud », car nous devrions l'héberger sur le même serveur que le site Web.

Ce composant gère aussi la transformation des données renvoyées par le composant « Key », de JSON en HTML. Le composant « GUI JS » s'occupe finalement d'ajouter les éléments graphiques nécessaires à l'utilisation de l'application « folksonomie » dans la page Web cliente.

Le second nœud « x : Linux Server » héberge deux composants « getag.hesge.ch : Application Server » et « DB : MySQL Server ». Ces deux composants pourraient être hébergés sur des serveurs distincts, mais pour répondre à l'un de nos objectifs (3.2.1), nous avons décidé de les grouper sur une seule machine. Le premier composant « getag.hesge.ch : Application Server » s'appuie sur le second composant « DB : MySQL Server » pour fournir l'information au composant « GUI JS ».

Le composant « getag.hesge.ch : Application Server » englobe trois composants « Application », « Key » et « TagCloud ».

- « Application » est le composant qui regroupe tout le code pour gérer le contenu de l'application « folksonomie ». Au travers de ce composant, nous pouvons valider, éditer ou bloquer un mot-clé. Lors de l'industrialisation, nous ajouterons dans ce composant toutes les fonctionnalités pour gérer les utilisateurs et les sites autorisés à utiliser l'application.

- Le composant « Key » fait l'interface entre la partie cliente « GUI JS » et la partie données stockées dans le composant « DB : MySQL Server ». « Key » valide les données reçues dans la requête HTTP et les formate pour soit interroger, soit mettre à jour la base de données en fonction de l'action demandée. Ce composant gère les cinq actions suivantes :

    - « Get » récupère la liste des mots-clés pour une page (URL) donnée

    - « Add » permet d'ajouter un mot clé dans une page donnée

    - « List » retourne une liste de tous les mots-clés commençant par la chaîne saisie par l'internaute. Cette action permet de faciliter la saisie de nouveaux mots-clés par l'internaute.

    - « Up » et « Down » sont deux actions qui fonctionnent de manière similaire. Ces actions permettent de noter l'importance d'un mot-clé pour une page.

- Le troisième composant « TagCloud » est un Web service qui sert à consulter par site l'ensemble des mots-clés enregistrés dans l'application. Ce Web service retourne soit une liste de mots-clés, soit le résultat d'une recherche par mots-clés. Le résultat de cette dernière retourne une collection de couples URL — titre de page, ainsi qu'une liste de mots-clés pour affiner ce résultat. Un composant « Parser XML » est nécessaire sur le nœud qui consomme ce Web service. Ce nœud pourrait être par exemple www.ge.ch. Ce composant ne fait pas directement partie du projet « folksonomie ».

4.2.3 Modèle physique de données

L'application « folksonomie » enregistre toutes ses données dans une base de données MySQL 5.0.51a. Ce moteur de base de données offre plusieurs mécanismes pour gérer l'enregistrement des données dans les différentes tables. MyISAM est le système le plus ancien pour gérer le stockage des données et donc le plus abouti [N_08].

InnoDB est plus récent. Il permet entre autres d'utiliser des mécanismes de transaction et de valider les relations entre les tables en implémentant le mécanisme de clés étrangères. De plus, lors de la mise à jour d'un tuple, InnoDB ne bloque que l'enregistrement impacté dans la table et donc facilite les accès concurrents.

MyISAM quant à lui bloque toute la table pour faire la même opération. Toutefois, MyISAM est moins gourmand en ressources processeur et mémoire car il bénéficie de nombreuses optimisations. Pour ce projet, nous avons comparé les deux solutions, MyISAM et InnoDB, en réalisant des tests de charge (4.3.2) et nous avons retenu InnoDB comme système pour gérer la base de données MySQL.

Dans le cadre de tous nos projets, nous appliquons les conventions suivantes pour nommer les tables et les champs. Le nom d'une table doit être significatif et il est toujours préfixé de l'acronyme « tbl ». Le nom des tables de jointure est composé des deux trigrammes des tables liées, préfixé par la lettre

« j » pour jointure. L'acronyme « log » signifie que la table n'est utilisée qu'à des fins d'enregistrement de données d'audit.

Chaque champ d'une table est défini en utilisant le trigramme de la table, suivi d'une lettre donnant la nature du contenu du champ (« c » : texte, « n » : numérique et « b » booléen) et d'un nom significatif. Par exemple : le champ « pag_c_url » est un champ de la table « page » (tbl_page) de type texte et qui contient un URL. Les clés étrangères reprennent le nom du champ auquel elles font référence.

Pour les champs de type date, nous appliquons le format suivant : année (sur 4 chiffres), mois, jour, heure, minute et seconde (sur 2 chiffres). Ce format a pour avantage de s'affranchir des différents paramétrages régionaux apportés au serveur ou hérités des utilisateurs. Et ce, tout en permettant de trier ou filtrer l'information de façon simple et efficace.

Pour gérer l'état d'un enregistrement, nous avons défini la convention suivante : « 0 » actif et valide, « 1 » actif non validé et « 2 » signifie que la donnée a été supprimée ou désactivée.

Pour les données de type texte, nous utilisons le format d'encodage UTF8_unicode_ci afin de nous abstraire des problèmes de caractères accentués et de la casse lors de recherches dans la base.

Certaines de nos applications peuvent être mises à disposition de tous. Pour cette raison, nous préférons utiliser l'anglais dans le code de nos projets.

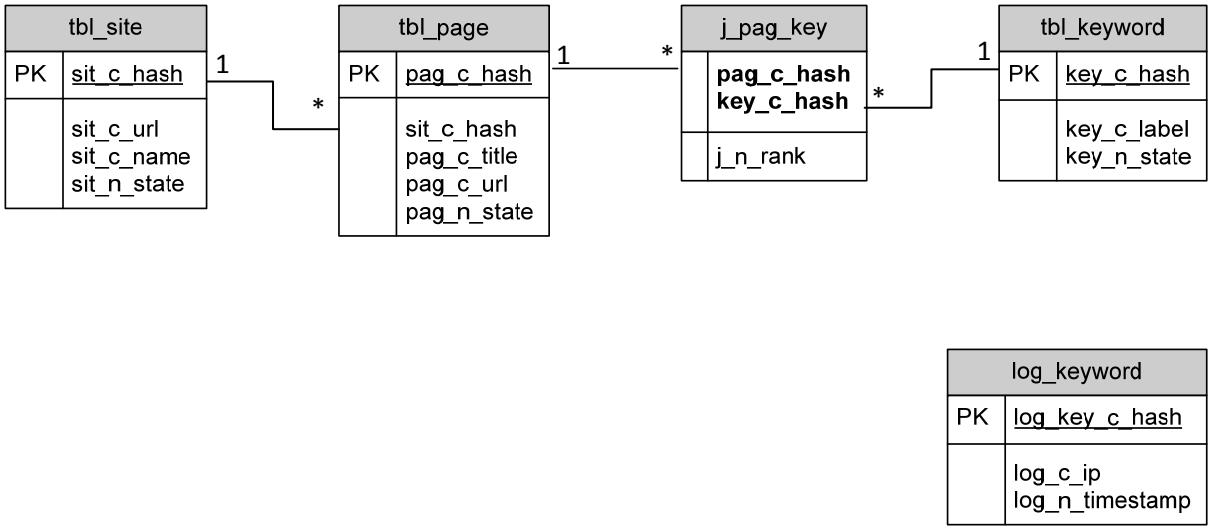

Figure 14 : Modèle Physique de Données

La base de données du projet « folksonomie » est composée de cinq tables (figure 14) :

- « tbl_site » contient la liste des sites (URLs racines) autorisés dans l'application. Un site est identifié par un hash. Chaque tuple est composé d'un URL, du nom du site et d'une variable numérique pour signifier son état.

- « tbl_page » contient la liste de toutes les pages (URLs) intégrant le composant « GUI JS », pour autant que la page ait été chargée au moins une fois. Chaque page est identifiée par un hash. Un enregistrement de la table « tbl_page » contient le titre et l'URL de la page, ainsi qu'une variable numérique pour signifier son état. Le champ « pag_c_title » est un champ qui est régulièrement mis à jour.

- « tbl_keyword » contient la liste de tous les mots-clés soumis par les internautes au travers du composant du « GUI JS ». Chaque mot-clé est identifié par un hash. Chaque tuple est composé du « mot-clé » et d'un champ numérique pour signifier son état.

- « j_pag_key » est la table de jointure entre les tables « tbl_page » et « tbl_keyword ». Chaque enregistrement contient le couple de clés étrangères nécessaires à cette jointure, plus un champ numérique permettant de connaître l'importance de cette liaison. La valeur par défaut de ce dernier champ est « zéro ». Cette table est la plus sollicitée du projet car le champ « j_n_rank » est constamment mis à jour par le vote des internautes.

- « log_keyword » est la table qui enregistre à des fins d'audit, « qui » soumet « quel » mot-clé. Seuls les nouveaux mots-clés sont enregistrés dans cette table. Chaque tuple est identifié par le hash du mot-clé. L'adresse IP de l'internaute et le timestamp sont ajoutés à cet enregistrement.

Des tables supplémentaires, par exemple une table utilisateur, pourront être ajoutées lors de l'industrialisation du projet « folksonomie ».

4.2.4 Code de l'application

Pour ce projet, nous avons retenu le PHP comme langage principal. Ce choix est basé sur plusieurs constatations. La première constatation est que nous n'avions encore jamais réalisé de projet avec ce langage sur le site officiel de l'État de Genève. Nos projets précédents étaient soit en ASP 3.0 (Active Server Pages), soit en JSP (JavaServer Pages) avec différents concepts architecturaux mettant parfois en œuvre des framework de développement.

L'ASP est un langage développé par Microsoft et qui est distribué avec les serveurs IIS (Internet Information Services) depuis 1996 (année de lancement du premier site officiel de l'État de Genève). Ce langage n'a plus évolué depuis la version 3.0 en 2000. Toutefois, l'ASP nous permet encore de développer rapidement de petits sites Web dynamiques, mais il ne nous permet plus de répondre à tous les besoins.

Le JSP offre plus de fonctionnalités que l'ASP, mais il est plus difficile à mettre en œuvre et ce langage nécessite toujours la mise en place d'une équipe plus conséquente pour la réalisation d'un projet. Depuis le début des années 2000, nous suivons l'évolution du développement du langage PHP. Dès 2008, nous avons réalisé plusieurs sites pour notre institution et pour d'autres clients, mettant en œuvre du PHP 5 au travers du framework Zend[72]. Ce langage et surtout ce framework, nous ont permis de très rapidement réaliser des sites de qualité, répondant aux besoins de nos clients. Par conséquent, le PHP nous semble adapté à ce prototype fonctionnel, car c'est un langage simple, performant, facile à mettre en œuvre et qui continue à évoluer.

Le second argument qui nous a poussé à choisir le PHP est lié à notre objectif (3.2.1) qui consiste à nous affranchir des problèmes de répartition de charge. IIS est connu pour ses problèmes de stabilité et il nécessite régulièrement un redémarrage du service. Un seul serveur IIS ne nous permet pas d'atteindre un niveau de service proche de 100%. Le déploiement de sites en JSP chez notre client fait l'objet de procédures précises. Le projet nécessiterait au minimum deux serveurs : un frontal Apache et un serveur applicatif distinct. Aucune des deux solutions utilisées à l'heure actuelle ne nous permet de répondre à notre objectif : s'affranchir des problèmes liés à la répartition de charge et de la haute disponibilité.

[72] http://framework.zend.com/

En choisissant le PHP pour le projet « folksonomie », nous allons pouvoir réaliser un premier projet mettant en œuvre ce langage et ainsi montrer les avantages du PHP par rapport aux deux autres langages déjà utilisés pour le site Web officiel de l'État de Genève. De plus, nous respecterons notre objectif sans déroger aux procédures déjà en place et sans baisse de qualité de service.

L'application « folksonomie » (figure 15), est découpée en quatre composants (4.2.2). Chaque composant regroupe un ou plusieurs scripts développés en PHP 5, exception faite pour le composant « JS » qui est développé en JavaScript. Tous les scripts PHP utilisent un référentiel commun « Inc » : Ce référentiel regroupe les paramètres communs au projet et une libraire de fonctions offrant un mécanisme pour se connecter à la base de données et traiter les exceptions.

Étant donné la nature prototype fonctionnel du projet « folksonomie », nous avons opté pour un développement en PHP de type procédural. De plus, nous voulions avoir une application capable de facilement monter en charge sur une architecture monolithique. Dans notre cas, la programmation orientée objet en PHP n'aurait rien apporté, mis à part une complexité accrue au niveau du code.

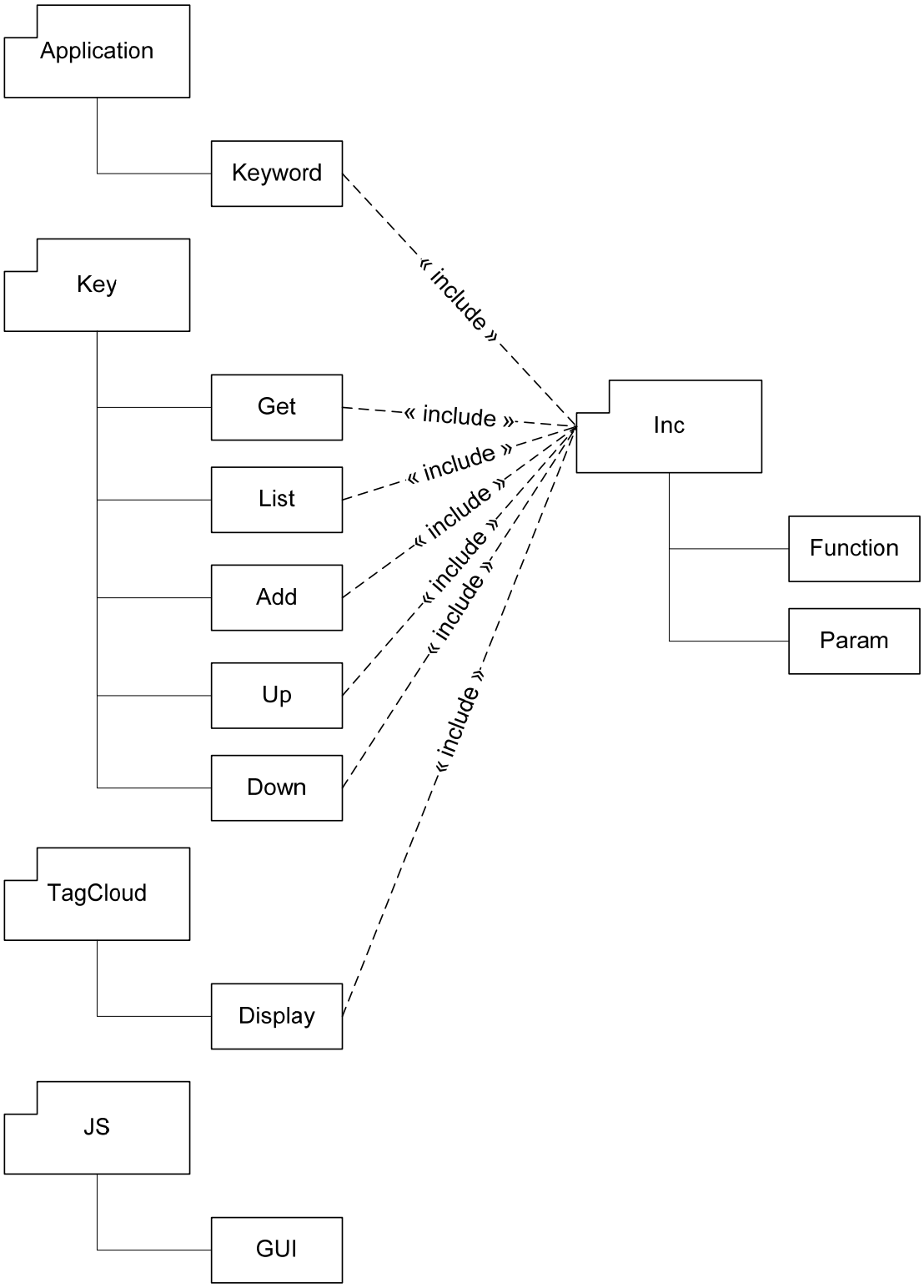

Figure 15 : Structure du code PHP

Le composant « Application », dans cette phase de prototype fonctionnel, ne contient qu'un seul script qui permet de valider, éditer et bloquer un mot clé.

Le composant « Key » a été divisé en cinq scripts, un par action, pour deux raisons. La première est de reproduire la structure (contrôleur – action) des URLs utilisés par des frameworks comme Zend et ainsi permettre à terme de facilement migrer sur un tel framework. De plus, ce choix nous évite aussi de devoir activer le « mod_rewrite » du serveur Apache. La non-utilisation de ce module nous permet donc de réduire la consommation des ressources sur le serveur et surtout nous évite d'écrire des « rewriterule » qui peuvent être complexes selon les besoins.

La seconde raison qui nous a poussé à faire ce choix est liée à l'optimisation de la consommation des ressources système. Compiler à la volée et exécuter des scripts qui pèsent entre 519 et 3'580 octets, nous semble moins gourmand en ressources système que de charger un script de 9'600 octets à chaque requête HTTP. De plus, ce script devra contenir un niveau de tests supplémentaires, afin d'exécuter l'action demandée par l'internaute. Avec l'architecture applicative que nous avons retenue, ce test est directement fait par le serveur Apache. Par manque de ressources, nous n'avons malheureusement pas réalisé de tests pour valider notre hypothèse d'optimisation de la consommation des ressources système lors de la réalisation de ce POC.

Chaque script du composant « Key » est autonome et commence par valider les données en entrée. Il exécute ensuite son action. Puis, il retourne dans le flux HTML une ligne de texte contenant l'appel de la fonction JavaScript à exécuter côté client (par le composant « GUI JS »). Cette fonction prend comme paramètre une chaîne de caractères structurée selon le standard JSON[73].

[73] http://www.json.org/

Le Web service « TagCloud » génère des flux XML, en fonction des paramètres reçus, respectant la DTD (Document Type Definition) suivante :

```
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT H (#PCDATA)>
<!ELEMENT L (#PCDATA)>
<!ELEMENT S (#PCDATA)>
<!ELEMENT ADR (#PCDATA)>
<!ELEMENT LBL (#PCDATA)>
<!ELEMENT A (L, H)>
<!ELEMENT C (L)>
<!ELEMENT R (H, L, S)>
<!ELEMENT U (ADR, LBL)>
<!ELEMENT URL (U+)>
<!ELEMENT TAG ((C, A?) | R)>
<!ELEMENT TC (TAG, URL?)>
```

Un « TC » (TagCloud) contient des « TAG » et des « URL » s'il y en a. Un « TAG » contient soit des « C » (mot-clé cliqué) et des « A » (mot-clé associé) s'il y en a, soit des « R » (mot-clé avec note). Les « C » (mot-clé cliqué) ne contiennent que des « L » (libellé), les « A » (mot-clé associé) contiennent des « L » (libellé) et des « H » (hash utilisé pour filtrer le résultat). Les « R » (mot-clé avec note) contiennent des « L » (libellé), des « H » (hash utilisé pour filtrer pour le résultat) et des « S » (note). Les « URL » contiennent au moins un « U » (URL) qui lui-même est composé d'une « ADR » (adresse HTTP) et d'un « LBL » (libellé titre de la page).

Pour ce Web service, nous avons volontairement choisi des balises avec des noms courts (3 caractères maximum), afin de limiter la quantité de données envoyées, ce qui diminue le temps de réponse et évite de charger la mémoire du serveur qui consomme ce Web Service.

Le dernier composant développé pour ce prototype fonctionnel est « GUI JS ». Ce composant, qui assure l'interface entre la page Web et le serveur applicatif, a été volontairement développé en JavaScript. Ce langage nous permet de facilement intégrer le GUI (Graphical User Interface – environnement graphique) de l'application « folksonomie » aux pages Web existantes du site officiel de l'État de Genève. Ce choix nous évite aussi de développer un « client » pour chaque technologie ou framework utilisés dans les pages Web clientes. Finalement, ce choix technologique nous permet de garder un contrôle complet sur le code exécuté côté client, car ce dernier est téléchargé depuis le serveur Web de l'application « folksonomie ».

Le code de cette application ne met pas en œuvre d'algorithmes complexes et il ne révolutionne pas l'algorithmique. Chaque fragment de code a par contre été pensé et optimisé pour diminuer les temps de réponse de l'application.

4.3 Validation de la réalisation technique

Dans une première phase, pour valider la réalisation technique du projet « folksonomie », nous avons réalisé plusieurs types de tests. Le premier concerne le GUI, c'est-à-dire l'intégration de l'application dans les sites Web cibles de l'État de Genève (figure 16). Le deuxième type de tests a servi à valider que l'application répondait dans un délai inférieur à 1 seconde dans 99% des cas. Le dernier test a permis de valider que le SLA de 99.95% peut être assuré sur la plate-forme « VMware vSphere 4 – Dell » hébergée dans nos salles machines.

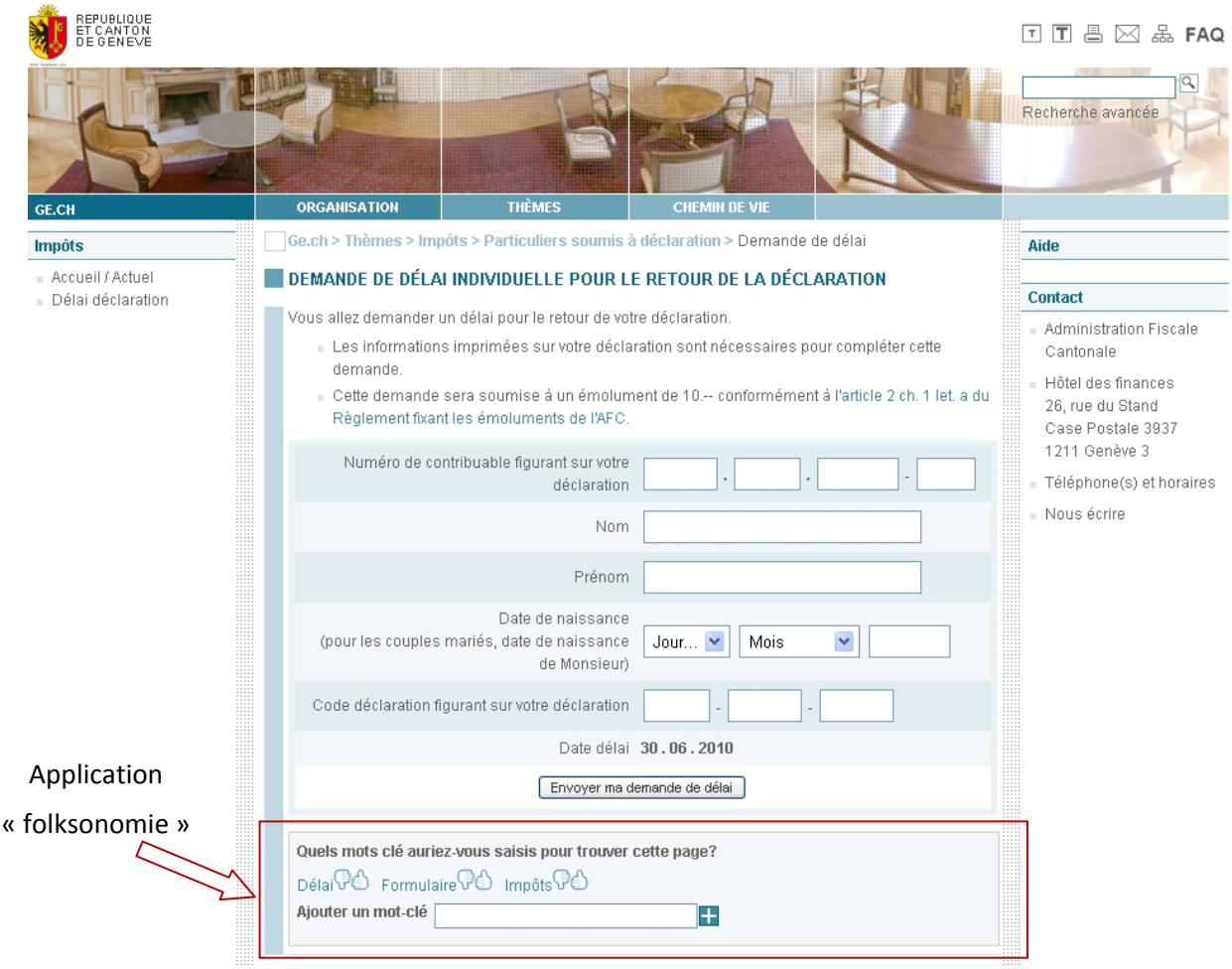

Figure 16 : intégration de l'application « folksonomie » dans une page du site officiel de l'État de Genève

Dans une seconde phase, en juin 2010, nous allons intégrer cette application dans des pages Web du site officiel de l'État de Genève. Ces tests nous permettront de valider l'intérêt des citoyens pour les nouvelles technologies dans un site officiel. Avec l'aide de spécialistes en information documentaire, nous validerons aussi la pertinence des mots-clés proposés par les internautes.

Nous allons décrire dans les sous-sections ci-dessous les tests que nous avons déjà réalisés.

4.3.1 Validation du design et test d'accessibilité

Pour le projet « folksonomie » nous avons la contrainte (3.2.2) de *respecter la charte éditoriale Web de l'État de Genève*. Nous avons donc fait valider cette contrainte par les spécialistes de la charte éditoriale Web, avant de présenter le projet aux membres du Comité Interdépartemental des Chargés de Communication. Ce comité, responsable de l'exploitation du site officiel de l'État de Genève, a montré un grand intérêt pour le projet. Il a été demandé, lors de la séance du 26 mars 2010, d'intégrer l'application « folksonomie » dans les pages Web du Département des constructions et des technologies de l'information (DCTI) et du Département des affaires régionales, de l'économie et de la santé (DARES).

Dans une seconde phase, nous avons collaboré avec une personne malvoyante[74] afin de valider l'accessibilité de l'application pour son handicap. Cette personne a pu très facilement, au travers des outils Jaws[75] (outil vocal de lecture d'écran) et une ligne braille, lire les mots-clés postés dans la page Web. Elle a aussi pu insérer des mots-clés sans difficulté. Cette personne n'a par compte pas pu noter les mots-clés à l'aide des images (thumb up et thumb down). Ce point sera évalué dans la section (4.4).

Pour faciliter l'accessibilité de l'application « folksonomie », nous avons aussi mis en œuvre les principes des normes WCAG 1.0 du W3C[76] (Web Accessibility Initiative) afin d'offrir un code de qualité et accessible. Nous n'avons par contre pas offert d'alternative pour les personnes ne supportant pas le JavaScript, car l'absence de cette application n'empêche ni l'utilisation du site, ni la navigation en son sein.

[74] Monsieur Julien Conti, chargé de mission pour l'accessibilité au CTI de l'État de Genève

[75] http://www.freedomscientific.com/jaws-hq.asp . Jaws est un outil capable de lire le contenu de l'information affichée à l'écran. Pour les pages Web cet outil se base sur le DOM (Document Object Model) de la page. A l'aide de touches de fonction, la personne malvoyante peut directement accéder aux titres, aux liens et aux formulaires de la page Web.

[76] http://www.w3.org/TR/WCAG10/

Finalement, nous avons testé sur différents browsers (Internet Explorer 6 et 8, FireFox 3.5 et 3.6, Chrome 4, Safari 4 et Opera 10) et sur différents systèmes d'exploitation (Windows XP, Vista et sur Ubuntu 9.10) le rendu graphique et le comportement fonctionnel du composant « GUI JS ». Seul Opera 10 nous pose un problème lié aux accents. Ce browser étant utilisé par moins de 0.26%[77] de nos internautes, nous n'avons pas, pour le moment, pris d'action corrective.

4.3.2 Tests de charge

Les tests de charge effectués sur le projet « folksonomie » ont deux objectifs. Premièrement, nous aimerions optimiser chaque script de l'application. Dans une seconde phase, notre souhait est de tester les limites de l'application et de la plate-forme.

Pour réaliser ces tests, nous avons installé l'outil Pylot[78] (version 1.26) sur trois terminal-serveurs, ce qui nous permet de simuler 150 utilisateurs virtuels. Cet outil open-source développé en python est distribué gratuitement sur internet. Il permet au travers d'un client lourd de facilement lancer une campagne de test. L'interface utilisateur permet de définir le nombre d'utilisateurs, le temps en millisecondes entre chaque requête HTTP, le temps pour monter en charge et la durée du test de charge. Lors du lancement du test, l'application charge un fichier XML contenant toutes les requêtes HTTP à exécuter durant le test. Pendant le test de charge, chaque utilisateur virtuel exécute de manière aléatoire les requêtes initialement chargées. A la fin du test, l'application génère un rapport montrant d'une manière générale le résultat du test de charge. Nous avons développé en plus, une petite application qui analyse les logs du test afin de ressortir par script le temps de réponse minimum, maximum et moyen. Ce second programme offre aussi la possibilité de sortir le pourcentage de requêtes qui prennent plus qu'un certain temps donné.

D'un point de vue volumétrique, pour ces tests nous avons pré-chargé dans la base de données plus de 100'000 enregistrements dans la table page. La table des mots-clés contient un peu moins de

[77] Valeur moyenne, pour les sites www.ge.ch, www.geneve.ch et www.ge.ch/votations-elections, relevée durant les 12 derniers mois sur Google Analytics, consulté le 4/10/2010

[78] http://www.pylot.org/

1'000 enregistrements. Tous ces enregistrements sont aussi présents dans la table réservée pour l'audit « log_keyword ». Chaque page est associée entre deux et cinq fois à des mots-clés, ce qui représente environ 400'000 enregistrements pour la table de jointure. Finalement, la table site ne contient qu'une seule adresse, celle du site officiel de l'État de Genève www.ge.ch.

Afin de valider les limites de l'application, nous nous sommes basés sur les statistiques de fréquentation des sites de l'État de Genève. Sur l'ensemble de ces sites, nous avons en moyenne environ 30'000 visiteurs uniques (un visiteur peut faire plusieurs visites par jour) quotidiennement. Ces internautes visitent en moyenne 120'000 pages tous les jours et ils téléchargent environ 2 millions de fichiers (image, feuille de styles, fichier JavaScript, etc.) associés aux pages visitées. Lors d'événements spéciaux comme des votations ou des élections par exemple, nous avons près de 80'000 visiteurs uniques. Ceci arrive quatre à six fois par an, hors événement exceptionnel. Ces jours-là, nos serveurs délivrent près d'un million de pages et 5 millions de fichiers sont téléchargés.

Pour valider nos tests de charge, nous avons estimé un nombre moyen de requêtes que l'application « folksonomie » peut avoir à traiter un jour normal et un jour spécial. Nous avons résumé ces données dans le tableau (VI) ci-dessous.

Script	**30'000 visiteurs**	**80'000 visiteurs**	
GUI JS	30'000	80'000	
Get	120'000	1'000'000	
List	30'000	80'000	~environ 20 requêtes par mot-clé ajouté
Add	1'500	4'000	5% des visiteurs
Up/Down	60'000	160'000	2 fois par visiteur
TagCloud	3'000	8'000	10% des visiteurs
Total	***244'500***	***1'332'000***	
Requêtes par seconde durant 10 heures	**6.79**	**37.00**	
Requêtes par seconde durant 4 heures	**16.98**	**92.50**	
Requêtes par seconde durant 1 heure	**67.92**	**370.00**	

Tableau VI : estimation de la charge

Dans le tableau (VI) ci-dessus, nous sommes partis du postulat pessimiste que chaque internaute chargeait chaque fois le composant « GUI JS ». Ce n'est généralement pas le cas grâce aux mécanismes de cache browser. Mais nous devons tenir compte que plus de la moitié de nos visiteurs sont de nouveaux internautes. Le script « Get » est exécuté lors de chaque chargement de page. En étant optimiste, nous avons estimé que cinq pourcent des utilisateurs de l'application « folksonomie » vont soumettre un mot-clé et qu'ils feront en moyenne vingt requêtes pour lister les mots-clés déjà enregistrés. Nous pensons qu'en moyenne chaque visiteur va évaluer (« Up » ou « Down ») deux mots-clés. Finalement, nous espérons avoir dix pourcent d'internautes qui utilisent le composant « TagCloud » une fois par jour. Nous avons calculé que l'application peut être sollicitée en moyenne 244'500 fois un jour normal et environ 1'332'000 fois un jour de forte affluence. Nous avons ensuite défini le nombre de requêtes HTTP par seconde que l'application doit pouvoir traiter. Le maximum estimé est de 370 requêtes par seconde, mais ce nombre devrait généralement être inférieur ou égal à 68 requêtes HTTP par seconde.

Pour valider les données décrites ci-dessus, nous avons effectué plusieurs fois des tests simulant des processus complets, c'est-à-dire :

1. Exécution du script « Get »
2. Trois chargements de listes de mots-clés au travers du script « List »
3. Ajout d'un mot-clé dans la base avec le script « Add »
4. Récupération de la liste des mots-clés de la page mise à jour utilisation du script « Get »
5. Évaluation du mot-clé avec les scripts « Up » et « Down »
6. Interrogation du Web service « TagCloud » tous les trente tests en moyenne

Entre chaque test nous changeons les paramètres suivants :

- Le nombre d'utilisateurs : 1, 15, 100 et 150
- Le système de gestion de la base de données MySQL : MyISAM et InnoDB

Les premiers tests nous ont permis de détecter les faiblesses de l'application. Afin d'améliorer les performances du projet « folksonomie », nous avons principalement travaillé sur le choix des indexes. Nous avons aussi simplifié certains codes en remplaçant par exemple la jointure entre la table « tbl_page » et « tbl_site » par un calcul de la clé étrangère « sit_c_hash » directement dans le code PHP « Get ». Finalement, nous avons rendu une partie du fichier XML, généré par le Web service « TagCloud », statique. Nous exécutons la requête SQL, qui prend entre 18 et 20 secondes une fois toutes les 4 heures et nous enregistrons le résultat au format XML directement sur le serveur. Avant de prendre cette décision, nous avons testé une solution utilisant une « vue » dans MySQL. Toutefois, cette solution ne nous offre pas une réelle amélioration, car une « vue » MySQL ne met pas en œuvre des mécanismes d'index pour faciliter la recherche. Dans ce Web service, nous avons aussi limité le nombre de mots-clés et URLs retournés par le système lors de requêtes pour un mot-clé donné.

Après ces améliorations, nous avons effectué à nouveau les tests pour vérifier que nous pouvons tenir nos objectifs quantitatifs définis ci-dessus. En nous basant sur les résultats du logiciel Pylot et l'analyse des logs, nous constatons qu'un utilisateur faisant en moyenne 7 requêtes par seconde obtient généralement un temps de réponse inférieur à 0.01 seconde, exception faite pour les interrogations du Web service qui prennent ordinairement moins de 0.06 seconde [Annexe I].

Quand nous faisons un test avec 150 utilisateurs faisant en moyenne environ 130 requêtes HTTP par seconde, le temps de réponse dans 99% des cas est inférieur à 0.027 seconde, pour une base MyISAM [Annexe J]. Toujours dans ce cas précis, moins de 0.23% des requêtes dépassent les 3 secondes et seulement 12 requêtes ont pris plus de 3.5 secondes.

Dans le cas d'une base de données InnoDB, 150 utilisateurs faisant en moyenne environ 130 requêtes HTTP par seconde, obtiennent un temps de réponse dans 99% des cas inférieur à 0.029 seconde [Annexe K]. Toujours dans ce cas, moins de 0.22% des requêtes dépassent les 3 secondes et seulement 5 requêtes ont pris plus de 3.5 secondes.

Ces tests nous ont permis de valider que l'application « folksonomie » peut traiter en une heure deux fois la charge globale prévue pour un jour normal sur le site officiel de l'État de Genève. De plus, la montée en charge n'occasionne pas de baisse significative de la performance. Ces derniers tests nous ont aussi permis de valider nos améliorations sur l'application, en matière d'indexage des champs MySQL et de mise en cache de la requête la plus pénalisante. Nous sommes ainsi passés d'un taux de 90% de requêtes HTTP répondants en moins d'une seconde à un taux de 99%.

Afin de valider les limites de l'application « folksonomie », nous avons utilisé la commande « ab » (Apache HTTP server benchmarking tool). En exécutant cette commande depuis une machine tiers, nous avons pu constater que le serveur tient une charge moyenne, générée par 75 utilisateurs concurrents, d'environ 730 requêtes par seconde. Cette charge soutenue n'a pas affecté les temps de réponse. Le temps maximum de réponse a été de 3.8 secondes et dans 99% des cas il était inférieur à 0.188 seconde.

Ces tests ne nous ont pas facilité le choix entre les deux systèmes de gestion de la base de données MySQL : MyISAM et InnoDB. En comparant les deux résultats [Annexes J, K], nous ne voyons pas une différence majeure. La base fonctionnant avec le système InnoDB répond en moyenne un millième de seconde plus rapidement. Les performances de cette base se dégradent légèrement moins rapidement que pour la base gérée avec MyISAM. Seulement 0.5% des requêtes effectuées sur la base InnoDB prennent plus d'une seconde. Sur le second système de gestion de base de données MySQL ce pourcentage passe à 0.54%. La base utilisant MyISAM offre de meilleures performances uniquement lors de l'exécution du script du Web service « TagCloud ». Contrairement aux autres scripts, ce dernier ne fait que des requêtes de type « SELECT ». Pour garantir un temps de réponse toujours le plus bas possible et surtout afin de prévoir un nombre de données toujours croissant, nous avons décidé d'utiliser le système InnoDB pour ce projet.

4.3.3 Tests de panne

Nous avons effectué un seul et unique test de panne sur l'infrastructure « VMware vSphere 4 – Dell ». Pour tester les mécanismes « VMware Fault Tolerance » et « VMware VMotion », nous avons brutalement arrêté une des machines physiques. En pingant en continu un des serveurs hébergés sur notre « Cloud », nous avons constaté à l'aide de la commande « ping <dest> -t », la perte d'un ping, soit une interruption d'une milliseconde du service. Les mécanismes « VMware Fault Tolerance » et « VMware VMotion » ont instantanément réagi et ils ont basculé tous les services sur le second serveur physique.

En cas de panne du SAN, n'ayant pas une infrastructure redondante pour cet élément, nous savons que l'ensemble des services serait interrompu pour une durée non déterminée.

Nous concluons qu'en l'état, nous ne pouvons pas héberger d'applications critiques sur ces serveurs, mais nous pouvons très bien héberger des projets ayant besoin d'un excellent service. La solution « VMware vSphere 4 – Dell » permet aussi de répondre à des projets qui ont des besoins en ressources systèmes à géométrie variable. Doubler le SAN a un coût qui devra être estimé pour garantir une redondance complète du système.

4.4 Améliorations à étudier

Les premiers points que nous allons étudier dans cette section afin d'améliorer le projet « folksonomie », concernent l'accessibilité. Nous y sommes contraints par l'Ordonnance fédérale « OHand » entrée en vigueur le premier janvier 2004[79]. Comme il est écrit dans le document « P028 – Directives de la Confédération pour l'aménagement de sites Internet facilement accessibles – Version 1.0 »[80] du 23 mai 2005, les nouveaux sites de la Confédération doivent respecter le niveau AA des normes WCAG 1.0 du W3C. Le canton de Genève a décidé d'appliquer la directive fédérale « P028 » depuis le premier janvier 2006.

[79] http://www.admin.ch/ch/f/rs/151_31/index.html, consulté le 19 avril 2010

[80] http://www.bk.admin.ch/themen/egov/00266/00267/index.html?lang=fr&download=M3wBPgDB_8ull6Du36WenojQ1NTTjaXZnqWfVp7Yhmfhnapmmc7Zi6rZnqCkkIN0gn,CbKbXrZ6lhuDZz8mMps2gpKfo, consulté le 19 avril 2010

Le second point d'amélioration que nous voulons étudier est lié à l'augmentation du nombre de sites et de pages que l'application pourrait être amenée à héberger. Le projet est destiné dans un premier temps au site officiel de l'État de Genève. Dans un deuxième temps, cette application pourra être proposée à tous les sites de l'État de Genève et donc référencer plusieurs millions de pages. Nous devrons alors étudier si le serveur, qui héberge l'application « folksonomie », est toujours capable de répondre en moins d'une seconde à 99% des requêtes. Si tel n'est pas le cas nous devrions étudier des solutions comme « MySQL replication », « MySQL cluster » ou même une solution Oracle. Dans ce dernier cas se poserait aussi la question du budget qui devra être plus conséquent. Les deux autres solutions permettent d'améliorer les performances en multipliant le nombre de serveurs de base de données à moindre coût. « MySQL replication » permet de mettre en place des mécanismes de type maître-esclaves. La base maîtresse s'occupe des enregistrements et les bases esclaves ne font qu'offrir des mécanismes de consultation de données. « MySQL cluster » permet de répartir la charge sur plusieurs serveurs de base de données de manière transparente pour l'utilisateur. Les deux solutions nous demandent une refonte complète de notre architecture serveur applicatif (4.1.2) et de notre architecture applicative (4.1.3).

Conclusion

Le concept de « Cloud Computing » n'en est qu'à ses débuts, mais il est déjà très prometteur. Offrir à moindre coût un data center dynamiquement scalable et élastique permet de répondre aux besoins en matière d'informatique de bon nombre d'entreprises. Malgré tout, il faut rester vigilant sur certains points comme la sécurité, la domiciliation des serveurs, l'élasticité ou le SLA.

Certains points comme la sécurité ont déjà été améliorés. Par exemple, pour améliorer la sécurité, de nouvelles offres permettent de créer un lien sécurisé et crypté entre l'entreprise et le prestataire de service[81].

D'autres points, comme l'élasticité nécessiteront encore beaucoup de travail de recherche pour répondre aux besoins des utilisateurs. Seul Google App Engine offre une solution de montée en charge et de réduction de puissance entièrement automatique et transparente pour le développeur. Les autres solutions nécessitent toutes des outils de monitoring externes capables de gérer la scalabilité des infrastructures de type « Cloud platform » et « Cloud infrastructure ».

La domiciliation des serveurs et le SLA sont les points qui font varier fortement la facture finale pour une société qui consomme des services externes hébergés sur un « Cloud ». Steve Ballmer, PDG de Microsoft, a dit lors d'une présentation à l'université de Washington[82] que "*…nous ne devrions pas nous soucier où les informations sont, …, mais il semble que les lois et les règlements se soucient vraiment où les informations sont.*". Amazon, Microsoft, Google et les autres acteurs du marché ne pouvant construire un data center dans chaque pays, une solution est de monter son propre data center. Microsoft et surtout les fabricants de hardware se positionnent sur ce marché en commercialisant des micros data center, comme la solution « VMware vSphere 4 – Dell » que nous avons utilisée pour notre projet.

[81] *Amazon Virtual Private Cloud (Amazon VPC)*, http://aws.amazon.com/vpc/, consulté le 29/04/2010

[82] Microsoft, Steve Ballmer, *The Cloud : Exciting New Possibilities*, http://www.microsoft.com/showcase/en/US/details/120eb97f-e7f8-4485-91dc-f6ada8e759e3, publié le 9/03/2010, visionné le 10/03/2010

Ce mémoire nous a permis de comprendre et d'expliquer les concepts qui font du « Cloud Computing » ce qu'il est. En choisissant une plate-forme de type « Cloud » pour notre projet basé sur le concept de Web 2.0, nous avons pu constater que les solutions ne sont pas encore matures et que comme nous venons de le dire ci-dessus, certains points doivent être améliorés. Toutefois, nous avons aussi pu remarquer que les solutions de « Cloud Computing » sont en amélioration constante afin d'offrir des plates-formes sûres et répondant aux attentes du marché. Nous allons donc continuer à surveiller le marché et préparer nos clients à franchir le pas de ce nouveau concept.

Grâce à ce mémoire, nous allons pouvoir suggérer à nos clients de nouvelles plates-formes pour leurs projets.

Ce projet de « folksonomie » nous a permis de tester et valider la solution « VMware vSphere 4 – Dell ». Cette plate-forme nous a permis d'atteindre nos objectifs (3.2.1) en respectant nos contraintes (3.2.2). Le point faible de cette application reste la base de données MySQL. Pour nos projets à venir, nous devrons trouver des alternatives (4.4) ou nous fixer des contraintes pour garantir l'exploitation des services ou applications hébergés sur un « Cloud ».

En juin 2010 les sites Web de l'État de Genève vont commencer à intégrer l'application « folksonomie ». Depuis avril 2010, je travaille à la définition, pour les cinq prochaines années, de l'infrastructure de type « Cloud » à déployer chez l'un de nos clients pour l'hébergement de ses sites Web.

En conclusion, j'aimerai dire que ce mémoire m'a permis de découvrir un domaine prometteur du monde informatique, de le comprendre, de l'analyser et ainsi de pouvoir dire modestement que je suis devenu une personne de référence, un spécialiste de cette technologie de « rupture » qu'est le « Cloud Computing ». Dans un avenir proche, de nombreux projets bénéficieront des opportunités nouvelles offertes par cette technologie.

Annexes

Bibliographie

[AFG_09] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica et Matei Zaharia. EECS Berkeley. Février 2009. *Above the Clouds: A Berkeley View of Cloud Computing*. 25 p.

[BDH_03] Luiz André Barroso, Jeffrey Dean, Urs Hölzle. IEEE. 2003. *Web Search for a Planet: The Google Cluster Architecture*. 7 p.

[CERN1] GridCafé.org. [En ligne]. http://www.gridcafe.org/. (Consulté le 10/07/2009)

[CT_01] Emiliano Casalicchio, Salvatore Tucci. IEEE. 2001. *Static and Dynamic Scheduling Algorithms for Scalable Web Server Farm*. 8 p.

[E_08] Thorsten von Eicken. Juillet 2008. *The Three Levels of Cloud Computing.* [En ligne] http://virtualization.sys-con.com/node/581961. (Consulté le 13/12/2009)

[F_02] Ian Foster. Argonne National Laboratory. 2002. *What is the Grid? A Three Point Checklist*. 4 p.

[F_74] George J. Feeney. ACM. 1974. *A panel session - Utility computing - A superior alternative?*. 2 p.

[G_09] Jeremy Geelan. Janvier 2009. Twenty-*One Experts Define Cloud Computing*. [En ligne] http://cloudcomputing.sys-con.com/node/612375. (Consulté le 12/08/2009)

[GL_09] Robert L. Grossman. IEEE. Mars / avril 2009. *The case for Cloud Computing*. 5 p.

[GS_08] Bernard Golden, Clark Scheffy. Wiley Publishing, Inc. 2008. *Virtualization for Dummies*. 50 p.

[L_08] Peter Laird. Septembre 2008. *Visual Map of the Cloud Computing/Saas/PaaS Markets*. [En ligne]. http://peterlaird.blogspot.com/2008/09/visual-map-of-cloud-computingsaaspaas.html. (Consulté le 04/07/2009)

[L_09] Neal Leavitt. IEEE. Janvier 2009. *Is Cloud Computing Really Ready for Prime Time? (IEEE).* 6 p.

[McK_09] Will Forrest - McKinsey & Company. Mars 2009. *Clearing the air on cloud computing*. 34 p.

[MG_09] Peter Mell, Tim Grance. NIST. Août 2009. *Draft NIST Working Definition of Cloud Computing (V15)*. 3 p.

[N_08] N. Newton. Avril 2008. *MySQL Engines: MyISAM vs. InnoDB*. [En ligne] http://tag1consulting.com/MySQL_Engines_MyISAM_vs_InnoDB. (Consulté le 16/04/2010)

[P_09] Guillaume Plouin. DUNOD. 2009. *CLOUD COMPUTING et SaaS*. 249 p.

[S_08] Michael Sheehan. Août 2008. *Introducing the Cloud Pyramid*. [En ligne] http://cloudcomputing.sys-con.com/node/609938. (Consulté le 10/01/2009)

[SUN_09] Sun Microsystems. Juin 2009. *Introduction to Cloud Computing Architecture*. 26 p.

[TCW_09] Tim Cowen, Jay Chaudhry, Michael G. Hill, Vincent Franceschini, Duncan Stewart, Panu Kause, Monique Morrow. Octobre 2009. *ITU Telecom World 2009 : Cloud Computing*.[Conférence] http://www.itu.int/WORLD2009/. (Participation le 8/10/2009)

[VRCL_09] Luis M. Vaquero, Luis Rodero-Merino , Juan Caceres, Maik Lindner. ACM SIGCOMM. 2009. *A Break in the Clouds: Towards a Cloud Definition*. 6 p.

[WIKI1] Wikipedia. 7 janvier 2010. *Folksonomie.* [En ligne] http://fr.wikipedia.org/wiki/Folksonomie. (Consulté le 1/02/2010)

[WTK_09] Lizhe Wang, Jie Tao, Marcel Kunze, Alvaro Canales Castellanos, David Kramer et Wolgang Karl. Rochester Institute of Technology. Octobre 2008. *Scientific Cloud Computing : Early Definition and Experience*. 18 p.

[WY_01] Joel L. Wolf, Philip S. Yu. ACM SIGMETRICS. 2001. *Load Balancing for Clustered Web Farms*. 3 p.

Liste des figures

Liste des tableaux

Amazon Elastic Compute Cloud (Amazon EC2)

http://aws.amazon.com/ec2/

1. *Présentation de la solution*

EC2 est une des solutions pionnières en matière de « Cloud Computing ». Avec sa solution EC2, Amazon permet à ses clients de louer à l'heure des instances de serveurs virtuels Linux ou Windows. Ces instances permettent d'exécuter des applications comme MySQL Enterprise, Oracle Database 11g, Hadoop, Apache HTTP ou IBM WebSphere Portal Server. Sur ces serveurs virtuels les clients d'Amazon peuvent aussi exécuter des centaines d'autres applications ou leurs propres applications.

Amazon commercialise des instances déjà préconfigurées appelées « Amazon Machine Images » (AMIs) et Amazon permet aussi de créer ses propres images de machines virtuelles.

En fonction de leurs besoins, les clients instancient autant de machines virtuelles que nécessaire. Amazon propose différents types d'instances en fonction des besoins. Elles peuvent offrir à l'utilisateur plus ou moins de mémoire, de puissance de calcul et d'espace disque.

Avec EC2 Amazon offre un certain nombre d'autres services à ses clients pour construire une solution capable de monter en charge et d'éviter les pannes.

- « Amazon Elastic Block Store » permet de conserver un espace de stockage indépendant des instances serveurs. Cet espace disque est partagé entre toutes les instances actives.
- « Multiple Locations » permet de choisir le lieu d'hébergement de ses instances parmi trois zones (États-Unis côte Est et côte Ouest ou Europe). Ce service permet d'éviter les pannes liées à un data centre particulier et de réduire le temps de latence réseau pour le client.
- « Amazon Virtual Private Cloud » (VPC) permet d'accéder à ses instances au travers d'un VPN et offre par conséquent une solution de « Private Cloud ».
- « Elastic Load Balancing » permet de répartir la charge entre les instances d'un même client.

2. SLA

Le « Service Level Agreement »[1] offert par Amazon est de 99.95% sur l'ensemble des zones géographiques. Le client peut demander des dédommagements pour autant que l'ensemble de ses instances aient été indisponibles pour plus de 5 minutes et qu'il n'ait pas pu créer de nouvelles instances. Amazon recommande de travailler au moins sur deux zones.

3. Technique

Amazon EC2 permet d'installer et de configurer sa propre instance de machine virtuelle en choisissant son système d'exploitation et ses applications et de créer son AMI. De plus Amazon offre des outils pour faciliter la création d'AMIs.

4. Prix

Le prix est difficile à calculer car chaque service est facturé séparément et son prix varie en fonction de la région géographique. Par exemple, l'hébergement d'une instance Linux de base coute 0.085 USD / heure sur la côte Est des États-Unis et 0.095 USD / heure sur la côte Ouest et en Europe. Le choix de la région géographique, du système d'exploitation et du type d'instance font varier les prix de 0.085 USD / heure à 3.16 USD / heure. Dans les coûts il faut ajouter les coûts de transferts de données, de stockage des données et tous les services supplémentaires.

Quant au prix de location d'une machine pour 24 heures il est le même que pour la location de 24 machines pendant une heure.

Amazon estime[2] le coût de l'hébergement d'un site web « marketing » sur deux zones à 1'400 USD par mois.

[1] http://aws.amazon.com/ec2-sla/

[2] http://calculator.s3.amazonaws.com/calc5.html

5. *Avantage / inconvénient*

Avantages :

+ Possibilité de créer ses propres AMIs, répondant à ses besoins et selon ses standards.

+ Multizone qui réduit les risques de pannes et diminue la latence réseau.

+ Nombreux AMIs disponibles.

+ Services supplémentaires déjà développés comme le VPC (Virtual Private Cloud).

Inconvénients :

- Prix flous, prix cachés (par exemple : licences).

- Fortes contraintes architecturales pour le développement d'application web scalable (par exemple : gestion de la session inter AMIs).

6. *En résumé*

Une solution robuste, idéale pour le calcul de type batch, pour l'hébergement de Web services ou pour l'hébergement de sites statiques. Pour les sites plus interactifs, Amazon EC2 n'offre pas une réelle solution comparée aux solutions des hébergeurs traditionnels, mis à part son implantation aux États-Unis et en Europe.

ElasticHosts

http://www.elastichosts.com/

1. *Présentation de la solution*

ElasticHosts est une solution européenne qui offre à ses clients une solution de location d'instances de serveurs virtuels Linux ou Windows payable à l'heure ou au mois. Le client peut accéder et gérer ses serveurs au travers d'un client VNC (Virtual Network Computing).

ElasticHosts offre plusieurs images pré-installées (par exemple : Debian, Ubuntu et Windows Server 2008), ainsi que des CD d'installation pour CentOs, Debian, Knoppix, Red Hat Fedora, Ubuntu, FreeBSD, OpenSolaris et des versions d'évaluation de Windows. ElasticHosts offre aussi la possibilité de télécharger d'autres systèmes d'exploitation.

En fonction de leurs besoins, les clients d'ElasticHosts peuvent allouer dynamiquement plus ou moins de puissance CPU (2 à 20 GHz) et de RAM (1'024 Mb à 8'192 Mb). Ils peuvent faire de même pour la gestion de leur espace disque.

2. *SLA*

Le « Service Level Agreement »[1] offert par ElasticHosts est de 100% en garantissant 0% d'interruption de service. Si cela se produit il offre cent fois le crédit du client en cas de problème.

3. *Technique*

ElasticHosts a basé sa solution de « Cloud Computing » sur la solution de virtualisation Linux KVM (Kernel Based Virtual Machine). Cette solution open-source permet de faire fonctionner plusieurs machines virtuelles sur un serveur physique en se servant de fonctionnalités de base de Linux pour la gestion du scheduler et de la gestion de la mémoire.

[1] http://www.elastichosts.com/blog/2009/08/04/elastichosts-launches-ultra-flexible-scalable-cloud-hosting/

4. *Prix*

ElasticHosts permet de payer soit à l'heure, soit au mois l'utilisation de ses ressources. Les prix commencent à 0.04 GBP / heure pour une machine de 2GHz avec 1Gb de RAM et 1Gb de stockage et le prix monte jusqu'à 0.523 GBP / heure pour une machine de 20GHz avec 8Gb de RAM et 1.8Tb de stockage. ElasticHosts recommande le prépaiement mensualisé pour garantir les meilleurs prix[2].

5. *Avantage / inconvénient*

Avantages :

+ Possibilité de créer son image, répondant à ses besoins et selon ses standards ou utilisation d'images existantes.

+ Basé en Europe.

Inconvénients :

- Puissance limitée.

- Les 2 data centres sont basés géographiquement au même endroit (autour de Londres).

- Petite startup.

6. *En résumé*

ElasticHosts est une solution encore jeune qui met en œuvre certains concepts du « Cloud Computing ». L'offre d'ElasticHosts doit encore être améliorée afin d'offrir de réelles fonctionnalités d'élasticité et de scalabilté (sans redémarrage lors de changement de settings).

[2] http://www.elastichosts.com/products/pricing

Microsoft Windows Azure Platform

http://www.microsoft.com/windowsazure/

1. *Présentation de la solution*

La plate-forme « Windows Azure Platform » est la solution de « Cloud Computing » de Microsoft. Cette plate-forme offre trois types de services :

- « Windows Azure » qui permet de faire fonctionner sur cette plate-forme aussi bien des solutions de traitement de données, que des applications Web. Cette plate-forme permet aussi de faire du stockage de données. Ce service offre la possibilité aux développeurs d'utiliser les langages du framework .NET, mais aussi d'autres langages comme PHP et Java. Afin de garantir un service optimum « Windows Azure » met en œuvre des mécanismes pour faciliter la scalabilité des infrastructures techniques.

- « SQL Azure » est le moteur de base de données relationnel de Microsoft porté sur le « Cloud ».

- « Windows Azure platform AppFabric » permet d'interconnecter des services hébergés sur le « Cloud » avec des applications existantes.

Tous ces services de « Cloud Computing » sont hébergés dans les data centres de Microsoft.

2. *SLA*

Microsoft garantit une disponibilité de 99.9% pour ses services[1] « Windows Azure », « SQL Azure » et « Windows Azure platform AppFabric ». Ce taux passe à 99.95% si l'utilisateur déploie son service dans au moins 2 zones géographiques distinctes.

[1] http://www.microsoft.com/windowsazure/sla/

3. Technique

Dans cette section, nous focaliserons cette étude succincte sur le service « Windows Azure ».

« Windows Azure » permet de facilement déployer des applications packagées dans un fichier « cspkg » sur le « Cloud » de Microsoft. Ce package est accompagné d'un fichier de configuration « cscfg ». Ce dernier permet de définir le type de VM utilisé (de small processeur 1 CPU 1,6 GHz 1,7 Gb de mémoire et 250 Gb de disque dur à extra-large avec 8 CPU, 15 Gb de mémoire et 2 Tb de disque dur) et le nombre d'instances (VM) à déployer pour le projet. Ce nombre d'instance peut ensuite être modifié en cours de production afin de répondre aux besoins. Microsoft recommande l'utilisation d'au moins 2 instances dans 2 zones géographiques distinctes afin de garantir un meilleur niveau de service.

4. Prix

Microsoft facture[2] tout. Le CPU coûte 0.12 USD par heure, le disque dur 0.15 USD par Go par mois et les 10'000 accès disque coûte 0.01 USD. L'outil d'estimation des prix prévoit un budget 1'959.49 USD par mois pour un site web d'information qui fonctionnera sur 5 instances. Ce coût baisse à 1'696.69 USD si nous nous limitons à 2 instances serveurs. Le calculateur prévoit aussi un budget de près de 55'000 USD pour le développement et le déploiement de l'application la première année.

[2] http://www.microsoft.com/windowsazure/tco/

5. *Avantage / inconvénient*

Avantages :

+ Supporte de nombreux langages.

+ Offre plusieurs zones d'hébergement.

+ Offre une base de données relationnelle.

+ Permet d'interconnecter des services de type « Cloud » avec des applications existantes.

Inconvénients :

- Scalabilité manuelle ou nécessite un programme tiers.

- Pas de session.

6. *En résumé*

Microsoft Windows Azure Platform est une plate-forme qui est en constante amélioration depuis son lancement fin 2008 en version « Community Technology Preview ». Microsoft doit encore travailler sur l'automatisation du mécanisme de scalabilté.

Google App Engine

http://appengine.google.com/

1. *Présentation de la solution*

App Engine est la plate-forme de « Cloud Computing » de Google. Cette plate-forme offre la possibilité d'héberger des applications Web, développées soit en Java, soit en Python, directement sur les infrastructures de Google. Google App Engine bénéficie de la grande expérience de Google en matière d'hébergement Web. Google avec sa solution de « Cloud Computing » App Engine garantit que les environnements d'exécution sont construits pour s'assurer que l'application s'exécute rapidement, de manière sécurisée, et sans ingérence liée à l'exécution des autres applications sur le système, même en cas de fort trafic.

2. *SLA*

Aucun « Service Level Agrement » n'est offert par Google, selon le point 11.2 des conditions d'utilisations de Google App Engine[1].

[1] http://code.google.com/appengine/terms.html, consulté le 17/01/2010

3. Technique

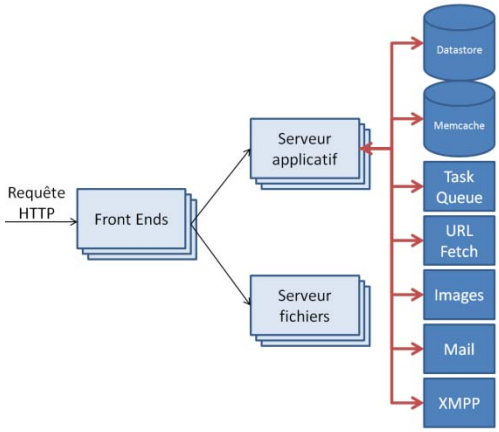

Figure : Architecture serveur et système de Google App Engine.

Pour sa plate-forme App Engine, Google a mis en place une architecture serveur et système qui lui permet de garantir un haut niveau de qualité. Google a clairement séparé la partie statique de la partie dynamique des applications hébergées chez lui. La partie applicative accède facilement à deux solutions de persistance : « Memcache » pour les données qui ont durée de vie courte et « Datastore » qui est le moteur de base de données de la solution App Engine. La plate-forme de Google offre d'autres services qui permettent entre autre d'accéder à d'autres ressources sur l'inter ou d'envoyer des mails par exemple.

Pour répondre à la montée en charge, Google multiplie les instances de serveurs applicatifs.

Sur sa plate-forme App Engine, Google a défini quelques contraintes techniques (par exemple : en Java impossibilité d'ouvrir un socket pour écrire dans un fichier, en Python les extensions en C ne sont pas autorisées) et opérationnelles (par exemple : le temps d'exécution d'une application est de 30 secondes maximum, une requête ne peut retourner que 1'000 enregistrements au maximum).

4. *Prix*

Gratuit pour les applications nécessitant moins de 500Mo de stockage et ayant moins de 5 millions de pages vues par mois.

Google facture[2] ensuite 0.10 USD l'heure de CPU entamée, 0.10 USD le giga de données entrantes et 0.12 USD le giga de données sortantes. Le stockage de données est facturé 0.15 USD le giga par mois. L'envoi d'email est facturé 0.0001 USD par destinataire.

5. *Avantage / inconvénient*

Avantages :

+ Plate-forme simple à prendre en main.

+ Peut servir 400 à 500 requêtes par secondes.

+ Mécanismes de scalabilité et d'élasticité transparent pour le développeur

Inconvénients :

- Choix des langages.

- Base de données propriétaire.

- SLA

6. *En résumé*

Google App Engine est une solution qui a été éprouvée depuis son lancement en 2008. Cette plate-forme est simple à prendre en main et offre malgré certaines contraintes techniques de grandes possibilités de développement.

Lors de test de la solution gratuite, nous avons constaté une latence d'environ 1 seconde lors de chaque requête HTTP à la partie dynamique. Nous avons aussi constaté que Google n'aime pas les « full table scan ».

[2] http://code.google.com/appengine/docs/billing.html#Billable_Quota_Unit_Cost

Appistry CloudIQ

http://www.appistry.com/

1. *Présentation de la solution*

Appistry CloudIQ est une plate-forme (de type middleware) qui permet de facilement déployer une application sur un « Cloud ». La solution proposée par Appistry se décompose en deux packages : « Appistry CloudIQ Manager » qui est la console qui permet de gérer tous les nœuds du cluster ainsi que de déployer les applications. « Appistry CloudIQ Engine » est le middleware qui héberge les applications. C'est aussi cette couche qui permet de gérer la scalabilté et qui garantit la fiabilité de la plate-forme.

Appistry CloudIQ peut aussi bien être déployé en interne sur des machines Linux ou Windows, que sur des solutions de « Cloud Computing » externes comme Amazon EC2 ou GoGrid.

2. *SLA*

Non applicable, car Appistry CloudIQ est une solution de type middleware et non un service.

3. *Technique*

« Appistry CloudIQ Engine » est un programme à installer sur chaque machine du cluster.

« Appistry CloudIQ Manager » est un service Web à déployer sur une des machines du cluster ou sur une machine tierce. Au travers de ce service Web l'utilisateur peut déployer ses packages spécialement préparés pour la plate-forme.

« Appistry CloudIQ Engine » supporte le Java, le .Net, le C, le C++. Ce middleware implémente certaines fonctionnalités pour faciliter la gestion des queues et pour permettre l'élasticité des applications. « Appistry CloudIQ Engine » offre aussi la possibilité de déployer des serveurs Web (Apache, Tomcat et IIS), des solutions de base de données (par exemple : MySql) ou des applications existantes, sans offrir de solution de répartition de charge.

4. *Prix*

Appistry offre une licence d'évaluation de sa solution pour 5 machines et 10 cores, mais communique peu sur le prix de ses licences. Selon Darryl K. Taft d'eWeek Europe[1], Appistry commercialise sa solution Manager + Engine pour 1'599 US$ / an.

5. *Avantage / inconvénient*

Avantages :

+ Facile à mettre en œuvre sur des machines en interne.

+ Multi plate-forme.

+ Supporte les grands langages de programmation du moment.

Inconvénients :

- N'offre pas une solution de répartition de charge lors de déploiement de serveur Web ou d'application existante.

6. *En résumé*

Appistry CloudIQ est une solution de middleware très simple à prendre en main et qui offre un grand potentiel. Cette solution est principalement dédiée pour les applications spécialisées dans le traitement de données et pour la mise en place de solutions de type SOA. Elle offre toutefois une solution pour le déploiement de serveurs Web, mais n'offre pas de solution de répartition de charge.

[1] http://www.eweekeurope.co.uk/news/appistry-tackles-multiple-cloud-apps-364, publié le 13/03/2009, consulté le 07/01/2010

AppScale

http://appscale.cs.ucsb.edu/

1. *Présentation de la solution*

AppScale est une implémentation open source de Google App Engine, développée par RACELab de l'université de Santa Barbara aux États-Unis. Cette plate-forme permet d'exécuter, sur son propre cluster, des applications développées pour Google App Engine.

Les buts d'AppScale sont premièrement de développer une plate-forme de type « PaaS » qui permette de déployer et de tester une application, avant son déploiement sur la solution propriétaire de Google. Le second but d'AppScale est d'étudier les interactions entre un « PaaS » et les couches inferieures d'une solution de « Cloud Computing »

2. *SLA*

Non applicable.

3. *Technique*

Cette plate-forme tourne sur des solutions de type « IaaS » mettant en œuvre les solutions de virtualisation comme Xen, KVM, Eucalyptus ou Amazon EC2. AppScale est une solution de type « PaaS » permettant d'héberger des applications développées en Java ou en Python. Pour stocker les données, AppScale supporte un grand nombre de bases de données : Hbase, Hypertable, MySQL Cluster, Cassandra, Voldemort, MongoDB, MemcacheDB.

4. *Prix*

Non applicable.

5. *Avantage / inconvénient*

Avantage :

+ Open source.

+ Plus souple que Google App Engine

Inconvénient :

- Pas connu

6. *En résumé*

Cette plate-forme est pour le moment développée à des fins de recherche scientifique et pour tester des applications destinées à fonctionner sur Google App Engine. En ouvrant son code et en permettant déjà l'utilisation de systèmes ouverts et connus pour la gestion de la persistance, AppScale promet de devenir une solution d'avenir.

VMware vSphere4

http://www.vmware.com/products/vsphere/

1. *Présentation de la solution*

VMware est un des leaders mondiaux en matière de virtualisation et il a lancé en avril 2009 vSphere4 - la technologie à la base de sa solution « vCloud ».

En résumant et en simplifiant très fortement le fonctionnement de vSphere4, ce produit permet de virtualiser un data centre en entier. Cette solution permet de rendre l'exécution d'une machine virtuelle totalement indépendante du hardware et par conséquent de réduire le risque de pannes.

Dans son package VMware offre plusieurs services :

- « VMware VMotion » et « VMware Storage VMotion » permettent de déplacer une machine virtuelle d'une machine physique à une autre sans interruption de service.
- « VMware High Availability » permet de redémarrer une machine virtuelle dans la minute en cas de panne de hardware ou de problème avec le système d'exploitation.
- « VMware Fault Tolerance » est le service qui permet de garantir une disponibilité continue de la machine virtuelle même en cas de panne de la machine physique.
- « VMware DRS » est le service qui gère la scalabilté de la machine virtuelle et qui permet d'ajouter des ressources à une machine virtuelle sans interruption de service.

VMware offre aussi des services pour garantir la sécurité de tout le système et assurer les backups.

2. *SLA*

Non applicable, car VMware vSphere4 est une solution de type middleware et non un service. En fonction du contrat, VMware peut offrir plus ou moins de support pour la mise en place de sa plate-forme.

3. Technique

Installer VMware vSphere4 est assez simple, tout comme déployer des machines virtuelles. Mettre en place une solution, tolérante aux pannes et offrant un maximum des services décrits ci-dessus, nécessite toutefois une infrastructure technique adaptée.

4. Prix

Le coût des licences annuelles[1] pour mettre en place la solution « vSphere » varie entre 3'500 USD et 5'000 USD par processeur de 12 cores maximum. Étant donné que pour garantir un service continu, il faut au minimum deux serveurs et un superviseur, le coût minimum des licences est de 10'500 USD par an.

5. Avantage / inconvénient

Avantages :

+ Solution robuste.

+ Solution scalable.

+ Solution fiable.

Inconvénients :

-Le prix des licences.

6. En résumé

Personnellement, j'ai pu tester l'hébergement de plusieurs serveurs virtuels sur une telle solution et je n'ai rencontré aucune difficulté. Le prix reste une barrière pour des petits projets.

[1] http://www.vmware.com/products/vsphere/buy/overview.html, consulté le 7 janvier 2010

Xen Cloud Platform

http://www.xen.org/products/cloudxen.html

1. *Présentation de la solution*

Xen Cloud Platform (XPC) est une future plate-forme de « Cloud Computing », développée par la communauté Xen et distribuée sous licence GPL. En novembre 2009 la version 0.1 a été lancée et elle devrait servir comme base de développement pour la première version officielle de la plate-forme XPC 1.0.

2. *SLA*

Non applicable, car Xen est une solution de type middleware et non un service.

3. *Technique*

Cette plate-forme se base sur la nouvelle version de l'hypervisor de Xen.

4. *Prix*

Non applicable.

5. *Avantage / inconvénient*

Avantage :

+ Open source.

+ Supporté par Citrix.

Inconvénient :

- Pas encore en service

6. *En résumé*

Cette plate-forme est encore en train d'être développée, le fait qu'elle soit open source est un atout indéniable.

Test de charge :

Résultat du test de charge pour 1 utilisateur faisant en moyenne 7 requêtes par seconde. Ce test a été effectué avec l'outil Pylot sur une base de données MySQL utilisant le système MyISAM. Cette base est chargée avec 100'000 enregistrements dans la table page, 1'000 mots-clés et environ 400'000 jointures entre ces deux tables.

Temps de réponse moyen : 0.009 seconde

99% des requêtes répondent en moins de : 0.011 seconde

Page	min	max	avg	hit	>1sec	pc >1sec	3sec	pc >3sec
Get	0.0092	0.025	0.0102	414	0	0.00%	0	0.00%
List	0.0072	0.0326	0.0081	1'917	0	0.00%	0	0.00%
Add	0.0081	0.0274	0.0100	639	0	0.00%	0	0.00%
Get hash	0.008	0.0218	0.0092	639	0	0.00%	0	0.00%
Up	0.0085	0.0115	0.0093	448	0	0.00%	0	0.00%
Down	0.0085	0.0102	0.0093	191	0	0.00%	0	0.00%
TagCloud	0.0104	0.1062	0.0645	13	0	0.00%	0	0.00%
Global	***0.0072***	***0.1062***	***0.0091***	***4'261***	***0***	***0.00%***	***0***	***0.00%***

Tableau : Analyse détaillée des requêtes HTTP par script

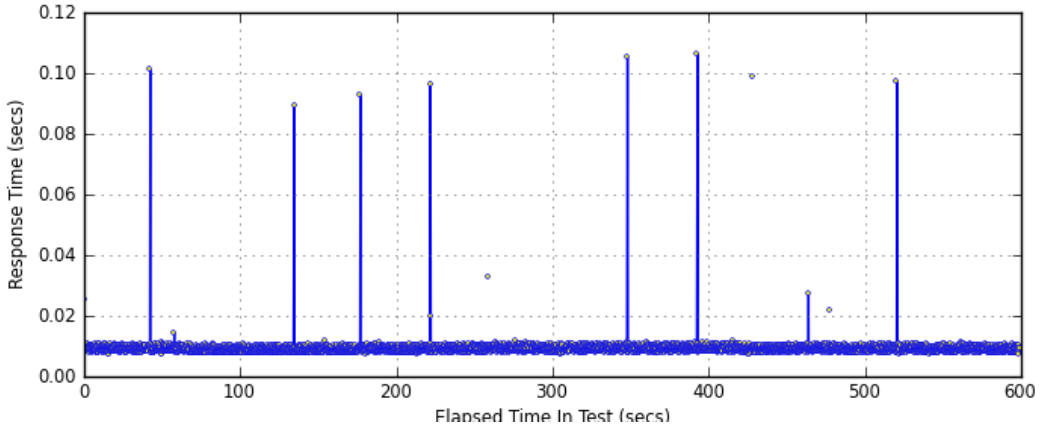

Figure : Graphique des temps de réponse

Test de charge :

Résultat du test de charge pour 150 utilisateurs faisant ensemble en moyenne 130 requêtes par seconde. Ce test a été effectué avec l'outil Pylot sur une base de données MySQL utilisant le système MyISAM. Cette base est chargée avec 100'000 enregistrements dans la table page, 1'000 mots-clés et environ 400'000 jointures entre ces deux tables.

Temps de réponse moyen : 0.029 seconde

99% des requêtes répondent en moins de : 0.027 seconde

Page	min	max	avg	hit	>1sec	pc >1sec	3sec	pc >3sec
Get	0.0083	3.1999	0.0304	234'796	1'278	0.54%	522	0.22%
List	0.0065	9.0679	0.0280	1'062'102	5'780	0.54%	2'488	0.23%
Add	0.0073	13.5512	0.0305	354'034	1'902	0.54%	850	0.24%
Get hash	0.0075	3.3783	0.0290	354'032	1'892	0.53%	818	0.23%
Up	0.0077	10.9929	0.0303	236'366	1'326	0.56%	578	0.25%
Down	0.0078	3.1712	0.0279	117'666	574	0.49%	260	0.22%
TagCloud	0.0091	3.4465	0.0809	7'824	42	0.54%	28	0.36%
Global	***0.0065***	***13.5512***	***0.0292***	***2'374'648***	***12'842***	***0.54%***	***5'556***	***0.23%***

Tableau : Analyse détaillée des requêtes HTTP par script

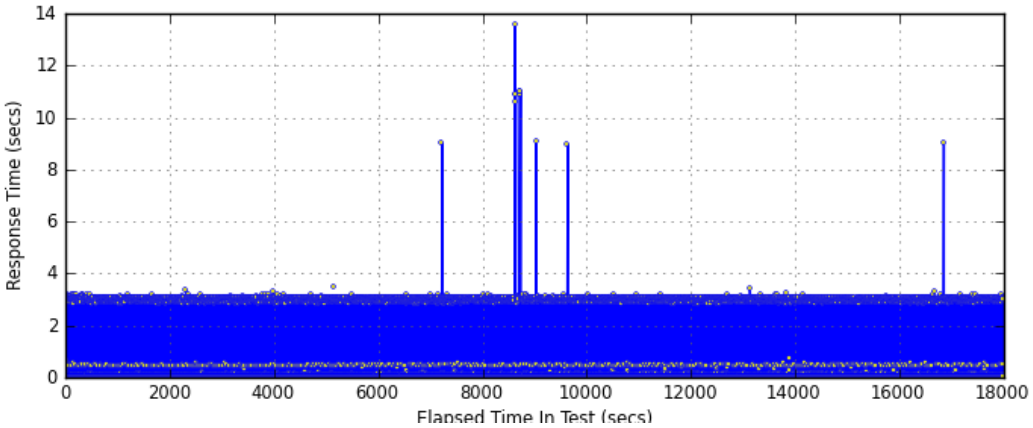

Figure : Graphique des temps de réponse

Test de charge :

Résultat du test de charge pour 150 utilisateurs faisant ensemble en moyenne 130 requêtes par seconde. Ce test a été effectué avec l'outil Pylot sur une base de données MySQL utilisant le système InnoDB. Cette base est chargée avec 100'000 enregistrements dans la table page, 1'000 mots-clés et environ 400'000 jointures entre ces deux tables.

Temps de réponse moyen : 0.029 seconde

99% des requêtes répondent en moins de : 0.029 seconde

Page	min	max	avg	hit	>1sec	pc >1sec	3sec	pc >3sec
Get	0.0082	3.1918	0.0310	234'946	1'264	0.54%	524	0.22%
List	0.0066	9.0412	0.0275	1'062'708	5'322	0.50%	2'262	0.21%
Add	0.0074	9.0808	0.0299	354'236	1'686	0.48%	784	0.22%
Get hash	0.0077	9.0234	0.0291	354'236	1'820	0.51%	786	0.22%
Up	0.0076	3.3445	0.0293	236'530	1'196	0.51%	552	0.23%
Down	0.0078	3.1786	0.0285	117'704	566	0.48%	238	0.20%
TagCloud	0.0092	3.3003	0.1570	7'828	44	0.56%	30	0.38%
Global	***0.0066***	***9.0808***	***0.0291***	***2'376'020***	***11'964***	***0.50%***	***5'196***	***0.22%***

Tableau : Analyse détaillée des requêtes HTTP par script

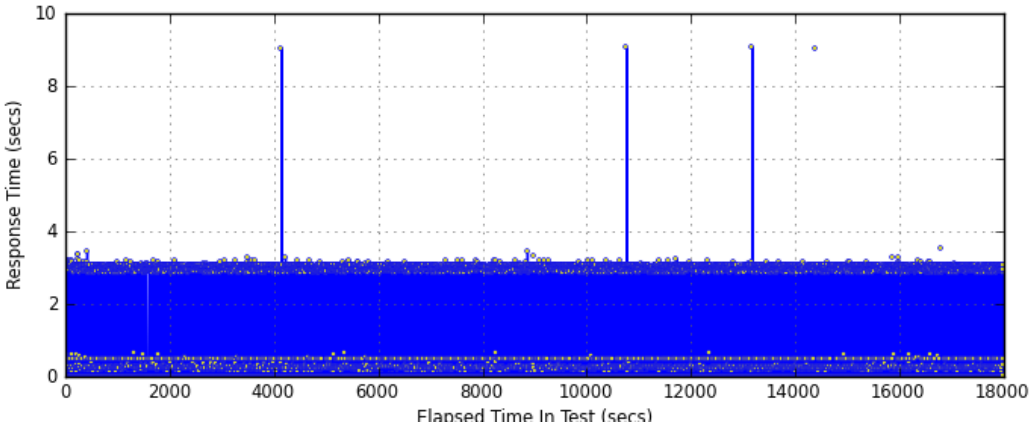

Figure : Graphique des temps de réponse

Résumé

Le « Cloud Computing » devrait d'ici deux à cinq ans devenir une des briques de base des plates-formes informatiques de prochaine génération. Le Laboratoire des Technologies de l'Information (LTI) de la Haute École de Gestion (HEG) de Genève, dont les missions sont de faire de la recherche et de valoriser cette dernière au travers de mandats, a voulu comprendre le concept de « Cloud Computing ».

La première partie de ce mémoire est consacrée à la définition du « Cloud Computing » et présente les cinq concepts fondamentaux de cette nouvelle technologie. Le deuxième chapitre présente les différents types d'applications « Cloud » en les classant en trois catégories (« infrastructure », « platform » et « software »). Les troisième et quatrième parties sont consacrées à la mise en pratique du concept. Le troisième chapitre est consacré à la définition des besoins, des objectifs à atteindre tout en respectant certaines contraintes. Dans cette troisième partie est choisie la plate-forme de « Cloud Computing » pour héberger un projet de « folksonomie ». Ce dernier, qui s'appuie sur les concepts du Web 2.0, consiste à demander à l'internaute de classifier les pages d'un site Web en utilisant ses propres mots-clés. Ce projet facilite la recherche d'information pour les autres internautes. Il leurs permet aussi de donner plus d'importance à certains mots-clés en les évaluant. La mise en œuvre de ce projet est décrite dans le quatrième chapitre.

Cette application Web 2.0 permet de constater les avantages, les inconvénients et les contraintes liées au développement sur un « Cloud » et de vérifier certains des concepts fondamentaux.

Mots-clés : Cloud Computing, pyramide du Cloud, folksonomie, définition, concept, implémentation

Summary

"Cloud Computing" is set to become one of the pillars of next-generation computing. The Laboratory of Information Technologies (LTI) at the University of Applied Sciences Western Switzerland (HEG) in Geneva, whose missions are to conduct research and development, aims to understand "Cloud Computing" concepts and conduct a pilot project using the technology.

This engineering thesis is divided into four chapters. The first chapter defines "Cloud Computing" and presents the technology's five fundamentals concepts. The second chapter presents the different types of "Cloud" solutions and classifies them in three categories ("infrastructure", "platform" and "software"). The third and fourth chapters cover practical aspects. The third chapter defines the requirements and objectives that will allow the LTI team to host a "folksonomy" project on a "Cloud". The project, based on Web 2.0 concepts, consists in asking Internet users to classify the pages of a Web site by using their own keywords. This project facilitates information retrieval for other Internet users. The application, also allows users to rank keywords thus giving more importance to better-ranked words. The implementation of the "folksonomy" project is described in the fourth chapter.

This Web 2.0 application will help determine the advantages, disadvantages and limitations specific to the development of applications on a "Cloud" and will validate the fundamental concepts of "Cloud Computing" defined in the first chapter.

Keywords: Cloud Computing, cloud pyramid, folksonomy, definition, concepts, application