Comment faire face aux problématiques de scalabilité de l'infrastructure dans une entreprise en forte croissance

Thibaut Lapierre - Digimind

Juin 2013 - Octobre 2014

Tuteur d'alternance: Stéphane Gras

Avant toute chose, je souhaite remercier :

- *M. Stéphane Gras, mon maître de stage, pour avoir su m'apporter de nombreux conseils, tant dans mon travail en lui-même que dans ma façon de le mener à bien.*
- *M. Eric et Cécile Lapierre, mes parents, pour leurs précieux conseils avisés tout au long de ce mémoire et de ces 22 dernières années.*
- *Melle Isabelle Plan, ma partenaire dans la vie, pour ses longues heures de relecture, son soutien quotidien et pour avoir su canaliser mon stress et le transformer en efficacité et motivation.*
- *Mme Hanane Blanco, Directrice des ressources humaines de Digimind, pour s'être souvenue de moi et m'avoir contacté lors de sa recherche de candidats pour la réalisation de cette mission*
- *M. Pierre-Arthur Mathieu, mon colocataire et amis depuis de nombreuses années, pour m'avoir aidé à rester concentré et pour m'avoir épaulé dans un grand nombre de décisions importantes.*
- *M. Sylvain Bauza, Ingénieur Cloud et virtualisation chez RedHat, pour m'avoir transmis sa passion pour OpenStack.*
- *L'école Supinfo, pour m'avoir donné la possibilité d'effectuer les projets personnels et professionnels m'ayant amené à ce travail.*

Merci.

TABLE DES MATIERES

INTRODUCTION

Dans la mesure où les entreprises se développent en même temps que leur nombre de clients, tant sur le plan du nombre de collaborateurs employés qu'en terme de ressources mises en œuvre, il est normal que leur parc informatique et son infrastructure évoluent en conséquence.

Le concept défendu ici est l'évolution d'une gestion des serveurs de manière individuelle, largement employée dans la plupart des infrastructures logicielles simples et peu utilisatrices de ressources matérielles, vers une gestion groupée et impersonnelle afin que la taille ne soit plus qu'un problème de puissance matérielle.
Cette idée porte un nom: "Pets vs. Cattle", qui illustre le fait de passer d'une gestion des serveurs semblables à des animaux de compagnie "Pets" (On leur donne une fonction spécifique, un nom et une adresse qui leur est propre) à gestion plus proche de l'animal destiné à l'élevage industriel "Cattle" où chaque serveur est remplaçable à la volée et est considéré comme une simple ressource matérielle.
Le principal avantage de ce mode d'administration est de pouvoir s'affranchir des contraintes techniques dues à la multiplication du nombre de serveurs et arriver ainsi à fournir les mêmes efforts pour l'administration d'un petit nombre de serveurs, que pour l'administration d'un très grand nombre de machines.
L'objectif est de pouvoir supporter d'importantes montées en charge et un nombre croissant de clients tout en maintenant la même qualité de service, et améliorer la disponibilité et la résilience aux pannes de l'application web développée. C'est un objectif très large, et qui nécessite bien d'autres mesures que celles décrites dans ce mémoire, mais toutes les démarches que j'ai entrepris cette dernière année vont dans ce sens.

Cette thématique correspond au travail effectué dans la société depuis mon arrivée. Il ne couvre bien entendu pas la totalité de ce sujet très vaste, mais tous les travaux entrepris vont dans cette direction, dans le but d'arriver à gérer le plus facilement possible un important nombre de machines destinées à supporter de fortes montées en charge.

Mon travail au sein de Digimind prend place après quatre années d'études à Supinfo dans les villes de Lyon, Montréal et Marseille au cours desquelles j'ai pu prendre part à des projets dans des contextes professionnels variés. Notamment une année en tant qu'administrateur systèmes et réseaux en alternance, ainsi qu'un an en tant qu'enseignant des technologies propres à l'utilisation de Linux en Data Center et du fonctionnement bas niveau des ordinateurs à Supinfo.

Mais même si ces deux précédentes expériences ont joué un rôle important dans la réalisation de mon travail à Digimind, ce sont plus précisément deux autres projets effectués à Montréal au Canada qui ont fait remonter la question d'échelle au centre de mes réflexions.

J'ai été amené dans le cadre d'un projet associatif de concours de Hacking à mettre en place une infrastructure d'entreprise-type comportant un nombre prédéterminé de failles. Une fois les différents serveurs et équipements mis en place, la nécessité de dupliquer ces environnements en fonction du nombre d'équipes participantes est devenu le centre de notre réflexion et j'ai ainsi pour la première fois été confronté à ce type de problématique. Je soulignerai cependant que les connaissances acquises et les expérimentations réalisées lors de ce projet m'auront amené à apporter des solutions radicalement différentes dans le cadre de mon travail à Digimind.

Toujours au Québec, mais dans un contexte diffèrent, j'ai réalisé dans le cadre d'un travail de chronométreur de courses cyclistes un programme de statistiques relativement simple afin d'analyser et traiter les temps effectués par les différents coureurs. Les courses étant habituellement constituées de vingt à trente coureurs au maximum, le fait que le programme de statistique réalisé nécessite un grand nombre d'interactions humaines n'était pas un problème, et ce, jusqu'à ce que la fédération de cyclisme me sollicite pour chronométrer le « Grand Défit Pierre Lavoie » (équivalent du Tour de France au Québec) constitué de près de mille-cinq-cents coureurs, et c'est devant cette toute autre dimension et ce changement d'échelle conséquent que j'ai dû adapter l'architecture et réécrire le fonctionnement de mon programme.

Ces deux expériences m'ont donc amené à approcher la problématique de la scalabilité des infrastructures d'entreprise sous deux angles bien distincts, tout d'abord d'un point de vue système et réseau lors du concours de Hacking ou la quantité de ressources matérielles attribuées et sa gestion étaient au centre des questions posées, mais aussi d'un point de vue logiciel et fonctionnel lors de mon travail pour la Fédération de cyclisme.

Ce sont ces deux approches distinctes qui, comme nous allons le voir, permettront de constituer la réponse apportée à la problématique posée, car les infrastructures technologiques d'une société sont bien entendu constituées de réseaux et de serveurs, et ce sera le point de vue de tout administrateur système et réseaux, mais aussi de logiciels, et ce sera le point de vue de tout développeur. Et c'est afin de traiter les deux aspects de cette problématique que j'appuierai les réflexions de cette dissertation avec deux projets entrepris au sein de l'entreprise constituants chacun une part de la réponse apportée à la problématique d'échelle et de scalabilité des infrastructures dans une entreprise en forte croissance.

Ce mémoire de fin d'étude se présente en plusieurs parties majeures. Après une introduction générale de l'entreprise et de certains de ses points clés, je présenterais une analyse du contexte dans lequel le projet à pris place, nous verrons dans ces deux parties que Digimind est une entreprise en forte croissance, et plus le nombre de ses clients augmente plus l'infrastructure mise en place pour délivrer l'application croît et se complexifie. Mais l'essor de l'infrastructure amène aussi la problématique de la scalabilité de celle-ci.
Afin de répondre à ce problème, et de simplifier la gestion du parc de serveurs et de leurs logiciels j'ai étudié les différentes méthodes usuelles dans le but d'apporter par la suite une solution originale et appropriée au contexte précédemment décris.

Ce mémoire sera donc la synthèse d'une réflexion autour du rapport d'échelle et de proportionnalité des infrastructures technologiques de l'entreprise.

PRESENTATION DE L'ENTREPRISE: DIGIMIND, UNE SOCIETE LEADER SUR LE MARCHE DE LA VEILLE STRATEGIQUE

Pourquoi réaliser mon stage de fin d'étude à Digimind?

Effectuer mon contrat de professionnalisation en alternance au cours de ma dernière année d'étude à Digimind aura présenté pour moi de nombreux aspects positifs. J'avais jusqu'à ce jour davantage travaillé dans de petites entreprises ou dans des services informatiques relativement réduits. Digimind représente pour moi la taille de société idéale, bénéficiant à la fois des avantages d'une Start-Up et de ceux d'une grande multinationale.

En effet, malgré le nombre grandissant de collaborateurs et les nombreux bureaux dans différents pays, les méthodes de management, l'ambiance et la façon de travailler restent plus proche d'une Start-Up que d'un grand groupe.

Tout ceci présente plusieurs avantages, le travail dans ce type d'entreprise offre en général aux employés plus de responsabilités et beaucoup plus de libertés que dans un gros groupe qui aura la plupart du temps des méthodes de management très verticales (plusieurs strates hiérarchiques nécessitant alors la validation et l'aval d'un supérieur pour chaque tâche effectuée) ainsi que des méthodes de travail entièrement procédurées (souvent de type ITIL).
Ce sont des processus de travail pouvant être très lourds pour l'ingénieur mais garantissant à la société et ses dirigeants des résultats beaucoup plus cadrés et proches de leurs souhaits initiaux (en théorie).

En revanche, le travail dans ces grosses sociétés offre d'autres avantages dont les sociétés de taille plus modeste ne jouissent pas: des ressources beaucoup plus vastes, des problématiques agissant sur une échelle complètement différente du travail dans une petite entreprise, et donc logiquement une infrastructure logicielle et matérielle proportionnelle à la taille de l'entreprise et à son nombre de clients.
C'est d'ailleurs ce point là que je développerai dans la problématique de mon mémoire.

Sachant pertinemment qu'un travail dans un grand groupe me correspondrait beaucoup moins, mais cherchant tout de même à travailler sur un nombre important de machines et des montées en charge importantes, le travail à Digimind s'est avéré être la meilleure des équations. J'ai ainsi pu bénéficier de beaucoup plus de libertés et d'autonomie de travail, ou encore avoir

un véritable impact sur les produits et les solutions utilisées dans la société, tout en répondant à des problématiques de montée en charge et de scalabilité bien plus larges que ce qui aurait été possible en intégrant une Start-Up.

Ce que propose l'entreprise

Le cœur de métier de Digimind est le développement d'applications web de veille stratégique et concurrentielle.
Digimind conçoit et développe une plateforme logicielle de veille concurrentielle et de e-réputation, qui permet aux entreprises de déployer et animer des projets de veille et de réputation digitale. Fondée en 1998, l'entreprise Digimind compte aujourd'hui une centaine d'employés répartis dans ses bureaux en Amérique du Nord, Europe, Asie et Afrique.

La mission de l'entreprise est donc d'aider ses clients à survivre sur leurs marchés respectifs, grâce à des outils qui leur permettront de franchir un seuil qualitatif important dans leur capacité à prévoir les marchés, cerner les besoins potentiels, identifier les innovations technologiques et anticiper les modifications de comportement des acteurs économiques, politiques et sociaux dans le but d'assurer leur compétitivité.

Afin de mieux comprendre les services proposés par les solutions Digimind, voici une définition de la veille stratégique qui facilite grandement la compréhension de la plateforme Digimind Intelligence :
David Coudol et Stéphane Gros, membres de l'URFIST de Toulouse donnent la définition suivante: *"La Veille Stratégique est un système d'aide à la décision qui observe et analyse l'environnement scientifique, technique, technologique et les impacts économiques présents et futurs pour en déduire les menaces et les opportunités de développement. Elle s'appuie essentiellement sur les informations ayant un caractère stratégique ou décisions importantes lui associant le terme de veille stratégique".* (D Coudol, S Gros. Veille Intelligence Economique)

La veille stratégique, parfois associée au concept plus institutionnel et politique d'"Intelligence Économique et Stratégique" (médiatisé par le rapport Carayon "Intelligence économique, compétitivité et cohésion sociale" commandité par le Premier ministre Jean-Pierre Raffarin en Janvier 2003) est une pratique qui s'est progressivement imposée au sein des entreprises évoluant en milieu fortement concurrentiel ou innovant. Elle a pour objectif de développer leurs

capacités anticipatives et réactives face aux évolutions rapides de leur environnement économique, et en particulier aux agissements de leurs concurrents.
Le but est de fournir aux clients toute l'actualité (passée et présente) concernant un sujet donné, puis de l'analyser, que ce soit de façon générale avec l'application "Digimind Intelligence" ou de façon plus spécifique dans les médias sociaux avec l'application "Digimind Social", ce qui représente un volume de collecte d'environ quarante millions de mentions par jours.

Le cœur de métier de l'entreprise se divise donc en trois fonctions simples:

- La collecte d'informations
- L'analyse des informations collectées
- La restitution des informations et des résultats de l'analyse au client

Avec plus de deux-cent clients principalement issus du "Global Fortune 500[1]", Digimind est le leader des logiciels de veille stratégique.
Mais l'entreprise tend à devenir plus accessible sur le plan tarifaire, dans le but de séduire des clients de taille plus modeste, et ainsi augmenter leur nombre. Une telle action permettrait de gagner des parts de marché laissées jusqu'ici à des entreprises concurrentes dans le secteur de la veille stratégique et concurrentielle.
Une autre grande avancée de Digimind en ce sens a été la sortie d'un nouveau logiciel plus simple et beaucoup moins onéreux, spécialisé dans les médias sociaux (Digimind Social).

Afin d'améliorer ses relations clients et de faciliter la recherche de nouveaux prospects, Digimind a choisi de créer des filiales à travers le monde entier. Aujourd'hui, Digimind compte six bureaux présents sur quatre continents, tous dirigés par et depuis la France. Il est également important de noter que la ligne directrice de l'entreprise est toujours tournée vers l'internationalisation puisqu'un nouveau bureau ouvrira ses portes à New York dans le courant de l'année 2015.

Nous pouvons apercevoir la répartition sur quatre continents des bureaux de Digimind sur la carte ci-dessous.

[1] Classement pas le magazine Fortune des 500 plus grosses entreprises en function de leur chiffre d'affaire
[2] Recherche pour l'analyse et l'interprétation automatisée d'extraits du Web. Analyse de sentiments,

Figure 1: Positionnement des bureaux de Digimind dans le monde

Les différents produits proposés par l'entreprise

L'entreprise développe plusieurs grands projets :

Le premier, et qui se trouve également être le produit historique "Digimind intelligence" (http://digimind.com/features-intelligence), est un logiciel général de veille stratégique permettant aux sociétés clientes de remonter, d'analyser et de rassembler sous forme de rapports une grande quantité d'informations fournies par différentes sources (Google, Forums, RSS, etc.)

(cf. Annexe Figure 6, Page 72)

Le second logiciel, sur lequel de gros efforts de développement ont été mis en oeuvre est Digimind Social (http://digimind.com/features-social). Créé récemment, il a été développé dans l'idée d'être plus simple d'utilisation que Digimind Intelligence, et est spécialisé dans la remontée d'informations provenant des médias sociaux. Il est davantage utilisé par les community managers et les équipes marketing des sociétés dans le but de contrôler leur réputation sur les médias sociaux, mais aussi celle des sociétés concurrentes et de certains évènements.

Il permet de scanner et d'analyser les réseaux sociaux afin de permettre aux entreprises clientes de contrôler et de mesurer leur image et réputation.

Plus épuré, ce produit a l'avantage d'être clé en main et plus simple d'utilisation.

On note également l'ajout d'une grande quantité de nouvelles fonctionnalités telles que le calcul d'un score d'influence pour chaque tweet, billet de blog, article, vidéo, etc; en fonction de critères comme le nombre de "followers", de mentions, de visiteurs ainsi que leur audience.
(cf. Annexe Figure 7, Page 72)

Un autre projet qui a été développé par l'équipe WMA (Web Mining Analytics) de Digimind est WhoGotFunded (http://www.whogotfunded.com/), qui après une veille finement paramétrée sur le logiciel Digimind Intelligence recense et classifie les différentes levées de fonds des sociétés auprès de fonds d'investissements.

Le dernier logiciel (qui est encore en cours de développement par l'équipe Recherche Et Développement de Digimind) se nomme "Digimind Express". Plus qu'un réel nouveau logiciel, cet outil reprend la simplicité et l'expérience utilisateur qui sont les points forts de Digimind Social ainsi que le vaste panel de fonctionnalités et de ressources du logiciel Digimind Intelligence afin de les combiner en un seul et même produit.

Faisant partie de l'équipe Exploitation de la société, j'ai eu l'occasion de travailler avec la totalité des produits proposés.

Histoire de l'entreprise

Digimind, (et il s'agira aussi du cœur de ma problématique), est une entreprise en plein changement.
L'entreprise effectue une transition progressive du stade de moyenne entreprise à celui de grande entreprise, chose que j'ai aussi pu constater durant l'année passée dans l'un de leurs services. Que ce soit de par la maturité de son produit historique ou encore via l'innovation des nouvelles technologies développées, sans oublier son nombre de clients croissant.
C'est donc une entreprise en pleine extension, et ce, comme je vais l'expliquer dans ce paragraphe, depuis sa création en 1998.

Afin de brièvement résumer l'historique de l'entreprise depuis sa création j'ai sélectionné plusieurs points et dates importants.

L’entreprise est créée en 1998 par un groupe de quatre étudiants Grenoblois dans le but de combler un manque sur le marché de l’époque, à savoir: Un logiciel de renseignement de pointe.

En 2002, Digimind Evolution est lancée: première solution de veille stratégique, elle permet aux entreprises d’obtenir des informations sur elles-mêmes et leurs concurrents par le biais d’Internet. C’est également l’année de l’ouverture du bureau de Paris pour les fonctions commerciales et marketing de l’entreprise.

L'expansion de la société à l’international débute quant à elle en 2006 avec l’ouverture d’un bureau à Rabat au Maroc, essentiellement destiné à la recherche et au développement, qui s’ajoute ainsi au bureau de Grenoble.

L'expansion de la société à l’étranger continue en 2008 avec l’ouverture d’un bureau commercial à Boston aux États-Unis. C’est aussi l’année du lancement de la septième version du logiciel Digimind Évolution qui apporte de nombreuses améliorations par rapport aux précédentes versions. Sans entrer dans des détails trop techniques, l’ergonomie de la plateforme a été entièrement repensée afin d'améliorer l'expérience utilisateur et la DCF (Digimind Content Factory) est créée. Il s’agit d’une très large base de données regroupant des sources sélectionnées au préalable et permettant ainsi aux clients de paramétrer plus rapidement leur plateforme de veille sans avoir à rechercher et sélectionner leurs sources d’information eux-mêmes.
Une autre version majeure de l’application “Digimind Evolution 10” est conçue en 2012.

En 2013, l’entreprise cherche à s'ouvrir à d’autres marchés afin de continuer sa progression et toucher ainsi un plus vaste panel de clients. C’est donc après plusieurs années de travail des équipes Recherche et Développement que Digimind pourra annoncer la sortie de Digimind Social et l’entrée de l’entreprise sur le marché de l’e-réputation.
J’ai pu assister au lancement de ce logiciel et voir les premiers contrats signés.

Plus récemment, en 2014, Digimind annonce l’ouverture d’une filiale en Angleterre à Reading, près de Londres, et également le lancement du développement de “Digimind Express” destiné à l'expérience utilisateur du nouveau logiciel Digimind Social ainsi que les fonctionnalités de la plate-forme historique de veille stratégique “Digimind Intelligence” à l’utilisateur final.

Le prochain point marquant de l'histoire de Digimind sera l'ouverture d'un nouveau bureau à New York en 2015. Ce bureau sera destiné à confirmer l'implémentation de Digimind en Amérique du Nord (qui avait débutée en 2008 avec l'ouverture du bureau commercial de Boston). L'objectif visé ici ne sera pas l'ouverture d'un second bureau commercial à quelques heures de Boston, mais la création d'un siège social pour la filiale Américaine de la société en vue d'ouvrir de nouveaux bureaux par la suite.

La division recherche et développement de Digimind

Le cœur de métier de l'entreprise est le développement et la maintenance du logiciel "Digimind Évolution" spécialisé dans la veille stratégique, la veille concurrentielle et l'analyse de données. La moitié des employés sont donc des Ingénieurs en Informatique, et pour la plus grande majorité d'entre eux spécialisés dans le développement.
Ces employés composent le pôle recherche et développement de la société, lui même scindé en deux bureaux: Rabat et Grenoble. Les autres bureaux, Paris, Londres, Boston, Singapour, plus petits, sont essentiellement constitués de commerciaux et de l'équipe marketing (pour les locaux de Paris).

Le pôle R&D de la société est lui même divisé en six équipes, quatre d'entre elles étant concentrées sur le développement des deux applications développées par Digimind (Deux équipes à Grenoble et deux équipes à Rabat travaillant respectivement sur les logiciels Digimind Évolution et Digimind Social).

Une autre équipe nommée WMA (Web Mining Analytics[2]), est davantage centrée sur la recherche et l'écriture d'algorithmes servant à l'analyse de données, l'extraction de concepts ou encore l'extraction des sentiments sur un texte donné. L'application de Digimind est aujourd'hui capable après avoir remonté une information de déterminer si le texte est davantage à consonance positive, neutre ou négative. Et ce, dans plus de soixante langues différentes.

[2] Recherche pour l'analyse et l'interprétation automatisée d'extraits du Web. Analyse de sentiments, extraction de mots clés, etc.

La dernière équipe, constituée de mon tuteur d'alternance (Stéphane Gras) et moi-même travaille sur l'exploitation de la plateforme et de l'infrastructure afin d'assurer la disponibilité de l'application ainsi que sa résilience face aux pannes et à la montée en charge.
Nous nous occupons donc de gérer les serveurs, le réseau ainsi que les bases de données de l'entreprise.

Il est important de souligner que les dirigeants de Digimind favorisent la recherche et développement au sein de la société: de nombreux projets ont vu le jour ces dernières années et la division Recherche et Développement représente près de la moitié des dépenses de la société.
L'objectif étant d'être toujours plus innovant et de rester compétitif techniquement.
En effet, depuis 1998 Digimind est l'acteur incontournable de la veille stratégique et a su sans cesse innover afin de ne pas se faire dépasser par les nouveaux acteurs du marché.

Ma place au sein de la société

Je suis arrivé au sein de l'équipe exploitation de Digimind en Juillet 2013 avec comme mission de remplacer et de pallier au départ de Sylvain Bauza, avec qui j'ai pu travailler pendant un mois afin qu'il me forme, m'explique son travail et les projets précédemment entrepris.
Sylvain a été le premier initiateur du projet OpenStack[3] chez Digimind. Il a donc effectué avant moi tout le travail de persuasion de la hiérarchie afin de les convaincre des avantages de cette solution.

Mes premiers objectifs en arrivant ont été l'industrialisation et la mise en place d'une plateforme OpenStack afin de virtualiser et d'automatiser la gestion des différents environnements de développement, de test et de pré-production de la société.
Ce qui a résulté en 5 environnements au fonctionnement le plus identique possible de l'environnement de production.

Ce projet a présenté pour moi un grand nombre de points positifs:
Passionné par OpenStack (j'ai construit mon projet de fin de Master1 sur cette solution) j'ai pu bénéficier d'une réelle première expérience en la matière.

[3] Les details de ce projets seront expliqués dans les chapitres suivants.

D’autre part, la mise en place des environnements de développement, tests et QA (Quality Assurance) au sein de la société est une responsabilité importante. En effet, sans ces environnements, les développeurs ne peuvent pas travailler et les projets entrepris n’avancent pas. Je suis sur ce point particulièrement reconnaissant envers Digimind de m’avoir confié un projet présentant autant de responsabilités et correspondant à un réel poste au sein de la société.

J’ai également, lors de mon travail à Digimind, eu la chance d'être relativement libre dans les choix concernant mes travaux et de pouvoir travailler de façon autonome. Cela m’a permis d'être régulièrement confronté à des choix aillant un fort impact sur le fonctionnement de l'entreprise et son produit final.

Organisation managériale et méthode de travail

Dans l'ingénierie informatique, deux grandes méthodes de management sont très largement utilisées :
La méthode ITIL, qui correspond à une hiérarchie verticale, est habituellement utilisée dans les grands groupes et les administrations car elle a un côté rassurant pour les dirigeants de par son aspect plus strict et ses démarches entièrement procédurales. Malheureusement elle peut s'avérer souvent lourde à implémenter et à vivre au quotidien car nécessitant un report constant à la hiérarchie et des procédures strictement définies pouvant souvent amener à un manque de liberté.

La seconde méthode, le plus souvent utilisée en Start-up, mais arrivant progressivement dans les grands groupes, est la méthode Agile, et plus précisément la méthode Scrum pour Digimind. C’est une méthode de management correspondant à une hiérarchie horizontale et donc fondamentalement opposée à ITIL qui offre davantage de souplesse et d’agilité pour le client mais aussi pour les développeurs et autres ingénieurs.
L’application de la méthode Agile Scrum diffère toujours assez largement suivant les sociétés. C’est en s'appuyant sur l’un des recueil de pratiques Scrum que les Scrum Master de chaque entreprise définissent les parties à appliquer au sein de la société. En effet, Scrum est une méthode Agile correspondant à un framework de bonnes méthodes de management sur lequel s’appuyer pour définir l’organisation managériale au sein de l'équipe.

Ces deux méthodes (ITIL et Agile) ont été étudiées au cours de ma dernière année de Master à SUPINFO. C'est précisément sur la méthode Agile, mise en place et appliquée à Digimind que j'ai pu approfondir mes acquis et surtout prendre part à une réelle mise en pratique des connaissances théoriques étudiées en cours. Ce sont précisément la logique et l'aspect humain de cette méthode de management qui m'avaient séduit lors des cours s'y référant.

Bien entendu, il existe une différence significative entre les cas théoriques étudiés en cours et leur mise en application concrète, mais c'est aussi ce que préconise la méthode Agile. Il est important d'y puiser les ressources managériales indispensables au bon fonctionnement de l'entreprise, sans toutefois s'embarrasser des aspects moins pertinents en appliquant chaque recommandation à la lettre.

Les comptes rendus quotidiens ou "Stand Up"

Chez Digimind, et comme la méthode Scrum le préconise, un compte rendu quotidien est effectué chaque matin. Le principe est simple, chacune des équipes se réunit, debout, pour une durée de moins de 10 minutes et chaque participant décrit les tâches effectuées le jour précédent et celles qu'il compte réaliser aujourd'hui. C'est un moyen pour chacun de connaître l'avancement de ses collaborateurs et de pouvoir par la suite les aider et proposer des modifications, corrections ou démarches à suivre.

Il est important de noter que les participants d'un Stand Up s'organisent pour se retrouver en fonction du sujet de leur travail et non pas toujours en fonction de leur équipe, par exemple, lors de mon travail sur la mise en place des environnements de développement je devais travailler de près avec les développeurs afin de répondre correctement à leurs besoins et rester proche de leur façon de procéder.
Pour cela j'ai donc effectué les Stand Up avec l'équipe des développeurs et ainsi pu les tenir au courant de l'avancement du projet et échanger sur d'éventuelles modifications, réorientations ou corrections de bugs.

La répartition des taches dans l'équipe R&D

La répartition des tâches au sein de l'équipe Recherche et Développement de Digimind se fait à l'aide d'une liste des tâches à effectuer nommée "Backlog".

Lors du Sprint Planning (cf. chapitre suivant : “La gestion du temps”) : les collaborateurs analysent la RoadMap[4] précédemment définie par la hiérarchie et les clients, et définissent ensemble une liste de tâches à effectuer (nommées “User Stories”) affin de constituer le Backlog.
Notées sur des Post-it et collées sur un mur, ces Users Stories seront ensuite pondérées[5] et chaque ingénieur pourra ensuite choisir la tâche qu’il a le plus envie de faire sur le moment.

Tout ce fonctionnement est bien entendu spécifique à chaque équipe et peut s’adapter à des équipes de toute taille. J’ai par exemple lors de mon premier projet dans la société travaillé seul, et donc élaboré un Backlog correspondant à ma mission, pondéré les taches et les ai menées à bien sur la durée d’un Sprint. Par souci pratique, j’ai continué à effectuer les “Stand Up” (ou comptes rendus quotidiens) avec les équipes de développeurs.

La gestion du temps

Dans l’informatique comme dans beaucoup d’autres secteurs, la gestion du temps lors de projets représentant de fortes charges de travail peut être problématique. Sous différents aspects la livraison en temps et en heure ainsi que le caractère à durée indéterminée d’un projet peuvent s'avérer problématiques. Pour palier à ce genre de problèmes une date de livrable est définie régulièrement. Nommé Sprint par la méthodologie Agile, ces périodes de temps définissent une deadline mais marquent aussi les différentes étapes d’un projet entre lesquelles il est possible de redéfinir sa direction et réorganiser la charge de travail.
Le principal intérêt d’avoir des dates précises pour changer la direction et revoir les points sur lesquels travailler est d’éviter de perturber l’avancement du projet entre chaque Sprint et ainsi que le projet aille « dans tous les sens ». La méthode Scrum préconise usuellement une durée de 2 semaines à 1 mois pour chaque livrable.
Par souci pratique et pour des questions de réactivité, la durée des Sprint à Digimind a été réduite à une semaine. C’est à dire qu’à la fin de chaque sprint une réunion de compte rendu est organisée afin de présenter les livrables, organiser les autres tâches et redéfinir la direction des différents projets en fonction des nouveaux objectifs définis pour le produit.

[4] La RoadMap est un planning de développement correspondant aux fonctions qui seront implémentées dans les versions futures du logiciel ainsi que l’avancement de leur réalisation.
[5] La pondération des tâches est dans l’objectif de leur donner une taille (S, M, L ou XL) afin de mieux visualiser la quantité de travail représenté par chacune d'elles.

Afin de visualiser plus simplement les efforts restant à délivrer en fonction du temps restant disponible avant la fin du sprint nous utilisons un “BurnDown Chart”. Le but est de visualiser à l’aide d’un simple graphique la quantité de taches effectuées en fonction du nombre de jours restant.

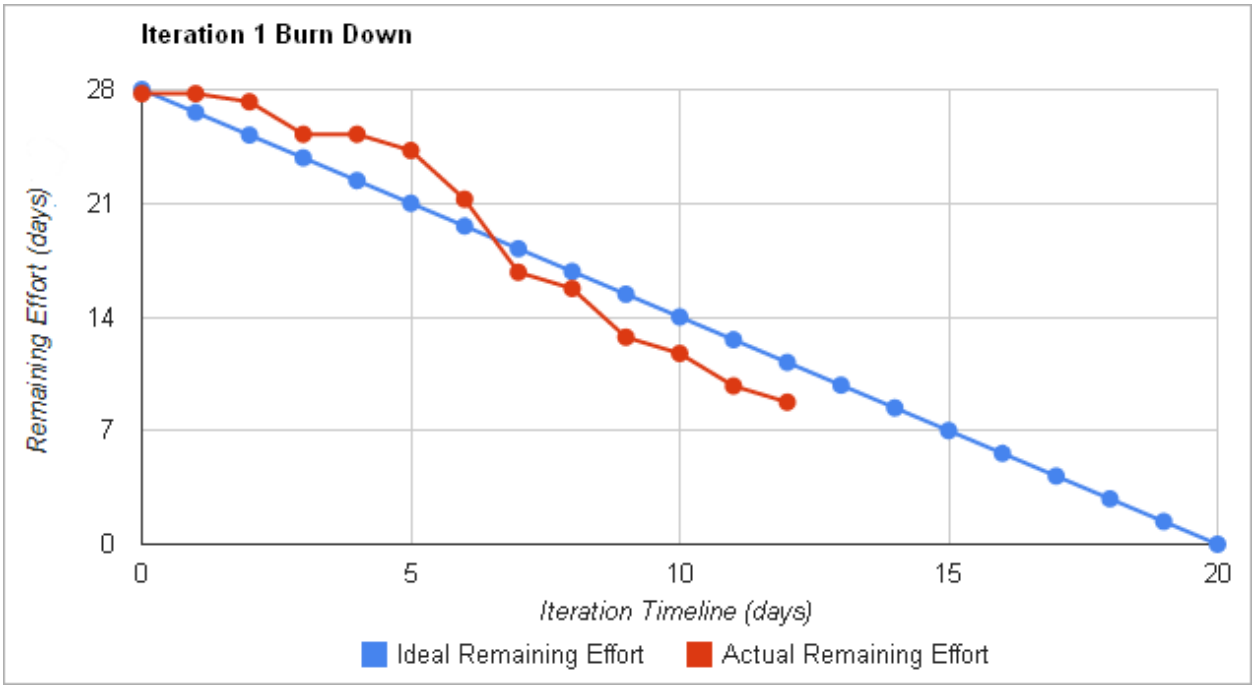

Figure 2: "BrunDown Chart" représentant l'évolution de la charge de travail en fonction du temps

Comme expliqué dans le paragraphe précédent chaque tâche est pondérée, il est donc logique qu’une tâche décrite comme importante soit davantage représentée sur le diagramme.
Sur ce schéma la courbe bleue représente les efforts idéalement fournis afin de maintenir une charge de travail continue au long du sprint. La rouge matérialise quand à elle les efforts développés par l'équipe.
Afin d'obtenir ce schéma il est bien sur indispensable d'avoir au préalable pondéré et correctement défini les tâches à effectuer durant le sprint. De façon évidente, aucune tâche ne doit être ajoutée en milieu de Sprint de façon à ne pas fausser les calculs effectués.

Cette méthode de visualisation des tâches restantes à effectuer, qui est aussi préconisé par la méthode Agile, est particulièrement pertinente dans la mesure où elle permet de visualiser tout au long du sprint et de façon très simple si la charge de travail effectuée est en dessus ou en dessous de la quantité idéale à fournir pour tenir les délais.

Afin de générer ce graphique c'est la solution RedMine qui est utilisée à Digimind. C'est une solution Open Source de gestion de planning Agile.

Forces et faiblesses de Digimind

Je vais présenter dans ce chapitre un diagramme montrant les forces et faiblesses de l'entreprise Digimind.

J'ai choisi de les présenter à l'aide d'une matrice SWOT (Strengths, Weaknesses, Opportunities and Threats) de l'entreprise, mettant en relief ses points forts comme ses points faibles, et ses opportunités comme ses menaces, le tout en essayant d'être le plus objectif possible.
Bien que surtout utilisée dans les études de marketing, cette matrice permet d'identifier les axes stratégiques à développer. Elle permet de s'assurer que la stratégie mise en place dans l'entreprise est pertinente et constitue une réponse satisfaisante à son environnement. Elle est donc pertinente dans ce contexte.

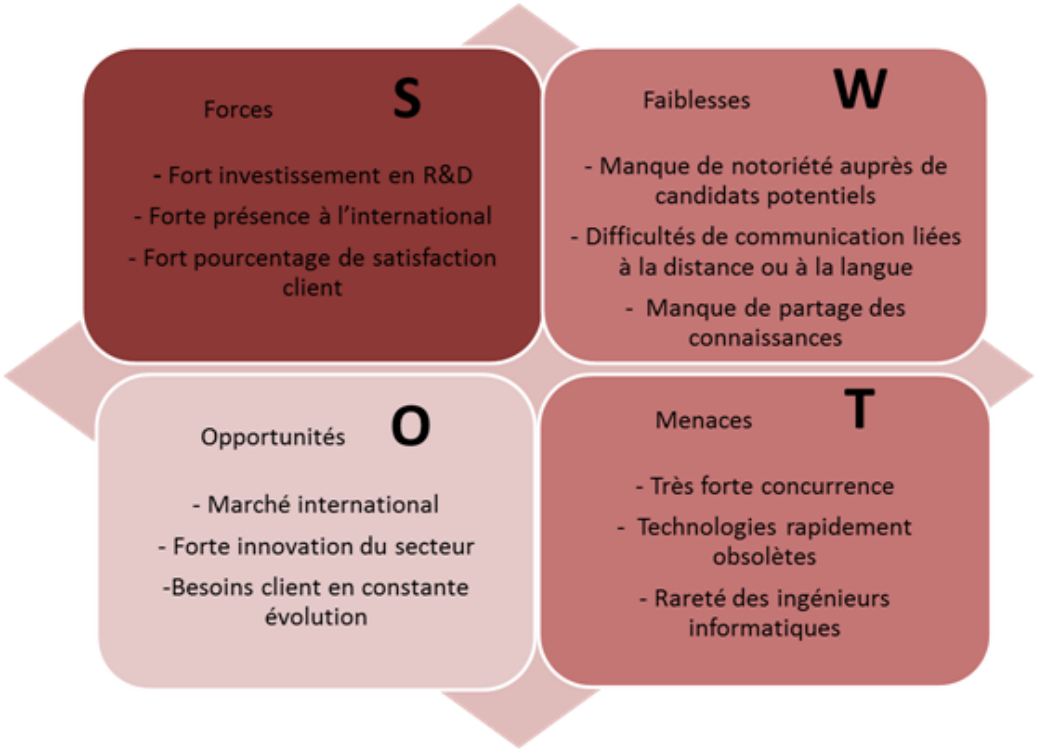

Figure 3: Matrice SWOT présentant les forces et faiblesses de Digimind

Chiffres Clés

Digimind s'est développée très rapidement, accroissant son chiffre d'affaires de plusieurs dizaines de pourcent par an depuis sa création.

Le chiffre d'affaires sur l'année 2013 a été de 7 Millions d'Euros pour environ 150 employés et deux cent clients. Afin d'expliquer ce chiffre, il est important de préciser que la solution développée par Digimind est uniquement vendue aux entreprises et que le coût moyen de la Licence d'utilisation du produit est relativement élevé (elle peut facilement dépasser les cent-cinquante-mille dollars par an). Les plus anciens clients de Digimind sont fidèles à la société depuis plus de 15 ans et 93% des contrats passés sont renouvelés, ce qui témoigne d'une très bonne satisfaction client.

Si je cite ces chiffres, c'est aussi pour amener le fait que d'un point de vue plus technique, la gestion de deux cent clients sur une application aussi complexe et nécessiteuse de ressources que celle développée par Digimind, est un challenge à part entière.

Ce chiffre de deux cent clients est à la fois énorme et ridiculement petit. Je m'explique: beaucoup d'entreprise développant des applications plus simples et destinées au grand public fournissent leurs solutions à plusieurs milliers de clients avec un chiffre d'affaire de deux fois inférieur, mais après les explications fournies sur l'activité principale de Digimind, il est facile de saisir la différence de contexte. Et il en est de même d'un point de vue technique où comparer deux architectures fournissant le même nombre de clients n'est en aucun cas pertinent tant que les services fournis par cette infrastructure ne sont pas correctement quantifiés.
Dans une certaine mesure, ce sont donc aussi ces chiffres qui nous aiderons à mieux cerner la problématique présentée.

ANALYSE DU CONTEXTE: UNE TRANSITION PROGRESSIVE VERS LA GESTION D'UNE INFRASTRUCTURE DE GRANDE ECHELLE

Présentation générale de l'infrastructure existante

Je vais ici décrire puis analyser les environnements de développements ainsi qu'une partie de l'environnement de production de la société. Nous verrons que ces points précis de l'infrastructure technologique de Digimind nécessitent une augmentation significative des ressources attribuées à leur fonctionnement afin de répondre aux besoins croissants de l'application développée.

Présentation des environnements de développement

Afin de maintenir la qualité de service d'une application web ainsi que sa disponibilité la pratique usuelle des sociétés délivrant un logiciel en tant que service (Software as a Service[6]) et de reproduire l'intégralité de l'infrastructure de production (celle accessible par les clients) dans une seconde infrastructure uniquement destinée au développement. Ainsi les développeurs travaillent sur cet environnement isolé de la production, afin de préparer la nouvelle version du logiciel et la déployer par la suite sur l'environnement de production.

À Digimind afin de répondre à la complexité des RoadMaps de développement et de permettre à plusieurs équipes de travailler simultanément, il existe (pour l'application Digimind Intelligence uniquement) un environnement de pré-production, un environnement destiné à la correction des bugs retournés aux supports par les clients, mais aussi de deux environnements sur lesquels sont effectués le travail des RoadMap 1 et 2, destinés au développement de nouvelles fonctionnalités définies au préalable dans la RoadMap. Le dernier environnement est utilisé à des fins de test et de QA (Quality Assurance), l'équipe de Rabat étant aussi en charge des tests de conformité et de qualité des nouveaux développements. Ce dernier environnement possède la particularité de devoir être accessible depuis les bureaux de Rabat.

Lors de mon arrivée à Digimind en Juillet 2013 une salle complète était attribuée aux serveurs des environnements de développement. Le fonctionnement est simple, chaque serveur installé correspond à une application précise avec une fonction spécifique définie.

[6] Le notion de "Software as a Service" (SaaS), qui sera décrite plus précisément dans les chapitres suivants, consiste en la mise à disposition du produit développé à la façon d'un service (web, le plus souvent). Son intérêt est d'éviter l'installation et l'intégration du logiciel chez le client.

Afin de répondre aux besoins évolutifs des développeurs, il s'est avéré indispensable d'induire davantage de scalabilité dans les environnements de développement (en quantité de ressources attribuées comme en nombre d'environnements isolés), car ils peuvent, par exemple, avoir besoin d'un nouvel environnement sur une courte durée de temps afin de développer une nouvelle fonctionnalité urgente, tout en laissant les autres environnements dans leur état actuel afin de reprendre leur travail plus tard.

Présentation du cloud d'analyse de données de l'environnement de production

Avant toute analyse portant sur cette technologie je trouve important d'en faire ici une rapide description qui sera plus largement étoffée dans les chapitres suivants.
Un Cloud de virtualisation est un ensemble de machines virtualisées, de réseaux et de stockages qui dans l'objectif d'une économie d'échelle sont mis en réseau afin de délivrer une même solution.
L'objectif est ici de mettre à disposition de l'entreprise une infrastructure (constituée au minimum des machines virtuelles de leurs réseaux et du stockage qui leur est associé) le plus simplement possible et en tant que service. C'est par la virtualisation de ces ressources qu'il devient possible de les délivrer d'une façon plus efficace, rapide et automatisée.

Pour des raisons de confidentialité des technologies et des solutions utilisées dans ce projet, les réels besoins ainsi que l'utilisation finale de cette solution et de cette infrastructure, ne peuvent être cités (ou alors de façon simplifiée et transformée) lors de l'abord des paragraphes y faisant référence.

Afin de garder une logique dans mon raisonnement et dans les explications détaillées ici, nous établirons les faits suivants:

- Chacune des 380 machines appartenant à ce cloud de machines virtualisées exécutent le même algorithme de traitement de données.
- Un dysfonctionnement de l'algorithme peut subvenir à tout moment et ne peut pas être anticipé.
- En cas de dysfonctionnement de l'algorithme, il ne peut pas être relancé, la machine doit donc être supprimée et remplacée par une nouvelle créée à partir d'un Template prédéfini.

- L’algorithme n'écoute sur aucun port, et un dysfonctionnement ne peut pas être détecté par les technologies fournies par notre fournisseur “Amazon AWS” ou par une technologie de loadbalancing telle qu’HAproxy.
- La totalité des communications entre le Cloud de machines virtualisées et le reste de l'environnement est chiffrée et ne doit pas être déchiffrée.
- Le projet sera nommé “cloud d’analyse de données”

Toutes les machines utilisées dans ce projet sont hébergées dans un cloud de virtualisation public[7] fournis par Amazon est nommé “Amazon Web Services”. J’abrégerais cette dénomination en “Amazon AWS” dans les paragraphes y faisant référence.

Chacune des machines fournissant le même service, un loadbalancer[8] frontal est chargée de repartir la charge de travail de façon égalitaire entre chacune des 380 machines. Ainsi la puissance d'exécution totale sera égale à la puissance d'une machine multipliée par le nombre total de l'ensemble. Cette technique vise à étendre les performances d'une application de façon horizontale (en multipliant les machines affectées à son exécution) plutôt que de façon verticale qui consiste quand à elle à étendre les performances de la machine exécutant l'application.

Pour aider à la compréhension de cette architecture, j'ai réalisé un schéma expliquant et simplifiant son fonctionnement. Il illustre la fonction primaire du cloud d'analyse de données employé à Digimind en visualisant en rouge les données entrantes non traitées et en vert les données sortantes de l'infrastructure après leur analyse et leur traitement.

[7] Cloud de virtualisation fournis par une société tierce.
[8] Un LoadBalancer est un répartiteur de charge, c'est à dire que sa fonction principale est de repartir les requêtes effectuées de façon égale sur les différentes machines virtuelles de l'infrastructure effectuant la même tâche.

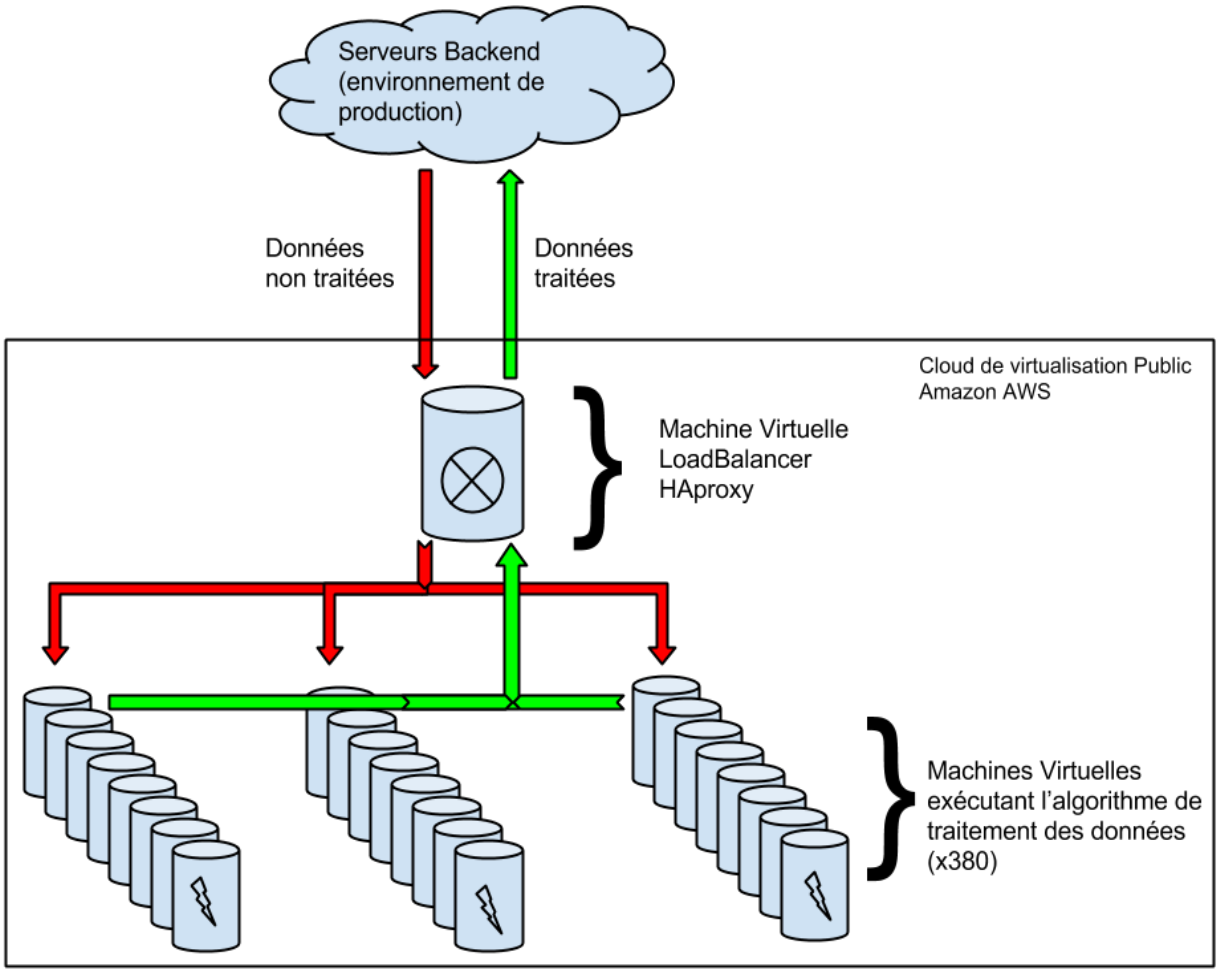

Figure 4: Schéma de l'architecture du Cloud d'analyse de données de Digimind en production

Afin d'assurer le bon fonctionnement de l'infrastructure un script est exécuté toutes les 20 minutes sur le loadbalancer : il envoie une donnée test à chacune des machines virtuelles et analyse la valeur retournée. Si la donnée retournée a correctement été traitée la machine est considérée comme saine et continue son travail. À l'inverse, si après analyse la donnée retournée n'est pas considérée comme correctement traitée, la machine virtuelle est entièrement supprimée pour être remplacée par une nouvelle machine.

Je précise à nouveau que, pour des raisons qui sont confidentielles, le programme ne peut pas être simplement redémarré sur la machine virtuelle corrompue, mais nécessite l'utilisation d'une nouvelle machine virtuelle.

Analyse des besoins de l'infrastructure existante

Analyse des besoins des environnements de développement

Comme décrit dans le chapitre précédent, les environnements de développement sont ici fournis aux développeurs par le biais de machines physiques sur lesquelles sont installées les applications nécessaires au bon fonctionnement de l'application.

Les principaux problèmes du mode de fonctionnement décrit précédemment sur les environnements de développement de Digimind sont:

- La résistance à la panne : en effet, sans sauvegarde des données de développement, les configurations et les données collectées sont perdues à la première panne de disque.

  C'est de ce problème là, qu'est tiré le premier objectif de ce projet: Une meilleure résistance face aux pannes.
- La souplesse et la capacité d'adaptation de l'infrastructure. En effet, les développeurs ayant des besoins très variables en terme de capacité matérielle pour les environnements sur lesquels ils développent, il est nécessaire que ces environnements puissent être supprimés, remis à zéro et redéployés à la volée.
- Dans une autre mesure, la consommation électrique du grand nombre de serveurs présents ainsi que la place qui leur était attribuée est une composante importante dans l'implémentation de ce projet. Ce qui nous amène au troisième point important: diviser le nombre de serveurs par deux au minimum afin de réduire d'autant leur consommation électrique et la place nécessaire à leur stockage.

Outre les défauts qui viennent d'être cités, cette pratique de gestion des environnements pose aussi d'autres problèmes: les développeurs ont besoin d'au minimum cinq environnements complets, ce qui comprend normalement environ douze serveurs distincts pour faire fonctionner l'application de la même manière en développement qu'en production. Si on multiplie ce nombre par les cinq environnements, cela nous donne environ soixante serveurs à administrer, installer, mettre à jour et réparer en cas de panne et ce uniquement pour les environnements de développement.
Nous pouvons ajouter à ce nombre pas moins de cent soixante serveurs physiques appartenant à l'environnement de production pour obtenir le nombre total de serveurs à monitorer et

maintenir, ce qui représente une charge de travail plus que considérable pour l'équipe exploitation constituée de mon tuteur d'alternance et moi-même.

C'est donc dans l'objectif de palier à ces problèmes, qu'il a été nécessaire de trouver un nouveau moyen de fonctionnement afin de simplifier la maintenance des environnements de développement, de réduire le nombre de machines utilisées et enfin réduire au maximum l'impact sur le travail des développeurs en cas de panne.

Analyse des besoins du cloud d'analyse de données en production

L'analyse détaillée dans ce paragraphe concerne quand à elle le Cloud d'analyse de donnée de la production de Digimind (dont le fonctionnement à été décrit dans le chapitre précèdent). Ce projet répond à la même problématique de scalabilité de l'infrastructure que celui sur les environnements de développement de la société.

Cette solution, déjà implémentée dans l'entreprise, présente plusieurs inconvénients que je vais décrire dans les paragraphes suivants.

Avec l'aide de mon tuteur d'alternance et de Mr Delalande, manageur de l'équipe R&D, j'ai identifié trois problèmes importants dans le fonctionnement de ce Cloud de machines virtualisées.
Dans un premier temps, le fait que le script de vérification du bon fonctionnement de l'algorithme soit exécuté toutes les 20 minutes: si un dysfonctionnement survient sur l'une des machines exécutant l'algorithme peu de temps après l'exécution du script, toutes les données transmises à cette machine pendant un laps de temps d'un maximum de vingt minutes ne seront pas traitées ni redistribuées: Elles seront définitivement perdues, ce qui n'est pas acceptable dans le fonctionnement nominal de l'application délivrée par Digimind.

Le premier objectif de ce projet est donc de réduire la durée d'indisponibilité au maximum afin de ne plus perdre de données.

Les deux autres problèmes identifiés sont dus à des soucis de scalabilité de l'infrastructure.
En effet, lorsque le projet a pris jour deux ans auparavant, une trentaine de machines virtuelles étaient utilisées lors du fonctionnement nominal de l'application. Avec l'augmentation du

nombre de clients et les deux nouvelles applications développées par Digimind qui utilisent elles aussi ce projet, il a été nécessaire d'accroître progressivement le nombre de machines virtuelles jusqu'aux trois cent quatre-vingts actuellement existantes.
Cette forte augmentation du nombre de machines exécutant l'algorithme pose problème lors de la vérification de leur fonctionnement. Le script initial qui fonctionnait parfaitement pour la vérification de 30 machines, nécessite aujourd'hui beaucoup trop de ressources et de temps (plusieurs heures) pour la vérification du fonctionnement des trois cent quatre-vingts machines virtuelles actuelles (je rappelle que le programme envoie à chaque machine une donnée test et analyse le résultat afin de déterminer son état).
D'autre part, le logiciel "Xinetd" utilisé pour la répartition des tests sur les différentes machines n'arrive que difficilement à gérer un si grand nombre de machines et retourne régulièrement une erreur correspondant à un troisième état "inconnu" des machines virtuelles.

Le second objectif est donc de trouver une solution permettant de réduire le temps d'exécution des tests afin de les rendre instantanés (ou en tous cas s'en approcher au maximum) et de supprimer le troisième état "inconnu" des machines virtuelles afin de ne plus perdre de ressources et de toujours savoir si leur état est "sain" ou "corrompu".

Analyses des solutions à apporter en fonction de mes compétences

Palier au manque de souplesse et de scalabilité des environnements de développement

L'une des solutions retenues pour palier aux problèmes précédemment cités est la mise en place et l'utilisation d'une plateforme de Cloud Computing[9].
C'est cependant une solution très générale puisque OpenStack ne correspond pas à une technologie à implémenter en particulier mais davantage à un framework ou un recueil de nombreux logiciels pouvant ou non être utilisées pour la création d'un Cloud de virtualisation.
Il m'a donc été nécessaire d'analyser de nombreuses fois les différents besoins afin de pouvoir déterminer les solutions les plus à même de correspondre aux besoins des développeurs et de l'infrastructure déjà en place.

[9] Une plateforme de Cloud Computing vise à fournir à l'entreprise une infrastructure entièrement virtualisée en tant que service à la demande, simple et rapide d'accès.

De nombreux paramètres sont à prendre en compte lors de l'élaboration d'un Cloud de virtualisation privée. Ils dépendent notamment du matériel à disposition pour la réalisation du projet, du nombre de machines virtuelles à mettre en place ou encore du type d'utilisation dont la plateforme de virtualisation Cloud fera l'objet.

N'ayant pas de précédentes expériences concernant l'implémentation d'OpenStack en entreprise, il m'a été nécessaire de passer énormément de temps à me documenter afin d'éviter au maximum de revenir sur des décisions prises au préalable et de perdre ainsi beaucoup de temps.
J'ai cependant dû expérimenter à plusieurs reprises et revenir sur certains choix importants nécessitant de nombreux changements.

Augmenter de façon significative la capacité globale de traitement du cloud d'analyse de données en production

Je vais dans ce paragraphe analyser et détailler les solutions proposées en réponse aux trois problématiques précédemment définies.

- Comment réduire au maximum le temps de détection du dysfonctionnement d'une des machines?
- Comment effectuer le plus rapidement possible le grand nombre de tests à effectuer?
- Comment supprimer le troisième état "inconnu" des machines virtuelles?

L'idée est ici de trouver une nouvelle façon de procéder à la vérification de l'état des machines de manière plus rapide, moins coûteuse en ressource et plus stable.

J'ai décidé de répondre à cette problématique non pas avec l'implémentation de nouvelles technologies plus performantes, mais en modifiant et en simplifiant l'architecture des tests de vérification.
Plutôt que d'utiliser le Loadbalancer afin de lancer les tests sur chacune des machines du cloud d'analyse de données, j'ai opté pour une architecture distribuée où chacune des machines teste elle-même le statut de son algorithme d'analyse de données afin de déterminer son état et s'éteint en cas d'état corrompu.

Cette nouvelle architecture divise la charge des tests à effectuer par le nombre de machines présentes, la charge reste donc toujours la même et n'évolue pas en fonction du nombre de machines.
Cette nouvelle architecture nécessite le développement d'un nouveau programme de vérification de l'état de la machine.

Analyse des spécificités de l'entreprise dans l'implémentation du projet

Spécificité des environnements de développement de Digimind

De part la complexité et la diversité de l'environnement de production de l'application Digimind qui est construit depuis plus de quinze ans, composé de différentes versions de Windows et de Linux, de près de cent soixante serveurs et de plusieurs dizaines de logiciels et de services différents, les environnements de développement sont eux aussi lourds et complexes.

Cette spécificité empêche les développeurs de travailler directement dans des machines virtuelles sur leurs postes de travail (qui est la façon de procéder d'un grand nombre de Startup) et nécessite la mise en place d'environnements de développement constitués de plusieurs serveurs destinés à cet usage.
Cette complexité est aussi due à la fonction même du logiciel développé par l'entreprise. En effet, la collecte et l'analyse d'un grand nombre d'informations nécessitent d'importantes bases de données et donc la mise à disposition d'importantes ressources matérielles.

Spécificité du cloud d'analyse de données de Digimind

La principale spécificité du cloud de traitement des données de Digimind en comparaison avec n'importe quel autre cloud de machines, réside dans l'algorithme d'analyse utilisé. En effet, il est impossible de déduire son bon fonctionnement par les vérifications usuellement mises en place dans ce genre de solutions.
Afin d'expliquer son fonctionnement, je dirais que l'algorithme d'analyse de données peut avoir deux états distincts: Fonctionnel et non-fonctionnel. En cas de dysfonctionnement, l'algorithme ne stoppe pas le traitement des données mais retourne des données corrompues.
Le problème principal se situe dans le fait que les données ressorties par le programme ne peuvent pas être analysées pour déterminer l'état de l'algorithme car l'intégralité de ces données (entrantes et sortantes) sont chiffrées et ne doivent pas être déchiffrées.

Par ailleurs, comme expliqué dans le paragraphe de description de cet environnement, en cas de dysfonctionnement de l'algorithme, la machine l'exécutant doit être entièrement supprimée et remplacée par une nouvelle car il ne peut pas être relancé sur cette même machine.

Du fait de cette spécificité, il est impossible d'utiliser une vérification basique de l'état de fonctionnement de la machine telles que celles proposées par les technologies courantes de loadbalancing.

Analyse personnelle par rapport à ce contexte

J'ai choisi de séparer cette analyse du contexte dans lequel la problématique que je vais développer ici prendra place, en deux parties à la fois très similaires et distinctes, que sont les environnements de développement et une partie spécifique à la production de l'entreprise composée par un ensemble de machines virtuelles effectuant la même tache de traitement de données.
Il est important de présenter ces deux points car bien que prenant place dans des environnements différents, ils présentent tout deux la même difficulté de passage à une échelle plus large en nombre de machines et donc en capacité de charge de travail traitée. C'est plus spécifiquement sur ce problème que je vais centrer mon analyse.

Dans le cas des environnements de développement, d'importantes modifications sont à effectuer sur la base même du fonctionnement de l'infrastructure, afin d'arriver à rendre le fonctionnement de ces environnements plus proche des besoins des développeurs. Il est ici nécessaire de pouvoir multiplier, élargir leur capacité de traitement, les sauvegarder et supprimer. Le tout avec le moins d'impact possible sur le travail de l'équipe d'exploitation qui pourra ainsi concentrer ces efforts sur la production de l'entreprise.

Dans le cas à la fois similaire et différent du Cloud d'analyse de données de la production de l'entreprise, des solutions permettant d'augmenter la capacité de traitement générale du cloud de machines sont déjà implémentées mais présentent aujourd'hui d'importantes limites sur le nombre de machines pouvant être géré. Il est donc nécessaire de revoir ici le mode de fonctionnement de ces solutions afin de pouvoir augmenter de façon significative la capacité de traitement du pool de machines.

PROBLEMATIQUE: COMMENT FAIRE FACE AUX PROBLEMATIQUES DE SCALABILITE DE L'INFRASTRUCTURE DANS UNE ENTREPRISE EN FORTE CROISSANCE ?

Explication du besoin de scalabilité des infrastructures

La taille de l'infrastructure d'une entreprise délivrant un produit de type Saas dépend de plusieurs facteurs tels que le nombre de clients, la complexité des solutions logicielles mises en œuvre et l'utilisation des ressources matérielles de celle-ci.
De ce fait, la plupart des entreprises démarrent leur activité avec une structure logicielle relativement simple et un nombre de clients limité, et n'ont besoin à ce moment que d'une infrastructure de taille réduite.

L'objectif de toute entreprise est généralement de servir un nombre toujours plus important de clients et de développer de nouvelles fonctionnalités (nécessitant la plupart du temps davantage de ressources matérielles). Il est donc normal et nécessaire que l'infrastructure supportant et distribuant les services de l'entreprise s'adapte en conséquence, afin de répondre à la demande croissante: typiquement en multipliant son nombre de machines. On parle alors de scalabilité horizontale.
La scalabilité verticale quand à elle, consiste à augmenter les performances d'une machine de façon à répondre aux besoins présentés, mais elle n'est plus très utilisée aujourd'hui. Un bon exemple de scalabilité verticale est le cas particulier du site internet leboncoin.fr délivrant son application par le biais d'un seul et unique serveur aux capacités surdimensionnées.
Je ne présenterais pas ici les nombreux inconvénients de la scalabilité verticale, mais la méthode la plus souvent utilisée par les entreprises (la scalabilité horizontale) consiste à multiplier le nombre de serveurs délivrant une même application et répartir les requêtes des différents utilisateurs sur le pool de machines à l'aide d'un Load Balancer.
Ainsi, en multipliant le nombre de machines, l'infrastructure peut faire face à une charge plus importante tout en réduisant les risques de panne matérielle, car si l'une des machines fait défaut, le Load Balancer re-divise la charge entre le nombre de machines restantes, qui continuent ainsi d'effectuer leur tâche.

L’essentiel de mon travail au sein de Digimind avait comme objectif de contribuer à l'élargissement de l’infrastructure de façon à ce que le nombre de machines gérées ait le moins d’impact possible sur le travail de l’équipe exploitation.

La mise à disposition d’une infrastructure en tant que service

Une entreprise qui fournit un logiciel en tant que service (SaaS) le met à la disposition de ses clients à la façon d'une ressource entièrement externalisée. Dans la plupart des cas, et c’est celui de Digimind, le logiciel est fourni en tant qu'application Web.

La notion d’“Infrastructure as a service” (Iaas) quand à elle, reprend le même principe mais délivre une infrastructure et non pas un logiciel. C’est à dire qu’une entreprise fournissant une infrastructure en tant que service administre et délivre pour ses clients des machines virtuelles, un ou des réseaux, du stockage et des Load Balancer suivant le type d’offre. (cf. Annexe Figure 8, Page 74)
C’est par exemple le cas d’Amazon avec leurs “Amazon Web Services” et d’autres sociétés telles que “Rackspace” ou encore “Softlayer” qui fournissent à leurs clients des infrastructures entièrement clés en main et administrées. Nous parlons donc de “Cloud Public ou Cloud externalisé”.

OpenStack, quand à lui est un ensemble de logiciels destinés à fournir une infrastructure en tant que service. Il est notamment utilisé par la Société Rackspace (qui a été l’une des premières compagnies fondatrices du projet) pour fournir ses services.
Lors de son implémentation et de son utilisation au sein de Digimind, ce service délivrant une infrastructure “Iaas” est hébergé et administré directement dans l’entreprise, nous parlons donc de “Cloud privé, ou interne”.

Analyse des éléments sur lesquels porter mon travail

Afin que les coûts de gestion d’une infrastructure ne soient pas proportionnels au nombre de serveurs qu’elle comporte, le point de rupture entre le management dit “classique” d’une infrastructure et la mise en place de solutions simplifiant le passage à une gestion à grande échelle est un élément clé dans l'expansion d’une société.

Je vais dans un premier temps concentrer mon travail sur les environnements de développement de la société. Comme expliqué dans le chapitre “Analyse des besoins de l’infrastructure existante”, afin de mieux répondre aux besoins des développeurs, il est important d’induire davantage de scalabilité dans les environnements de développement, afin de pouvoir multiplier les environnements de travail de façon isolée sur une infrastructure en tant que service, où les machines, le réseau et le stockage sont distribués à la volée.

La seconde partie de mon travail sera davantage concentrée sur le cloud d’analyse de données de la production de l’entreprise. Comme expliqué précédemment, il est nécessaire de revoir et de modifier son fonctionnement afin qu’il puisse supporter un plus grand nombre de machines virtuelles.
Ce travail va donc porter sur l'amélioration de la scalabilité horizontale de ce cloud de machines afin qu’il puisse traiter une quantité plus importante de données en moins de temps.

Connaissances à acquérir

Étant donné que l’objectif de ces travaux consiste à délivrer et administrer une infrastructure en tant que service interne de l’entreprise, une importante connaissance de ses différentes composantes est requise. En d’autres termes, avant de déployer de façon automatisée, il est nécessaire de savoir mettre en place le réseau, virtualiser les machines et implémenter le système de stockage de cette infrastructure.

L’importante quantité de nouvelles notions à aborder et de nouvelles connaissances à acquérir pour la réalisation de ces projets, ne peuvent venir qu'après une très bonne maîtrise des systèmes de type Unix et du fonctionnement bas niveau des ordinateurs en général. Deux thématiques dont j’ai fait ma spécialité et qui représentent mon principal centre d'intérêt depuis plus de cinq ans.

Le réseau est sans doute la partie la plus complexe de ce travail, et c’est dans ce domaine que j’ai le plus de connaissances à acquérir,
Il faut dans un premier temps savoir implémenter et connaître le fonctionnement des différentes technologies nécessaires à la virtualisation de réseaux telles que OpenVSwitch qui est destiné à virtualiser des switchs, ou encore le L3Agent d’OpenStack servant à virtualiser des routeurs.

Dans un deuxième temps, et car en réseaux rien ne fonctionne toujours de façon idéale du premier coup, il est également important d'avoir de bonnes connaissances des différents programmes d'audit, d'administration et de sécurité réseau tels que TCPdump utilisé pour sniffer les paquets réseaux ou encore Nmap pour l'audit et IP pour l'administration.

Une autre caractéristique importante des infrastructures délivrées en tant que services est l'aspect stockage. En effet, c'est un composant de l'architecture qui, à cette échelle et répondant à cette problématique est nouveau pour moi (à posteriori aussi l'un des plus intéressants et enrichissants).
Afin de répondre à la problématique posée de scalabilité et de maintient de services de l'infrastructure, le stockage en tant que part de l'infrastructure doit lui aussi pouvoir étendre facilement sa capacité tout en restant hautement disponible et résilient aux multiples pannes pouvant survenir (disques durs hors service, coupures électriques, etc.).
Pour cela, il est nécessaire d'aborder un grand nombre de connaissances en stockage, et notamment en stockage distribué et redondé.

La virtualisation sous ces différents aspects est au cœur des travaux entrepris et de la problématique présentée. En effet, l'une des solutions les plus simples afin de délivrer simplement une machine et un réseau, est la virtualisation de celui-ci.

Dans le cas des machines virtuelles, l'objectif est ici d'obtenir une plus grande souplesse en changeant le rapport du système d'exploitation à la machine physique et donc de partir d'une association système d'exploitation / machine, à une déconsidération de celui-ci en tant que simple programme.
Dans le cas de la virtualisation du réseau, l'objectif est de fournir ici de façon virtuelle et standardisé un réseau classique et ainsi, de la même façon que pour les machines virtuelles, de pouvoir le supprimer, déployer et dupliquer à l'infini en s'affranchissant ainsi des limites physiques imposées par les routeurs, Switchs et Load Balancer physiques.

Afin de correctement répondre à la problématique posée, des connaissances en Load Balancing, en haute disponibilité, ainsi que dans les technologies de cloud en général sont essentielles, notamment concernant le travail sur les environnements de développement. N'ayant encore jamais entrepris l'installation d'un cloud de virtualisation privé dans une société, je vais devoir aborder en plus de tous ces aspects, beaucoup de processus et technologies

davantage spécifiques à la technologie mise en place, tels que la connaissance des outils internes d'OpenStack et comment interagir avec eux (par exemple à l'aide du langage Python).

Une autre partie des connaissances à acquérir est composée de tous les aspects du fonctionnement spécifique de l'entreprise comme les méthodes de travail des développeurs.
En effet, d'une part les besoins de Digimind seront différents de ceux d'une entreprise d'un autre secteur d'activité ou d'une taille différente, et d'autre part les technologies implémentées pour répondre à cette problématique devront s'interfacer parfaitement avec les technologies, le matériel et les différentes personnes de l'entreprise.

Il est donc important d'appréhender chacun des aspects de l'entreprise.
Ce qui est aussi le cas du travail à fournir sur le cloud d'analyse de données de la production de l'entreprise, car même si la connaissance des méthodes de travail des développeurs est beaucoup moins pertinente dans ce cas, il est essentiel de bien cerner le fonctionnement initial de l'ensemble de technologies déployées pour pouvoir y apporter une amélioration.
D'autre part, étant donné la spécificité de la solution déjà en place et du mode de fonctionnement de l'infrastructure, il est aussi nécessaire de développer certaines fonctions spécifiques de son fonctionnement et (pour les raisons évoqués lors du chapitre sur les décisions prises) donc l'apprentissage du langage Python, ce qui rejoint là aussi le travail sur les environnements de développement de la société.

L'ensemble de ces connaissances me permettra d'aborder l'implémentation d'un cloud de virtualisation OpenStack ainsi que la scalabilité d'un cloud de machines en production, en me reposant sur des bases solides d'administration de système Linux et de fonctionnement bas niveau des ordinateurs permettant d'aborder ces nouvelles notions plus aisément.

METHODES HABITUELLEMENT UTILISEES POUR UNE SITUATION PRESENTANT DES SIMILITUDES: LES SOLUTIONS DE CLOUD COMPUTING PUBLIQUES ET PRIVEES

Méthodes usuelles dans la gestion des environnements de développement

La mise en place d'environnements de développement au sein d'une société dépend d'un grand nombre de paramètres tels que la complexité de l'architecture logicielle déployée ou le nombre d'équipes développeurs travaillant sur le projet. Il n'existe donc pas de méthodes dites usuelles.
Par exemple, dans le cas d'une architecture logicielle relativement simple et sur la même équipe de développeurs, il est fréquent de voir les développeurs travailler dans des machines virtuelles exécutées en local directement sur leurs machines, puis de déployer leur code sur une infrastructure de test afin de procéder la vérification du fonctionnement avant la mise en production.
Dans ce cas de figure, c'est aux développeurs que revient le rôle de mettre en place leurs environnements de développement.

Dans les cas d'une architecture plus complexe, comme c'est le cas à Digimind, les développeurs ne peuvent pas virtualiser le composant nécessaire aux développements sur leurs machines. Il est donc nécessaire de leur fournir une infrastructure identique à l'infrastructure en production afin qu'ils puissent développer.
Pour délivrer ces environnements, il existe ici aussi plusieurs solutions. Pour des besoins qui restent relativement simples et un nombre réduit de développeurs, il est d'usage d'installer les environnements de développement sur un ensemble de machines dédiées à cet usage, ce qui est le cas de Digimind.
Un des principaux inconvénients de ce mode de fonctionnement, comme nous l'avons vu dans le paragraphe sur l'analyse des besoins des environnements de développement de la société, est le manque de souplesse de l'infrastructure.
En effet, les développeurs ont des besoins très variables concernant ces environnements. Afin de lancer le développement d'une nouvelle fonctionnalité, ils peuvent être amené à avoir besoin d'un nouvel environnement identique à celui en production, tout en gardant un autre environnement pour la résolution des bugs remontés par les clients. Dans le cas d'une société aillant une équipe de QA (Quality Agreement) entièrement concentrée sur les tests des produits

développés, il est aussi nécessaire de leur fournir un environnement dédié. (Ce qui est le cas de Digimind)
Dans ce cas de figure, l'une des solutions usuellement apportées au problème, et de mettre en place des environnements de développement en tant que service au développeurs, et ainsi en pouvant à la volée dupliquer, sauvegarder et redéployer leurs environnements nous répondons aux besoins que nécessitaient le développement de l'application.
Toujours dans ce cas de figure, plusieurs types d' "Infrastructure as a Service" existent : la mise en place de cette solution en interne à l'aide d'un cloud privé ou l'externalisation de ces ressources à l'aide d'un cloud public.

Comparaison de différentes solutions en vue de fournir une plateforme de Cloud Computing

Les Cloud de virtualisation privés

La mise en place d'une solution de cloud privé au sein d'une entreprise, est une procédé vaste et complexe et le marché des offres proposées a été longtemps largement dominé par la société qui est aussi l'un des acteurs historiques de cette part de marché: VMware.
En effet, la solution développée et vendue par la société VMware, est l'une des premières solutions apparues sur le marché apportant aux entreprises l'implémentation d'un cloud privé. Elle dispose d'un grand nombre d'avantages tels que la maturité et le support vendu par la société.

Cependant, même si c'était sans aucun doute l'une des solutions la plus utilisée ces dernières année, ceci est en train de radicalement changer avec l'arrivée il y a quatre ans d'un acteur aujourd'hui majeur dans ce marché: Le projet OpenStack.
Ce projet Open Source est aujourd'hui le concurrent le plus sérieux à la solution développée par VMware pour l'implémentation d'un cloud de virtualisation privé.
Initialement mis au point par une collaboration entre la Nasa et la société d'hébergement américaine Rackspace, compte aujourd'hui une grande partie des principaux acteurs du marché de l'informatique dans ses contributeurs.
C'est de part sa nature Open Source que la collaboration entre des entreprises telles que Red Hat, HP, Dell, IBM, Rackspace, Cisco ou encore Symantec est possible.

Mais d'autres alternatives telles que les projets CloudStack ou Eucalyptus, eux aussi Open Sources existent.

Les Cloud de virtualisation Public

Il existe cependant une autre solution, vers laquelle se tournent beaucoup d'entreprises et qui est notamment très utilisée chez Digimind, (j'ai d'ailleurs eu l'occasion de travailler dessus à plusieurs reprises).

D'un point de vue radicalement différent, l'externalisation de toutes les ressources virtualisées de l'entreprise sur une plateforme cloud gérée par une entreprise tierce (telle qu'Amazon avec leur plateforme Amazon Web Service EC2) est une option souvent étudiée. En revanche, elle offre aussi plusieurs inconvénients:

- Le prix,
- L'externalisation des données.

On ne peut pas réellement comparer la plateforme Amazon Web Services à OpenStack même si les deux fonctionnements sont en tous points similaires, étant donné que l'approche est différente. L'implémentation d'une plateforme de Cloud Computing avec OpenStack est destinée à fournir les mêmes services qu'Amazon AWS, sauf quand dans le cas d'AWS le cloud de virtualisation est entièrement hébergé et administré par Amazon. Le client n'a ainsi plus qu'à créer ses différents environnements et à lancer ses machines virtuelles par le biais de l'interface web ou de l'API fournie par Amazon.

En implémentant une plateforme OpenStack chez Digimind, j'ai fourni à l'entreprise son propre Cloud de virtualisation privé, administrable et utilisant les machines de l'entreprise, ce qui offre un certain nombre d'avantages tels que la personnalisation ou la sécurité des données.

A l'inverse, l'utilisation d'une plateforme de virtualisation de type Cloud Privé tel qu'Amazon Web Services, permet à l'entreprise de confier à une entreprise tierce le management de sa plateforme de virtualisation et de s'affranchir ainsi d'un grand nombre de contraintes techniques. Cependant, cette solution présente d'autres points négatifs tels que le coût, qui fluctue en fonction des besoins de l'entreprise.

Cette solution est donc entièrement viable et beaucoup plus simple à mettre en place mais représente un coût d'utilisation pouvant être très important notamment concernant l'espace disque utilisé. L'application développée par Digimind utilise d'importantes bases de données nécessitant une quantité importante d'espace disque. Cette solution ne peut donc pas être utilisée dans ce contexte.

Dans un autre contexte, ou la virtualisation de machines Windows serait dominante, Microsoft dispose lui aussi de sa plateforme de Cloud Computing publique nommée Microsoft Azure, mais ce choix n'est pas pertinent dans le cas de Digimind dont l'infrastructure est très largement composée de serveurs Linux.
En revanche, une solution de Cloud Computing arrivée récemment sur le marché est celle de Google, bénéficiant de prix inférieurs à ceux usuellement pratiqués sur le marché cet acteur du Cloud Computing est en voie de s'installer en tant que Leader. Il propose de plus un grand nombre de fonctionnalités manquantes à la concurrence telles qu'un Load Balancer plus efficace ou un réseau plus performant. Mais ce qui fait de Google Compute Engine un acteur de choix dans le marché des Infrastructures en tant que services est très certainement sa fonction « Live Migration » permettant de migrer les machines virtuelles utilisée d'un hyperviseur[10] à un autre sans stopper son fonctionnement. En effet, les autres fournisseurs tels que Amazon EC2 ou RackSpace doivent parfois effectuer des opérations de maintenance sur leurs hyperviseurs pouvant résulter en une indisponibilité temporaire de la machine virtuelle utilisée.

Comparaison des solutions de loadbalancing et d'auto-scalabilité pour le cloud virtualisé d'analyse de données de l'environnement de production

Il existe un très grand nombre de solutions permettant le loadbalancing et l'auto-scalabilité d'un important nombre de machines. Je vais présenter dans les paragraphes suivants les solutions habituellement les plus utilisées dans ce cas de figure.

Les solutions intégrées aux Amazon Web Services

La première des solutions qu'il est intéressant d'étudier dans ce cas de figure, est l'importante quantité de services fournis par Amazon dans cet objectif.

Amazon propose en effet en plus du service EC2 (Amazon Elastic Compute Cloud) qui est le plus utilisé pour la virtualisation des machines utilisées, d'autres services destinés à fournir aux entreprises une façon simple de repartir la charge et d'implémenter une scalabilité horizontale dans leur infrastructure.

[10] Serveur physique utilisé pour la virtualisation de machines virtuelles.

Par exemple, “Route 53” est un service de nom de domaine (DNS) scalable et hautement disponible directement intégré dans la plateforme Amazon; Le service Amazon “CloudWatch” fournit quant à lui des remontées statistiques sur l’utilisation et la santé des services déployés.

Le recours aux services fournis par Amazon correspond logiquement aux technologies usuellement employées lors de l’utilisation du cloud de virtualisation fourni par Amazon afin d’assurer le loadbalancing et la scalabilité de l’infrastructure. Ces services possèdent de nombreux avantages, par exemple le fait qu’ils soient directement intégrés à la plateforme et ne nécessitent donc aucune installation supplémentaire, ni de machines dédiées pour faire tourner le service.
De plus, elles sont particulièrement faciles à utiliser et peuvent être entièrement configurées via l’interface web fournie ou encore via l’API mise à disposition par Amazon.

Cependant toute l’architecture des services de loadbalancing et d’auto-scalabilité repose sur la détermination du bon fonctionnement des applications installées sur les machines et des ressources utilisées.
Prenons l’exemple le plus simple qui soit: un serveur web destiné à fournir une page HTML. Nous commençons avec un pool de 3 serveurs web et le service de loadbalancing fourni par Amazon: le service répartit les requêtes sur ces 3 serveurs.
Si le service http de l’un des serveurs tombe en panne, il est automatiquement remplacé par une nouvelle machine.
Et si un trop grand nombre de clients se connecte sur la page et que les ressources du serveur sont saturées, de nouvelles instances de ce serveur sont lancées afin de s’adapter à la charge de travail : il s’agit du fonctionnement usuel.

Cependant, comme expliqué précédemment, dans le cas de ce projet l’algorithme de traitement des données exécuté par les machines ne s'arrête pas lors d’un dysfonctionnement mais continue à traiter les données et à retourner des résultats corrompus. Et dans la mesure où les services mis à disposition par Amazon ne peuvent déterminer ce dysfonctionnement, je n’ai pas pu les utiliser dans ce projet.
Nous allons donc comparer d’autres solutions plus administrables et non intégrées à Amazon.

Comparaison des solutions de Load Balancing tierces

Une autre alternative aux services fournis par Amazon est d'implémenter directement les technologies de loadbalancing de manière plus classique par leur administration.

C'est aussi le procédé originellement implémenté dans ce genre de circonstances et les outils mis en place par les prestataires de Cloud Public sont le plus souvent directement développés à partir de ces technologies Open Source.

La technologie historique dans ce cas d'application est sans aucun doute HAproxy. Ce logiciel Open Source, vieux de prés de quinze ans est aujourd'hui la solution logicielle de Load Balancing la plus aboutie. Ces principaux avantages sont la stabilité et la maturité de son code source, le peu de ressources matérielles que demande son fonctionnement et des fonctions très avancées d'ACL[11] permettant un routage avancé des paquets distribués par le Load Balancer.

[11] Les "Acces Control Lists" sont des règles permettant ou non l'accès à une ressource. (ici l'accès aux machines derrière le Load Balancer)

EXPOSE DES DECISIONS PRISES ET DES INTERVENTIONS MENEES AFIN DE RESOUDRE LES PROBLEMES: VERS DES SOLUTIONS DE SCALABILITE HORIZONTALE PLUS PERFORMANTES

Durant ce contrat de professionnalisation de plus d'un an j'ai réalisé plusieurs projets ayant le même objectif: Répondre aux problématiques de scalabilité des infrastructures de l'entreprise.

Afin d'illustrer mon travail à Digimind j'ai choisi de présenter dans ce mémoire deux projets répondant à la même problématique et s'inscrivant dans la même démarche de scalabilité des infrastructures de l'entreprise, ou "Comment passer de la gestion de quelques serveurs à plusieurs centaines (dans le cas présent) avec le minimum d'impacts sur la charge de travail de l'administrateur système", afin de répondre à un besoin croissant de ressources, machines et services dus à l'augmentation du nombre de clients et à la complexité croissante des applications fournies par l'entreprise.

Les décisions exposées ici ont été prise dans l'objectif de pouvoir augmenter les ressources matérielles (autrement dit le nombre de machines physiques) affectées à une infrastructure de la façon la plus transparente possible et ainsi répondre plus aisément aux besoins croissants des applications fournies par cette même infrastructure. Pour cela il a été nécessaire de choisir des technologies pouvant être répliquée et de déterminer une architecture globale de l'infrastructure pouvant supporter un nombre toujours plus important de machines sans impact sur son fonctionnement. Nous verrons ici deux cas de figure distinct,

Le premier projet présenté afin d'illustrer ces propos est un travail sur les environnements de développement de la société, comme je l'ai expliqué dans le paragraphe précédent, sur "L'analyse des besoins des environnements de développement de la société".
Il est nécessaire afin de répondre aux besoins des développeurs d'avoir des environnements de développement qui soient scalables, redéployables et duplicables à la volée.
Il est aussi nécessaire d'augmenter la résilience de ces environnements face aux pannes notamment avec l'implémentation de snap-shots incrémentaux[12] et par un système de stockage entièrement distribué, redondé et scalable à l'infini (ou presque).

[12] Sauvegarde incrémentale d'un état instantané de la machine virtuelle.

Le second projet, quand à lui, est un travail sur une plateforme de Cloud Computing (Amazon AWS) servant à l'analyse et au traitement de données.
Mes travaux concernant ce projet se concentrent sur l'amélioration de la solution d'auto-scaling déjà existante mais ne pouvant plus gérer l'importante quantité de machines virtuelles ajoutées.
En effet, le programme de gestion automatique de la scalabilité de cet ensemble de machines avait à l'origine été écrit par mon prédécesseur et mon tuteur d'alternance pour la gestion d'un petit nombre de machines (à l'origine entre trente et cinquante).
L'objectif de mes travaux sur cette infrastructure est donc ici de gérer un parc de plus de quatre-cents machines virtuelles tout en m'affranchissant des contraintes de tailles imposées par le fonctionnement initial.

Les choix effectués lors de ces différents travaux et projets ont tous pour objectif de simplifier et de rendre possible la gestion d'un important parc de machines et d'infrastructures complexes.

L'implémentation d'OpenStack comme plateforme de Cloud Computing privée

Présentation du déroulement du projet

OpenStack est une solution Open Source de cloud de virtualisation privé. L'objectif est de virtualiser le plus aisément possible une importante quantité de machines ainsi que leurs réseaux et leurs environnements respectifs.
Ce projet relativement vaste et complexe s'est déroulé sur une durée de sept mois et est composé de trois phases principales (chacune suivie d'une phase plus ou moins importante de support et de résolution des problèmes)

- La première phase a consisté en la réalisation d'un POC (proof of concept) et en l'apprentissage de la solution et des différentes technologies associées à ce projet.
- La seconde phase a consisté en l'implémentation du projet au sein de l'environnement de développement de la société.
- Et la troisième phase a été l'industrialisation de la solution en vue de son utilisation quotidienne.

Le choix de la plateforme de virtualisation et de sa technique de déploiement

Même si le choix de l'utilisation d'OpenStack en tant que plateforme de cloud Computing a été décidé avant mon arrivée à Digimind, il est particulièrement pertinent de le détailler ici.
Par ailleurs, il m'a été demandé à la fin du projet de revenir sur cette décision et d'effectuer une analyse comparative des différentes solutions concurrentes afin de s'assurer qu'OpenStack soit toujours la solution la plus à même de répondre aux besoins de l'entreprise.

Le choix d'OpenStack comme plateforme de virtualisation est un choix lourd et difficile car une importante quantité de jours/hommes est nécessaire à son implémentation et à sa maintenance au quotidien.
En revanche, de part la complexité de l'infrastructure logicielle de Digimind, OpenStack reste la solution la plus appropriée.

Dans le cas d'une infrastructure plus simple et uniquement de type Linux j'aurais cependant très largement recommandé l'utilisation de Docker implémenté sur une architecture distribuée par des technologies telles que Consul, Etcd ou Zookeeper.
Le principal intérêt de ce type de technologies appelées "containers" par rapport à un cloud de virtualisation plus classique tel qu'OpenStack réside dans ses performances (en effet, plus de virtualisation ici mais une simple isolation des services par chroot) ainsi que dans sa facilité de gestion et d'implémentation.
Facilité d'implémentation qui est cependant à nuancer dans le cas d'architectures complexes comme celle de Digimind rendant son implémentation particulièrement délicate, voire impossible.

Une autre des raisons pour laquelle l'implémentation d'un réel cloud de virtualisation est préférable à l'utilisation de containers avec Docker est le comportement même des logiciels et solutions implémentées.
En effet, alors que l'Hyperviseur KVM (j'expliquerai ce choix et cette technologie de façon plus détaillée dans le paragraphe: "Choix de l'hyperviseur") d'OpenStack tend à émuler un système le plus strictement identique possible à une installation sur un poste physique, les containers quand à eux isolent les services sur une machine utilisant le même kernel (noyau Linux) ce qui peut changer le comportement de certaines applications spécifiques.
Il est donc préférable pour un fonctionnement le plus identique possible à l'environnement de production d'utiliser de véritables machines virtuelles.

J’ai également dû effectuer un choix stratégique important concernant les différentes distributions d’OpenStack fournies par les entreprises éditrices de la solution.
C’est après m'être longuement documenté sur les différentes distributions fournies que j’ai procédé à des essais d’implémentation dans des machines virtuelles directement sur mon poste de travail.

Parmi les différentes solutions testées j’ai notamment retenu trois modes d’implémentation, par ordre de préférence:
Le premier est RDO, la distribution OpenStack de RedHat qui offre un grand nombre d’avantages par rapport aux solutions concurrentes: l’installation s’effectue à l’aide du gestionnaire de configuration Puppet qui est un outil particulièrement reconnu et très largement utilisé dans le monde de l’administration système et qui apporte une couche de simplicité non négligeable dans l’implémentation de la solution.
De plus, RedHat Enterprise Linux (RhEL) étant le système d’exploitation utilisé par les serveurs de production de Digimind, l’usage de cette distribution s’inscrit dans une logique de continuité de l’exploitation de l’infrastructure.

Le second mode d’installation retenu est celui d’Ubuntu qui bien que non automatisé et donc beaucoup plus lourd et complexe à implémenter, est très largement mieux documenté.

Le troisième et dernier mode d’installation est une implémentation de la solution directement depuis le code source de l’application. Cependant, même si son installation à des fins de tests aura été particulièrement formatrice, cette dernière ne sera pas retenue pour des raisons de complexité d’implémentation et de maintenabilité dans le temps.

Une fois cette phase de test effectuée, j’ai procédé à l’implémentation de la distribution d’OpenStack fournie par RedHat sur les serveurs dédiés destinés à son fonctionnement.

J’ai cependant dû revenir sur cette décision lors du choix de l’architecture réseau, car contrairement à ce qui était indiqué dans la documentation officielle fournie par RedHat et après de longues discussions avec les développeurs de RedHat, il s’est avéré que le type de réseau que j’avais choisi de mettre en place (ce choix est détaillé dans la paragraphe: “Le choix de

l'architecture réseaux à mettre en place") n'était pas encore supporté par la distribution d'OpenStack fournie par RedHat.
Le réseau étant la seule étape de l'implémentation non virtualisable lors de mes test.

J'ai alors choisi de reprendre l'implémentation d'OpenStack en me basant sur la distribution fournie par Ubuntu.
Il est important de préciser que l'architecture réseau en question est aujourd'hui supportée par la distribution délivrée par RedHat.

Choix technologiques inhérents à l'implémentation d'une plateforme de cloud computing

Le choix du système de stockage

L'un des choix le plus critique lors de la mise en place de la plateforme a été le choix du système de stockage.
L'un des premiers prérequis à respecter dans ce projet a été la disponibilité des données, l'objectif étant qu'en cas de panne d'un ou plusieurs disques durs, les développeurs puissent continuer à travailler en toute transparence et sans perdre leurs données.
A cette contrainte vient s'ajouter le fait qu'un important nombre de machines doit pouvoir accéder simultanément aux données sans ralentissement.
La dernière des contraintes à respecter a été de type budgétaire, l'acquisition d'une baie de stockage spécialisée ou de contrôleurs RAID (redondance des disques) n'est pas envisagée, et le matériel à disposition de type "grand public" est particulièrement sujet aux pannes.

En résumé, j'ai eu besoin d'une solution open source et gratuite, distribuée, s'interfaçant avec du matériel grand public et pouvant gérer et exposer une importante quantité de données (plus d'1Téraoctet) sans ralentissement.
J'ai donc sélectionné deux technologies (GlusterFS et Ceph) correspondant à ces critères et effectué des tests "Benchmark" afin de déterminer laquelle utiliser.
Les deux solutions testées à ce moment:

- GlusterFS, qui est d'après sa description officielle (initialement en anglais) " Un système de fichiers unifié, poly-protocole, scalable et au service de nombreux PB[13] de données "

[13] Ici PB est à prendre dans le sens de petabytes

- Ceph, lui aussi d'après sa description officielle est " Un système de fichier objet distribué conçu pour offrir une excellentes fiabilité, évolutivité et performances "

Il est important de préciser que les tests susmentionnés ont été effectués il y a aujourd'hui plus d'un an, et d'après un récent retour d'expérience il semblerait qu'à ce jour GlusterFS soit plus performant que Ceph. (RedHat.org. Distributed Storage Performance For Openstack Clouds: Red Hat Storage Server Vs. Ceph Storage)
Ce test compare cependant Ceph à la solution RedHat Storage qui est le projet dans lequel GlusterFS à été intégré lors de son rachat par RedHat et il n'est donc pas entièrement pertinent.

Un benchmark plus poussé de Ceph peu aussi être consulté dans la partie Bibliographie et Webographie: (Mark Nelson. Ceph Performance Part 1: Disk Controller Write Throughput)

Ces deux technologies ont été récemment rachetées par l'entreprise RedHat et sont maintenant intégrées à leur solution RDO (Acronyme sans signification particulière) qui est la distribution OpenStack de RedHat.

Le choix du type de virtualisation

Le choix du type de virtualisation et donc de l'Hyperviseur a été beaucoup plus rapide et simple que le choix du système de stockage. En effet, il y a un an de ça la compatibilité d'OpenStack avec les différents hyperviseurs était assez limitée. Il a cependant été nécessaire d'en comparer un grand nombre en fonction de leur utilité, de leur performance, des fonctions supportées, mais surtout en terme de simplicité d'utilisation.
En effet, le projet étant déjà de nature assez volumineuse, j'ai trouvé important de choisir un hyperviseur simple à implémenter et entièrement supporté par OpenStack.

Le choix s'est donc naturellement porté vers KVM qui a été le premier hyperviseur compatible avec OpenStack, et même s'il n'est pas le plus simple d'utilisation, il reste le plus performant, fonctionnel et le mieux documenté.

Le choix de l'architecture réseaux à mettre en place

L'infrastructure réseau d'une plateforme de virtualisation Cloud est sans aucun doute le point le plus critique à aborder dans ce type de projet.

En effet l'adressage d'un grand nombre de machines virtuelles sur un réseau physique, ainsi que la gestion des ressources réseaux est un élément clé de l'implémentation d'une plateforme cloud mais est aussi l'endroit où les choix à faire auront les plus lourdes implications sur l'installation.
Par exemple, un réseau sous dimensionné par rapport à l'infrastructure mise en place causera des ralentissements importants sur l'application et risque donc d'entraver les développeurs dans leur travail.

L'essentiel du travail consiste sur ce point à adapter les fonctions et l'utilisation de la plateforme en fonction des ressources matérielles à disposition. Il est par exemple absurde de créer un cluster de stockage distant pour les données des machines virtuelles en cours d'utilisation, si les cartes réseau utilisées ne peuvent pas supporter la charge en conséquence.

Un autre exemple, lui aussi tiré de mon expérience, est la prise en compte de l'inter compatibilité des technologies réseau adoptées pour le projet avec le matériel déjà en place: alors que l'utilisation de Vlan s'est avérée nécessaire pour isoler les différents environnements d'un point de vue IP, les switchs utilisés n'étaient pas compatibles.
J'ai alors dû trouver une autre façon de les isoler, après une nouvelle recherche de la section réseau de la documentation officielle d'OpenStack, je me suis orienté vers les tunnels GRE (Generic Routing Encapsulation) qui forment un protocole d'encapsulation développé par Cisco. Ce protocole est usuellement utilisé lors de la mise en place de VPN. (cf. Annexe Figure 8, Page 73)

Un des inconvénients de ce type d'architecture est qu'il n'est pas supporté par la distribution d'OpenStack fournie par RedHat, et c'est donc après ce choix d'architecture que j'ai du réorienter le choix de la distribution mise en place vers celle délivrée par Ubuntu.

Le schéma disponible en Annexe correspond à l'implémentation du protocole GRE dans un cloud OpenStack afin d'isoler les différents projets et environnements d'un point de vue. Je ne donnerais pas davantage d'explications techniques dans ce mémoire.

Tout ces exemples peuvent paraître logiques et évidents une analysés, mais nécessitent une importante réflexion en amont du projet, et engendrent la plupart du temps de nombreuses identifications et résolutions de problèmes après les premières journées d'utilisation.

Les choix sur l'architecture matérielle de la solution mise en place

L'architecture d'une plateforme de virtualisation OpenStack se compose de plusieurs nodes essentielles à son fonctionnement (une node étant correspondant à un serveur).

La première node est la node d'administration du cloud, c'est elle qui recevra les programmes de gestion de la plateforme, ainsi qu'une base de donnée stockant les paramètres et les informations nécessaire a son fonctionnement. Elle mettra aussi a disposition une interface web destinée a simplifier l'administration du Cloud de machine par le biais d'une interface graphique.

La seconde node va contenir tout les services réseaux de la plateforme, c'est ici que seront virtualisé les switchs et les routeurs des environnent déployés, elle sert aussi de passerelle vers l'extérieur pour tous les éléments internes du cloud de virtualisation (machines virtuelle, services OpenStack, etc.)

La troisième node, qui est aussi la plus conséquente en nombre de serveurs est la node de virtualisation, c'est elle qui sera en charge d'héberger les hyperviseurs destinés à l'exécution des machines virtuelles. La capacité de ses ressources physiques (mémoires vive, CPU) sera équivalente aux ressources qu'il est possible d'allouer aux machines virtuelles.

La dernière node de l'architecture est celle attribuée au stockage des données de celle ci. Comme pour la node virtualisation, sa capacité de stockage qu'elle sera à même de délivrer sera équivalente aux ressources disque allouées aux machines virtuelles.
Vous pouvez apercevoir dans le schéma ci-dessous les quatre nodes composant une plateforme de cloud privé OpenStack.

Il est important de préciser que la première node d'administration du cloud de virtualisation est ici représentée par les mentions "OpenStack Dashboard" et "OpenStack Shared Services"

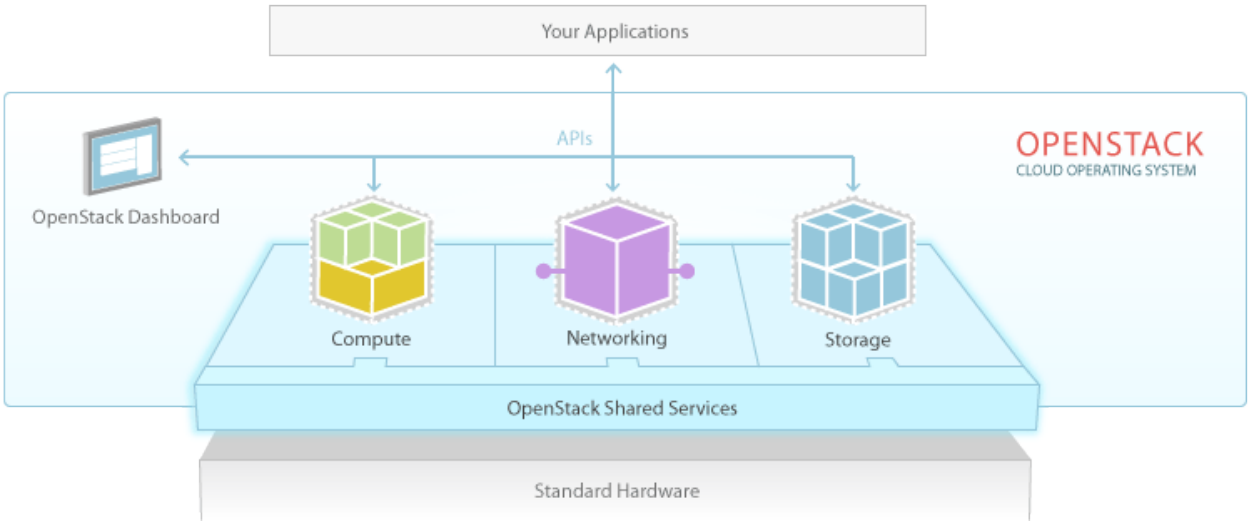

Figure 5: Présentation des différents composants constituants le projet OpenStack

La première étape des choix concernant l'architecture d'une plateforme de cloud privé est son dimensionnement, c'est a dire déterminer le nombre et la capacité des machines constituant chacune des nodes de l'installation.
Cette étape concerne en particulier les nodes de virtualisation et de stockage de l'infrastructure. C'est après avoir déterminé les besoins en mémoire stockage et processeur d'un environnements typique existant et en le multipliant par le nombre maximal d'environnements virtualisé à l'aide de cette plateforme qu'il m'a été possible de dimensionner correctement la quantité de ressources matérielle à attribuer aux nodes virtualisation et de stockage.

La seconde étape est la nécessité ou non rendre certain services hautement disponibles.
La mise en haute disponibilité d'une node ou d'un service nécessite la duplication (au minimum) de la node concernée.

Durant cette implémentation j'ai choisi de commencer par une infrastructure simple et non redondé et d'implémenter par la suite peu à peu la notion de haute disponibilité dans chacun des services.
J'ai donc déterminé une priorité dans les composants de la plateforme à redonder. Le premier des services nécessitant une mise en haute disponibilité est le stockage qui est sans aucun doutes l'aspect le plus critique d'une plateforme Cloud. En effet, ils est indispensable de ne pas perde les données dans le cas d'une panne d'un serveur.

Le second composant critique sur lequel j'ai choisi de me pencher pour l'implémentation d'une redondance est le node réseau d'OpenStack qui est aussi le plus complexe à dupliquer et redonder.

J'ai choisi pour cela un mode redondance dis actif/passif c'est à dire que lors du fonctionnement nominal les deux nodes réseaux ne fonctionnent pas en parallèle et la charge n'est pas repartie, mais cependant en cas de panne du premier serveur c'est le second qui prend le relais. La raison qui m'a amené à faire ce choix est ici très simple, il n'existe pas de façon d'implémenter une haute disponibilité active/active du composant réseau d'OpenStack sans matériel spécifique par exemple des routeurs physiques conçus à cet effet et bien trop chers pour le gain effectif dans cette implémentation.

Impacts stratégiques et managériaux

Avec l'intégration d'une nouvelle plateforme de virtualisation il a été nécessaire de modifier certaines procédures et façons de fonctionner dans l'entreprise, plus précisément au pôle R&D.

Après plusieurs pannes électriques et de longs moments nécessaires au rétablissement de la plateforme, j'ai décidé de rédiger une procédure de redémarrage des différents services la composant.
En effet, la plupart des serveurs nécessitant l'opérabilité d'un autre serveur pour un bon fonctionnement il est nécessaire d'établir un ordre de redémarrage à respecter en cas de panne.

Le redémarrage de l'ancienne infrastructure de type "baremetal" (tous les services installés sur des machines physiques) n'étant pas fréquent, cette procédure n'était pas indispensable.
Mais avec l'ajout des nouvelles fonctionnalités de redémarrage et de redéploiement à la volée de la nouvelle plateforme apportées par OpenStack lors du passage à des environnements de type virtualisé, l'ajout d'une procédure de redémarrage à respecter est devenue indispensable.

La mise en place de cette plateforme de virtualisation a eu un impact direct et important sur le travail des développeurs.

Travaux sur le cloud d'analyse de données en production

Présentation du projet

Le projet de modification du fonctionnement du cloud d'analyse de données a pris place quelques temps après l'implémentation de la plateforme de cloud de virtualisation OpenStack.

Ces deux projets possèdent un grand nombre de similitudes de par les technologies employées et s'inscrivent dans la même logique de scalabilité de l'infrastructure tout en possédant aussi quelques différences significatives.
J'ai travaillé ici sur un Cloud de virtualisation dit "Public", à savoir qu'il est fourni par la société Amazon et non pas implémenté en interne sur les serveurs de l'entreprise, et contrairement au projet précédent il prend place dans l'environnement de production en non pas de développement.

Étant donné que ce projet prend place directement dans l'environnement de production, j'ai choisi de diviser son processus en cinq phases notables, indispensables à son bon déroulement, et ce dans le but de minimiser au maximum l'impact négatif de mon travail sur le travail des clients.

- Identification et analyse des problèmes.
- Documentation et recherche des solutions pouvant êtres apportées.
- Développement et réalisation de la solution apportée
- Test grandeur nature de plusieurs semaines afin de m'assurer du bon fonctionnement et de la viabilité de la solution proposée.
- Procéder à la mise en production de la solution que j'ai apportée aux problèmes rencontrés précédemment.

Le choix des technologies à implémenter

Le choix du langage de programmation

Pour la réalisation de ce projet il a dans un premier temps été nécessaire de déterminer le langage le plus adapté aux besoins du programme. C'est à dire, un langage qui puisse interagir avec l'API mise à disposition par Amazon pour la gestion des machines virtuelles et qui puisse également interagir facilement avec le programme de gestion des données et analyser son résultat.

Mais d'autres paramètres ont aussi été nécessaires pour faire ce choix, tels que ma connaissance dudit langage de programmation ou la rapidité d'apprentissage de celui-ci dans le cas contraire (Une longue formation à un nouveau langage pour la réalisation de ce projet n'étant pas envisageable).

C'est en prenant en compte tous ces facteurs que j'ai choisi d'utiliser le langage Python pour la rédaction de ce programme.

Un autre argument en faveur de ce langage est que le code source du projet OpenStack est lui aussi entièrement rédigé en Python. Ce langage de programmation s'inscrit donc dans une logique de continuité durant mon contrat de professionnalisation après mon précédent projet de cloud privé OpenStack.

Réflexion sur le choix du Load Balancer à utiliser

Comme expliqué au préalable, le Load Balancer sert à répartir la charge de manière égale sur les différentes machines, ici la technologie de loadbalancer utilisée se nomme HAproxy.
C'est une technologie développée il y a prés de dix ans et présentant plusieurs avantages notables par rapport aux autres technologies de la sorte:

- La gestion de la répartition d'une quantité très importante de requêtes pour un usage minime de la mémoire vive du serveur
- Une stabilité longuement éprouvée

L'une des spécificités notables du pool de machines attribué à ce projet est l'ajout et la suppression constante d'un important nombre de machines, et l'un des principaux inconvénients (sans doute le seul) d'HAproxy est la nécessité de redémarrer le logiciel à chaque ajout de nouvelles machines afin que la configuration soit prise en compte.

J'ai alors réfléchi à la mise en place d'une autre solution de loadbalancing, et après une recherche et une analyse approfondie des différentes solutions de remplacement, j'ai choisi d'effectuer un POC (proof of concept) de la technologie HiPache, elle aussi open source et développée par l'entreprise DotCloud (ayant aussi développé Docker), gage de fiabilité.

Contrairement à HAproxy cette technologie à l'avantage de stocker sa configuration dans une base de données Redis (base de donnée NoSQL très légère et performante) et présente donc l'avantage de pouvoir ajouter de nouvelles machines au pool de serveurs sans recharger sa configuration.
En revanche HiPache est développé avec NodeJS (langage JavaScript) et son impact en terme de mémoire utilisée est donc plus important.

Après avoir testé cette technologie en situation réelle pendant plusieurs jours, il s'est avéré que l'utilisation de la mémoire vive utilisée augmentait de façon irrationnelle avec le temps, ce qui est en réalité dû à un problème de fuite de mémoire dans la librairie "node-http-proxy" avec laquelle est développé HiPache.

Après ce test j'ai donc décidé que l'impact de cette fuite de mémoire était bien plus important et problématique que de recharger la configuration à chaque ajout de machine et j'ai donc choisi de garder HAproxy malgré son défaut actuel.

Le choix de la technologie d'automatisation de l'infrastructure

A l'heure de l'automatisation de la gestion des infrastructures il existe une quantité assez improbable d'outils destiné a automatiser la gestion d'un parc de machines conséquent.
La fonction première de chacune de ces solutions est d'appliquer une configuration spécifique ou un ensemble de requêtes sur un ensemble de machines. Chacune des solutions effectue cette tache de manière plus ou moins poussée et plus ou moins avancée.

Étant donné leur nombre et leurs fonctionnalités toutes différentes, le choix de la technologie d'automatisation est un choix complexe dans lequel il est indispensable de rester concentré et de ne pas perdre de vue les besoins initiaux de l'infrastructure.
Car même si l'implémentation d'une technologie extrêmement complète peut être séduisante il est essentiel de s'attarder sur le facteur du temps nécessaire à sa mise en place à son d'apprentissage ainsi que sur son évolutivité par la suite.

Dans le cas du cloud d'analyse de donnée de Digimind les besoins sont extrêmement basiques: A savoir, ne modifier que le fonctionnement des machines virtuelles en cours d'exécution lors du déploiement de mes modifications car celles nouvellement crées le serons a partir d'un Template préalablement stocké sur la plateforme AWS.

Mais malgré la simplicité de ce besoin, le fonctionnement atypique de ce cloud de virtualisation impose quelques spécificités auxquelles un logiciel d'automatisation et de gestion d'un parc de machine classique ne saurait apporter de solution.
En effet, avant d'automatiser une fonction sur un ensemble de machine il faut dans un premier temps en établir une liste, et l'une des spécificité de ce cloud virtualisé est que les machines ainsi que leurs adresses changent en permanence et à une vitesse rendant impossible l'exécution de la requête sur l'intégralité des machines à un instant donné.

Au vu de cette spécificité et des besoins très basiques de la tâche à automatiser, j'ai choisi d'utiliser pour ce projet non pas une technologie et un logiciel clé en main mais directement la librairie sur laquelle sont basés un grand nombre des solutions d'automatisation actuelles: Python-Fabric.
L'un des inconvénients de cette solution est la difficulté à mettre en oeuvre des tâches complexes, mais je bénéficie ici d'un avantage non négligeable: je n'ai qu'un tâche simple à automatiser.
Son principal avantage en revanche est la souplesse et l'adaptabilité de son utilisation, étant donné que c'est une librairie il est nécessaire de développer un programme l'utilisant et ainsi créer de toute part son comportement,
A l'aide de l'API fournie par Amazon, j'ai donc pu récupérer les adresses des machines en cours d'exécution au moment de chaque requête d'exécution de la tâche définie.

Le choix du changement d'architecture et de la suppression d'une technologie existante

Changer l'architecture et le mode de fonctionnement existant pour une nouvelle architecture distribuée est sans doute l'un des choix les plus important de ce projet. En effet, comme expliqué précédemment, le principal problème de fonctionnement de l'infrastructure existante est le temps mis en oeuvre à la vérification du bon fonctionnement de l'algorithme sur chacune des machines (il peut facilement être supérieur à une heure en fonction du nombre de machines non fonctionnelles).

En choisissant une architecture distribuée pour le programme de vérification que j'ai développé, chaque machine exécute indépendamment le programme toutes les minutes (contre vingt minutes précédemment) et ne doit supporter que la charge de ses propres vérifications. Ainsi, si la machine détecte un dysfonctionnement du programme de traitement de données qu'elle

exécute, elle se coupe afin de ne plus recevoir de données et est par la suite supprimée avant qu'une nouvelle ne prenne le relais.

Impacts stratégiques et managériaux

La réalisation de ce projet aura permis d'augmenter considérablement le nombre de machines virtuelles utilisées et ainsi d'augmenter tout autant la capacité de traitement du cloud d'analyse de données.

Le nombre de clients de la société ayant un impact direct sur la quantité de données à traiter, il est aujourd'hui possible pour l'infrastructure de production de supporter un nombre plus important de clients.
De plus l'ajout récent d'une nouvelle fonctionnalité de l'application nécessite le traitement d'une quantité beaucoup plus importante de données et sa mise en place n'aurait pas été possible sans les modifications effectuées.

Une autre conséquence directe de ce projet est la diminution considérable (divisée par 20) des données corrompues remontées par le cloud d'analyse de données. Ces données n'étaient jamais remontées jusqu'au client, mais tout de même perdues. La réalisation de ce projet aura donc permis (je l'espère) une meilleure satisfaction client.

DEMONSTRATION D'UNE ORIGINALITE DANS L'ELABORATION ET LA MISE EN ŒUVRE DE LA SOLUTION : UNE SOLUTION ADAPTABLE ET TOURNEE VERS L'AVENIR

En comparaison à une mise en place plus basique et de type "baremetal" des environnements de développement et de tests d'une société, utiliser un Cloud privé offre de nombreux avantages et représente une approche radicalement différente de la question.

L'utilisation d'OpenStack pour la virtualisation des environnements de développement

L'intégration d'une solution de Cloud privé au sein même d'une société peut s'effectuer d'une multitude de façons différentes. Dans ce cas de figure, j'ai choisi de l'implémenter de la façon la plus indépendante possible du reste de l'infrastructure afin de fournir aux développeurs des environnements de travail isolés au fonctionnement se rapprochant le plus possible de l'environnement de production.

Dans cet objectif, et comme expliqué lors de la présentation du choix de l'architecture réseau mise en place, il a été nécessaire de choisir et mettre en place des solutions adaptées et permettant une parfaite isolation des environnements afin que les travaux de développement entrepris sur un premier environnement n'influe en aucun cas sur les autres. Il a pour ça été nécessaire virtualiser un réseau complet pour chaque infrastructures déployées ainsi que des adressages IP spécifiques et des routeurs virtuels uniques à chacun d'eux.

Démonstration du caractère original et critique de la solution

OpenStack est une solution jeune et il existe aujourd'hui d'autres solutions plus matures et plus simples à mettre en place pour l'établissement d'un cloud de virtualisation.
Par exemple, la solution Vsphere proposée par la société VMware permet aux entreprises le déploiement d'un cluster de virtualisation plus simple à administrer et plus stable. On pourrait de ce fait remettre en cause le bien-fondé de l'implémentation d'OpenStack au sein de la société, mais cette solution Open Source offre de nombreux avantages par rapport aux autres acteurs du marché.

Le premier point sur le lequel je souhaite insister est la liberté d'utilisation et de modification de la solution. Puisque OpenStack est une technologie Open Source, il est totalement possible (et conseillé) de modifier à notre guise le code de l'application, d'en comprendre chacun des mécanismes et ainsi d'arriver à une intégration en toute transparence pouvant correspondre et s'adapter à chacune des spécificités de l'infrastructure préalablement existante.

Le fait qu'il s'agisse d'une technologie Open Source offre également d'autres avantages tout aussi importants: la capacité d'innovation ainsi que la rapidité d'évolution et d'adaptation.
En effet, OpenStack est aujourd'hui riche d'une communauté de plusieurs milliers de développeurs et les principaux acteurs de l'informatique moderne contribuent au projet. Des entreprises telles qu'IBM, Red Hat, HP, RackSpace mais aussi de nombreux centres de recherche tels que le CERN contribuent grandement au projet, le plus souvent en employant des développeurs et des ingénieurs de tout type.
En résulte un projet gigantesque dont les capacités d'innovation et la rapidité d'évolution surpassent très largement un produit propriétaire tel que Vcenter, où une seule société emploie des ingénieurs pour travailler sur un projet qui leur est propre.
Une seule entreprise ne peut pas aujourd'hui surpasser le travail d'un projet communautaire auquel contribue un si grand nombre d'acteurs majeurs du marché de l'informatique. Les coûts d'un développement destiné à concurrencer un tel produit seraient beaucoup trop importants et impossibles à soutenir pour une seule multinationale.

Ce qui nous amène au troisième et dernier point: Le prix.
Pour financer le développement de leur logiciel, les entreprises développant des solutions propriétaires se doivent de facturer un prix de licence particulièrement élevé aux clients utilisateurs de leur solution afin de maintenir leur seuil de rentabilité. OpenStack, de par son aspect Open Source est gratuit en termes d'utilisation, d'implémentation et de modification.

Il est en revanche important de nuancer ce dernier point; car si l'implémentation d'une plateforme OpenStack ne présente aucun coût de licence, il est crucial de disposer d'une main d'oeuvre qualifiée en la matière et disposant d'un nombre suffisant de jours/homme pour l'implémentation du projet. L'acquisition de compétences internes pouvant parfois être difficile et complexe, de nombreuses entreprises choisissent pour ce genre de projets de faire appel à des entreprises spécialisées dans l'intégration de cette technologie, qui sont aussi les principaux

acteurs du développement d'OpenStack (HP, IBM, RedHat), et c'est bien ici que se trouve pour eux tout l'intérêt de contribuer au projet.
Ainsi donc, même si l'on peut qualifier cette technologie de "gratuite" en comparaison à une autre technologie fonctionnant par le biais de licence, il est important de nuancer ces propos et de bien comprendre l'ampleur et le coût de l'implantation d'un tel projet.

Il est aussi important de souligner que si aujourd'hui encore le logiciel de cloud computing Vsphere est considéré comme plus mature et plus stable qu'OpenStack, il sera impossible pour VMware de continuer à dominer le marché aussi bien sur le plan de l'innovation que sur celui de la stabilité de leur solution. En effet, une équipe de développeurs embauchés par une seule et même entreprise ne peut concurrencer les dix-neuf mille deux cent vingt-six (19 226) personnes ayant contribué au projet à ce jour.
Vous trouverez en suivant ce lien la liste exhaustive des entreprises et des personnes contribuant au projet OpenStack: http://stackalytics.com/report/members

Les spécificités de mon travail sur le cloud d'analyse de données en production

L'une des principales spécificités de la solution apportée à l'amélioration de la scalabilité horizontale du cloud de virtualisation sur lequel a porté mon travail réside dans son fonctionnement nominal.
C'est cet aspect qui, d'un point de vue technique, différencies sa réalisation des autres solutions mise en oeuvre pour ce genre de problématiques. Et c'est donc aussi certainement le point le plus intéressant du travail effectué sur ce cloud de machines virtuelles.

Il est aujourd'hui usuel de travailler sur des problématiques de disponibilité et de scalabilité d'infrastructure ou la fonction nominale est d'assurer la continuité et la qualité de service en cas de panne. Mais j'ai pu travailler ici à une toute autre échelle, où le fonctionnement nominal des machines déployées est de se corrompre avec le temps et l'objectif du travail sur cet environnement de compenser en permanence le nombre de machines corrompues afin de maintenir la capacité d'analyse du cloud de machines.
C'est sur cette problématique bien définie que j'ai concentré mon travail, et tous les choix précédemment exposés vont dans ce sens.

Dans ce but j'ai été amené à redéfinir l'architecture du programme initialement développé pour gérer cette infrastructure. J'ai aussi changé son langage de programmation et supprimé la technologie sur laquelle il se reposait.
Ce sont ces modifications de fond qui permettent aujourd'hui de gérer un nombre de machines virtuelles très largement supérieur et de réduire la marge d'erreur de cette solution.

Les avantages et inconvénients de cette solution

La mise en place d'une plateforme de Coud Computing dans l'objectif de répondre à un besoin de souplesse et de scalabilité d'une infrastructure est un choix stratégique important, il est nécessaire de s'assurer que les coûts de mise en place de l'infrastructure ne soit pas supérieure au coût de la gestion du nombre de serveurs qu'elle virtualise.
Cependant, dans certaines conditions, l'utilisation de ce type de solutions peut s'avérer indispensable afin d'augmenter la réactivité du parc de machines à certains facteurs.
Par exemple, dans le cas du Cloud d'analyse de données de l'environnement de production, l'utilisation d'une telle solution s'est avérée indispensable à son fonctionnement. Il est en effet impossible de déployer des machines physiques tout en répondant aux besoins de volumétrie et de réactivité présentés ici. En effet, suivant son contexte d'utilisation il peut être nécessaire de supprimer et de redéployer la totalité des machines travaillant à l'analyse des données en le moins de temps de possible.

ANALYSE DE L'APPROCHE CHOISIE: UNE SOLUTION AJUSTABLE AU CHANGEMENT D'ECHELLE DE L'ENTREPRISE

Résultats

C'est après 3 mois de mise en place de la solution que les développeurs ont commencé à migrer leurs environnements de développement et de test vers les nouveaux environnements virtualisés. J'ai continué à travailler sur ce projet les quatre mois suivants afin d'obtenir la plateforme la plus stable et performante possible.
Ainsi il est devenu possible pour les développeurs de travailler sur un environnement spécifique à une RoadMap, de dupliquer cet environnement pour les besoins de l'équipe de test et de continuer leur travail sur le premier. L'implémentation de cette plateforme aura aussi permis au développeur de sauvegarder l'état actuel des environnements liés à leur développement à l'aide de Snap-shots incrémentaux et ainsi revenir à un précédent état sauvegardé en cas de besoin.

Mais l'un des point les plus important du résultat de ce projet, et c'est aussi cet aspect précis qui répond à la problématique, est l'implémentation d'une forme simple de scalabilité au sein de l'architecture afin de répondre à des besoins temporaires. La possibilité de créer un nouvel environnement de développement en quelques clics aura permis aux développeurs d'augmenter leur productivité et d'éviter de longues heures de suppression et de redéploiement de leur code lors du passage d'un contexte de développement à un autre.

Cependant, un important aspect négatif de cette technologie remet en cause son implémentation au sein de la société, à savoir, le coût et la complexité de la maintenance du projet au quotidien, qui avec une nouvelle version tous les 6 mois nécessite le travail d'une personne à temps plein dans son exploitation.

Dans un autre contexte, mais toujours dans un dynamique d'amélioration de la scalabilité des infrastructures, mon travail sur le cloud d'analyse de données de la production aura permis une forte augmentation de la quantité des données traitées sur un temps donné, ce qui amène à plus de liberté vis a vis du client final dans son nombre d'informations remontées, mais aussi vis à vis de l'entreprise concernant le nombre de clients lui-même.

D’autre part, la diminution considérable des données corrompues remontées par le pool de machines aura permis d’augmenter la qualité finale du produit en évitant que le client ne perde des informations cruciales.
Dans une autre mesure, l’augmentation de la capacité générale de traitement de cet ensemble de machines virtuelles aura été nécessaire dans la mise en place d’une nouvelle fonctionnalité développée par l'équipe travaillant sur le logiciel “Digimind Social”, à savoir la possibilité d’initialiser un compte lors de des débuts du client sur la plateforme remontant des données passées. En effet, la grande quantité d'informations à traiter à un instant donné provoquait une surcharge de requête qu’il est aujourd’hui possible de gérer.

Analyse du champ d’application

Les champs d’application d’un cloud de virtualisation dans une société sont multiples, ils peuvent, sous certaines conditions, remplacer n'importe qu'elle infrastructure d'une société. OpenStack, qui a été l'une des solutions choisie dans ce projet, s'inscrit justement dans une logique d'adaptabilité et d'interopérabilité avec la plupart des composants dominant le marché des infrastructures techniques d'entreprise.

Dans le contexte plus précis des environnements de développement de la société, cette plateforme de Cloud Computing composée des différentes solutions choisies (OpenStack, Ceph, OpenVSwitch, etc.) remplace la totalité de la précédente infrastructure en délivrant à la fois un réseau et ces différents composants, des systèmes d'exploitations, un système de stockage distribué ainsi que l'ensemble du système d'authentification.
Ainsi, en maitrisant chacun des différents aspects de cette plateforme, il est parfaitement possible d'appliquer son fonctionnement à d'autres environnements de la société telles que la production et de remplacer ainsi la plupart des infrastructures délivrées par une entreprise.

Mise en perspective dans d’autres contextes

Après plusieurs mois d’utilisation d’OpenStack je peux déduire que l'implémentation d’une plateforme de ce type prendrait un sens d'autant plus important lors de son implémentation dans d’autres contextes nécessitants de plus importants besoins volumétriques, notamment en nombre de machines virtualisées et de machines physiques.
Il peut être lourd de maintenir une telle solution, et c'est principalement pour cette raison qu'aujourd'hui la grande majorité des implémentations de ce type de solution sont destinées à

devenir des plateformes de Cloud Computing public afin de louer à des entreprises tierces les ressources mises en place. C'est notamment le cas de la société OVH qui après un très lourd travail interne d'implémentation d'OpenStack loue les ressources de sa plateforme par le biais de sa solution d'hébergement pour les sociétés.

Ce n'est pourtant pas pour autant que l'implémentation de ce type de solution en interne dans une société n'est pas pertinent. En effet, beaucoup d'entreprises et notamment les grands groupes nécessitent des infrastructures très réactives en terme de scalabilité mais aussi en terme de volumétrie de ressources matérielles. C'est dans ce cas d'application que l'implémentation d'une plateforme de la sorte prend tout son sens et permet aux entreprises de mieux maîtriser les coûts de gestion et d'administration de leur parc informatique

REFLEXION SUR LE STAGE ET LE MEMOIRE: DES PERSPECTIVES REDEFINIES

Évaluation personnelle

Lors de ce contrat d'alternance, les membres de la société et ma hiérarchie m'ont beaucoup laissé la liberté d'entreprendre et d'innover avec de nouvelles solutions, ce qui m'a permis d'expérimenter un très grand nombre de technologies que je n'avais jamais abordées jusqu'ici et de réfléchir à de nouvelles problématiques que je n'avais jusqu'à ce contrat de professionnalisation encore jamais approchées. Je peux notamment citer l'aspect réseau qui a sans aucun doute été le plus complexe à acquérir dans ce projet.

La mise en place d'une plateforme de virtualisation de type Cloud au sein de Digimind a été un projet extrêmement vaste et complexe. J'ai malgré tout eu la chance d'être accompagné le premier mois par mon prédécesseur sur le projet, qui a su m'aider dans un grand nombre de décisions à prendre.

N'ayant auparavant jamais eu l'occasion de travailler dans une entreprise éditrice de logiciels à une telle échelle je n'avais encore jamais eu à gérer de problématiques de scalabilité aussi avancées. Ces questions spécifiques aux entreprises disposant d'infrastructures conséquentes me passionnent pourtant depuis la première fois que je m'y suis intéressé lors des projets de fin d'année de ma troisième année à Supinfo. Ce contrat de professionnalisation a donc été pour moi l'occasion idéale de mettre en pratique les nombreux essais menés à bien ces dernières années.

Bilan des acquis

Connaissances acquises lors de l'implémentation d'une plateforme de Cloud Computing OpenStack pour les environnements de développement

Les connaissances à acquérir lors de la mise en place d'un cluster de virtualisation open source tel qu'OpenStack sont très vastes et s'étendent dans de nombreux domaines.
Qu'il s'agisse de la programmation Python (qui est le langage constituant 99% du code source d'OpenStack), les architectures réseau virtualisées ou encore les modules kernels destinés à la

virtualisation et l'isolation des ressources, ce sont autant de points que je n'avais encore jamais étudiés lors de mes anciens stages en tant qu'administrateur système ou en cours.

Ce contrat de professionnalisation a été extrêmement formateur, tant sur le plan technique que sur le plan stratégique et managérial, dans le sens où j'ai pu travailler pour la première fois dans un grand groupe ayant comme corps de métier l'informatique et étant constitué essentiellement d'ingénieurs en informatique.

Connaissances acquises lors de mes travaux sur le cloud d'analyse de données de l'environnement de production de la société

L'un des aspects les plus importants des connaissances que j'ai acquises lors de mes travaux sur ce projet est sans aucun doute la programmation en langage Python.
En effet, même si ce langage n'est pas nouveau pour moi, ce projet a été la première occasion de l'utiliser dans un contexte professionnel et de sortir ainsi des tâches classiques d'un administrateur système pour me rapprocher du travail de développeur.
J'ai aussi appris à maîtriser et implémenter les différents aspects d'une infrastructure de plus de trois-cent machines virtuelles aillant la même fonction. J'ai pu travailler sur des problématiques de répartition de charge, de haute disponibilité, et de réactivité de la scalabilité de l'infrastructure.

Perspectives professionnelles en relation avec les compétences acquises

Lors de ce contrat de professionnalisation chez Digimind, j'ai pu mettre en application une technologie qui me passionne, mais plus important encore, j'ai eu l'occasion de rencontrer la communauté constituant les utilisateurs, intégrateurs et développeurs de cette solution. Notamment lors des "MeetUp" souvent organisés par Sylvain Bauza, travaillant avant moi chez Digimind et qui contribue aujourd'hui au développement d'OpenStack comme employé chez RedHat.
Mon souhait est aujourd'hui de continuer à travailler sur cette technologie, même si Digimind a aujourd'hui arrêté son utilisation.

A l'issue de ce contrat de professionnalisation mes perspectives professionnelles se sont très largement précisées, à savoir: Je souhaite pouvoir accompagner les entreprises dans

l'évolution de leur infrastructure en réponse à des besoins croissants et un nombre de clients grandissant.
Aujourd'hui, énormément d'entreprises disposent ou délivrent des solutions SaaS (Software as a Service): elles commencent petites avec un ou deux serveurs et seront forcément confrontées à ce problème à un moment donné.
L'objectif est dans un premier temps d'acquérir davantage d'expérience en tant qu'employé dans un groupe participant au développement d'OpenStack. Puis, dans un second temps, de proposer à terme une solution facilitant et automatisant cette phase de changement pour les entreprises.
Ce sont les deux projets que j'ai présentés dans ce mémoire qui ont été déterminants dans la définition de mes perspectives professionnelles.

L'implémentation d'une plateforme OpenStack au sein de Digimind m'a très largement conforté dans l'idée de me spécialiser dans cette technologie.
Quant au projet d'automatisation de la scalabilité du cloud d'analyse de données, il m'a fait redécouvrir un autre aspect de l'informatique qu'est la programmation en langage Python et le travail de développeur toujours en rapport avec des problématiques de scalabilité des infrastructures, de loadbalancing et de haute disponibilité.

C'est donc après avoir travaillé sur ces deux projets que j'ai pris la décision de travailler en tant que développeur de la solution OpenStack, ce qui me permettra d'exploiter les connaissances que j'ai acquises sur OpenStack et en développement Python mais aussi de mettre à profit mes précédentes expériences en tant d'administrateur en systèmes Linux et Unix.
Cette décision de perspective professionnelle est une nouvelle orientation que je prendrai le 17 novembre chez Hewlett Packard en Irlande.

CONCLUSION

Afin de conclure cette dissertation présentant mon travail au sein de Digimind, je souhaite avant tout rappeler l'intérêt de cette problématique pour les entreprises en pleine expansion.
Car l'objectif et la volonté que poursuivent chaque entrepreneur et chaque comité de direction à la tête d'une société sont l'expansion de celle-ci, qu'elle gagne en notoriété comme en nombre de clients, et en bénéfices dégagés autant qu'en services et innovations apportés aux clients.
Cette transition d'échelle est une étape capitale de l'accélération économique de la société et comme les autres aspects liés à ce changement, il est primordial que l'infrastructure technologique soit elle aussi capable de supporter tous les éléments de cette transition d'échelle.

Dans l'informatique tout est une question d'échelle, et c'est cette même notion qui a permis à l'informatique de gagner une si forte notoriété en si peu de temps.
Il y a une image, initialement employée par Jeff Bezos, l'un des fondateurs d'Amazon, sur laquelle j'aime m'appuyer pour illustrer ce propos:
Imaginez que vous ne sachiez faire qu'un ensemble de tâches simples, diriger vos bras et vos jambes vers le haut, le bas, la gauche, et ainsi de suite.
Maintenant imaginez que vous puissiez effectuer ces tâches 100 000 fois plus rapidement que n'importe quel autre personne: Vous pourriez alors, en une fraction de seconde, vous lever, prendre votre ordinateur, le déposer dans une autre pièce et revenir à votre place. Et du point de vue d'une autre personne, vous auriez fait disparaître votre ordinateur par ce que relativement à sa perception, vous auriez été bien trop rapide.
C'est exactement ce que font les ordinateurs, en effectuant des taches très simples à une échelle de vitesse si éloignée de la notre, ils peuvent apparaître comme magiques. (J Bezos, The Playboy interview)

Nous avons dans un premier temps de l'histoire de l'informatique travaillé à l'augmentation de cette différence d'échelle. Dans une optique de scalabilité verticale, nous avons rendu les ordinateurs toujours plus rapides afin d'accentuer cet effet.
Aujourd'hui, nous commençons à peine le travail de scalabilité horizontale et de changement d'échelle de l'outil informatique, non plus de façon verticale en augmentant les performances d'une machine unique, mais de façon horizontale en démultipliant leur nombre dans un fonctionnement commun.

Dans la réponse à cette problématique de scalabilité des infrastructures, il a été démontré que l'évolution des technologies de Cloud Computing joue un rôle essentiel et moteur car cette technologie permet d'adapter la taille de l'infrastructure d'une entreprise au nombre de ses clients et aux ressources matérielles nécessaires à l'évolution de l'application fournie. Mais, d'une manière sous-jacente, ce que démontre aussi cette solution, c'est qu'il est aujourd'hui possible de remplacer les équipements réseaux et les systèmes d'exploitation d'une infrastructure complexe par de simples programmes reconstituant leur fonctionnement initial.

Demain, ces programmes seront des librairies directement intégrées aux logiciels développés et nous nous dirigeons donc vers des infrastructures entièrement logicielles et directement intégrées aux produits finaux. [14]

Et dans la mesure où nous n'en sommes aujourd'hui qu'aux balbutiements de ces avancées, il est important d'imaginer et de garder en tête la dimension qu'elles peuvent prendre: La scalabilité globale, distribuée, et à l'échelle mondiale des infrastructures techniques d'une société par le biais de ses produits développés.

[14] Un très grand nombre de technologies telles que ZeroMQ, Etcd ou encore node-http-proxy existent déjà dans cette optique et sont d'ores et déjà employées par de nombreuses sociétés: DotCloud, Heroku.

BIBLIOGRAPHIE ET WEBOGRAPHIE

Jeff Bezos, Steve Jobs, Lee Lacocca, Bill Gates, David Geffen, Malcolm Forbes, Ted Turner. (2012*). The Playboy Interview: Moguls*. (J. Bezos, Intervieweur)

Quentin Hardy. (Novembre 2011). *Google: Scale Changes Everything*. Forbes Magazine.

Gartner. (2008). *Mastering The Hype Cycle.* Harvard Business Press.

Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Antony Rowstron. (s.d.). *SCRIBE : A large-scale and decentralized application-level multicast infrastructure*. Disponible sur http://research.microsoft.com/pubs/65210/jsac.pdf (Initialement consulté en Février 2013)

David Coudol, Stéphane Gros. (2009). *La microéconomie est une branche de la science*. Disponible sur http://www.veille-intelligence-economique.fr (Site actuellement hors ligne)

Bernard Carayon, Jean-Pierre Raffarin. (Juillet 2003). *Intelligence économique, compétitivité et cohésion sociale*. Disponible sur http://www.ladocumentationfrancaise.fr/var/storage/rapports-publics/034000484/0000.pdf

OpenStack.org. (s.d.). *OpenStack Project Documentation*. Disponible sur http://docs.openstack.org/ (Initialement consulté en Juillet 2013)

OpenStack.org. (2014). *Top 10 Automotive Manufacturer Makes the Business Case for OpenStack*. Disponible sur http://www.openstack.org/assets/pdf-downloads/openstack-automotive-case-study.pdf (Initialement consulté en Octobre 2013)

Nate Silver. (2012). *The Signal and the Noise: The Art and Science of Prediction.* Allen Lane.

Sam Alba. (2013, Jan 13). *PaaS under the hood, episode 5: Distributed routing with Hipache*. Disponible sur http://blog.dotcloud.com/under-the-hood-dotcloud-http-routing-layer (Initialement consulté en Mai 2014)

Jérome Petazzoni. (28 Novembre 2012). *PaaS under the hood, episode 1: kernel namespaces*. Disponible sur http://blog.dotcloud.com/under-the-hood-linux-kernels-on-dotcloud-part (Initialement consulté en Mai 2014)

Jérome Petazzoni. (5 Décembre 2012). *PaaS Under the Hood, Episode 2: cgroups*. Disponible sur http://blog.dotcloud.com/kernel-secrets-from-the-paas-garage-part-24-c (Initialement consulté en Mai 2014)

RedHat.org. (Octobre 2013). Distributed Storage Performance For Openstack Clouds: Red Hat Storage Server Vs. Ceph Storage. Disponible sur http://www.principledtechnologies.com/Red%20Hat/RedHatStorage_Ceph_1113.pdf (Initialement consulté en Octobre 2013)

Mark Nelson. (9 Octobre 2013).Ceph Performance Part 1: Disk Controller Write Throughput. Disponible sur http://ceph.com/community/ceph-performance-part-1-disk-controller-write-throughput/ (Initialement consulté en Octobre 2013)

ANNEXES

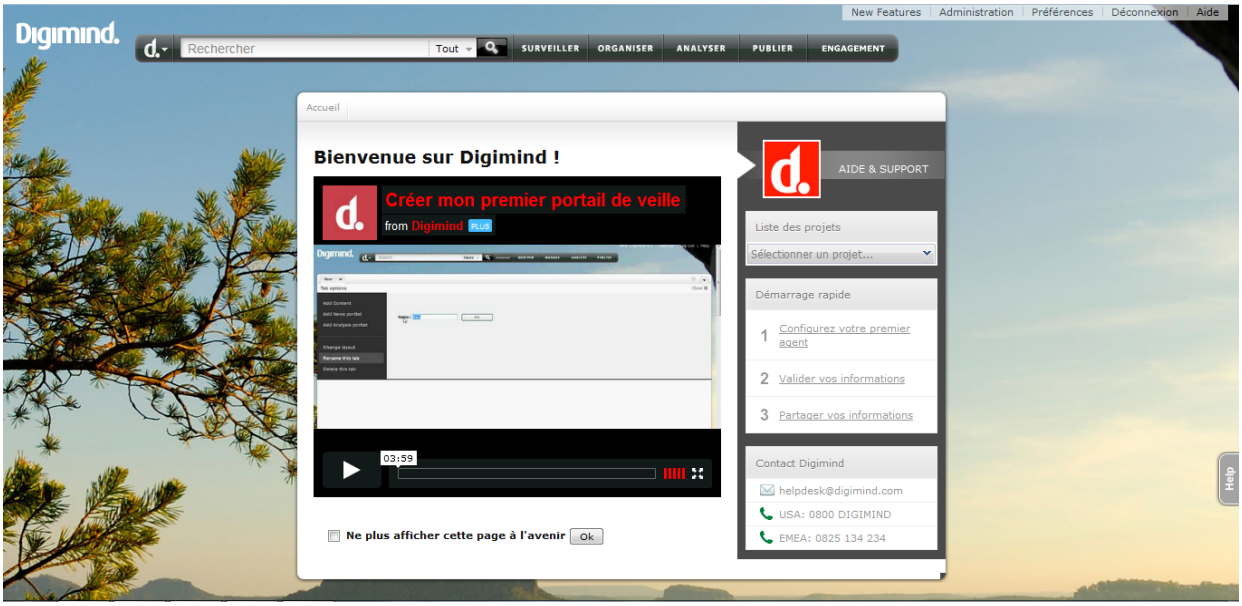

Figure 6: Page d'accueil de l'application "Digimind Intelligence"

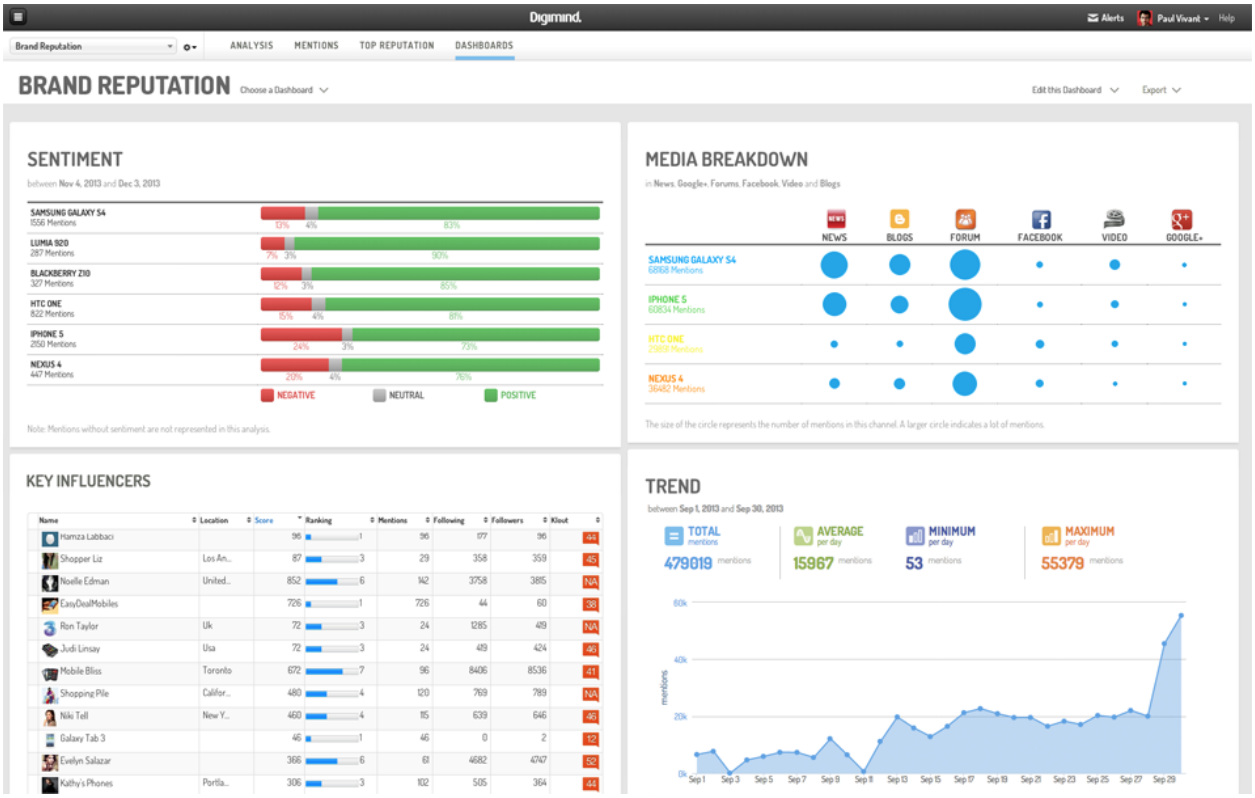

Figure 7: Page d'accueil de l'application "Digimind Social"

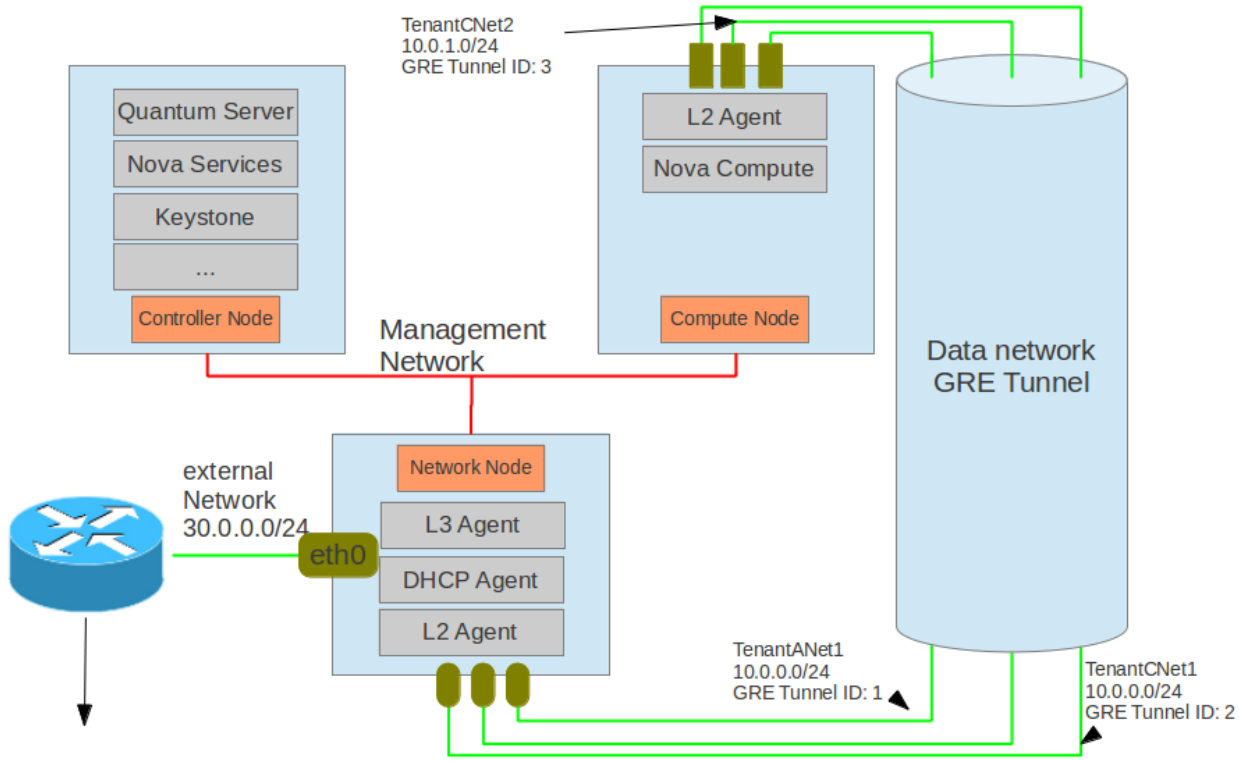

Figure 8: Schématisation de l'utilisation de tunnels GRE dans l'implémentation d'un réseau pour un cloud de virtualisation privé OpenStack.

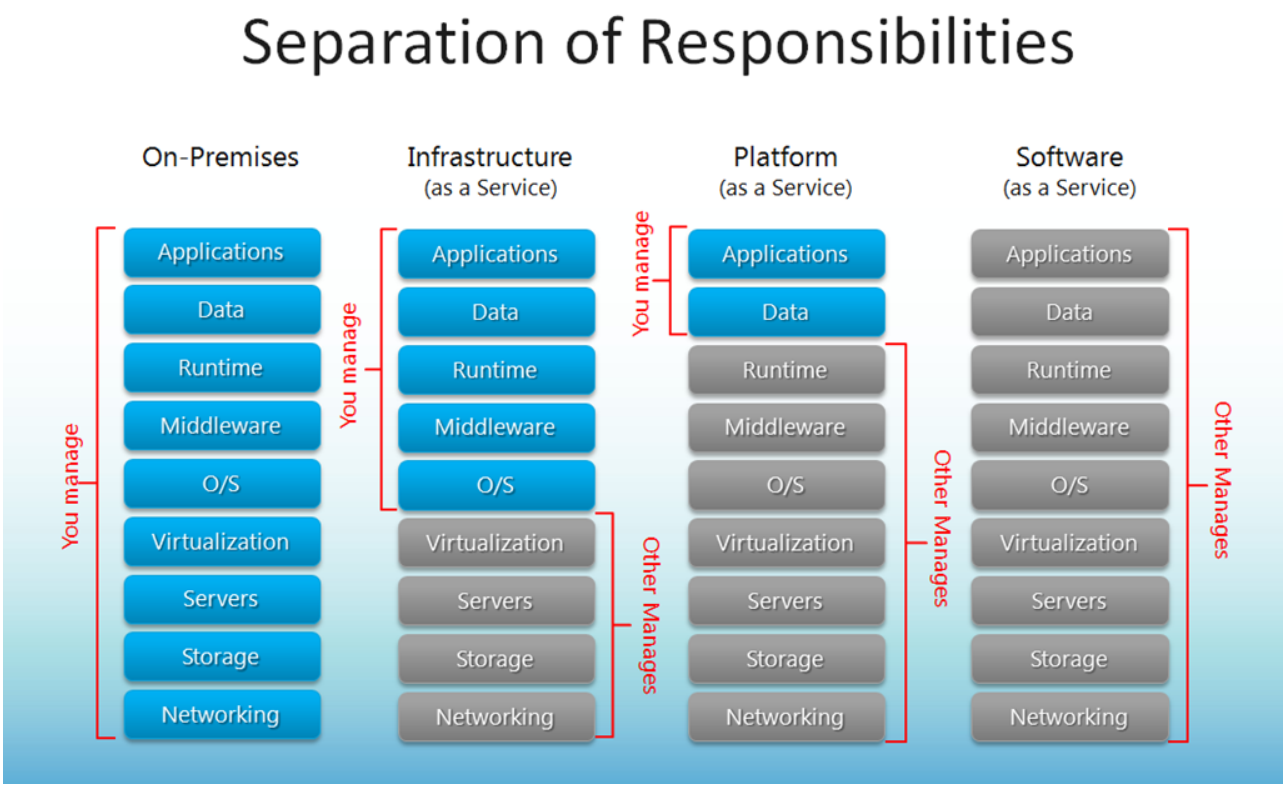

Figure 9 : Visualisation de la répartition des responsabilités dans les différents cas d'externalisation (IaaS, Paas, SaaS)

Ce schéma résume les différents modèles de services en Cloud Computing :

- «On Premise » est le modèle d'administration classique où la totalité de l'infrastructure et de l'applicatif sont administré par le client.
- Le modèle « Infrastructure as a Service » (IaaS) fournit aux clients des machines et leurs réseaux. Plutôt que d'avoir à acheter du matériel (serveurs, équipement réseau), les utilisateurs peuvent acheter une solution IaaS en étant facturés en fonction de leur consommation.
  Exemples: Amazon Web Service (AWS), Microsoft Azure, Google Compute Engine (GCE)
- Le modèle « Plateform as a Service » (PaaS) fournit en plus de tout ça le système d'exploitation sur lequel le client n'aura plus qu'à installer ses applications.
  Exemples: Apprenda
- Le dernier modèle « Software as a Service » fournit directement l'application au client qui n'aura plus qu'à l'utiliser.
  Exemples: Google Gmail, Microsoft 365, Salesforce