République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur
Et de la Recherche Scientifique
Université SAAD DAHLEB BLIDA
Faculté des sciences
Département d'Informatique

Mémoire de fin d'étude pour l'obtention du diplôme de Master en Informatique

Option : Ingénierie du Logiciel

CloudMoodle

Une plateforme de type Cloud pour un centre d'examen en ligne basée sur la plateforme Moodle.

Réalisés Par :
- Belkhiri Abdellatif
- Ferhati Adel

Supervisé Par
- Dr. Djamal Bennouar

Président de jury : Dr. Chrif Zahar
Examinateur : M. Sudommou Redha
Examinateur : M. Baouia Abdelhakim
Promoteur : Dr. Djamal Bennouar

2011/2012

Table des matières

Dédicaces

Je dédie ce modeste travail :
Aux deux personnes les plus chères à mes yeux, à ma mère et mon père
Qui ont tout sacrifiés pour leurs enfants, qui ont veillés à notre éducation, qui, sans eux après ALLAH je ne serai pas ce que je suis.

A tous mes sœur, mes frère : Rachid, Mounir, Mourad et leurs femmes. A Khaled, Chaker, Salah Eddine, Adam, Yakoub .
A la fille la plus chère dans le monde
TOUTOU-HANDELLALI
A mon binôme Adel et toute sa famille
A tous mes amis de Tébessa, et mes collègues de l'université, avec qui j'ai partagé de bons moments.
A toutes les personnes que je connaisse et que je n'ai pas citées.
Qu'Allah accepte notre travail. Inchaa Allah.

Abdellatif

Remerciement

Nous tenons à remercier ALLAH qui nous a donné la force de faire ce modeste travail,

Quelques lignes ne pourront jamais exprimer la reconnaissance que nous éprouvons envers tous, ceux qui, de près ou de loin, ont contribué, par leurs conseils, leurs encouragements et leurs aides à l'aboutissement de ce travail.

Nos vifs remerciements accompagnés de toute nos gratitude vont d'abord à notre promoteur monsieur ***DR.DJAMAL BENNOUAR*** pour premièrement nous avoir proposé ce thème et pour les conseils qu'il n'a cessé de nous prodiguer, son aide et surtout la confiance qu'il a mise en nous pour la réalisation de ce travail. *Merci Monsieur.*

Nos remerciements vont aussi à Nos parents, Nos grands mercis vont aussi à Nos amis : pour Bilal Berkani, Bilal Gridi, Hamza bouchrit.

Nous citerions plus particulièrement tous les enseignants de département d'informatique de l'université de BLIDA.

Nous ne saurions aussi oublier nos amis et nos collègues de loin et de prés.

Résumé

Ce travail met en œuvre le concept de Cloud Computing pour réaliser une infrastructure de elearning hautement disponible et pouvant facilement passer à l'échelle désirée. Cette infrastructure appelée *CloudMoodle* est basée sur la plateforme de télé-enseignement Moodle. L'architecture de base de CloudMoodle, comprends un serveur frontal, un ensemble de serveur Moodle de Base et une base de données commune à tous les serveurs Moodle. Les divers serveurs Moodle se partagent la charge de traitement des requêtes émanent des clients. CloudMoodle supporte plusieurs politiques d'équilibrage de charge qu'il est possible de sélectionner dans une opération d'administration du Cloud pour mieux répondre à la catégorie des clients et à leur nombre.

Mots clé : *cloud computing, client léger, loadbalacing, télé-enseignement, repartions des charges, e-learning.*

Abstract

This work implements the concept of cloud computing to achieve a highly available infrastructure and elearning can easily switch to the desired scale. This infrastructure is called CloudMoodle based distance learning platform Moodle. The basic architecture of CloudMoodle, includes a front-end server, a set of Basic Moodle server and a database common to all servers Moodle. The various Moodle servers share the processing load of requests emanating from clients. CloudMoodle supports several load balancing policies that can be selected in an operation administration Cloud to better meet the category of customer and their number.

Keywords : *cloud computing, client léger, loadbalacing, télé-enseignement, repartions des charges, e-learning.*

Introduction et Motivation

Introduction et Motivation

Avec l'avènement d'outils de plus en plus puissant pour le e-learning, nous sommes en droit de penser que dans quelques années ces technologies d'aide à l'enseignement seront utilisées de plus en plus. L'utilisation actuelle de ces plateformes pour une examinassions en ligne concerne souvent un nombre réduit d'étudiants. La généralisation de l'examen en ligne nous fait penser à la possibilité de la mise sur pied futur de centre d'examens en ligne. Dans un tel centre d'examens, un nombre très important d'étudiants compté par centaines, voire par millier peuvent passer un ou plusieurs examens simultanément.

Si nous considérons le cas d'une plateforme assez complète comme Moodle, utiliser un seul serveur pour supporter l'interaction avec un grand nombre d'apprenants et d'examinés ne pourrait être possible. Déjà, nous avons observé que le temps de réponse pour une vingtaine d'examiné est déjà assez important, notamment lorsque Moodle active la technologie Ajax qui réalise un trafic intense entre client HTTP et serveur HTTP en background sans que l'usager ne s'en rende compte **[BEN, 12]**. L'utilisation directe de plusieurs serveurs nécessiterait une gestion très importante et complexe des enseignants et des étudiants.

Cette gestion consiste au minimum à designer pour chaque enseignant le serveur sur lequel il doit travailler et y mettre ses cours, Un enseignant intervenant dans des départements différents pourrait se voir affecter à plus d'un serveur. Pour les étudiants c'est le même scénario qui se répète, Ils doivent avoir un compte sur chaque serveur contenant les cours qui les intéressent. Dans ce contexte, l'étudiant devra à chaque fois de déplacer d'un serveur à un autre, de même pour l'enseignant.

La solution préconisée consiste à détacher les clients de l'emplacement exact de leurs ressources, Ces clients doivent faire face toujours à un seul serveur. Ce serveur cacherait en fait une batterie de serveur dotée chacun d'un service de téléenseignement. Si nous considérons une plateforme eX, l'utilisateur sera toujours en face d'un serveur frontal qui fournirait tous les services de eX et doit se comporter exactement comme eX du point de vue de l'utilisateur. Le serveur frontal est en fait un serveur eX virtuel. Le client s'adresse à une plateforme eX viruelle et réellement

il ne sait pas que ce n'est pas ce serveur virtuel qui traite les requêtes. Ce sera un des serveurs eX que la virtualisation a cachée et a caché la réalité de l'architecture qui prend en compte les services eX. Cette virtualisation est appelée actuellement dans le monde du Business et le mondes des services le Cloud Computing.

L'intérêt d'une approche de virtualisation, ou du cloud si on utilise ce terme qui est actuellement fortement propulsé par les entreprises de services, est de cacher l'architecture réelle du système prenant en charge le elearning. Dans ce contexte le passage à l'échelle se fera sans impact sur l'utilisateur. L'utilisateur ne verra en fait qu'un gain de performance. Dans ce contexte Il serait possible d'ajouter autant de serveur eX réel que nécessite le centre d'examen. Ceci ne changera en rien les pratiques des apprenants et enseignants. Le passage à l'échelle pourra même toucher le serveur frontal lui-même. Le rôle principal de ce serveur sera l'équilibrage des charges sur les divers serveurs réel.

Objectif du travail

L'objectif principal de ce travail est de proposer une architecture d'un cloud pour le e-learning. Cette architecture devra être mise à l'échelle de manière très simple. Ainsi l'ajout d'un serveur réel de téléenseignement devra être complètement transparent au client et devra se faire de manière très souple sans remise en cause de l'architecture en place et de toute l'activité qui aurait été menée.

L'architecture devra permettre au cloud une résistance forte aux pannes. Ainsi si un serveur réel tombe en panne ou si le serveur frontal tombe en panne, le Cloud doit toujours rester disponible et la prise en charge des pannes devra se faire de manière transparente sans aucun impact sur le travail du cloud.

Le Cloud à concevoir et réaliser devra supporter les 2 approches de déploiement : privé et communautaire. Ainsi, à titre d'exemple, pour réaliser des examens, le Cloud devra selon les besoins être configuré pour intégrer les ressources de plusieurs départements, ou de plusieurs facultés, ou de plusieurs Université, de manière absolument transparente aux apprenants des divers départements, facultés et Universités

Pour une première conception et réalisation du cloud pour le e-learning, et pour des raisons pratiques, le cloud sera basée uniquement sur une seule plateforme de e-learning à déterminer dans le contexte d'une étude comparative. Une étude approfondie de la plateforme choisie et une étude très fine de son interaction avec les apprenants et les enseignants est nécessaire. La conception du Cloud à réaliser ne devra pas être fermée. Elle devrait être ouverte pour le support d'autres plateformes de téléenseignement. Le Cloud pour le e-learning devra dans sa forme finale proposer plusieurs plateformes de télé-enseignement. Le choix de l'un ou l'autre dépendra des exigences des enseignants et des étudiants.

Parmi les fonctionnalités du Cloud, représenté du point de vue de l'utilisateur par un serveur frontal, il y a l'équilibrage des diverses charges affectées aux divers serveurs réels Une étude des principes de l'équilibrage des charges et des politiques adoptées est ainsi nécessaire, pour permettre le choix d'une ou de plusieurs politique à mettre en œuvre lors de la configuration du Cloud pour le e-learning.

Organisation du mémoire

Ce mémoire est divisé en deux parties :

Une **première partie** composée de quatre chapitres, elle se focalise sur les importantes notions qu'on va utiliser dans notre travail. Dans le premier chapitre, nous présenterons des généralités sur e-learning et les plateformes. Le deuxième chapitre sera une étude comparative entre les principales plateformes e-learning afin de choisir la meilleure plateforme. Le troisième chapitre représente le cloud computing, ses caractéristiques, ses types et ses différents services. Le chapitre 4 présentera les notions fondamentales de l'équilibrage de charge, notamment dans le contexte de l'accès aux serveurs

Une **deuxième partie** composée de deux chapitres, chapitre cinq pour la conception, dans cette dernière on se présente les solutions qu'on va proposer. Sixième chapitre pour l'implémentation de la solution proposée et on détaille les différentes étapes d'installations, configurations de plateforme moodle et problèmes rencontrées.

Chapitre 1

E-learning

CHAPITRE 1:E-learning

Introduction

Actuellement, le monde connait une grande révolution technologique qui a fait un grand changement dans plusieurs domaines tel que l'enseignement .l'utilisation de l'informatique dans ce dernier a donné naissance à une nouvelle forme de formation qui s'appelle l'e-learning. Ce mode d'apprentissage basé sur les services offerts par le web, connait une évolution importante. Il permet aux apprenants de suivre leurs formations indépendamment du lieu et du temps et aux développeurs de créer des plateformes technologiques pour la création des cours, le suivi des utilisateurs, les communications synchrone et asynchrone.

1.Généralités sur l' e-learning

1.1.Terminologie

Le terme e-learning est un terme en anglais à plusieurs termes équivalents : Apprentissage en ligne, Apprentissage virtuel, Apprentissage électronique, E-formation, Formation à distance, Enseignement A Distance (EAD), Formation Ouverte et A Distance (FOAD), Formation Ouverte.

1.2.Définition

Le mot "**e-learning** " en anglais (**"e-formation" en français)** est composé de :

1) le préfixe 'e' qui signifie « électronique »qui indique qu'un dispositif électronique est utilisé dans la formation, et après apparition de l'internet et les réseaux informatique « e » Désigne « en ligne ».
2) et le mot formation qui désigne une relation entre enseignant et l'étudiant basée sur L'échange des connaissances et des informations.

Le '' **e-learning** '' est une méthode de formation qui utilise un réseau local, étendue ou internet pour diffuser, interagir ou communiquer en environnement distribué et accédé, à des sources par téléchargement ou en consultation sur le net. Il peut faire intervenir du **synchrone** (service de chat, téléconférence,…) ou de

l'asynchrone (tests effectués par les apprenants, messagerie,…), des systèmes de tutorats, des systèmes à base d'autoformation, ou une combinaison des éléments évoqués. Le e-learning dépend donc de l'association de contenus **interactifs** et multimédia, de supports de distribution (PC, internet, intranet ,extranet), d'un ensemble d'outils logiciel qui permettent la gestion d'une formation en ligne et d'outils de création de formation **interactives.** En d'autre terme le e-learning est l'utilisation des nouvelles technologies multimédias et de l'Internet, pour améliorer la qualité de l'apprentissage en facilitant l'accès à des ressources et des services, ainsi que les échanges et la collaboration à distance. **[OUB 03]**

D'après la **commission européenne 2008**: « la e-formation consiste à utiliser les nouvelles technologies multimédia et l'Internet, pour améliorer la qualité de l'apprentissage en facilitant l'accès à des ressources et des services, ainsi que les échanges et la collaboration à distance »

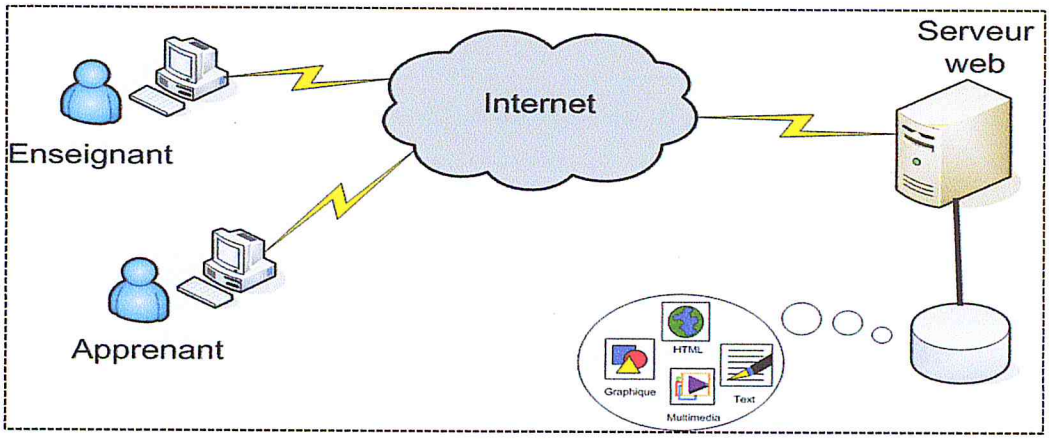

Figure 1 : Apprentissage à distance **[MIE 05].**

1.3.Qui utilise l' e-learning ?

L'e-learning est un domaine qui a pris une grande importance dans la vie moderne, il est appliqué dans deux secteurs :

- Les grandes entreprises pour former leurs employés plus rapidement.
- Les universités pour former leurs étudiants et faciliter l'apprentissage.

1.4.Les modes de communication dans l' e-learning

L'enseignement en ligne peut passer par trois modes de communication qui sont cité si dessous

1.4.1.L'enseignement par communication synchrone

La synchronisation entre enseignant et apprenant suppose que l'enseignant et l'élève soient mobilisés en même temps, l'un pour la transmission et l'autre pour l'acquisition, ou pour des échanges. **[BOU 05]**

Ce type de formation nécessite une connexion simultanée des participants à une session de formation (temps réel), l'échange de la formation est réalisé par : Web conférence, Visioconférence, Le chat, Téléphone.

1.4.2.L'enseignement par communication Asynchrone

Cette formule recouvre les cours de formation multimédia que l'apprenant peut utiliser de manière autonome, qu'ils soient en ligne, en local ou un mixte des deux, cette formule qui donne le plus de souplesse a l'apprenant.**[BOD 05]**

Ce type de formation ne nécessite pas une connexion simultanée. L'échange de la formation entre les apprenants se fait par : Forum de discussion, Email, Messagerie instantané, Des documents diffusés sur le web .Ce mode de communication est le plus utilisé car il élimine la contrainte de temps.

1.4.3.L'enseignement mixte blended Learning

Le blended e-learning en anglais ou bien l'enseignement multimodales qui est une combinaison des deux modes précédent (synchrone et asynchrone) ou l'apprenant consulte les cours déposés par l'enseignent et poser leur question par le chat, forum, ou par email.

1.5.Pour quoi l'e-learning ?

E-learning a une grande importance dans les universités et les entreprises car la formation traditionnelle à poser beaucoup de problèmes entre autres:

- Insuffisance de formateurs spécialisés.
- Manque de salles de formation, qui parfois sont très chargées ce qu'influence sur la qualité de la formation.
- L'absence du formateur reporte la formation.
- La constitution des groupes homogènes est difficile car les niveaux de besoin de formation varient d'un étudiant à un autre.
- Perte de temps.
- Les frais de déplacement et d'hébergement sont très élevés.

Le e-learning a donné une solution à la plus part des problèmes posés précédemment.

	Mode présentiel	**E-learning**
Contenue pédagogique	Utilise le support papier, qui doit satisfaire les besoins de l'ensemble des apprenants.	Les documents numériques Satisfont chaque apprenant.
Cout	Salle de formation/Equipement de projection/Salaires des formateurs, des participants…etc. Cout liée au support de formation (manuel)/ Frais de déplacement et hébergement.	Les couts sont liés seulement aux équipements informatiques.
Le déplacement	Important avec la perte de temps.	Pas de déplacement donc pas de perte de temps.
La rapidité	Moins rapide	Plus rapide.
Type d'enseignement	Collective.	Individuel ou bien collective.
Qualité de formation	Bien.	Très Bien.
Les ressources Pédagogiques	Les livres, internet, les bibliothèques.	partage, réutilisation et échange des unités existantes.
Mode de communication	Synchrone	Synchrone, asynchrone, mixte

Tableau 1 : *Comparaison entre le mode présentiel et l' e-learning.*

D'après le tableau précèdent, l'e-learning est le meilleur choix pour améliorer la qualité de formation et pour l'enseignement, il devient plus efficace car il permet de :

- Proposer une formation homogène et personnalisable.
- Offrir une formation juste à temps, chaque étudiant peut choisir son temps de formation
- Eliminer les frais de déplacement et d'hébergement.
- Donner un rythme d'apprentissage libre.
- Une formation sur le lieu de travail.
- Une formation plus universelle tout le monde peut participer.**[BOD 05]**

1.6.Les avantages

L'apprentissage en ligne présente beaucoup d'avantage pour apprenant et pour le formateur

1.6.1.Pour l'apprenant

- L'apparent a le droit de choisir les orientations, les conditions de son apprentissage **(le lieu, le temps)** et déterminer son propre agenda de déroulement des cours à apprendre dans sa formation (par exemple la durée de chaque module) ,et en il peut préciser ses besoins qu' ils sont influencés par ses désirs ,son niveau actuel ,sa qualité professionnel , l'exigences du travail...).
- L'apprenant a dépassé le rôle d'un récepteur d'information et de savoir dans le mode présentiel, il devient un participant à la formation dans l'e-learning.
- L'apprenant peut évaluer ses capacités de maîtrise des différents cours et outils à n'importe quel moment pendant sa formation.
- L'absence du problème de manque de concentration qu'il était posé dans le système classique.et qui influence sur la qualité de formation.
- Donne à l'apprenant un sentiment de liberté et de confiance en lui-même.

- Aide les gens qui ne pouvant pas déplacer à l'université comme les travailleurs et les handicapés.
- Donner la possibilité de former individuellement ou collectivement.
- Réduire la perte de temps de déplacements et en plus la durée de la formation est moins long que en présentiel.
- Gérer automatiquement le parcours de l'apprenant (les notes)
- Eliminer tous les couts liés au déplacement et hébergement.
- Donner à l'apprenant la possibilité de poser n'importe quelle question dans le chat ou les forums.
- Bénéficier d'une formation plus rapide et durable.

1.6.2.Pour l'enseignant

- Minimise les couts liés aux équipements si on a un grand nombre d'étudiants pour le formateur.
- Un formateur peut s'adresser à un grand nombre d'apprenants tout en assurant une relation individualisée avec chacun d'eux.
- Eviter la répétition du cours, la surcharge et les problèmes de stresse posés au mode traditionnel.
- Développer des ressources pédagogiques que l'on peut partager, réutiliser et échanger.

1.7.Les inconvenants

1) Les équipements multimédia :

l'université qui veut appliquer le systèmes e-learning, doit posséder des équipements puissants pour permettre la création de contenue de cours (des PC , serveurs, terminaux, retours, câbles, des logiciels et des réseaux avec une bonne bande passante) ces équipements peuvent avoir plusieurs problème tels que :

- Les pannes des ordinateurs, serveurs, terminaux, retours.
- Perturbation des réseaux de communications.
- Les virus qui attaquent les documents électroniques et les systèmes d'exploitation

- Le piratage

2) Ce monde virtuel fait disparaitre le contact visuel alors l'absence de l'enseignant et de l'apprenant ce qui montre la perte du partage, et les relations humaines.
3) L'e-learning limite les interactions entre les apprenants.
4) L'utilisation de l'outil informatique est obligatoire.
5) Manque de repères visuels.

2.Les plateformes

2.1.Définition d'une plateforme

Une plateforme en informatique est une base de travail à partir de laquelle on peut écrire, lire, utiliser, développer un ensemble de logiciels. **[BOU 09]**

2.2.Définition d'une plateforme e-learning

Une plateforme de formation : « est un système informatique destiné à automatiser les diverses fonctions relatives à l'organisation des cours, à la gestion de leur contenu, au suivi des progrès des participants et à la supervision des personnes responsables des différentes formations » **[CAR 07].**

Elle peut être vue comme une mémoire organisationnelle qui permet non seulement de capitaliser les ressources pédagogiques liées au contenu de la formation mais aussi les informations concernant les acteurs (leurs spécificités, leurs parcours, etc.), ainsi que la gestion administrative (inscription, notes, etc.) de la formation. **[ABE 03]**

Une plateforme e-learning regroupe l'ensemble des applications et logiciels informatiques utilisés au service de l'enseignement et de l'apprentissage. L'expression englobe tant **les plateformes de gestion des cours** et de **gestion de contenu** que les logiciels outils. **[GTI 07]**

2.3.Intérêt d'une plateforme e-learning

Une plateforme e-learning a une grande importance dans la formation des apprenants

- Elle dirige la formation ouverte et à distance.
- Elle gère les individus, les groupes, les sous-groupes, les programmes et les contenus par module.
- Elle regroupe les outils nécessaires aux trois principaux acteurs de la formation : apprenant, tuteur, administrateur. Elle est basée sur les techniques de travail collaboratives.
- Elle utilise les outils de communication synchrone (chat, forum de discussion) et asynchrone (envoi d'e-mail…).
- Elle utilise les outils d'évaluation (exercices, qcm. …).
- Elle suit individuellement le parcours de l'utilisateur (les notes).
- Elle effectue la gestion des espaces de cours, des utilisateurs et le contrôle des accès.

2.4.Les composants d'une plateforme e-learning

Une plate-forme e-learning est composée de 2 systèmes imbriqués : le **LMS**, qui prend en charge la gestion de la « masse des étudiants »; et le **LCMS**, qui prend en charge la « masse des contenus ». **[FRE 09]**

2.4.1.Learning Management System (L.M.S)

L.M.S a beaucoup de termes équivalents qui sont : **SGC** : Système de Gestion de Cours, **MLE**: Managed Learning Environnent,**VLE**: Virtual Learning Environnent, **CMS**: Course Management System, **LSS**: Learning Support System.

Définition :

LMS (Learning Management Système) permet la gestion de formation par internet. Il prend en charge la couche logistique de la plate-forme e-learning **[FRE 09].** LMS est un site web qui héberge du contenue didactique et facilite la mise en œuvre de stratégie pédagogique. **[OUB 03]**

LMS est une solution axée sur l'intégration, la diffusion et la gestion de contenu pour la formation à distance, il désigne une suite de fonctionnalités conçues pour assurer la présentation, le suivi, la production de rapports et la gestion d'un contenu d'apprentissage, des progrès des élèves et de leurs interactions. **[MAO 04]**

L'objectif principal d'un LMS est la gestion des apprenants en gardant la trace de leurs progressions et performances à travers tous les types d'activités d'apprentissage et d'entrainement. **[BOU 11]**

2.4.2.Learning Content Management System (L.C.M .S)

Définition :

LCMS: Learning Content Management System gère le contenu de l'apprentissage (les Objets Pédagogiques) qui est servis au bon apprenant au bon moment. Comprendre la différence entre les deux types peut être une tâche difficile car la plus part des systèmes LMCS incluent un système LMS et peuvent inter-opérer avec des LMS externes**. [BRA 09]**

2.5.Les acteurs d'une plateforme e-learning

Les différents acteurs des plateformes e-learning sont cités ci-dessous avec leurs rôles.

Acteur	Rôle
Apprenant	▪ Transformer les informations en connaissances. ▪ Gérer son scénario et ses activités d'apprentissage. ▪ Réaliser des activités servant à son évolution. ▪ Auto-évaluer ses activités. ▪ Communiquer et échanger des informations.
Enseignant	
Concepteur	▪ La création d'un contenu pédagogique multimédia. ▪ La création des parcours pédagogiques types ou individualisés. ▪ La création des moyens de suivi des activités des étudiants.
Orienteur	▪ La création des cursus et le livret des apprenants. ▪ L'intégration des événements dans leur agenda. ▪ a gestion des ensembles de modules pédagogiques regroupés pour un objectif bien spécifié. ▪ la gestion particulière des plannings destinés aux cas de formations organisées par les entreprises.
Tuteur	▪ Rendre disponible les informations pour l'apprentissage. ▪ Aider les apprenants à progresser. ▪ Gérer le fonctionnement de la communauté d'apprenants. ▪ Présenter des informations. ▪ Gérer des médias donnant de l'information. ▪ Analyser et évaluer le contenu de document.
Evaluateur	▪ la création des tests sous forme de Questions à choix unique ou multiple, Questions ouvertes. ▪ le suivi et l'évaluation de l'apprenant.
Administrateur	
Administrateur Techniques	▪ L'installation /La maintenance (détection et la correction des bugs).
Administrateur Institutionnel	▪ La gestion des inscriptions. ▪ Il gère les droits d'accès. ▪ La gestion des ressources pédagogiques. ▪ Planifier le déroulement des activités. ▪ Diagnostiquer le déroulement des événements. ▪ Organiser des équipes ou des groupes.

Tableau 2: *les acteurs avec leurs rôles.* **[MAD 05][PAQ 02][OUB 03].**

2.6.Les critères du choix d'une plateforme e-learning

	Le critère	Signification
Critère liée la plateforme	L'aspect fonctionnel	Toutes les plateformes ont les mêmes fonctionnalités de base (forum, agenda …etc.) la différence est dans la richesse des outils proposés.et les améliorations à ajouter pour passer d'une version à une autre.
	Les langues. facilité d'utilisation	Doit être facile à manipuler par l'utilisateur et disponible en plusieurs langues.
	L'intégration possible avec d'autres applications	Intégration avec le système de gestion des cours et des utilisateurs (création des cours, importation des utilisateurs).
	L'interopérabilité	Respecter certaine norme et La compatibilité avec SCORM[1] **[ELA 05]** est très importante.
	Open source ou commercial ?	Chaque plateforme nécessite des ressources humaines et matérielles donc pas de solution gratuite.
Critère interne liée au l'université qu'elle veut faire un projet e-learning	L'objectif du projet e-learning pour l'université	On doit répondre à quelques questions :S'agit-il de quelques cours isolés ou existe-t-il une stratégie plus large ? Les cours en ligne sont-ils utilisés en complément des cours donnés en présentiel ?
	Le nombre des utilisateurs	un nombre important d'utilisateurs en ligne réalisant des activités signifie non seulement des tâches d'administration plus importantes mais aussi la nécessité d'une capacité de montée en charge suffisante pour répondre sans faille aux requêtes simultanées des utilisateurs.
	Les compétences et les ressources nécessaires pour l'installation et la maintenance	Pour déterminer ces compétences et ressources il faut : avoir les autres partenaires de l'université avec les détails de leurs projets. Est-ce qu'elle a des moyens internes pour gérer l'installation et la maintenance.
	Moyens financières	Quels moyens financiers utiliser ?

Tableau 3 : *Les critères du choix d'une plateforme.* **[BEA 03]**

[1] SCORM : est un modèle s'applique aux contenus multimédia principalement web, à vocation d'enseignement ou de formation, ainsi qu'aux plates-formes qui les administrent. Elle allie des propositions techniques à des aspects pédagogiques.

2.7.Les catégories des plateformes e-learning

2.7.1.Les plateformes open source

Ce sont des logiciels libres à dire leur modification, leur étude, leur utilisation et l'accès à leur code source sont autorisées, ce sont des logiciels gratuits et parmi ces plateformes on cite Claroline, Moodle, Sakai, Dokeos, Ganesha. (On va détailler quelques une dans le chapitre suivant).**[BEA 07]**

2.7.2.Les plateformes payantes (propriétaires)

L'accès au code source, l'utilisation, l'étude de ces plateforme ne sont pas autorisées (des logiciels non libres) donc il y a une limitation de diffusion et de modification parmi ces plateformes payante on trouve: WebCT, Blackboard, Edumatic, Learn eXact, iTutor, Angel. **[BEA 07]**

Conclusion

L'évolution importante de système e-learning a impliqué des améliorations importantes dans le mode d'apprentissage mais il n'a pas remplacé totalement le mode présentiel.

Les plateformes existantes, que ce soit open source ou propriétaires, offrent aux utilisateurs un ensemble d'outils de gestion et d'administration: création des cours, gestion des apprenants, élaboration de parcours pédagogique, ainsi elles disposent de modules complémentaires qui peuvent enrichir les fonctionnalités disponibles : rendu de travaux, visibilité des cours, gestion des notes.

Moodle est une plateforme à une grande utilisation dans le domaine e-learning à cause de sa simplicité et ses caractéristiques que nous détaillerons dans le chapitre suivant.

Chapitre 2

La plateforme Moodle

Chapitre 2 : la plateforme Moodle

Introduction

Moodle est une plateforme open source dédiée au domaine du e-learning qui permet de créer des interactions entre enseignants, apprenants et ressources pédagogiques. Ce dispositif joue un rôle très important dans l'amélioration de la qualité de formation en ligne à cause de sa richesse fonctionnelle.

Les caractéristiques, la description générale et les acteurs ainsi que d'autres exemples de plateforme ce sont les points que nous aborderons dans ce chapitre.

1.Définition de moodle :

Moodle est l'acronyme de **M**odular **O**bject-**O**riented **D**ynamic **L**earning **E**nvironnement, en français un environnement d'apprentissage dynamique modulaire et orienté objet. Modeler est aussi un verbe traduisant les manières agréables d'agir qui mène souvent à la réflexion et à la créativité.

Moodle est une plateforme e-learning **LMS**(Learning management système) gratuite et open source créée en 2002 par un groupe des développeurs à sa tête **Martin Dougiamas** l'administrateur de la plateforme WebCT (c'est une plateforme e-learning payante) à l'Université de Curtin en Australie dans le cadre de ses recherches doctorales, Elle favorise un cadre de formation socioconstructiviste. Elle a un système de gestion de contenu (SGC), qui permet la mise en place de cours en ligne et assure la gestion des ressources pédagogiques et les activités d'apprentissage, La création d'environnement d'apprentissage en ligne favorisant les interactions entre des pédagogues, des apprenants et des ressources pédagogiques. Elle offre un certain nombre de liberté tel que (utiliser et modifier le code source).

Elle est utilisée dans de nombreuses universités et organismes de formation, qui dispose d'une communauté active sur internet.

2.Les caractéristiques de moodle

1) Moodle a une interface simple, modifiable, efficace et compatible.
2) Moodle est un site web qui peut prendre des milliers d 'utilisateurs et cours dans plusieurs langues.
3) Moodle est très simple à prendre et facile à utiliser soit par l'enseignant où bien par les étudiants il ne demande pas une grande expérience pour le maitriser.
4) Moodle fonctionne sans modification sur tous les systèmes d'exploitation.
5) L'architecture modulaire de moodle facilite le développement et permet une grande flexibilité pour ajouter des fonctionnalités à différents niveau.
6) Est un système généraliste il permet de gérer, créer et planifier le contenue d'un cours
7) Moodle possède un système interne pour mettre à jour ses bases et faciliter la mise jour d'une version à une autre.
8) Moodle contient un système complet d'abstraction de la base de données qui supporte plusieurs serveurs de base de données.
9) Le niveau de sécurité de moodle est très élevé car les formulaires sont tous vérifiés, les données validées et les cookies encryptés.
10) La communauté des utilisateurs et développeurs est internationale, large et active.
11) Elle possède plusieurs outils de communication (forum, chat, sondage, wiki ...). **[BOU 11][BOU 09]**

3.Les utilisateurs de la plateforme moodle

Le tableau suivant présente les différents utilisateurs de la plateforme Moodle avec leurs rôles :

<table>
<tr><th>Utilisateur</th><th>Définition</th><th>Les droits</th></tr>
<tr><td>Administrateur</td><td>C'est Le responsable de la plateforme, il peut faire n'importe quelle opération sur la plateforme</td><td>▪ Contrôle la création des cours.
▪ Ajouter les utilisateurs.
▪ Attribuer les rôles.
▪ Créer les groupes.
▪ Modifier les thèmes… etc.</td></tr>
<tr><td>Etudiant ou l'apprenant</td><td>C'est lui qui suive les cours de la formation</td><td>▪ Créer un compte.
▪ S'inscrire à un cours.
▪ Participer à l'activité d'apprentissage.
▪ Consulter les ressources.
▪ Voir ses notes et modifier son profil.
▪ participer à des forums.</td></tr>
<tr><td>Enseignant

Responsable de cours</td><td rowspan="3">L'enseignant est l'utilisateur le plus important qui oriente et gère le contenue cours et les activités, on peut distinguer

3catégories d'enseignants

Responsable de cours

Enseignant (teacher)

Enseignant non editor (tuteur)</td><td>▪ Inscrire les étudiants et les enseignants aux différents cours.
▪ Gérer l'autorisation pour les invités.
▪ Supprimer les étudiants après une durée de validité d'inscription.
▪ Retirer les privilèges à certains enseignants.
▪ ajouter des activités et des ressources</td></tr>
<tr><td>Enseignant (teacher)</td><td>▪ ses cours sont créés par l'administrateur.
▪ il ne peut modifier que ceux qui lui ont été attribués.</td></tr>
<tr><td>Enseignant non editor (tuteur)</td><td>▪ Il n'a pas le droit d'écrire ni de modifier le contenu de cours.
▪ Il peut participer aux forums et faire la notation.
▪ Il ne peut pas ajouter de ressources ni d'activités
▪ Il peut évaluer les activités</td></tr>
<tr><td>Invité (guest)</td><td>C'est un utilisateur qui peut accéder à la plateforme avec un certain nombre de droit limité</td><td>▪ Il peut seulement lire le cours.
▪ Il ne peut pas participer aux activités ou forums</td></tr>
</table>

Tableau 4 : *les utilisateurs Moodle.*

4.Etudes comparatif entre les plates-formes

4.1.Présentations de quelques plateformes

4.1.1.Claroline

Est une plateforme open source de formation à distance LMS (Learning management système) et de travail collaboratif, elle été créé (2001) à l'Institut de pédagogie et des multimédias de l'Université catholique de Louvain. Elle s'appuie sur les technologies libres et gratuites PHP et MySQL.et elle est compatible avec les environnements Linux, Macintosh et Windows. Traduite dans 35 langues.

Cette plateforme est utilisée pour créer et administrer des cours par internet, elle offre un ensemble des fonctionnalités qui permettent de réaliser des exercices (un générateur de quiz),des outils de discussion synchrone (chat) et asynchrone (forum),un espace de document pour les apprenants, un calendrier, gestion des groupes, un système de suivi et de control pour les utilisateurs et donne aussi des fonctionnalités avancées des statistiques et un wiki comme environnement collaboratif.**[CLA 12]**

Figure 2: *la plateforme Claroline.*

❖ **Les caractéristiques :**

- L'utilisation de cette plateforme est très simple et rapide par les étudiants que les enseignants.
- Elle donne beaucoup de facilité pour créer un cours rapidement
- Elle a été développée sur base de l'expertise pédagogique des professeurs et en fonction de leurs besoins.
- Elle offre une gestion sobre et intuitive des outils et des espaces d'administration
- Elle intègre les standards SCORM et IMS pour une intégration simple et rapide des contenus d'apprentissage dans des parcours pédagogiques. **[DOS 08]**

❖ **Quelques inconvénients :**

- Les apprenants peuvent s'inscrire dans un group mais ne pas se désinscrire.
- Les outils de communications asynchrones (chat, messagerie) ne sont pas très ergonomiques.
- Manque des outils de communication synchrone (seulement l'outil discussion).
- Le SCORM n'est pas complètement interprété par la plateforme.
- Les package produit par les langages de modélisation comme IMS-LD ne fonctionnent pas.

4.1.2.Ganesha

Est une plateforme de téléformation ou LMS Crée en 2001. Ce logiciel permet à un service d'entreprise ou un centre de formation de mettre à la disposition de stagiaires des modules e-learning ainsi que des outils collaboratifs (émail, forum, chat, partage de documents) et d'assurer un tutorat en ligne.

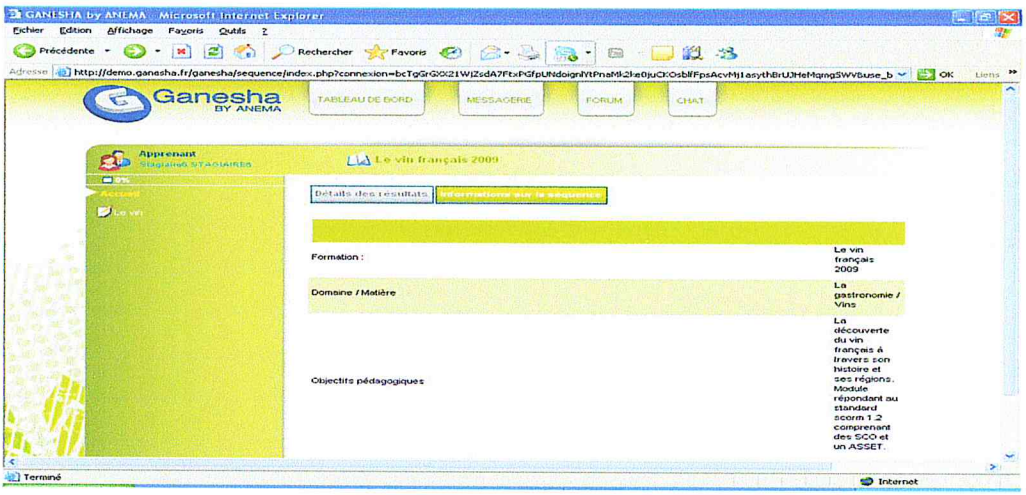

Figure 3: *Interface de la plateforme Ganesha*

❖ **Les caractéristiques :**

- Elle offre des outils de communications synchrone (chat) et asynchrone (email).
- Elle donne une zone interactive de dépôt des documents pour laisser des commentaires sur les documents.
- l'intégration de contenus pédagogiques (cours, questionnaires) qui ont été créés avec des logiciels ou des éditeurs externes à la plateforme.

❖ **Quelques inconvénients :**

- Authentification : mots de passe des utilisateurs non cryptés.
- La modification des paramètres de cours ne se fait pas de manière indépendante
- Messagerie interne limitée.
- Le Chat est peu ergonomique et n'a pas de traces.
- Wiki, blog, vidéoconférence non implémentés.

4.1.3.Sakai

Est une plateforme d'enseignement à distance développée en 2004 par quatre université américain (indianuniversity, massachusettsinstitute of technology, stanforduniversity et university of Michigan), leur objectif est de créer un système de

gestion de cours libre et de qualité équivalent au mieux aux autres produit du marché. Elle se base beaucoup plus sur l'aspect de travail collaboratif que sur l'aspect suivi pédagogique. Elle offre des fonctionnalités pédagogiques des fonctionnalités collaboratives, elle contient les outils suivants: gestion de profiles ressources, gestion de groupes, annonces, Messagerie à diffusion interne et externe, Wiki, blog, sondage, agenda, chat et forum, statistiques d'utilisation, gestion des notes, gestion des devoirs.

Les caractéristiques :

- Une interface légère et facile à utiliser.
- Elle propose un guide pour l'ensemble des outils qui facilite leur utilisation.
- La flexibilité des outils et de l'environnement combinée à ses possibilités d'extension permet l'exploitation de la plateforme pour des contextes pédagogiques variés.
- Le rythme de progression beaucoup plus élevé par rapport aux autres plateformes.

Quelques inconvenants :

- Pas de possibilité d'importer des ZIP avec dézippage en ligne.
- Importations de modules IMS 1.1.2 pas toujours opérationnelles selon le logiciel auteur
- Importation SCORM pas implémentée.
- Manque de documentations en français pour les utilisateurs
- L'installation de cette plateforme est complexe parce qu'elle est développée en java ce qui demande un administrateur compétent dans ce langage.

5.Les points clés pour choisir une plateforme

1) La disponibilité de la documentation en ligne de l'installation et de l'utilisation de la plateforme par les enseignants et les apprenants.
2) La facilité de l'Installation et de la gestion de la plateforme.
3) Intégrer plusieurs outils de communications synchrones et asynchrones.
4) Le nombre d'utilisateurs gérés par la plateforme.
5) Outils collaboratifs dédiés aux échanges autour d'apprentissages communs.
6) Adaptabilité et modularité de la plateforme.
7) Intégration de spécifications techniques et de standards comme l'AICC/SCORM le LOM et IMS-LD
8) Importante communauté d'utilisateurs et de développeurs, dynamique et d'envergure internationale.

6.Pourquoi on a choisi Moodle

Le tableau suivant compare les différentes plateformes

	Moodle	**Claroline**	**Sakai**	**Ganesha**
Documentation	1	1	-1	-1
Installation et gestions	1	1	-1	-1
Le nombre d'utilisateur	1	1	1	-1
Outils de communication.	1	1	1	1
Outils collaboratifs	1	1	1	1
Adaptabilité et modularité	1	-1	-1	-1
Communié importante	1	1	-1	-1
Intégration des standards et des spécifications	1	-1	-1	-1
Totale	**8**	**6**	**-2**	**-4**

Tableau 5 : *Comparaison entre les plateformes.*

+1 : la plateforme est répond au critère / -1 : la plateforme ne répond pas au critère.

D' après Cette étude comparative entre les différentes plateformes de formation et les résultats obtenus nous allons choisir la plateforme moodle car elle répond à tous les critères.

Conclusion

Moodle est une plateforme qui constitue une véritable classe virtuelle où les utilisateurs (apprenants, enseignants) peuvent trouver beaucoup de fonctionnalités qui facilitent l'apprentissage, elle propose aussi des différents outils de communications (chat, forum).

Chapitre 3

Le cloud computing

Chapitre 3: Cloud Computing

Introduction

Dès l'apparition de l'informatique, celle-ci a connu un sort remarquable dans la mesure où plusieurs nouvelles approches émergent, et parmi ces nouvelles, on trouve le Cloud Computing ou « informatique dans les nuages ». C'est une nouvelle approche qui encourage et prône la location à outrance de services informatiques sous ses diverses formes, au lieu de les acheter et réaliser les diverses opération de suivi et maintenance. Les services à louer sont de plus accessible de n'importe où et à n'importe quel moment.

*Du point de vue technologique, le Cloud se base fondamentalement sur la virtualisation des services. Les services peuvent être des services logiciels ou des services matériels. Comme avancées technologique, le Cloud n'apporte en réalité aucune nouveauté. Pire encore, les adeptes du logiciel libre sont fondamentalement hostiles à cette approche qui est propulsé par les grandes compagnies détentrice de ressources informatique (logiciel et matériel) énorme qu'il faut rentabiliser.***[SOG 09]**

1.Définitions

Premièrement, et en vue de l'importance de la virtualisation dans le cloud computing, nous allons définir la virtualisation, puis on définit le cloud computing.

❖ **Virtualisation** :

La virtualisation a été la première pierre vers l'ère du Cloud Computing. En effet, cette notion permet une gestion optimisée des ressources matérielles dans le but de pouvoir y exécuter plusieurs systèmes « virtuels » sur une seule ressource physique et fournir une couche supplémentaire d'abstraction du matériel.

❖ **Cloud Computing** :

- « Le Cloud Computing permet un accès facile et à la demande par le réseau à un ensemble partagé de ressources informatiques configurables (serveurs, stockage, applications et services) qui

peuvent être rapidement provisionnées et libérées par un minimum d'efforts de gestion ou d'interaction avec le fournisseur du service ». **[NIST]**

- « Une interconnexion et une coopération de ressources informatiques, situées au sein d'une même entité ou dans diverses structures internes, externes ou mixtes, et dont le mode d'accès est basé sur les protocoles et standards Internet». **[WYG 10]**
- « Le Cloud Computing peut être perçu comme un système d'exploitation distribué sur des milliers de machines. Cet OS distribué, que l'on représente par ce fameux nuage, assure l'abstraction de l'infrastructure (matérielle, réseau, etc.) et a pour rôle d'héberger et d'exécuter des applications ou des services mais aussi de stocker des données. »

2.LES CARACTERISTIQUES ESSENTIELLES

L'utilisation de ressources à distance n'est pas nouvelle. Le time sharing utilisation partagée d'un ordinateur en langage Basic avait fait son apparition en 1966. On parlait alors de la « prise de calcul » à côté de la prise de courant. Dès le début des années 1970, les activités « service bureau » ou « traitement à façon » partageaient des traitements comme les payes ou les facturations sur des infrastructures communes avec souvent une facturation à l'usage. Plus récemment, sous le nom outsourcing, l'hébergement et l'exploitation des applications des entreprises à distance se sont largement développés. Ces activités n'avaient pas changé l'architecture des systèmes. Les gains provenaient d'une mise en commun de locaux et de moyens humains et techniques spécialisés dans l'exploitation de systèmes et d'applications existantes. **[NIST]**

Le Cloud Computing se différencie par les cinq caractéristiques essentielles suivantes :

2.1.Accès aux services par l'utilisateur à la demande

La mise en œuvre des systèmes est entièrement automatisée et c'est l'utilisateur, au moyen d'une console de commande, qui met en place et gère la configuration à distance.

2.2.Accès réseau large bande

Ces centres de traitement sont généralement raccordés directement sur le backbone Internet pour bénéficier d'une excellente connectivité. Les grands fournisseurs répartissent les centres de traitement sur la planète pour fournir un accès aux systèmes en moins de 50 ms de n'importe quel endroit.

2.3.Réservoir de ressources (non localisées)

La plupart de ces centres comportent des dizaines de milliers de serveurs et de moyens de stockage pour permettre des montées en charge rapides. Il est souvent possible de choisir une zone géographique pour mettre les données « près » des utilisateurs.

2.4.Redimensionnement rapide (élasticité)

La mise en ligne d'une nouvelle instance d'un serveur est réalisée en quelques minutes, l'arrêt et le redémarrage en quelques secondes. Toutes ces opérations peuvent s'effectuer automatiquement par des scripts. Ces mécanismes de gestion permettent de bénéficier pleinement de la facturation à l'usage en adaptant la puissance de calcul au trafic instantané.

2.5.Facturation à l'usage

Il n'y a généralement pas de coût de mise en service (c'est l'utilisateur qui réalise les opérations). La facturation est calculée en fonction de la durée et de la quantité de ressources utilisées.

3.LES MODELES DE SERVICES

Trois modèles de services peuvent être offerts par le Cloud:

3.1.Software as a Service (SaaS)

Ce modèle de service est caractérisé par l'utilisation d'une application partagée qui fonctionne sur une infrastructure Cloud. L'utilisateur accède à l'application par le réseau au travers de divers types de terminaux (souvent via un navigateur web). L'administrateur de l'application ne gère pas et ne contrôle pas l'infrastructure sous-jacente (réseaux, serveurs, applications, stockage). Il ne contrôle pas les fonctions de l'application à l'exception d'un paramétrage de quelques fonctions utilisateurs limitées.

3.2.Platforme as a Service (PaaS)

L'utilisateur a la possibilité de créer et de déployer sur une infrastructure Cloud PaaS ses propres applications en utilisant les langages et les outils du fournisseur. L'utilisateur ne gère pas ou ne contrôle pas l'infrastructure Cloud sous jacente (réseaux, serveurs, stockage) mais il contrôle l'application déployée et sa configuration.

3.3.Infrastructure as a Service (IaaS)

L'utilisateur loue des moyens de calcul et de stockage, des capacités réseau et d'autres ressources indispensables (partage de charge, pare-feu, cache). L'utilisateur a la possibilité de déployer n'importe quel type de logiciel incluant les systèmes d'exploitation.

L'utilisateur ne gère pas ou ne contrôle pas l'infrastructure Cloud sous jacente mais il a le contrôle sur les systèmes d'exploitation, le stockage et les applications. Il peut aussi choisir les caractéristiques principales des équipements réseau comme le partage de charge, les pare-feu, etc.

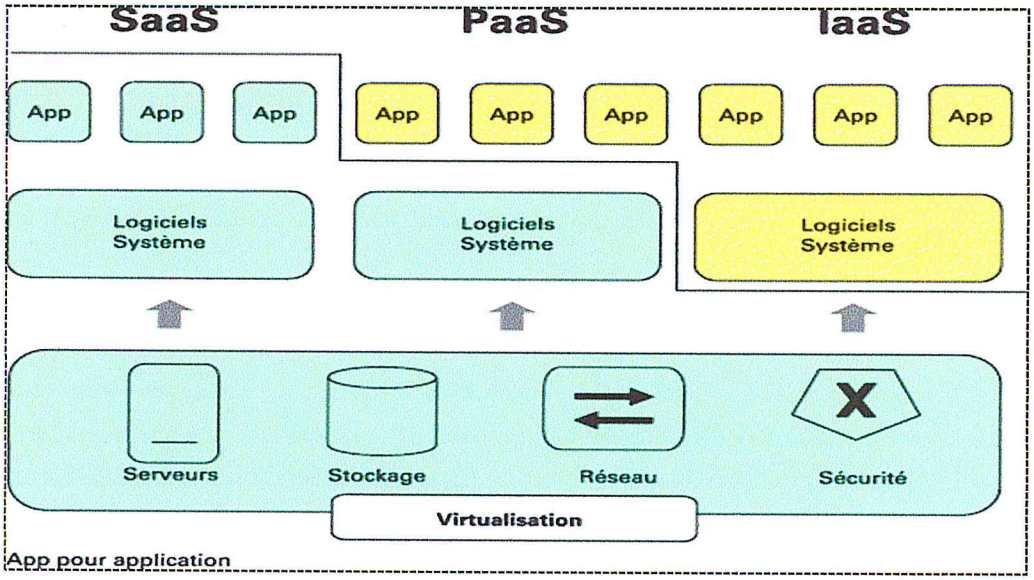

Figure 4 : *Modèles de service du Cloud Computing.* **[NIST]**

4.LES MODELES DE DEPLOIEMENT

On distingue quatre modèles de déploiement **[SOG 04].**

4.1.Cloud privé

L'infrastructure Cloud est utilisée par une seule organisation. Elle peut être gérée par l'organisation ou par une tierce partie. L'infrastructure peut être placée dans les locaux de l'organisation ou à l'extérieur.

4.2.Cloud communautaire

L'infrastructure Cloud est partagée par plusieurs organisations pour les besoins d'une communauté qui souhaite mettre en commun des moyens (sécurité, conformité, etc.). Elle peut être gérée par les organisations ou par une tierce partie et peut être placée dans les locaux ou à l'extérieur.

4.3.Cloud public

L'infrastructure Cloud est ouverte au public ou à de grands groupes industriels. Cette infrastructure est possédée par une organisation qui vend des services Cloud.

4.4.Cloud hybride

L'infrastructure Cloud est composée d'un ou plusieurs modèles ci-dessus qui restent des entités séparées. Ces infrastructures sont liées entre elles par la même technologie qui autorise la portabilité des applications et des données. C'est une excellente solution pour répartir ses moyens en fonction des avantages recherchés.

5.LE CLOUD COMPUTING DES DIFFERENTS ACTEURS

Il serait trop de citer tous les acteurs du Cloud Computing présents sur le marché actuel. Parmi les principaux acteurs: Amazon, Google, Salesforce.com, VMware et Microsoft. Dans cette partie, nous présentons la solution (**AWS**) apportée par Amazon et les solutions (Google App Engine, Google App) apportées par Google.

5.1.Amazon (AWS)

Amazon Web Services apparue en 2006, AWS est une plateforme d'hébergement de services d'infrastructure pour le Web, dans le cloud.

L'offre **AWS** est principalement constituée des services suivants:

- **EC2** (ElasticComputeCloud): C'est le cœur de l'infrastructure AWS, il permet de créer et gérer des machines virtuelles.
- **ELB** (ElasticLoadBalancing): permet de distribuer automatiquement le trafic entrant vers les applications sur de multiples instances Amazon EC2. Il vous permet d'accroître encore plus la tolérance de panne de vos

applications, en fournissant de manière transparente l'équilibrage de charge nécessaire pour répondre à la demande du trafic entrant.

- **CloudWatch** : est un service web qui permet la surveillance des ressources du cloud Amazon. Il offre aux clients une vue sur l'utilisation des ressources, la performance opérationnelle, et de la demande globale, y compris des indicateurs tels que l'utilisation du processeur, entrée/sortie disque, et le trafic réseau.
- **Auto Scaling** : est activée par Amazon CloudWatch et permet d'augmenter ou diminuer automatiquement la capacité Amazon EC2 en fonction des conditions que nous définissons.
- **Elastic IP** : sont des adresses IP statiques conçues pour donner une grande liberté dans la gestion des instances. Une adresse IP élastique est associée à un compte et non à une instance particulière, on dispose du contrôle de cette adresse jusqu'à sa libération explicite
- **EBS** (**E**lastic**B**lock**S**tore): offre un stockage persistant pour les instances EC2 d'Amazon.
- **S3** (**S**imple **S**torage **S**ervice): permet le stockage d'applications
- Web, quelque soit la taille de cette application.
- **SimpleDB**:est un service Web simple qui fournit les fonctionnalités essentielles d'une base de données (consultation élémentaire en temps réel, requêtes simples de données structurées).
- **RDS** (**R**elational**D**atabase**S**ervice): permet de mettre en place une base de données sur le cloud, de l'exploiter et d'en ajuster la capacité, le tout très simplement. Nous pouvons démarrer une instance de base de données et

bénéficier de l'intégralité des fonctionnalités d'une base de données MySQL.

- **SQS (Simple Queue Service)** : est une file d'attente fiable et capable de monter à l'échelle. Utilisée pour transmettre des messages entre différents services, elle facilite la création d'un flux de travail automatisé.
- **VPC (Virtual PrivateCloud)** : permet l'extension du réseau de l'entreprise vers un cloud privatif pris en charge par AWS.
- Cloud Front (**CDN: ContentDeliveryNetwork**):est un service Internet de livraison de contenu, libre-servicet facturé à l'utilisation. Les développeurs peuvent distribuer du contenu par l'entremise d'un réseau mondial de sites de débit amélioré qui offre un faibletemps d'attente et de grandes vitesses de transfert de débit de données. **[JIN 02][MAJ 10]**

Le diagramme suivant représente l'architecture de AWS :

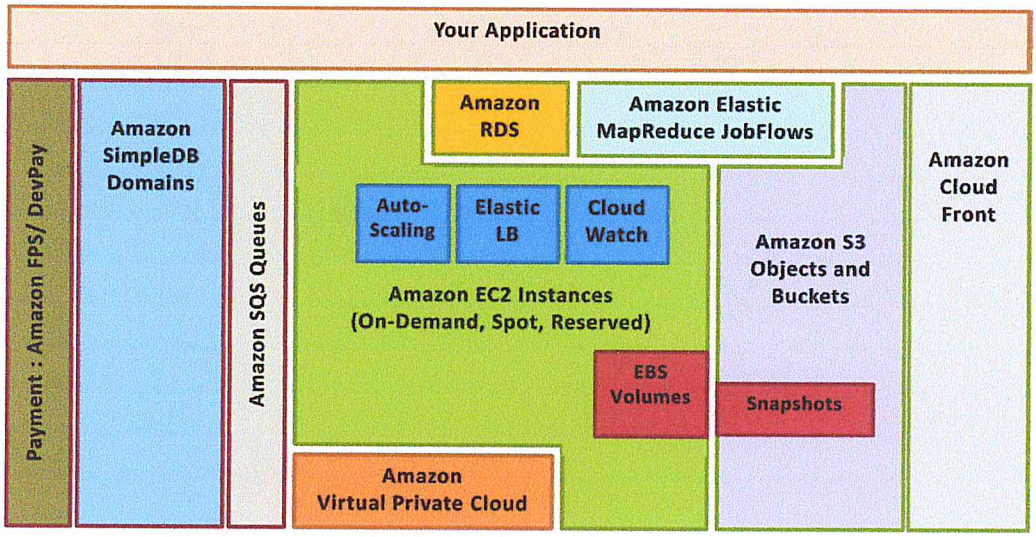

Figure 5: *Architecture de AWS.* **[MAJ 10]**

5.2.Google

En 2008, Google a lancé son Cloud public orienté pour les services Web offrant une plate-forme (PaaS) nommée « Google App Engine » et permettant l'hébergement d'applications Python ou Java, ainsi que des applications SaaS regroupées dans la gamme « Google App ». **[WYG 10]**

5.2.1.Google App Engine

La platefeorme « Google App Engine » met à disposition sur les Clusters de serveurs de Google appelés des « Sandbox » et dans lesquelles pourront s'exécuter des applications Java ou Python.

Ces sandbox sont réparties sur plusieurs noeuds de serveurs en fonction de la charge. L'écriture de fichiers, la création de sockets ou encore la création de ce thread n'est pas permise. De plus, les processus tournant plus de 10 secondes sont automatiquement coupés.

Il sera possible d'exécuter des applications Java incluant la JVM, les servlets Java et le langage de programmation Java (mais pas de support de J2EE). La plate-forme propose également l'environnement d'exécution python de Google.

La plateforme inclut aussi des services fournis sous forme d'API permettant d'envoyer des mails, de manipuler des images, d'utiliser les comptes de Google pour les identifications au sein d'une application, de communiquer au travers du protocole XMPP, etc.

- *Datastore :* Un service de base de données. Cela permet la mise à disposition d'un système de base de données performant, distribuant les informations via un moteur de recherche et une gestion des transactions.

5.2.2.Google APP

Google met à disposition plusieurs applications SaaS personnalisables par le client. Il est possible de le faire fonctionner sur son propre nom de domaine et de personnaliser l'affichage graphique aux couleurs de son entreprise par exemple. Cette offre inclut :

- *un service de messagerie* identique à Gmail mettant à disposition 25 Gb d'espace de stockage par utilisateur, une application d'agenda permettant de collaborer efficacement grâce au service *Google Agenda*.
- *Google Document :* une suite bureautique Web permettant le stockage et l'édition de documents, de feuilles de calcul, de schémas ou de présentations.
- *Google Sites :* met à disposition un CMS à la demande pouvant concevoir des sites Web d'équipe sans aucun développement.
- *Google Vidéos :* pour l'hébergement d'un Web TV privé.

Conclusion

Comme pour toute nouveauté technologique, il faut attendre les réelles expériences des entreprises pour pouvoir mesurer ce nouveau modèle.

Nous pouvons dire que le cloud est une simple évolution de nos idées et de la vision de l'informatique d'aujourd'hui, mais une réelle révolution dans ses usages futures.

Chapitre 4

LoadBalancing

Chapitre 4 : LoadBalancing

Introduction

Du client au serveur, le parcours est parfois long et très souvent semé d'embuches. Par exemple, dans notre travail, pour une centaine d'examinés passant en même temps leurs examens, la charge du travail serait lourde pour l'ensemble de serveurs de cloud, donc il faut penser au LoadBalancing « Equilibrage de charge » pour assurer la performance et la haute disponibilité.

Dans ce chapitre, vous retrouverez les outils et les méthodes de répartition de charge.

1.Définition

- Littéralement "répartition de charge" ou "équilibrage de charge".
- La répartition de charge est donc la capacité d'utiliser plusieurs serveurs qui délivrent le même service et font le même travail.

Une solution de loadbalancing peut intervenir à deux niveaux :

- **Niveaux 3/4 :** Réseau et transport (protocole TCP/IP).
- **Niveau 7 :** Applications (protocole HTTP, RDP…).

2.Les outils de répartition de charge

Il existe deux manières pour faire la répartition de la charge :

2.1.La répartition de charge sans répartiteur

La méthode la plus simple d'opérer une répartition de charge consiste à dédier des serveurs à des groupes d'utilisateurs prédéfinis. si cette méthode est simple à mettre en œuvre pour un Intranet, elle est très complexe, voire impossible, pour des serveurs Internet. Dans ce cas, un paramétrage spécifique du DNS (Domain

Name Server) au niveau réseau et/ou l'optimisation applicative (niveau 7) sont un premier pas vers une démarche de répartition de charge sans répartiteur.

2.2.La répartition de charge avec répartiteur

La mise en œuvre d'un répartiteur de charge est la méthode la plus pertinente et efficace lorsqu'il s'agit d'engager une démarche de répartition de charge de la population des utilisateurs sur plusieurs serveurs, voire plusieurs sites.

Le répartiteur peut indifféremment prendre la forme d'un matériel spécifique, d'un logiciel installé sur un serveur.

3.Les différents niveaux d'action

Matériel ou logiciel, le répartiteur de charge (ou load balancer) est la couche supplémentaire qui permet d'optimiser et de réguler le trafic, tout en soulageant les serveurs, en répartissant la charge selon des algorithmes prédéfinis (niveaux 3/4 et 7) ou selon des fonctions intelligentes capables de tenir compte du contenu de chaque requête (niveau 7).

3.1.La répartition de charge de niveau 3/4 (Réseau)

La répartition de charge de niveau 3/4 consiste à travailler sur les paquets réseau en agissant sur leur routage (TCP/IP). Le répartiteur de niveau 3/4 intervient donc à l'ouverture de la connexion TCP puis aiguille les paquets en fonction des algorithmes retenus, selon 3 méthodes :

- **Le routage direct**

Avec cette méthode, le répartiteur "distribue les cartes" : il se charge de répartir les requêtes sur une même adresse entre les serveurs locaux. Les serveurs répondent ensuite directement aux clients.

- **Le tunneling**

Le tunneling est une évolution du routage direct qui permet de s'amender de la problématique de proximité des serveurs grâce à la mise en place de tunnels entre des serveurs distants et le répartiteur.

- **La translation d'adresses IP (NAT)**

Dans ce cas, le répartiteur centralise tous les flux, les requêtes (comme dans le cas du routage direct, il les répartit entre les serveurs), mais aussi les réponses. Il "masque" ainsi l'ensemble de la ferme de serveurs, qui ne nécessitent aucun paramétrage particulier.

3.2.La répartition de charge de niveau 7 (Applicatif)

En s'attaquant à la couche applicative, le répartiteur de charge ne se contente plus d'aiguiller "aveuglément" les requêtes. Il analyse le contenu de chaque requête HTTP, RDP ou autre pour décider du routage.

4.Les fonctions principales d'un répartiteur

Un répartiteur, quel qu'il soit, assumera au moins deux missions principales : l'équilibrage de charge entre les serveurs et la surveillance de ces derniers.

4.1.Equilibrer la charge sur les serveurs

En pratique, le répartiteur s'appuie sur des algorithmes de répartition. Il en existe de nombreux qui répondent à des critères bien différents : les durées de sessions, les variations de charge, les capacités de serveurs, etc. Parmi eux, on peut citer :

- **Round Robin**

  Dans ce cas, le répartiteur utilise chaque serveur à la suite, selon un ordre cyclique. Si les serveurs sont de capacités inégales, il est possible d'ajouter une pondération (weighted round robin). C'est la méthode la plus efficace pour des serveurs Web.

- **Least Conn**

  Cet algorithme consiste à envoyer la nouvelle requête sur le serveur le moins chargé. Idéal pour des sessions longues, cette méthode est inadaptée à des serveurs dont la charge varie d'une seconde à l'autre.

- **Le hachage d'url ou d'adresse IP**

  Plus déterministe, cette méthode permet de créer une affinité plus ou moins efficace (sous certaines conditions) entre un client et un serveur.

4.2.Surveiller l'état des serveurs

En cas de défaillance de l'un des serveurs, toutes les requêtes seront réparties automatiquement entre les autres serveurs On parle alors de haute disponibilité applicative.

Les méthodes de surveillance des serveurs prennent de nombreuses formes telles qu'un **ping**, une tentative de connexion

TCP ou bien encore l'envoi d'une requête **HTTP** (niveau 7 uniquement). En effet, un serveur défaillant peut parfois répondre au ping mais pas aux connexions TCP ou accepter ces dernières mais refuser de répondre aux requêtes HTTP.

4.3.Les fonctions "intelligentes" du répartiteur applicatif

Pour répondre à certains besoins particuliers, le répartiteur de niveau 7 dispose de fonctions supplémentaires, issues de sa capacité d'analyse des requêtes.

La persistance

La persistance répond au cas où toutes les requêtes d'un même utilisateur doivent impérativement être adressées à un même serveur sur une session donnée, afin de conserver en mémoire les informations relatives à cette session (le "contexte" de l'utilisateur).

Différentes méthodes permettent la mise en œuvre de la persistante

- **La redirection *:***

  C'est l'une des solutions les plus économiques. Elle consiste à faire en sorte que l'application redirige la demande vers l'adresse locale du serveur concerné (en cas de panne du serveur, l'utilisateur sera tout de même redirigé vers ce dernier).

❖ **L'affinité de sessions :**

Le répartiteur associe un utilisateur à un serveur. La manière la plus simple étant d'identifier le serveur sur lequel s'est connectée l'adresse IP de l'utilisateur lors de la dernière connexion.

❖ **L'apprentissage de cookie :**

À l'arrivée d'une requête utilisateur, le répartiteur vérifie la présence du cookie de l'application dans l'en-tête de la requête et compare les valeurs de ce dernier à sa table de sessions, pour rediriger ensuite l'utilisateur vers le bon serveur. S'il n'y a pas de correspondance, la requête sera dirigée vers n'importe quel serveur, via l'algorithme choisi. Lors de la réponse du serveur à la requête du client, le répartiteur collectera l'information d'identifié cation du serveur (par apprentissage) pour ensuite l'enregistrer dans sa table de sessions. A la seconde requête du client, l'enregistrement alors effectué permettra de rediriger le client vers le bon serveur.

❖ **L'insertion de cookie :**

Sur le fond, le principe est le même que précédemment, à ceci près que le cookie est inséré directement sur la machine utilisateur par le répartiteur (qui n'a plus alors à maintenir une table de sessions), lors de la réponse du serveur.

A partir de la seconde requête, le cookie est lu par le répartiteur et les requêtes sont alors immédiatement redirigées vers le même serveur tout au long de la session.

Conclusion

Grâce aux outils de répartition de charge on peut optimiser les performances des serveurs. Il faut que ces outils fournissent la haute disponibilité et l'équilibrage de charge.

Chapitre 5

Conception

Chapitre 05 : Conception

Introduction

Dans la partie précédente, nous avons présenté les notions importantes sur le cloud computing, la répartition des charges, le e-learning ainsi que la plateforme Moodle.

Nous allons, à travers ce chapitre, proposer les solutions possibles pour un cloud orienté service de télé-enseignement. Ce cloud sera dorénavant référencé par le terme *CloudMoodle*, Il sera principalement basé sur la plateforme Moodle **[BEN 12].** Par la suite la mise en œuvre de l'une ou de l'autre de ces solutions (c'est le comportement interne du Cloud) sera définie dans le contexte d'une activité d'administration du Cloud. Nous présenterons aussi dans ce chapitre les éléments de conception de ce qu'on appellera le serveur frontal.

Pourquoi avoir choisi Moodle ?

Dans le monde du télé-enseignement, comme présenté précédemment, il y a un nombre important de plateformes. L'une des plus simples est Claroline. L'une des plus complètes est Moodle. Ainsi si nous arrivons à relever ces défis et prendre en charge cette diverse technique, la réalisation d'un Cloud basée sur une des autres plateformes serait très simple et le support d'un Cloud multiplateforme ne serait pas plus difficile. Les techniques d'interaction spécifique à Moodle seront revues en détail dans le chapitre réalisation dans lequel nous mettrons en évidence notre réponse à chacune de ces techniques.

1.Rappel de l'architecture traditionnelle de la plateforme Moodle

Cette architecture est valable pour la quasi-totalité des plateformes de e-learning qu'elle soit basée sur le langage PHP ou sur la technologie Java. C'est une architecture 3 tiers ordinaire *(Figure 6)*, dans laquelle il y a la partie présentation, la partie métier et la partie persistance de l'information. La connexion est directe entre les clients et le serveur de l'application *moodle*. Dans le cas d'une forte activité, le serveur pourra être surchargé et ne peut pas répondre dans les délais aux diverses requêtes. Ainsi un dimensionnement du serveur est nécessaire pour chaque communauté utilisant Moodle.

Cette architecture est facile à installer et configurer, moyennant une connaissance moyenne du langage PHP et de la technique d'ajout de modules spécifique à PHP. Une telle installation est préférable, recommandée pour un nombre très réduit de clients.

Cette architecture simple est malheureusement une solution faible qui possède plusieurs inconvénients. A titre d'exemple, cette architecture n'est pas hautement disponible. Si le serveur HTTP qui héberge Moodle s'arrête, tout le service de télé-enseignement s'arrête. Si nous désirons mettre à jour la plateforme il est nécessaire d'arrêter le serveur. Si nous désirons augmenter la puissance de calcul il est nécessaire de refaire toute une installation. Ainsi cette architecture simple *(Figure 6)* n'offre ni la disponibilité ni l'élasticité.

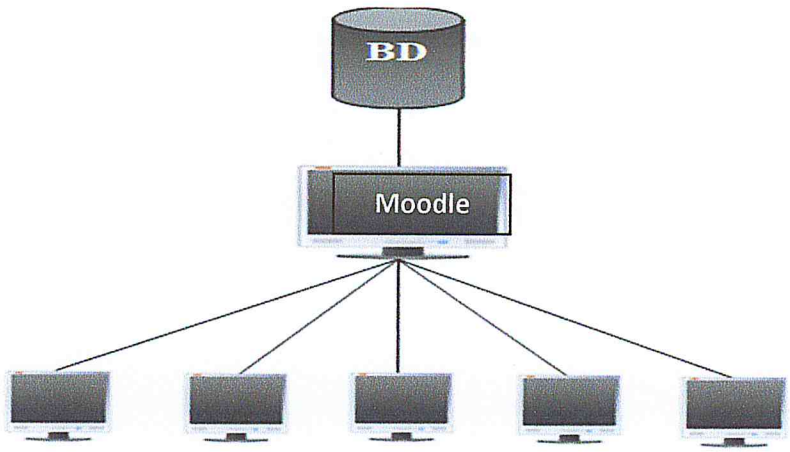

Figure 6 : *Schéma ordinaire.*

2.Architecture de type Cloud

Dans une architecture de type Cloud, l'architecture simple est maintenue du point de vue du client. Le client l'impression d'être en face de Moodle dans son architecture simple. Ainsi le serveur de télé-enseignement CloudMoodle parait exactement le même que le serveur personnel (easyPhP, Lamp ou Wamp) hébergeant Moodle.

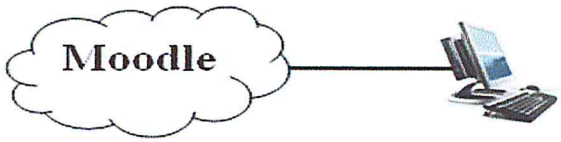

Figure 7 : *la nouvelle approche.*

L'intérieur du Cloud

Le cloud vu de l'intérieur peut avoir des structures diverses selon les objectifs désirés et les performances à atteindre. Toujours est-il que cette structure ou cette architecture devra répondre aux caractéristiques générales du Cloud Computing tel que la haute disponibilité (Résistance aux pannes, maintenance transparentes), le passage totalement transparent à l'échelle (ajout et retrait de serveurs, réorganisation de l'architecture).

Selon les objectifs, cette architecture pourra être plus ou moins différente. Dans ce qui suit nous présentons certaine solution architecturale du *CloudMoodle*.

2.1.Architecture de Base

L'architecture de Base est composée des éléments suivant *(Figure 8)*:

- *1 serveur Frontal*: C'est vers ce serveur que seront réellement dirigées toutes les requêtes des clients HTTP. Ce serveur frontal, selon l'architecture réelle mise en œuvre dans le Cloud, se chargera de diriger les requête vers un ensemble de serveurs, hébergeant chacun l'application Moodle. C'est ce serveur frontal que nous allons concevoir et réaliser pour pouvoir obtenir l'architecture du CloudMoodle. C'est une sorte de Proxy spécifique à Moodle.
- *1 ensemble de serveur Moodle*. Cet ensemble de serveurs est souvent appelé une ferme de serveur. Le serveur frontal cache en fait une ferme de serveurs Moodle.

❖ **Rôle du serveur Frontal**

Pour les clients, le serveur frontal parait comme étant un serveur Moodle. En réalité ce n'est qu'un intermédiaire entre la ferme de serveurs Moodle et le client. La ferme doit paraitre pour le client comme étant un seul serveur. Ainsi l'ajout de serveur à la ferme de serveurs ne fera qu'augmenter la puissance de la ferme de serveurs et ses

performances. Cependant pour permettre cet accroissement de la puissance et l'augmentation des performances, il est nécessaire que le frontal dirige les requêtes client vers les serveurs Moodle de façon assez intelligente permettant par exemple d'équilibrer la charge entre les divers serveurs de la ferme des moodle. Ainsi, lorsque il reçoit une requête il devra déterminer quel est le serveur le moins chargé pour lui assigner la requête à traiter.

❖ **Avantage de l'architecture de base**

- Répartition les charges entre les serveurs.
- la capacité de connexion et traitement est plus importante et peut être facilement augmentée. Elle est proportionnelle au nombre de serveurs présentés dans le CloudMoodle.
- il est possible de dimensionner convenablement le CloudMoodle en fonction des besoins.
- Si un serveur de la ferme tombe en panne, on peut le retirer sans aucune influence sur l'architecture.

❖ **Exigences**

- Le serveur frontal doit être très puissant.

❖ **Inconvénients**

- Si le serveur frontal tombe en panne, tout le cloud service s'arrête. Une solution à ce problème est nécessaire.

❖ **Comportement du serveur frontal**

En fait dans le contexte de cette architecture de base, le serveur frontal pourra avoir divers comportement dont l'un devra être sélectionné dans le contexte d'une opération administration du cloud. Un comportement correspond à une architecture et aussi à la politique d'aiguillage de requêtes aux serveurs dans le contexte d'une solution d'équilibrage de charge. Certains comportements sont généraux et sont valable pour n'importe quelle plateforme de télé-enseignement, voir pour n'importe quelle application centrée sur le Web. D'autres comportement sont spécifique à Moodle et devront être re-réalisé pour chaque nouvelle plateforme. Le serveur frontal devra donc être conçu pour fournir une interface d'administration qui permettra au moins de lui indiquer l'architecture

dans laquelle il doit opérer et la politique de gestion de charge. En réalité d'autres opérations d'administration du cloud sont nécessaires, notamment l'ajout ou la suppression de serveurs de la ferme des serveurs.

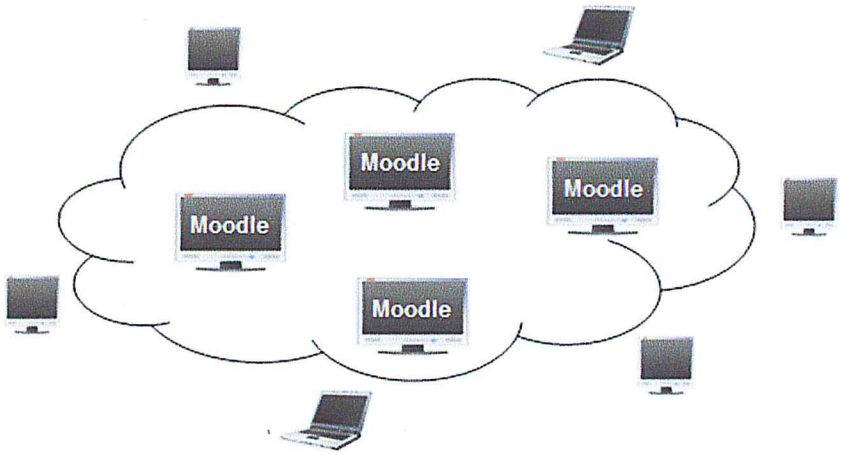

Figure 8 : *schéma générale de cloud.*

❖ **Détail de la ferme des serveurs Moodle dans l'architecture de base**

Un aspect important de l'architecture de base doit attirer notre attention. Cet aspect se situe au niveau de la ferme des serveurs. En effet au niveau de chaque serveur moodle se cache en réalité deux serveurs *(Figure 9)* : Un serveur HTTP (ou Web) hébergeant l'application Moodle et Un serveur de Base de données contenant les informations de Moodle (enseignants, étudiant, cours, activités, documentations, test, forum, chat etc..). Cette structure pose néanmoins un problème dans la réalisation du Cloud, notamment du serveur frontal et rend complexe la réalisation de ce dernier. Pour illustrer cette problématique, prenons l'exemple suivant.

Supposons qu'un enseignant veut créer un test, Il s'adresse au serveur frontal, ce dernier dirige l'enseignant vers l'un des serveurs de la ferme où l'enseignant va créer le test. Si les étudiants de cet enseignant sont appelé a faire le test, il est nécessaire que le frontal les dirige vers le bon serveur. Cette situation a deux inconvénients important :

- *Le frontal doit obligatoirement garder les informations liant étudiant, enseignant, cours et test à son niveau pour qu'il soit capable de retrouver pour chaque étudiant le bon serveur.*

- *Les étudiants qui doivent passer le test le feront sur un seul serveur, ce qui surchargera ce serveur et laissera les autres au repos. Il ne peut y avoir ainsi aucune possibilité de mettre en œuvre une politique de gestion de la charge.*

2.2.Architecture avec frontal replicateur d'information

Une autre solution existe, dans laquelle le frontal ne garde pas les informations de liaison entre les enseignants, cours, étudiants et tests. Dans cette solution le frontal devra à chaque fois répliquer les informations sur toutes les bases de données de tous les serveurs. Si l'utilisateur (enseignant, étudiant etc..) rajoute une donnée dans n'importe quel serveur désigné par le serveur frontal, il faut mettre à jour tous les serveurs restants. Cette opération pourraient être très couteuse, notamment lorsque le nombre de serveurs devient important. Ainsi dans cette solution le passage à l'échelle pourrait devenir un problème

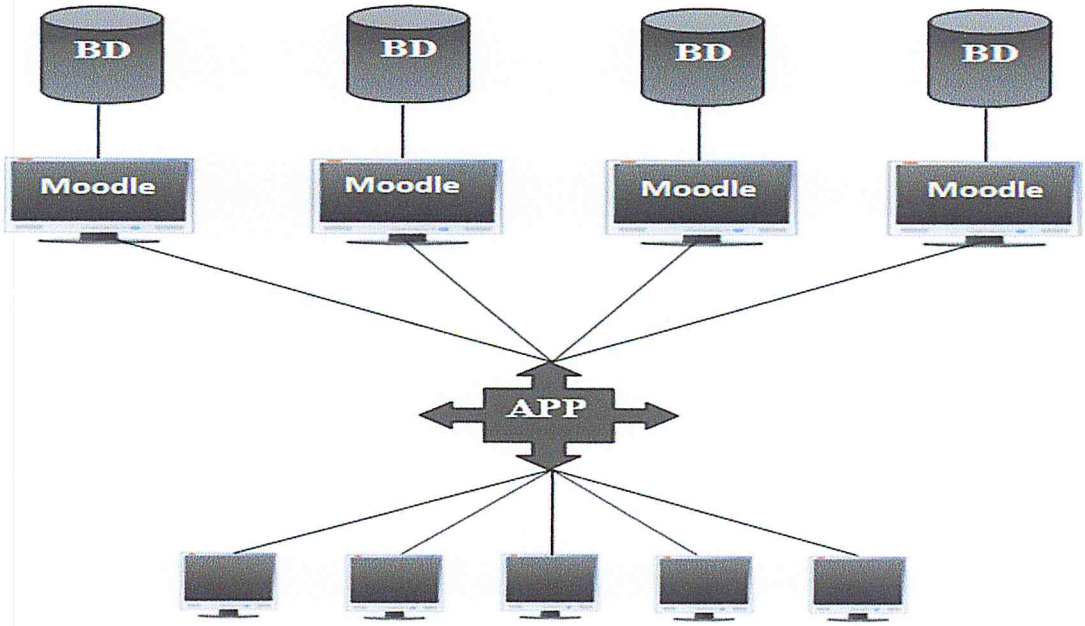

Figure 9 : *Solution proposée 01.*

2.3.Architecture de base améliorée

Cette architecture est une idée que nous avons élaborée nous-même et n'est pas indiquée ou recommandé par Moodle, vu que Moodle de par sa conception n'est pas un Cloud et qu'il y a toujours une configuration fixe qui dit que les serveurs sont indépendant et que pour chaque serveur Moodle il est nécessaire d'associer un serveur de base de donnée.

Dans cette solution que nous avons vérifiée expérimentalement, nous utilisons une seule base de données pour tous les serveurs comme le montre *(Figure 10)*. Ainsi toutes les plateformes moodle installés sur les divers serveurs Web sont reliées à une et une seule base de données (*Figure 10*). De ce fait les serveurs Moodle interagissent avec un seul serveur de base de données qui pourrait à son tour un cloud de gestion de données.

❖ **Comment cette architecture a été validée**

La validation de cette architecture a été faite de manière expérimentale, après qu'on a séparé la base des données de la plateforme moodle dans un serveur propre, nous avons installé la plateforme moodle dans la ferme des serveurs, puis on a configuré chaque plateforme pour la connecter à la base des données qu'on a séparé., afin de nous assurer que la réponse d'une requête vers un serveur est la même vers un autre serveur de la ferme.

Cette proposition nous a permis d'éviter tous les problèmes qu'on a rencontrés avec la proposition précédente dont la mise à jour des bases des données.

❖ **Avantages et inconvénient de l'architecture de base améliorée**

Cette architecture possède de réels avantages par rapport à la première.

- *Elasticité et passage à l'échelle*: l'augmentation du nombre de serveurs de la ferme sera facile.
- *Haute disponibilité* : en cas d'une panne de l'un des serveurs de la ferme, cloud reste toujours en marche.
- *Le frontal est moins complexe et plus rapide* car il ne s'occupe plus de la réplication des données.

Toujours est-il que cette solution possède des Inconvénients, notamment le problème de l'arrêt du cloud si le frontal s'arrête ou si le serveur de base de donnée s'arrête. Une solution type Cloud pourrait être imaginée pour la partie base de données et pour la partie frontale pour résoudre le problème de haute disponibilité

L'architecture de base améliorée semble être plus intéressante que la première malgré les inconvénients cités et qui possèdent des solutions qu'il faudrait juste mettre en place comme nous le verrons par la suite.

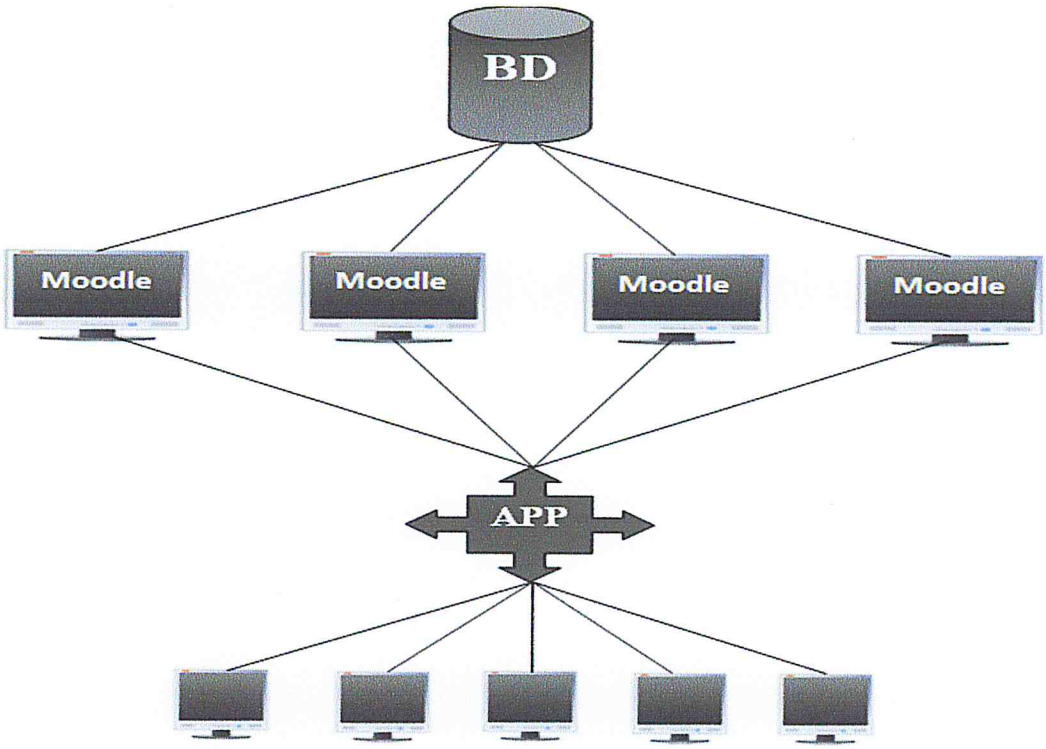

Figure 10 : *Solution proposée 02.*

2.4.Architecture à base d'un cloud pour le frontal

Le but de cette architecture est de résoudre le problème de haute disponibilité du serveur frontal et ne pas permettre au Cloud Moodle de s'arrêter si le serveur frontal s'arrête. Le problème crucial qui se pose c'est comment réaliser le Cloud qui se charge de la partie frontale. Il est clair que la solution n'est pas de réaliser un frontal de frontaux, car nous retomberons dans le même problème. Une approche totalement différente est nécessaire.

Pour réaliser le cloud du frontal, nous allons faire appel à un service très sollicité d'Internet. C'est le service de DNS et plus exactement le DNS doté d'un service de gestion de charge appelé aussi un Round Robin DNS.

❖ **Rappel sur le DNS**

De par son architecture, le DNS est un service hautement disponible. La configuration basique d'un serveur DNS au niveau du client DNS nécessite la spécification d'au moins deux serveurs DNS. Le deuxième est automatiquement sollicité dans le cas de l'absence du premier.

Dans un contexte ordinaire, à chaque nom DNS (exemple *www.univ-blida.dz, elearning.univ-blida.dz, mail.univ-blida.dz)* correspond une adresse IP. Parfois, plusieurs services, donc serveurs, sont installés sur une même machine. A titre d'exemple nous pouvons trouver le service de messagerie et le serveur web installé sur une même machines Chaque service étant accessible avec un nom DNS particulier (*www.univ-blida.dz* pour le web et *mail.univ-blida.dz* pour la messagerie), il est alors d'usage de déclarer au niveau du serveur DNS ces divers noms symboliques associés à une même adresse IP qui est celle de la machine qui héberge les deux services. Ainsi, dans cette situation nous avons une machines ayant plusieurs surnoms**.** L'adresse IP représente le nom (ou référence) réel d'une machines. Chaque nom mène à la même machine dans le processus de résolution de nom que le DNS supporte.

Lorsque le serveur de nom (DNS) est sollicité pour la résolution d'un nom (rechercher l'adresse IP qui correspond à un nom symbolique donné tel que *elearning.iuniv-blida.dz*), il recherche le nom dans sa base de donnée, trouve la correspondance avec une adresse IP et renvoie l'adresse IP. Si la machines a été déclarée avec divers noms symbolique, la résolution de chaque nom symbolique produit la même adresse IP qui est celle de la machine.

❖ **Le Round Robin DNS**

Avec le Round Robin DNS, la situation est autre. L'objectif est de spécifier un même nom, mais à chaque fois le serveur doit renvoyer comme réponse une adresse IP différente de la précédente, donc une machine différente. Ainsi si nous avons plusieurs serveurs frontaux, chaque serveur frontal possède une adresse spécifique unique sur Internet ou dans un Intranet. Lorsque ces serveurs sont découvert par un round Robin DNS, à chaque fois qu'une résolution de nom est sollicitée par une requête utilisant le nom général du cloud (par exemple *elearning.univ-blida.dz*) le Round Robin DNS principal, ou le serveur Round Robin DNS de secours (ou secondaire), renvoie une l'adresse IP d'un frontal différent du premier. Ainsi grâce au Round Robin DNS, les requête du point de vue du client semblent être adressée à une même machine, alors qu'en réalité, plusieurs requêtes successives sont prise en charge chacune par un serveur frontal différent. Grace à cette technique nous pouvons facilement mettre sur pied un cloud pour le frontal afin d'assurer sa haute disponibilité. Le passage à l'échelle au niveau frontal est très simple, car il faut juste déclarer au Round Robin DNS les nouveaux serveurs frontaux qui feront partie du cloud de la partie frontal.

L'architecture basée sur le Round Robin DNS est ainsi composé de plusieurs frontaux, chacun possédant sa propre adresse IP. Ces frontaux possèdent cependant le même nom DNS, par exemple **elearning.univ-blida.dz.** Ce sera au serveur DNS d'équilibrer la charge entre les serveurs frontaux. **Cette architecture** assurera ainsi une haute disponibilité. Si l'un de serveurs frontaux tombe en panne, les requêtes sont redirigées vers les autres serveurs frontaux disponibles.

Le serveur DNS **lbnamed**[1] de la Stanford University semble être la solution la plus intéressante pour le round Robin DNS. En effet ibnamed ne se suffit pas seulement de rendre à chaque requête une ip differente (une ip d'un des serveurs frontaux du cloud des frontaux), mais il permet aussi de détecter si un de ces serveurs n'est plus disponible suite à un problème ou devient disponible.

[1] http://www.stanford.edu/~riepel/lbnamed/lbnamed-2.3.2.tar.gz

❖ **Variante de l'architecture du Cloud basée sur un Cloud de Frontaux**

En utilisant un Cloud de frontaux, nous pouvons imaginer diverses architectures qu'il est possible de mettre sur pied dynamiquement. L'architecture globale est reconfigurable. Cet aspect de reconfiguration pourrait être réalisé automatiquement en réponse à des situations bien précise comme elle peut être décidée dans le contexte d'une activité d'administration du cloud.

➢ **1ere Variante :**

Liaison total entre le cloud frontal et la ferme des serveurs Moodle.

Dans cette architecture tous les serveurs de la ferme sont à la disposition de chaque serveur de la partie frontale. Nous avons ainsi un réseau totalement maillé allant des frontaux vers les serveurs de la ferme comme le montre la (*Figure 13).*

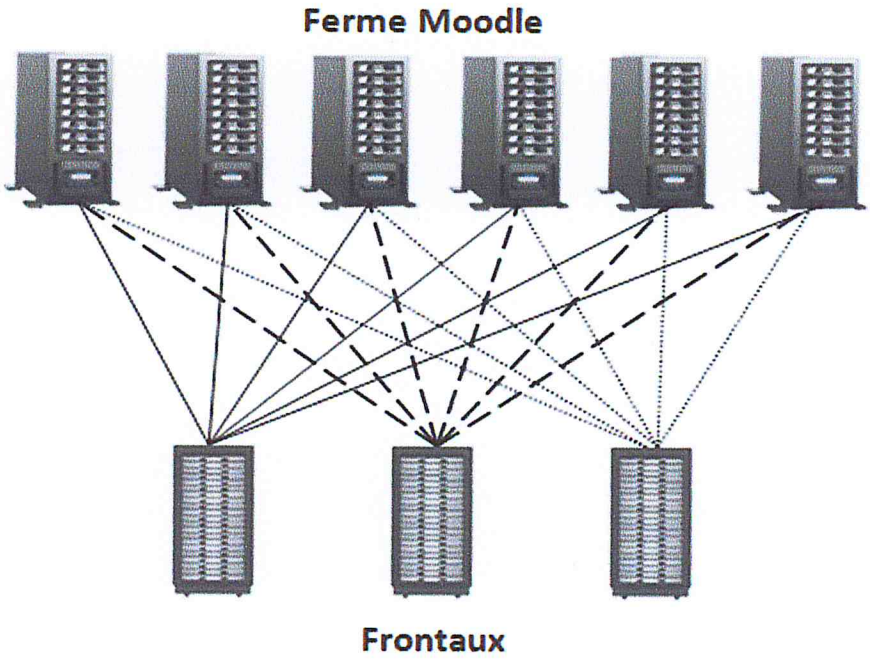

Figure 13 : *Liaison entre le cloud frontal et la ferme.*

Le problème de cette architecture réside dans la nécessité pour les frontaux de coordonner leur effort dans l'opération de gestion de charge. Pour bien cibler un serveur moins occupé, le serveur frontal doit avoir l'information sur la charge qui a été affecté par

les autres frontaux aux serveurs de la ferme Moodle. Réaliser un frontal collaborateur entrainera une complication de la réalisation du frontal. Une démarche modulaire (ou basé sur le concept de composant logiciel) pourrait nous mener à la réalisation d'un frontal qui aura par exemple le module de collaboration comme un module optionnel pouvant être installé ou non.

❖ **Architecture 1 frontal pour une partie de la ferme**

Dans cette architecture chaque serveur frontal est associé avec un nombre bien précis de serveur de la ferme des serveurs. Les sous-ensembles de serveurs Moodle associé aux frontaux sont disjoints. Lorsqu'un frontal ne devient plus disponible, il est nécessaire d'affecter équitablement les serveurs qui lui étaient associé aux autres serveurs frontaux. L a reprise du serveur frontal implique la récupération de ses serveurs moodle. (*Figure 14*)

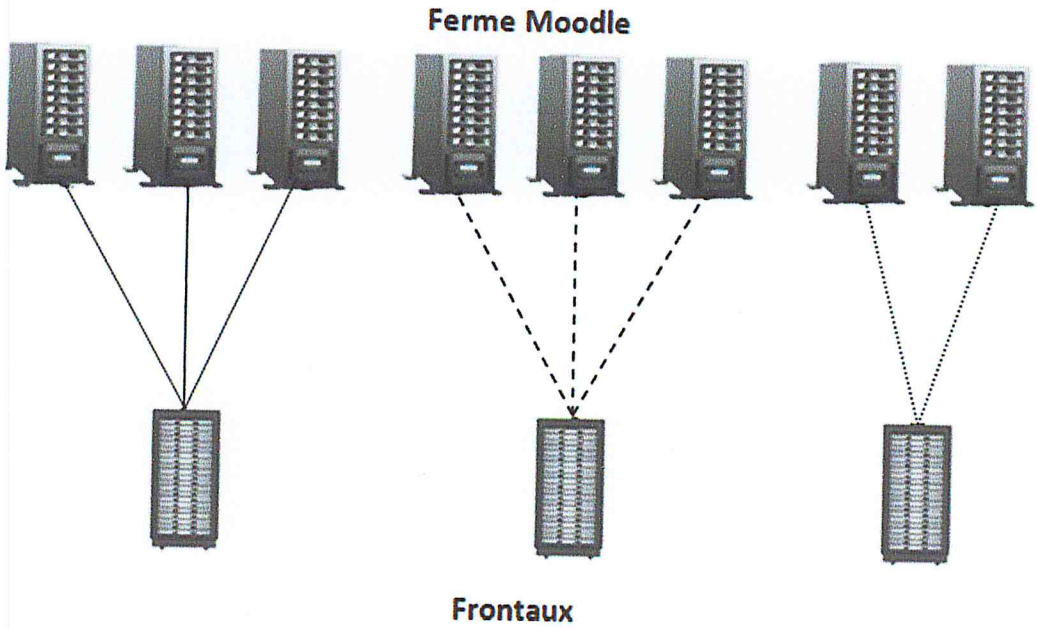

Figure 14 : *Architecture un frontal pour une partie de la ferme.*

3.La répartition des charges dans le CloudMoodle

La définition d'une architecture a pour objectif de répondre à divers critère, notamment le critère de haute disponibilité. L'architecture aura aussi un impact sur la complexité du logiciel qui représentera le frontal. Cependant, quel que soit la complexité ou la simplicité du serveur frontal, la fonctionnalité de gestion, répartition et équilibrage de la charge est une fonction fondamentale du serveur frontal. Dans ce qui suit nous allons passer en revue quelques approches que nous proposons pour la répartition et l'équilibrage de charge. Certaines de ces approches seront implémentée et représenterons ainsi une part du comportement programmable du frontal.

3.1.Algorithme à base de tables de Hashage

Cet algorithme construit une table de correspondance à partir de l'adresse IP du client pour rediriger celui-ci vers le même serveur à chaque nouvelle requête, ceci durant un laps de temps prédéterminé. (*Figure 15)*

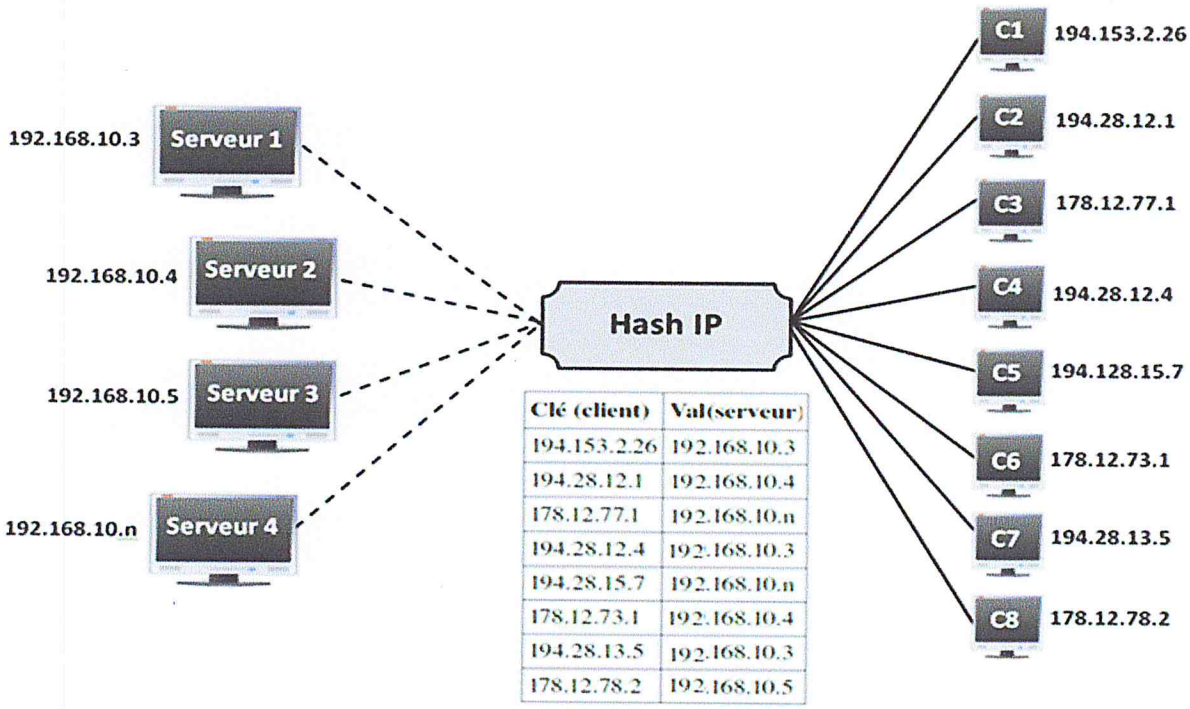

Figure15 : *Hash IP.*

Cet algorithme peut poser des problèmes lorsque les clients accèdent au Cloud Moodle en passant par un même proxy *(Figure 16)*. Ces mêmes clients seront redirigés vers le même serveur, vu que pour le Cloud tous ces clients sont représentés par un même client qui est le proxy qu'ils utilisent pour accéder au Cloud. Si le nombre de client passant à travers un même proxy est important, le cloud sera déséquilibré et sa performance sera en dégradation pour ces clients passant à travers un proxy.

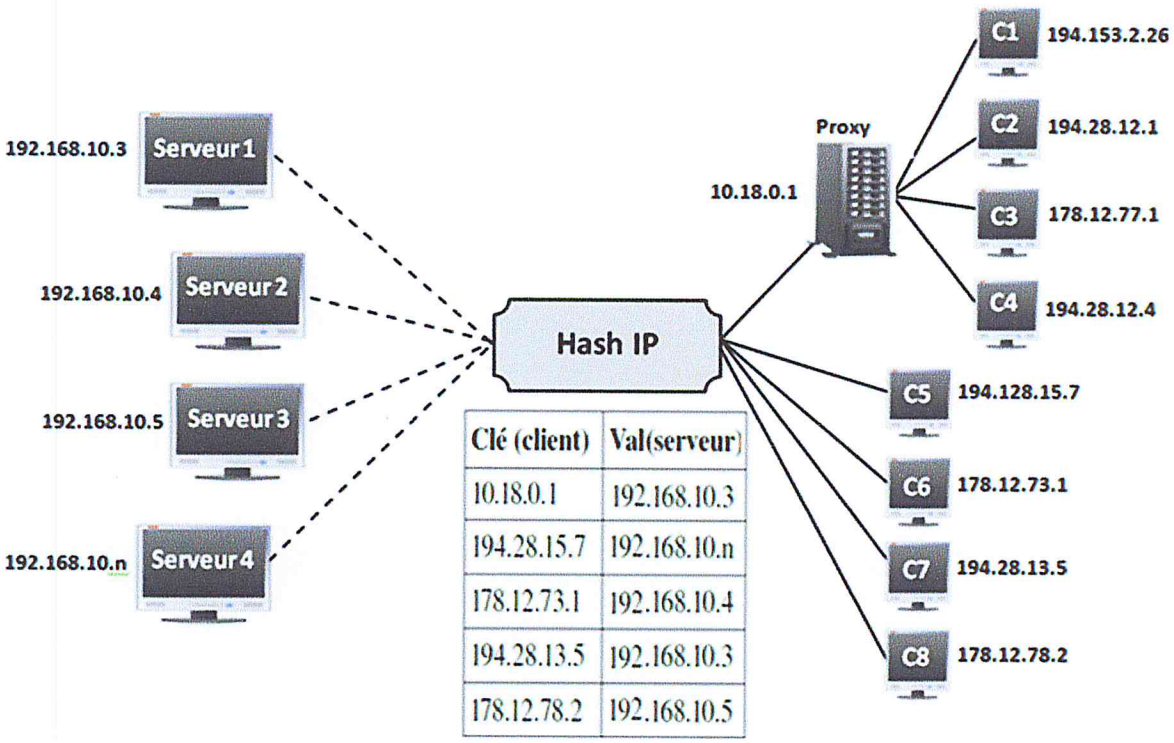

Figure 16 : *Hash IP à la présence de proxy.*

3.2.Algorithme du premier répondeur

Les requêtes clients sont envoyées simultanément à tous les serveurs et seule la réponse du premier qui répond sera considérée.

3.3.Round Robin

Cet algorithme maintient une liste de serveurs. La répartition round-robin consiste à n faire le tour des serveurs. Lorsque le dernier de la liste a été chargé, nous revenons au premier de la liste. *(Figure 17)*. Dans le round robin ordinaire, la décision de déterminer quel est le serveur quoi sera destinataire de la prochaine requête est simple : c'est le

suivant. Cependant, dans le contexte de notre travail, nous avons pu identifier plusieurs variantes du round robin.

3.3.1.Association en Round-Robin de clients au serveur

Dans cet algorithme, lorsqu'un nouveau client est déterminé, on l'associe au serveur suivant de la liste circulaire et toutes les requêtes du client iront vers ce serveur jusqu'à ce que le client termine sa session ; Les informations de sessions doivent être accessibles à cet algorithme. Nous tenons à remarquer à ce niveau que le travail ordinaire dans Moodle se fait toujours dans le contexte de session. Très rare sont les requêtes qui sont exécutées hors session. Même un visiteur anonyme possède une session avec des privilèges très limités.

Si on suppose qu'on a trois serveurs (S1,S2, S3) et quatre clients (C1, C2, C3, C4), C1est dirigé vers S1, C2 vers S2, C3 vers S3 et C4 vers S1. Si le client C1 envoie une autre requête, celle-ci est dirigée vers le serveur S1.

3.3.2.Round-Robin appliqué au niveau requête

Dans cet algorithme les requêtes d'un même client sont reparties entre l'ensemble de serveurs. Lorsque les requêtes appartiennent à une même session, il est nécessaire que le frontal garantisse qu'au niveau du serveur ou la requête est dirigée, la requête sera prise en charge dans sa session ordinaire. Pour atteindre cet objectif, il est nécessaire d'ouvrir la même session sur tous les serveurs Moodle. Dans cette approche, le frontal doit examiner le contenu des requêtes et agir en conséquence.

3.3.3.Association des clients au serveur en se basant sur un indicateur de charge

Cette approche a été introduite pour améliorer l'équilibrage de charge au niveau des serveurs de la ferme des serveurs Moodle. L'idée principale est se doter d'une liste pondérée de serveurs. Le prochain serveur à sélectionner n'est pas forcément le suivant. Ce sera le serveur le moins chargé. Ainsi avant de décider avec quel serveur un client sera associé, on consulte la liste et nous sélectionnons le serveur le moins chargé. A chaque fois qu'un client est associé à un serveur, l'état de charge de ce serveur est augmenté de 1. Lorsque le serveur est déchargé du client, sa pondération diminue de 1. L'affectation d'un client à un serveur et l'opération de désaffectation du client du serveur se base sur des informations de session.

3.3.4.affectation de requête aux serveurs basée sur l'état des serveurs

Nous suivons les mêmes procédures que l'approche précédente. La charge d'un serveur est maintenue dans une variable d'état associé à chaque serveur. Cette fois ci, cette variable est augmentée à chaque fois qu'une requête est dirigée à un serveur. La variable est décrémentée est décrémenté après réception de la réponse.

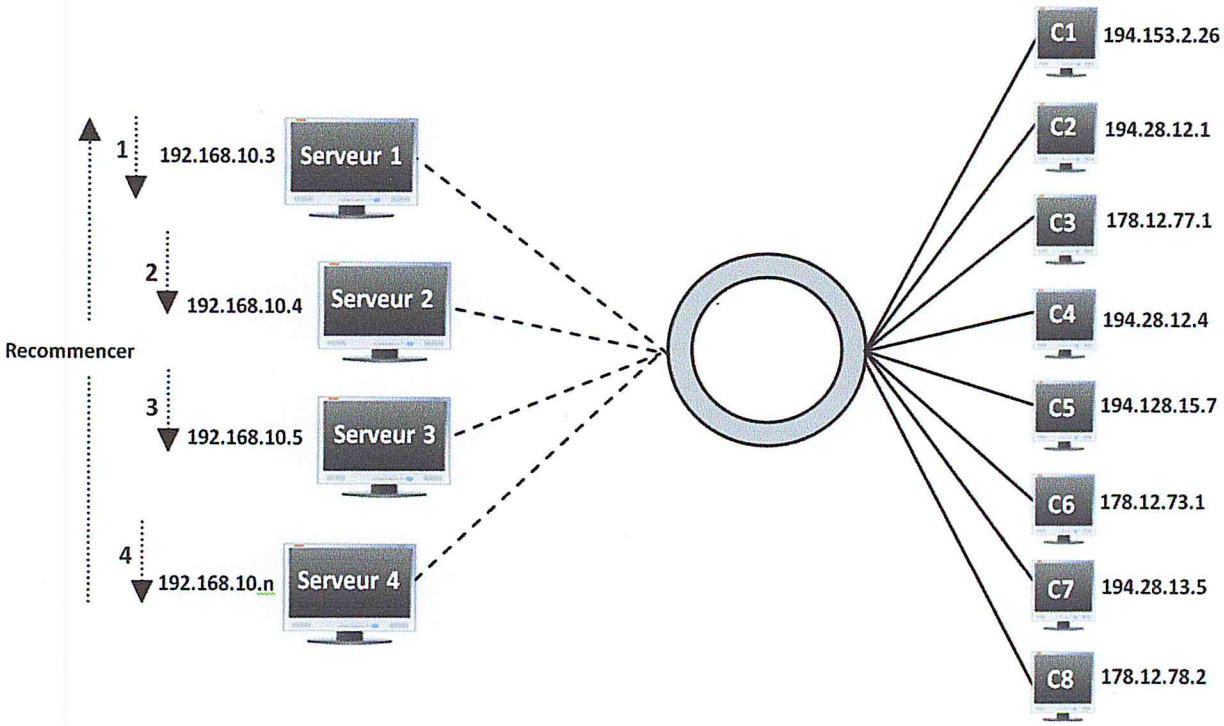

Figure 17 : *Round Robin.*

3.3.5.Affectation du client jusqu'au chargement d'un serveur

Dans cette politique le frontal dispose d'une liste de serveur doté de plusieurs pondérations fixe et variables. Chaque serveur est associé à deux poids fixe et une charge actuelle. Les deux poids fixe sont respectivement la charge maximale et la charge moyenne d'un serveur. Si des serveurs de puissance distincts sont utilisés, la charge maximale et moyenne de chaque serveur serait en principe différente de celles d'un autre

serveur. La fixation des charges maximales et moyennes est une opération d'administration du frontal.

Lorsqu'un client envoie une requête, il est alors associé au serveur courant si sa charge courante est inférieure à la charge maximale. Le passage au serveur suivant n'aura lieu que lorsque la charge moyenne du serveur courant aurait été atteinte. Lorsqu'un client n'est plus associé au serveur (fin de session) la charge courantes est alors réduite de 1. Lors de l'affectation d'un serveur au client, le suivant n'est pas obligatoirement le suivant dans l'ordre mais celui qui suit et ayant une charge inférieure à la moyenne.

Lorsque la charge moyenne de tous les serveurs aurait été atteinte, l'affectation continue à se faire en déterminant le serveur le moins chargé. Une fois le Max atteint au niveau d'un serveur, celui-ci ne sera plus disponible pour les autres requêtes. Le frontal pourrait dans ce cas avertir les clients de l'indisponibilité temporaire du cloud.

Il faut tout de même noter à ce niveau que les politiques basées sur la charge réelle pourraient faire face à un problème de la précision de l'information concernant la charge. Ainsi, il est tout à fait possible de trouver un serveur trop chargé alors qu'en réalité il ne l'est pas, car il se trouve dans une situation d'attente du résultat d'une requête du serveur de base de donnée ou du cloud représentant la base de donnée

4.Conception du Frontal

Dans cette section nous allons présenter les aspects liés à la conception du serveur frontal.

Rappel des Objectifs

Le rôle principal du serveur frontal est d'appliquer une politique bien précise de soumission des requêtes aux serveurs de la ferme des serveurs Moodle dans le contexte d'une architecture bien précise. Nous venons de présenter divers approches de soumission de requêtes aux serveurs. Chaque approche représente en fait une politique de soumission des requêtes. Le basculement d'une politique à une autre se fait dans le contexte de l'administration du serveur. Ainsi globalement le frontal est composé de deux grandes parties

- La partie administration
- La partie métier du frontal

4.1. La partie administration

C'est dans cette partie qu'il est possible de configurer le frontal pour qu'il puisse mettre en œuvre une politique bien précise et pour lui indiquer quels sont les serveurs de la ferme qu'il doit considérer dans sa politique. La partie administration comporte aussi toutes les actions de suivi, d'ajout et suppression de serveurs. Le diagramme des cas d'utilisation schématise les fonctionnalités principales de la partie administration du frontal.

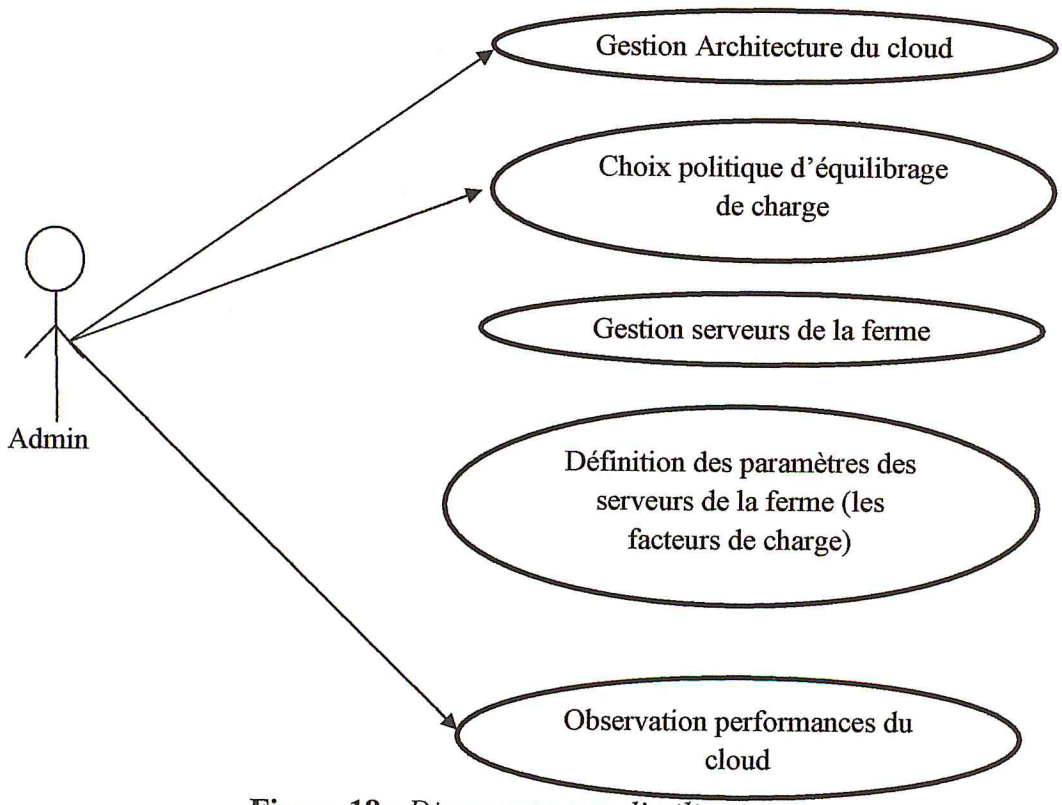

Figure 18 : *Diagramme cas d'utilisation.*

4.2. La partie métier

Le comportement de la partie métier dépend de la politique de l'interaction entre le frontal et les serveurs de la ferme. Elle englobe deux types de fonctionnalités :

- Les fonctionnalités fondamentales.
- Les fonctionnalités de surveillance de l'architecture.

4.2.1.Fonctionnalités fondamentales

Dans cette catégorie de fonctionnalité nous retrouvons ce qui suit :

- Interception des requêtes HTTP émanent des clients.
- Analyse des requêtes lorsque la politique de fonctionnement l'exige.
- Gestion de session si la politique l'exige.
- Déterminer, selon la politique, le serveur vers lequel la requête sera dirigée.
- Interception des réponses aux clients.
- Analyser si nécessaire les éléments de réponse.
- Déterminer le client destinataire de la réponse.
- Diriger la réponse vers le client.

4.2.2.Fonctionnalités de surveillance

Ces fonctionnalités ont pour objectif la surveillance de la structure de la ferme afin de permettre d'une part une haute disponibilité et d'autre part assurer la performance du cloud. Parmi ces fonctionnalités nous retrouvons :

- Détection de serveur hors d'usage.
- Détection d'un serveur qui serait de retour après panne ou de nouveau serveur Un serveur serait retiré de l'architecture ou ajouté.
- Restructuration de l'architecture suite à l'arrivée ou l'indisponibilité d'un serveur

L'architecture globale du frontal est indiquée dans la figure suivante :

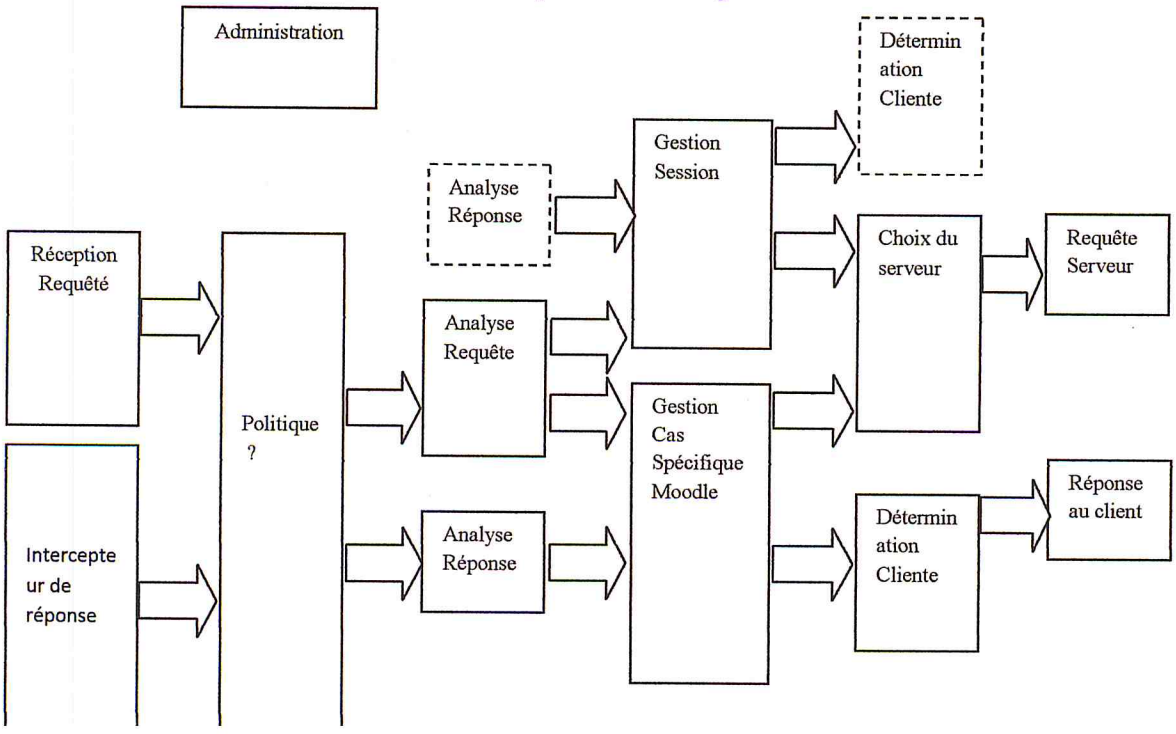

Figure 19: *Architecture globale du frontal*

Remarque : Pour la simplicité du schéma, les boites en pointillé représente la boite en trait plein ayant le même nom. C'est le même composant.

5.Stratégie de conception

Notre objectif est de réaliser un serveur pouvant être configuré pour fonctionner selon la politique choisie. Si nous optons pour une stratégie de conception réalisation monolithique (toutes les fonctionnalités dans une seule application), la réalisation du frontal deviendra difficile, complexe et certainement mal maitrisable. De plus nous savons que ces politiques sont disjointes. Lorsqu'une politique est choisie c'est toute l'application qui balance sur cette politique. Les éléments des autres politiques sont présents mais non fonctionnels.

UN autre élément fondamental de la stratégie à mettre sur pied pour la conception est le fait que notre application est une Application accessible par l'intermédiaire d'un serveur HTTP . Dans ce contexte toute opération de reconstruction ou mise à disposition d'élément de l'application se fait au niveau du serveur, de manière complètement transparente au client

A la base de ces remarques nous optons pour la stratégie suivante : Réaliser pour chaque politique un frontal dédié. Au niveau du site si nous avons n politique à mettre en œuvre nous aurons alors n frontaux, chaque frontal spécialisé pour la réalisation efficace de la politique choisie. Mais comment passer réellement d'une politique à une autre après le choix. La solution, qui est de niveau technologique est très simple. Du point de vue du client il y a une application qui s'appelle moodle. Ce sera le nom du frontal. Chaque politique **i** porte le nom de **moodle_i**. Lorsque l'administrateur choisi une politique **x** pour le frontal, le **moodle_x** sera transformé en moodle. Plusieurs techniques de transformations simples existent. Les plus usuelles sont :

- Recopier l'application **moodle_x** dans la zone moodle.
- Redéfinir un lien entre moodle et **moodle_x**.

Dans cette stratégie, le composant état frontal sera supprimé de toutes les applications représentant le frontal dans une politique particulière. Chaque frontal ne contiendra que le composant qui muni permettent fonctionner efficacement.

Nous tenons en fait à remarquer que nous sommes arrivé à cette solution après avoir fait diverses implémentations, durant lesquelles nous étions à chaque fis confronté a des problèmes spécifiques. Ces problèmes n'existent que dans certaines politiques. Alors pourquoi les considérer et les résoudre dans le contexte d'une politique dans laquelle ces problèmes n'ont pas lieu d'exister. La conclusion à laquelle nous sommes arrivés est la suivante: IL est plus efficace de réaliser plusieurs applications légères que de réaliser une grosse application.

Conclusion

Dans ce chapitre, nous avons présenté tous les approche qui il y'en a proposé, les détails, les avantages, les inconvénients de chaque approche. Et on a choisi la deuxième approche, puisque on a jugé après plusieurs essaies et après la recherche sur l'internet à des problèmes similaires que notre choix est le plus près à réaliser.

Dans le chapitre suivant, nous présentons l'implémentation de ce système.

Chapitre 6

Réalisation

Chapitre 6 : implémentation

Introduction

Un cloud offrant une application comme service est souvent associé à un type d'application particulière, malgré le fait qu'un Cloud peut délivrer de multiples applications comme services. Un cloud offrant une application comme service, nécessite pour sa mise en place une compréhension assez fine de l'application et de l'environnement nécessité pour sa mise en œuvre. En général, un cloud est accessible dans le contexte d'Internet. Techniquement c'est une application qui centrée sur le Web. Cette application va permettre aux clients de type HTTP (navigateur Web notamment d'accéder à ces services). Ainsi l'accès à ces services se fait dans le contexte du protocole HTTP. Ainsi pour la mise en place (réalisation) d'un Cloud deux aspects sont à maitriser :

- Le protocole HTTP
- L'application à fournir comme service par le Cloud. Dans notre cas c'est Moodle.

La compréhension de HTTP n'est pas vraiment un défi ou est très délicate. C'est un protocole Standard et la documentation officielle (RFC 1945 pour HTTP1.0 et RFC 2616 pour HTTP1.1) existe sur Internet en plus d'un nombre très important de documents accessible sur Internet.

Le défi à relever concerne la plateforme Moodle. La documentation d'installation, de mise en œuvre et d'exploitation existe et est même très riche. Cependant la documentation et conception semble être difficile d'accès et non détaillé. Dans notre cas le détail très fin indiquant la politique suivi par Moodle dans l'élaboration des diverses requête HTTP dans ses divers états est nécessaire pour pouvoir réaliser de manière efficace la fonctionnalité du frontal relative à l'analyse du contenu d'une requête. Pour relever ce défi, nous avions eu recours de manière intense à un analyseur de requête HTTP. C'est par l'analyse fine des informations reportés par l'analyseur des requêtes HTTP que nous avions pu comprendre comment Moodle élabore la plupart des requête dans les diverses conditions. Il faut

noter que cette approche pourrait ne pas prendre en compte certain état spécifique des requêtes. IL est en effet nécessaire de faire un tour de tous les états de Moodle pour pouvoir affirmer que le frontal est sans faille.

Dans ce qui suit nous allons commencer par un bref rappel des concepts fondamentaux du protocole http et qui seront utilisé dans la réalisation du frontal.

1.Le protocole http

C'est un protocole de niveau application, basé sur le protocole de niveau transport TCP. Il Permet d'accéder aux diverses ressources gérées par un serveur HTTP (Web). Il Fonctionne selon le principe de *requête/réponse* : Le client HTTP transmet une *requête* comportant des informations sur la ressource demandé. Le serveur renvoie les ressources demandées ou un message d'erreur dans le cas de l'impossibilité d'accès aux ressources demandées.

HTTP est un protocole SANS ETAT Chaque couple *requête/réponse* est totalement indépendant des autres couples *requête/réponse*.

Les ressources sont accessibles à l'aide de Méthode. Les méthodes du protocole HTTP1.0 sont **Get, Post et Head.** La Version 1.1 de HTTP enrichit la version 1.0 par 5 nouvelles méthodes **Options, Put, Delete, Trace** et **Connect**.

Une requête transmise au serveur comprend Une ligne de requête (*request-line*) qui indique la méthode suivie d'une ou plusieurs lignes d'en-têtes, Chacune comportant un nom d'entête et une valeur.La requête peut optionnellement contenir un contenu exploité souvent pour transmettre des paramètres de la méthode **POST**.

Une Réponse http est composée d'une ligne d'état contenant la version de HTTP utilisée.et un code d'état. Cette ligne d'état est suivie d'une ou plusieurs lignes d'en-têtes, chacune comportant un nom d'entête et une valeur.

Le protocole HTTP ne possède pas de mécanisme de session. Pour pallier ce manque, *Netscape* [1] a défini le concept de *cookies* afin de fournir à HTTP un mécanisme de gestion d'états. Les *cookies* vont permettre au serveur de mémoriser des données du côté client. Cela permet un suivi du client d'une requête à l'autre et d'implémenter ainsi la notion de session. **[BEN].**

[1]- Netscape: navigateur web de la société *Netscape Communications*

2.Technologies utilisés

Malgré que Moodle est réalisée complétement à l'aide de PHP et utilise un serveur Web ordinaire tel que Apache, le frontal par contre a été réalisé dans un environnement technologique différents.

La technologie utilisée est fondamentalement basée sur Java. La construction de l'interface du frontal est réalisée en JSP (*Java Server Page*). La logique Métier est réalisé en Java. Le serveur HTTP utilisé est Apache Tomcat qui est doté d'un moteur d'exécution de Servlet Java.

Nous avons utilisé la technologie Java pour deux raisons principale :

- Connaissance Préalable de Java : Java est une technologie que nous avions apprise, mise en œuvre et exploitée durant notre cursus Universitaire, notamment en M1 dans le module POO et en M2 dans le module Architecture Logicielle. Changer de langage juste pour changer ne nous ai pas paru intéressant.

- Haute Performance des Servlet Java: Les Servlet Java sont des taches résidente en mémoire. Contrairement à PHP, il n y a pas de phase de chargement, interprétation et exécution. Une Servlet Java s'exécute directement, ce qui pourrait avoir une conséquence intéressante sur la performance du frontal en termes de temps de réponse.

La figure 20 montre les divers serveurs utilisés dans le Cloud

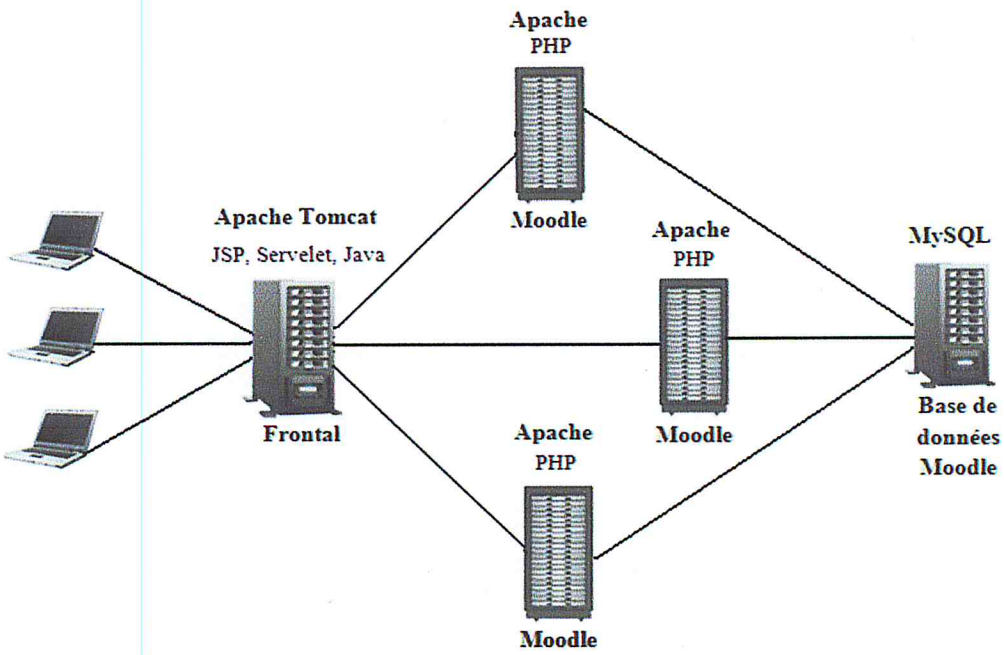

Figure 20: *Divers serveurs utilisés dans le cloud.*

Pour avoir un *cloudmoodle* portable (plus exactement un frontal portable, car c'est le frontal que nous réalisons) nous avons utilisé deux systèmes d'exploitation pour tester le *cloudmoodle* : Microsoft Windows XP et Linux UBUNTU. Parfois nous n'utilisons que du Windows XP, Parfois Uniquement du linux et parfois nous utilisons simultanément les deux systèmes (par exemple : Le frontal sur Windows XP, la ferme de serveurs Moodle utilise Linux Ubuntu, le serveur de base de données utilise Linux Ubuntu.

2.1.Configuration de la plateforme Moodle

La plateforme moodle est une application PHP qui utilise le gestionnaire de base de donne '`MySQL`' pour stocker ses données dans une base de données baptisée '`moodle`' qui contient 250 tables.

Pour atteindre notre Objectif principale (le schéma proposé dans le chapitre précédant – *Figure 8*), il faut que nous modifiions les paramètres de la plateforme 'moodle' et du 'MySQL'.

La *Figure 21* illustre notre objectif :

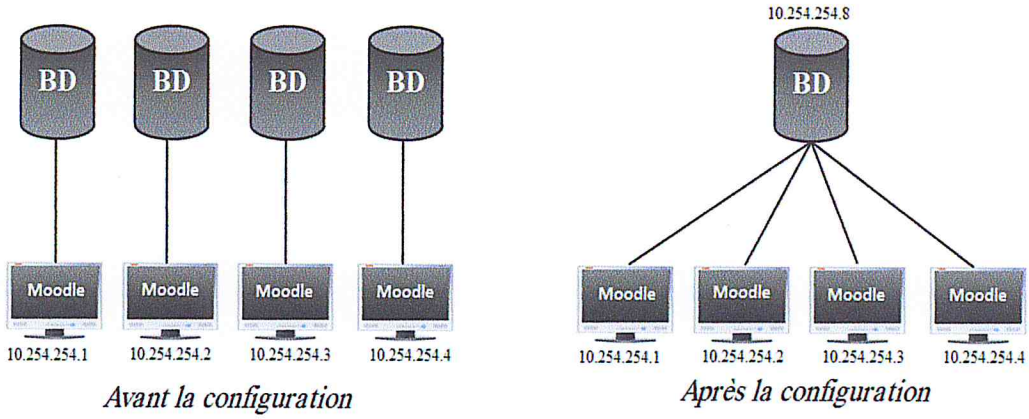

Figure 21 : *schémas illustre notre objectif de la configuration.*

2.1.1.Configuration du moodle :

- On se dirige vers le fichier 'confige.PHP' qui se situe dans le dossier 'moodle' qui se trouve sur 'C:/wamp/www/' sur Windows, et sur le lien '/var/apache2/localhost/' sur Ubuntu server.
- La variable '$CFG->dbhost=localhost' signifie l'adresse IP du server qui contient la base de donnée, par défaut il est initialisé par 'localhost', donc cette adresse doit être remplacé par l'adresse IP du serveur qui contient la base de donnée. Par exemple si on veut mettre le serveur '10.254.254.8', un serveur de base de donnée, il faut que nous modifions la variable '$CFG->dbhost=localhost' du fichier 'confige.PHP' sur tous les serveurs pour être comme ceci : '$CFG->dbhost=10.254.254.8'.

2.1.2.Configuration du MySQL:

Le SGBD MySQL est un système qui ne permet pas aux utilisateurs l'accès à ses donnes qu'à celles qui sont bien défini sur sa table de privilège.

Les utilisateurs des MySQL sont définis par un nom de l'utilisateur, mot de passe, de quel serveur les requêtes sont venues et sur quelle base de données et avec quel droit d'accès. Donc il faut que nous ajoutions les nouveau users.

❖ **Sous Windows**

On peut ajouter un nouveau utilisateur sur Windows soit par l'outil 'phpmyadmin' à partir de l'onglet 'privilège' puis 'ajouter un utilisateur'.

Ou, par l'invite de commande de Windows 'cmd' comme suit :

- La variable 'path' du système doit être initialisé par l'emplacement de 'mysql.exe'. par exemple dans notre cas :

```
Path='C:\wamp\bin\mysql\mysql5.5.16\bin';
```

- Après le lancement de 'cmd', on exécute les commandes suivantes :

```
mysql -u root -p
```

✓ La commande au-dessus est pour démarrer MySQL en mode 'root', il va nous demander le mot de passe, nous l'insérons s'il y' en a.

```
CREATE USER 'root'@'10.254.254.1';
```

✓ Cette commande est pour créer un utilisateur 'root' sans un mot de passe du serveur '10.254.254.1', nous répétons cette commande tant que le nombre de serveur du cloud. Puis on le donne son privilège par la commande suivante :

```
GRANT ALL PRIVILEGES ON *. * TO 'root'@'10.254.254.1'
WITH GRANT OPTION;
```

✓ Puis on exécute ces commandes pour relancer MySQL :

```
flush privileges ;
exit ;
sudo service mysql reload
```

Remarque : tous ces commandes SQL doivent être exécutées seulement sur le serveur qu'on veut le mettre comme un serveur de base de données,
Le '10.254.254.8' dans notre cas.

Maintenant si on a bien appliquée ces commandes notre application moodle va démarrer avec succès.

❖ **Sous Ubuntu Server**

Tant que Ubuntu server est un système d'exploitation sans une interface graphique donc on ne peut ajouter un utilisateur a MySQL qu'en utilisant le Shell ou le terminal.

La manière d'exécution de la commande sous le terminal d'Ubuntu server est pareille à celles de Windows .Mais si on lance moodle, un message d'exception de MySQL va nous retourner par le navigateur. (*Voir la figure 22*)

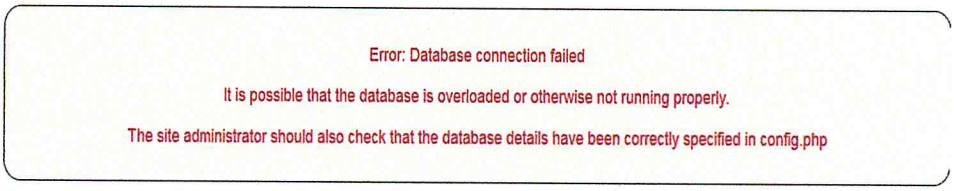

Figure 22 : *Fenêtre d'erreur de moodle.*

Problème… !:

Le SGBD MySQL d'Ubuntu server est par défaut hors ligne, il n'accepte que les requêtes viennent de 'localhost'.

Solution :

Les paramètres de MySQL doivent être reconfigurés seulement dans le serveur qui contient la base de données comme suit :

- On accède à le fichier de configurations du MySQL qui se trouve à `'/etc/mysql/my.cnf'` par la commande suivante :

```
Sudovi /etc/mysql/my.cnf
```

- on localise les lignes de configuration au-dessous de l'entête `[mysqld]` et on met la ligne `'skip-networking'` comme un commentaire comme suit :`'#skip-networking'`.
- Puis on modifier l'adresse d'assemblage `'bind-address=127.0.0.1'` comme suit :

```
'bind-address= [l'adresse IP de notre serveur de BD]'
```

Exemple :

'bind-address= '10.254.254.8'' (voir la figure 9)

- Nous enregistrons les modifications par ' : 'wq' , et on relancer MySQL Par la commande :'/etc/init.d/mysql restart'

```
[mysqld]
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket          = /var/run/mysqld/mysqld.sock
port            = 3306
basedir         = /usr
datadir         = /var/lib/mysql
tmpdir          = /tmp
language        = /usr/share/mysql/English
bind-address    = 65.55.55.2
# skip-networking
```

Figure 23 : *le contenue de la partie [mysqld] du fichier my.cnf*

Maintenant si on a bien appliquée ces commandes notre application moodle va démarrer avec succès.

Problème :... !

- Logiquement Si on veut ouvrir une session sur moodle il faut que nous utilisions le pseudo et le mot se passe du compte 'administrateur' du serveur qui porte la base de données (il a été créé lors de l'installation du moodle), puisque ces derniers sont stockés sur la base.

  Si on essaie d'accéder, moodle refuse l'accès... !

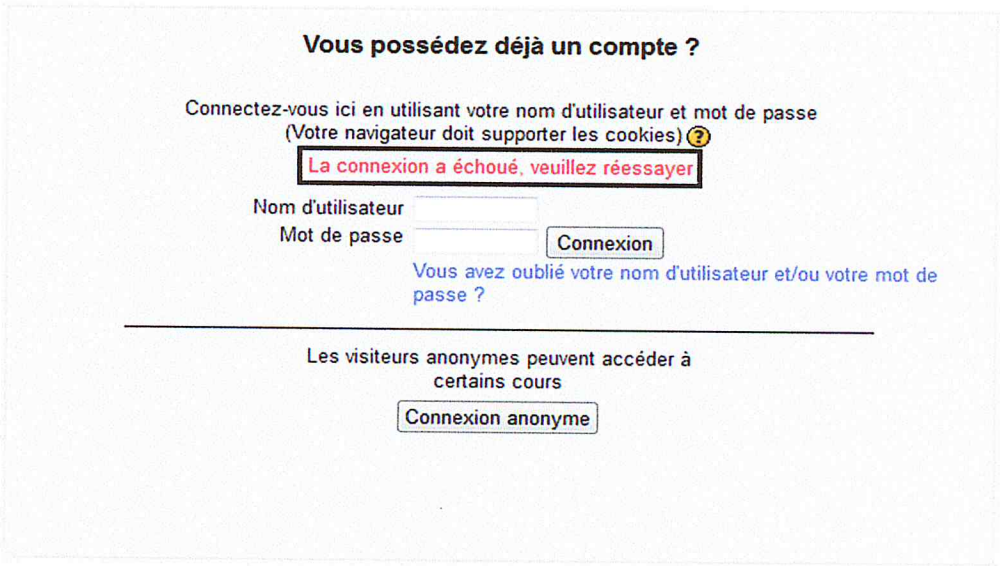

Figure 24 : *une fenêtre d'échec d'ouverture de la session.*

- Si on crée un utilisateur 'user1' sur le serveur '10.254.254.1', les données vont stockées sur la base de '10.254.254.8' et l'utilisateur 'user1' ne peut ouvrir son session que à partir le serveur '10.254.254.1'…… !

Solution :

Avant de voir la solution il faut comprendre comment la plateforme 'moodle' stocke les mots de passe, et pour cela nous allons créer un utilisateur nommé 'admin' avec un mot de passe 'Lotfi_18091987', et on consulte la base de données pour voir qu'est-ce que moodle a stocké dans la base de données. (La Figure 11 illustre le principe.)

policyagreed	deleted	suspended	mnethostid	username	password	idnumber	firstname	lastname
0	0	0	1	guest	d3f6e93efac33e06813718bb048a8706		Guest user	
0	0	0	1	admin	69a8d300a237cbc412f770f6735c2103		Admin	User

Figure 25 : *la table 'user' de la base de données moodle.*

Moodle utilise l'algorithme 'MD5' pour crypter le mot de passe à l'aide d'une clef qui se trouve au fichier 'config.PHP', donc il faut modifier la clef de tous les serveurs en le mettant comme la clef du serveur qui contient la base.

Voilà la clef de notre serveur de base de données (Kf]?&:{V=:!5Jf&=T}iIRa*HnaX;P)

Elle est la valeur de la variable

'$CFG->passwordsaltmain' dans le fichier 'config.PHP'

Il faut que nous modifions tous les variables '$CFG->passwordsaltmain' dans le fichier 'config.PHP' dans tous les serveurs du cloud. afin il devient être comme suit :

```
$CFG->passwordsaltmain = 'Kf]?&:{V=:!5Jf&=T}iIRa*HnaX;P';
```

Constat

Après ces configurations n'importe quel utilisateur possède une session sur moodle peut accéder à son compte de n'importe quel serveur ... et voilà le but.

3.Ingénierie inverse de l'interaction entre le client et le serveur Moodle

N'ayant pas de document décrivant avec le détail requis le contenu des interaction entre Moodle et le client Moodle dans les divers états par lesquels passe Moodle, nous étions obligé de faire une ingénierie inverse pour découvrir le contenu des requête. Cette opération a été rendu possible grâce à un outil d'analyse du trafic réseau, notamment le trafic http entres les clients http et les serveurs http. L'analyseur du trafic http utilisé est HTTP Analyser.

3.1.IEInspector HTTP Analyzer

IEInspector HTTP Analyzer est un outil pratique qui nous permet de surveiller, de tracer, analyser et déboguer HTTP / HTTPS trafic en temps réel. Il est utilisé par des entreprises leader de l'industrie, notamment Microsoft, Cisco, AOL et Google.

Il peut assurer la traçabilité et l'affichage large éventail d'informations, y compris la tête, calendrier, le contenu, les cookies, les chaînes de requête, des postes de données, la demande et de réponse Stream, redirection d'URL et plus encore. Il a une demande de constructeur qui vous permet de l'artisanat un HTTP / HTTPS demande. Il fournit également des informations de cache et la session de compensation, ainsi que de code de statut HTTP information et de plusieurs options de filtrage. Un bon outil de développement pour l'analyse des performances, le débogage et le diagnostic. **[TOD 10]**

Maintenant, on passe vers l'implantation du serveur frontale en utilisant java.

4.Implémentation du serveur frontal

Le serveur que nous sommes en train de réaliser est une application web programmé par le langage de programmation java et déployé sur le conteneur de servlet Apache Tomcat.

Une application web java utilise la classe '`HttpServlet`' pour communiquer avec les clients, en utilisant les objets '`HttpServletRequest`' pour recevoir la requête du client et '`HttpServletResponse`', pour répondre le client.

Puisque notre application sert à dissimuler plusieurs serveurs, nous allons utiliser l'objet '`HttpURLConnection`' pour l'échange des requêtes et les réponses entre les clients du notre serveur et les serveurs en question.

Cette opération d'échanges a connu plusieurs étapes, que nous allons citer et détailler par la suite en mettant en exergue les problèmes que nous avons rencontré durant l'implémentation de chaque algorithme.

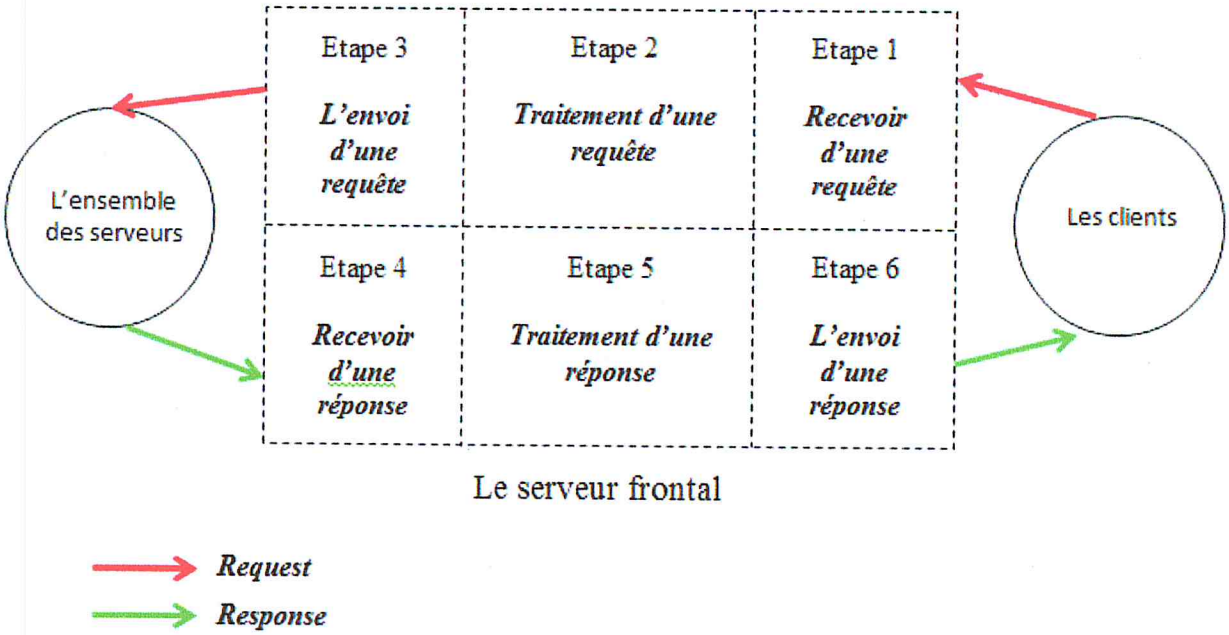

Figure 26 : *Schémas résume les étapes du traitement.*

4.1. Les étapes d'échanges de la requête et de la réponse

4.1.1.Recevoir d'une requête

Lorsque un client envoi une requête vers notre serveur, la servlet * qui se charge de contrôler l'application reçoit cette requête par le paramètre 'HttpServletRequest' de la méthode doGet(), si la requête est de type GET, ou par le paramètre 'HttpServletRequest' de la méthode doPost () si elle est de type POST.

Lors de la réception de la requête, il faut extraire la ligne de la requête, l'entête et le corps si elle est de type POST comme suit :

4.1.2.Extraire la ligne

Les lignes de la requête de l'application moodle ne se composent généralement que de l'URL et des paramètres, donc on utilise la méthode 'getRequestURL()' de l'objet 'HttpServletRequest' pour le chemin de

ressource et la méthode 'getQueryString()' du même objet pour les paramètre.

Example:

```
Stringligne=request.getRequestURL().toString+ "?"+request.getQuerySt
ring();
```

❖ **Extraire l'entête**

On extrait les noms des entête de la requête par la méthode 'getHeaderNames()' et la valeur de chaque entête par la méthode 'getHeader(String object);' et on met le tout dans une HashMap ;

❖ **Extraire le Corps**

Le corps contient les données postées, on les récupère par la lecture du flux entrant de l'objet 'HttpServletRequest' par la méthode 'getInputStream()' et on les met dans une table d'octet .

* **Remarque :**

Pour assurer que toutes les requêtes du client sont reçues par la servlet qui s'occupe du traitement, il faut que nous nommions le projet 'moodle' en mettant la valeur de 'url-pattern' égale a '/' dans le fichier web.xml.

La réception de la requête est de même pour tous les algorithmes qu'on va implémenter.

4.2.Traitement de la requête

Après la réception d'une requête, plusieurs traitements vont s'effectuer sur cette dernière avant de l'envoyer. Ces traitements dépendent du serveur qui va recevoir la requête et l'algorithme, donc comment choisit-on ce serveur… ?

Avant les traitements, l'application fait appel à l'algorithme Round Robin (voir la page pour déterminer le prochain serveur. Dès que l'application récupère l'adresse du serveur, elle effectue des traitements tout dépend des algorithmes qu'on va citer ci-dessous :

4.2.1.Les traitements qui dépendent de l'algorithme Round Robin pour une affectation par requête

Cet algorithme consiste à envoyer la 1ère requête vers le 1er serveur, et la 2eme requête vers le 2em serveur et ainsi de suite, lorsque l'algorithme aboutit au dernier serveur, elle revient au premier.

Pour que les serveurs du cloud acceptent les requêtes, il faut que chaque requête porte un cookie qui a été créé par ces serveurs.

Donc le serveur frontal doit être mémorisé les cookies de chaque utilisateur pour chaque serveur.

- Puisque l'application moodle utilise un cookie nommé '*MoodleSession*' avant l'authentification et deux autres cookies après l'authentification nommé '*MoodleSession*' et '*MoodleID* 'et autre cookie appelé '*MoodleSession*' après la fin de la session. Notre serveur doit être mis à jour pour chaque changement de cookie… donc on a un *problème* des *cookies*.
- Tous les requêtes ont des pages des références qui sont de la forme suivante : [Referer: http://@du host/moodle/la page à partir de laquelle le document est demandé] où l'adresse est par défaut est l'adresse du serveur frontal.

Donc on a un *problème* des entêtes *Referer*.

- L'application moodle utilise une technique pour renforcer la sécurité de ses requêtes, cette technique est un code nomme '*sesskey*', il est inclus dans les réponses dont le type de ses contenues sont html, ce code a le même rôle que les cookies mais il n'est pas injecté dans les entêtes, il est inclus dans les paramètres d'URL ou dans l'entête '*Referer*' ou dans les données postées, il se pourrait aussi changer après l'authentification…donc notre serveur doit être mis à jour pour chaque changement de sesskey… donc on a un *problème* des *sesskey*.

Dans cette étape, on a traité ces problèmes comme suit :

- Pour *les cookies*, on inspecte chaque entête de la réponse, si on trouve l'entête '*set-cookie*', on renvoie la requête de cette réponse vers tous les autres serveurs du cloud et on récupère les cookies et on les enregistre dans un vecteur '*Vector*' avec l'adresse du client et l'adresse du serveur qui envoi ce cookie comme clé.
- Pour l'entête 'Referer', il faut changer l'adresse du Host par l'adresse du serveur qui va recevoir la requête.

- Pour le problème du *sesskey*, on le recherche dans chaque réponse de type html
  Et on retire le sesskey qui est un code qui se compose de 10 chiffres et lettres, on le mémorise de la même façon que les cookies.

4.2.2.Les traitements qui dépendent de l'algorithme Round Robin pour une affectation par client

Cet algorithme consiste à faire connecter chaque client a un serveur parmi l'ensemble des serveurs des cloud, autrement dit, tous les requêtes d'un client seront envoyé vers le même serveur que l'application l'a affecté lors de l'envoi de sa première requête. Donc il n'y en a pas encore les problèmes de *cookie* et *sesskey*, il nous reste que le problème de *Referer*, on le traite de la même façon que le cas précèdent.

4.2.3.Les traitements qui dépendent de l'algorithme Round Robin Améliorés pour une affectation par requête

Les traitements qui se déroulent dans cet algorithme sont les mêmes de ceux du traitement du 1er algorithme sauf le scénario de choix du serveur, il est comme suit :

- ✓ si la différence de MAX et de MIN du vecteur d'état est inférieure de son moyen, on laisse l'algorithme Round Robin décide quel est le prochain serveur.
- ✓ sinon on choisit le serveur qui a le minimum des requêtes en cours des traitements comme prochain serveur.

4.2.4.Les traitements qui dépendent de l'algorithme Round Robin Améliorés pour une affectation par client

Les traitements qui se déroulent dans cet algorithme sont les mêmes de ceux du traitement du 2^{em} algorithme sauf le scénario de choix du serveur, il est le même que l'algorithme précèdent.

4.2.5.L'envoi de la requête

Après les traitements nécessaires sur la requête et avant d'envoyer cette requête, il faut insérer l'entête de la requête sur l'objet '`HttpURLConnection`' en utilisant la méthode '`setRequestProperty`', pour les données à poster, la méthode d'envoi doit être indiqué par la méthode '`setRequestMethod ("POST")`'et les données sont transmises au serveur par le flux de sortie '`OutputStream`' après de les avoir converti en byte.

L'authentification sur moodle est une requête de type POST qui a une réponse contient les cookies de la session et un code de réponse égale à 303 qui signifie redirection automatique vers une ressource indiquée sur l'entête 'Location' de la réponse.

L'objet '`HttpURLConnection`', lorsque il reçoit le code de réponse 303, il renvoi automatiquement une requête vers le lien qui a été indiqué sur l'entête 'Location' mais sans aucun entête, moodle, de sa part, ne sais plus de quel client vient cette requête, puisque elle ne contient aucun cookie.

Par conséquent, il répond par une page d'accueil... donc, on a un *problème* de la *redirection automatique*.

On peut désactiver *la redirection automatique* en utilisant la méthode `setInstanceFollowRedirects (false)`.

4.2.6.L'envoi de la requête au l'algorithme Round Robin pour une affectation par requête

Lorsqu'un client fait un test sur la plateforme moodle, il est en train d'envoyer des informations, donc les requêtes sont des types POST.

Quand moodle reçoit ce genre des requêtes, il répond par le code '*303*' et le message '*SEE OTHER*' ou '*VOIR AUTRE*', cela signifie qu'il faut envoyer la prochaine requête vers un fichier PHP qui a été créé après la réception de la requête précédente, et nous comptons sur l'algorithme Round Robin qui, à chaque requête, il nous donne le prochain serveur, d'après ce scénario, la deuxième requête va être envoyer à un autre serveur qui ne contient pas le fichier PHP …donc on reçoit une réponse de type '*404 NOT FOUND*'.Et cela nous impose vraiment *un problème*.

Solution **:**

Lorsqu'on détecte un code de réponse égale à 303, on pause l'avance de l'algorithme Round Robine.

Remarque :

- L'envoi de la requête sur Round Robin Améliorés avec une affectation par requête est pareil à celui de Round Robin avec affectation par requête.
- L'envoi de la requête sur Round Robin avec une affectation par client et Round Robin Améliorés avec une affectation par client est déroulé sans la pause de Round Robin puisque le client s'est connecté au même serveur jusqu'à la fin de la session.

4.2.7.Recevoir de la réponse

Le serveur frontal reçoit la réponse d'une requête à partir le flux d'entre '`BufferedReader`' de l'objet '`HttpURLConnection`' si le type de réponse est html et par '`inputStream`' sinon.

4.2.8.Traitement de la réponse

Le code html d'une réponse contient les liens de ses ressources, qui seront utilisés plus tard par le navigateur pour récupérer ces ressources.

Les liens de ces ressources contiennent l'adresse IP du serveur qui envoie cette réponse, si on les laisse tels qu'ils sont, les prochaines requêtes vont être envoyé

directement vers le serveur qui envoi la réponse sans passer par notre serveur frontal…par conséquent on a *problème* des liens.

Pour résoudre ce problème, il suffit de remplacer tous les adresses IP d'une réponse de type html par l'adresse IP de notre serveur frontal.

4.2.9.L'envoi de la réponse

L'envoi de la réponse vers le client est similaire à la réception de la réponse, on utilise 'writer' pour les réponses dont le type est html et 'OutputStream' pour les autres types.

5.Haute disponibilité

Notre serveur doit assurer la disponibilité du serveur dans le cloud, autrement dit, lorsqu' une ressource n'est pas disponible (le code : 404 not found) ou une perte de connexion dans l'un des serveurs (panne d'un serveur), l'application renvoi la requête à un autre serveur sans que le client s'en rendre compte, si tous les serveurs tombent en panne, un message va s'afficher au client en lui informant qu'il faut contacter l'administrateur du serveur.

Conclusion

Lors de la réalisation de notre système qu'on a implémenté sur Windows et sur Ubuntu serveur, on a rencontré beaucoup de problèmes que certains parmi eux ont été difficile et qui nous ont pris beaucoup de temps pour les résoudre, cela ne signifie pas qu'on a découvert tous les problèmes puisque il reste la phase de maintenance qui est très importante pour développer et découvrir les ambigüités de n'importe quel logiciel.

Conclusion Générale

Conclusion générale

Nous avons proposé une approche Cloud qui apporte beaucoup de facilités dans l'utilisation des services de elearning. Dans ce contexte, nous avons installé et configuré une plateforme moodle dans chaque serveur que nous voulons intégrer dans le Cloud, puis nous avons développé un LaodBalancing que nous pouvons considérer comme le noyau de notre application.

Le LaodBalancing se charge de cacher l'ensemble des serveurs du Cloud, et équilibrer les charges entre les différents serveurs.

Enfin, dans le cadre de notre travail, ce projet reste un petit premier pas vers la réalisation d'un Cloud qui supporte toutes les plateformes e-learning.Il nous a permis d'acquérir de nouvelles connaissances comme : Cloud, équilibrage de charge et Linux.... , il nous a permis aussi d'approfondir nos connaissances au : **JAVA** qui est un langage orienté objet, le **protocole HTTP, JSP**... etc.

Bibliographie

Bibliographie :

[ABE 03] : Abel M.H., Lenne D., Moulin C., and Benayache A. "Gestion des ressources pédagogiques d'une e-formation." Documents Numériques 7 (2003): 111-128.

[BEA 03] : Béatrice Lecomte, IFRES-LabSET, ULg Colloque "Le choix d'un Learning Management System : une question institutionnelle", Liège – 27 & 28 novembre 2007

[BEN 12] Bennouar Djamal 2012, cours du master 2, architecture de logiciel, chapitre 4 , le protocole http. document numérique.

[BEN 12] : Bennouar D., Introduction au téléenseignement et à la conduite des eTest, Rapport, Département d'Informatique, USDB, 2012

[BOD 05] : BODET Gaël, Daoud Sabrina, Amalric Pierre-Henri « Comment réussir la mise en Place d'un projet e-learning ? », 2005.

[BOU 09] : BOUHADDOUZE .YIKHLEF .T « Environnement de développement (Langage C) intégré pour la plateforme d'enseignement à distance MOODLE »mémoire de fin d'étude INI 2008 /2009 Oued Smar Alger.

[BOU05] : BOUSBIA .Nabila. « Contribution Théorique Et Méthodologique A L'élaboration D'un Environnement De Foad (Formation Ouverte Et A Distance). » Thèse de magistère, INI 2005, Oued Smar, Alger.

[BRA 09]: Brandon Hall. "LMS and LCMS Demystified." Brandon Hall Research, 2009.

[CAR 07] : P. Caron, Ingénierie dirigée par les modèles pour la construction de dispositifs pédagogiques sur des plateformes de formation, Thèse soutenue le 18 Juin 2007 pour l'obtention du Doctorat de l'université des sciences et technologies de lille.

[CLA 12]: P. Caron, Ingénierie dirigée par les modèles pour la construction de dispositifs pédagogiques sur des plateformes de formation, Thèse soutenue le 18 Juin 2007 pour l'obtention du Doctorat de l'université des sciences et technologies de lille.

[DEV 04] : Devine J. "Creating, sharing and re-using e-Learning Content." Dublin, Ireland: IADT: Dun Laoghaire Institute of Art, Design & Technology, 2004.

[DOS 08] : DossouAnani Koffi DOGBE-SEMANOU, Anne Durand , Marie LEPROUST , Hélène VANDERSTICHEL « Etude Comparative de plates-formes de formation distance »dans le cadre du Projet @2L Octobre 2007 Corrections mineures : avril 2008.

[ELA 05] Claire FAGE, eLearning Agency, Vous avez dit « SCORM », Version : 0.9 Date : 22/9/2005

[FRE 09] : FREDERIC CURMIN « L'amélioration Contenue de la Qualité des Contenus Pédagogiques dans un LCMS : approche, évaluation et procédure de validation »Mémoire en vue de l'obtention du Master 2 soutenu le 17/09/2009, Université de Lille 3, France

[GTI 07] : GTIENA. "Normes, standards et interopérabilité pour les environnements numériques d'apprentissage." Montréal, Canada.: CREPUQ, 2007.

[JIN 02] : JineshVaria,Livre Blanc adaptation non officielle de"Architecting for the Cloud" BestPractices,2002.

[MAD 05]: MADJAROV.I « Des services web pour le e_learning »Laboratoire des Sciences de l'information et des Systémes,madjarov@iut-gtr.univ-mrs.fr, Date de publication : 28 Octobre 2005

[MAJ 10]: Majid B, En route pour les Amazon Web Services (AWS), www.neoxia.com Publier le 25/01/2010.

[MAO 04]: El_MaouhabMedjekFaiza et BenabderrahimOthmane « Services web Technique, démarche et outils, 2004.

[MIE 05] : MIELNIKOFF.M « Qu'est-ce que l'E-earning ? »Editer par le centreRégional d'Innovation et de Transfert de Technologie, France,Le 07/11/2005.

[NIST] : National Standard Institute http://www.nist.gov/itl/cloud/index.cfm)

[OUB 03] : OUBAHSII« Elément pour la modélisation d'une plate-forme d'enseignement distance »;Annexes aux actes de la conférence EIAH 2003,Strasbourg,Avril2003.

[PAQ 02] : Paquette G. "L'ingénierie pédagogique, Pour construire l'apprentissage en réseau." Québec: Presse de l'université de Quebec, 2002-490.

[SOG 09]: SOGETI Enterprise Services Consulting, Etat de l'art Cloud Computing, Mars 2009.

[TOD 10] Todoroms, Site Web : http://fr. httpanal/hf-fs-fs-ieinspector-http-analyzer-full-edition-v6-1-1-313, 2012.

[WYG 10]: WYGwam, Bureau d'expertise technoloqiqueLe Cloud Computing : Réelle révolution ou simple évolution ?, 2009.