UNIVERSITES PARIS-SUD XI,
En cohabilitation avec
L'Observatoire de Paris, l'Université Pierre et Marie Curie (Paris VI), et
l'Université Denis Diderot (Paris VII)

RAPPORT DE MASTER

Mention : Information, Systèmes et Technologies

Spécialité Professionnelle : Outils et Systèmes de l'Astronomie et de l'Espace

(Niveau M2)

DEVELOPPEMENT D'UNE INTERFACE LOGICIELLE POUR LE CALCUL SCIENTIFIQUE A DISTANCE : APPLICATION A L'HELIOSISMOLOGIE

Présenté par
Mohamed Yacine SAIDI

Réalisé sous la direction de
Dr. Laurent Gizon
au

Max Planck Institute for Solar System Research
Max-Planck-Strasse 2, 37191 Katlenburg-Lindau, Germany

Septembre 2006

Developpement d'une interface logicielle pour le calcul scientifique a distance : application a l'heliosismologie

Par Med.Yacine SAIDI

Résumé

Ce travail s'inscrit dans le cadre du projet européen HELAS dont l'un de ses objectifs est de rendre accessible à la communauté scientifique certains codes de calcul pour l'analyse et l'interprétation des données héliosismiques, via une interface la plus simple et la plus commune possible (un navigateur Internet). À cet effet, nous considérons, dans un premier temps, un code de calcul de noyaux de sensibilité pour l'héliosismologie temps-distance (code MATLAB).

Cette étude synthétise la procédure d'approche et les choix de conception d'une solution dans un contexte systémique. En effet, l'interface n'est que le mécanisme d'accès aux informations, sa mise en œuvre fait appel à des outils logiciels et matériels complexes et parfois fondamentalement différents : MATLAB, Serveur Web, Base de données, Grappes de calculs, etc.

Pour relier tous ces outils en une entité fonctionnelle répondant à nos besoins, nous avons implémenté une solution en langage PHP, qui fait appel à des technologies sous-jacentes ouvertes, gratuites, et multiplateformes. Cette approche est fonctionnelle et a été prévalidée par une première phase de tests.

Mots Clés

Héliosismologie, Analyse temps-distance, Noyau de sensibilité, MATLAB, MATLAB Compiler, Application n-*tier*, Serveur Web, Apache, MySQL, PHP, Calcul distribué, grappe de calcul.

Table des matières

Table des figures

Liste des tableaux

Liste des acronymes

C	Contrainte.
CGI	Common Gateway Interface.
CSS	Cascading Style Sheets.
ESA	European Space Agency.
FP	Fonction principale.
FS	Fonction secondaire.
FTP	File Transfert Protocol.
HELAS	European HELio- and ASteroseismology Network.
HTML	Hyper Text Markup Language.
HTTP	HyperText Transfer Protocol.
HTTPS	HyperText Transfer Protocol Secure.
JNI	Java Native Interface.
JRE	Java Runtime Environment.
JSP	JavaServer Pages.
JVM	Java Virtual Machine.
LAN	Local Area Network.
LSF	Load Sharing Facility.
MATLAB	MATrix LABoratory.
MDI	Michelson Doppler Imager
MPI	Message Passing Interface.
MPS	Max-Planck-Institut für Sonnensystemforschung
MPS	Max-Planck-Institut für Sonnensystemforschung.
NCSA	National Center for Supercomputing Applications
OSC	Ohio Supercomputer Center
PBS	Portable Batch System.
PHP	Pre Hypertext Processor / Personal Home Page.
PNNL	Pacific Northwest National Laboratory
QoS	Quality of Service.
SDO	Solar Dynamics Observatory.
SGBD	Système de Gestion de Base de Données.
SGBDR	Système de Gestion de base de données Relationnel.
SOHO	Solar and Heliospheric Observatory

SQL	Structured Query Language.
SSH	Secure SHell.
SSL	Secure Socket Layer.
TCO	Total Cost of Ownership.
URL	Uniform Resources Locator.

CHAPITRE I
Contexte scientifique

« La musique est une mathématique sonore,
la mathématique une musique silencieuse. »
Edouard Herriot (1872-1957).

I.1 Les oscillations solaires

Les oscillations solaires de cinq minutes ont été découvertes par Leighton, Noyes et Simon en 1962 [1]. La nature de ces oscillations a été élucidée en 1970 par Ulrich [2] qui les a interprétées comme des ondes acoustiques internes. En 1975, Deubner [3], grâce à des observations améliorées, confirma que le spectre de puissance des oscillations est conforme à la théorie de Ulrich. Les oscillations solaires permettent d'étudier les couches internes du Soleil, en suivant une approche similaire à celle que les géophysiciens utilisent pour sonder l'intérieur de la terre à partir de mesures sismiques. Dans le cas solaire, on parle d'héliosismologie. La source d'excitation des oscillations solaires est la convection turbulente près de la surface du soleil [4].

Les propriétés des ondes solaires permettent d'estimer les conditions de température, de densité, et de composition chimique à l'intérieur du Soleil, ainsi que sa rotation interne. Les données observationnelles fondamentales de l'héliosismologie moderne sont des images Doppler de la vitesse à la surface du Soleil projetée le long de ligne de visée, $\Phi(\mathbf{x},t)$, où t est le temps et $\mathbf{x}$ le vecteur position sur la surface du solail. Les deux principales séries de données sont fournies soit par le réseau global d'observations terrestre GONG[1], soit par l'instrument MDI[2] embarqué à bord du satellite SOHO de l'ESA en orbite autour du point de Lagrange L1 depuis 1996. Les images MDI sont obtenues en mesurant le déplacement Doppler de la raie d'absorption photosphérique Ni 6788 Å. Chaque minute, une image Doppler du disque solaire de taille 1024×1024 est enregistrée. La cadence d'une minute permet de mesurer toutes les oscillations solaires, puisque leurs fréquences sont inférieures à la fréquence de Nyquist du signal (8,3 mHz).

[1] http://gong.nso.edu/

[2] http://soi.stanford.edu/

L'analyse des données heliosismiques se fait dans l'espace de Fourier. En faisant l'approximation que la surface du Soleil soit localement plane au voisinage d'une position donnée, nous pouvons définir le système de coordonnées cartésiennes $\mathbf{x}=(x,y)$, tel que x pointe dans la direction de la rotation du soleil, et y pointe dans la direction du pole nord. Les composantes harmoniques du signal sont extraites par application de la transformée tridimensionnelle de Fourier :

$$\tilde{\Phi}(\mathbf{k},\omega) = \int_A \mathrm{d}^2\mathbf{x} \int_0^T \Phi(\mathbf{x},t)\ e^{i\mathbf{k}\cdot\mathbf{x}-i\omega t}\mathrm{d}t\ ,$$

où A est une petite région de la surface du Soleil, $\mathbf{k}=(k_x,k_y)$ est le vecteur d'onde horizontal, et $\omega = 2\pi\nu$ est la fréquence angulaire.

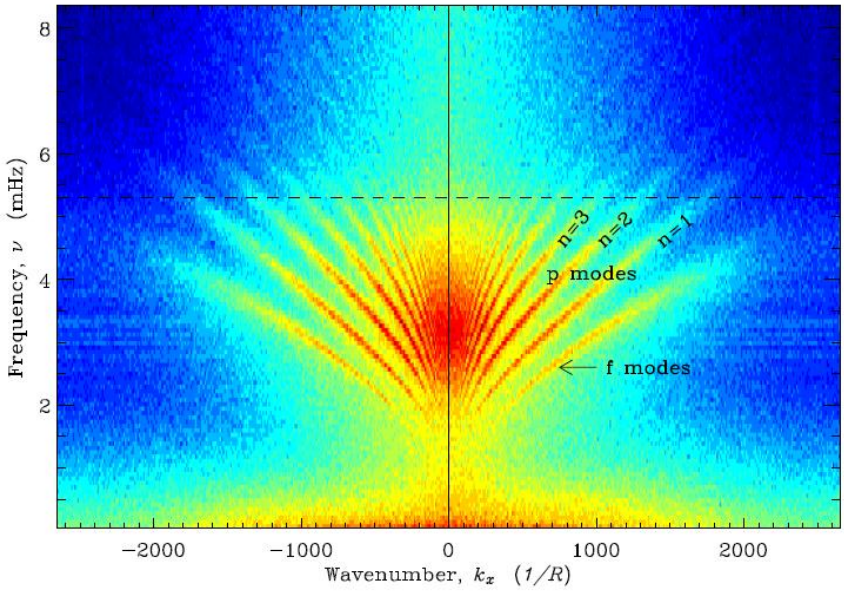

Figure 1 – Spectre de puissance des oscillations solaires.

La Figure 1 montre la coupe $k_y = 0$ du spectre de puissance $P(\mathbf{k},\omega) = \left|\tilde{\Phi}(\mathbf{k},\omega)\right|^2$ obtenu à partir d'une série temporelle d'images MDI d'une durée de 32 heures. La puissance est repartie selon différentes branches de la relation de dispersion, qui sont repérées par le nombre quantique n. La branche de plus basse fréquence (n=0) correspond aux ondes de gravité de surface, appelées aussi modes *f*. Ces derniers ont presque la même relation de dispersion que celle des ondes observées à la surface des océans, $\omega^2 = gk$, où $g = 274\ \mathrm{ms}^{-2}$ est la gravité à la surface du Soleil, et k est le nombre d'onde horizontale. Les modes f se propagent horizontalement. Toutes les autres branches, dénotées p_n, correspondent aux modes acoustiques, ou modes p, dont la force de rappel est la pression. L'indice n correspond au nombre de zéros des fonctions propres radiales des oscillations acoustiques. Plus k est petit et n est grand, plus les ondes acoustiques pénètrent plus profondément à l'intérieur du Soleil. Une propriété physique essentielle est que les ondes acoustiques dont les vitesses de phases horizontales (ω/k) sont comparables se propagent à des profondeurs similaires à l'intérieur

du Soleil. Pour des fréquences au dessus de la fréquence de coupure acoustique (5.3 mHz, ligne pointillée), les ondes ne sont pas piégées à l'intérieur du Soleil, mais se propagent librement dans l'atmosphère solaire. Pour plus de détails, voir [5],[6].

I.2 Méthode temps-distance

L'héliosismologie locale est un ensemble de méthodes d'analyse qui permettent d'obtenir des images tridimensionnelles de l'intérieur solaire. Dans la suite de notre étude, on s'intéressera seulement à une méthode particulière appelée analyse temps-distance [7].

L'héliosismologie temps-distance, a pour principe la mesure des temps de parcours des ondes qui se propagent entre deux points quelconques à la surface du Soleil. Le temps de parcours entre deux positions $\mathbf{x}-\boldsymbol{\Delta}/2$ et $\mathbf{x}+\boldsymbol{\Delta}/2$ à la surface du soleil, est extrait à partir de la fonction de covariance

$$C(t)=\int_0^T \Phi\left(\mathbf{x}-\boldsymbol{\Delta}/2,t'\right)\ \Phi\left(\mathbf{x}+\boldsymbol{\Delta}/2,t'+t\right)\mathrm{d}t'\,,$$

où T est la durée des observations. La fonction de covariance permet de moyenner, en phase, les ondes aléatoires ; elle s'apparente à un sismogramme solaire. Le temps de parcours pour les ondes se propageant de $\mathbf{x}-\boldsymbol{\Delta}/2$ vers $\mathbf{x}+\boldsymbol{\Delta}/2$, noté $\tau(\mathbf{x},\boldsymbol{\Delta})$, est mesuré à partir de la restriction de C au domaine $t>0$ [8].

Les temps de parcours contiennent la signature sismique des inhomogénéités à l'intérieur du soleil. Les écoulements de plasma dans le Soleil rompent la symétrie entre $\tau(\mathbf{x},\boldsymbol{\Delta})$ et $\tau(\mathbf{x},-\boldsymbol{\Delta})$, tandis que les perturbations de température et de densité affectent le temps de parcours moyen, $\left[\tau(\mathbf{x},\boldsymbol{\Delta})+\tau(\mathbf{x},-\boldsymbol{\Delta})\right]/2$.

Les temps de parcours sont mesurés pour chaque paire de points (pour chaque $\mathbf{x}$ et $\boldsymbol{\Delta}$). Un problème inverse devra ensuite être résolu pour interpréter les mesures de temps de parcours et estimer les paramètres physiques qui caractérisent la structure 3D et la dynamique de l'intérieur solaire.

I.3 Noyaux de sensibilité pour temps de parcours

Afin d'effectuer les inversions linéaires des temps de parcours, il est d'abord nécessaire de résoudre le problème direct. Le problème direct consiste à déterminer la relation entre un temps de parcours et les propriétés internes du soleil. Nous désignons par q_α l'ensemble des quantités physiques : l'indice α réfère, par exemple, à la vitesse du son, la température, la densité, etc. D'après les travaux de Birch et Gizon [9],[10], le problème direct linéaire se présente sous la forme d'une intégrale de volume :

$$\tau(\mathbf{x},\boldsymbol{\Delta})=\sum_\alpha \int_\odot \mathrm{d}^3\mathbf{r}\ K_\alpha(\mathbf{r};\mathbf{x},\boldsymbol{\Delta})\,\delta q_\alpha(\mathbf{r})\,,$$

où le noyau $K_\alpha(\mathbf{r};\mathbf{x})$ donne la sensibilité de τ à la perturbation interne $\delta q_\alpha(\mathbf{r})$. Les perturbations δq_α, mesurées par rapport à une moyenne horizontale, sont supposées être petites en valeurs relatives. Jusqu'à présent, toutes les études supposent que les δq_α sont indépendants du temps. Le détail des calculs de noyaux est exposé dans la référence [11].

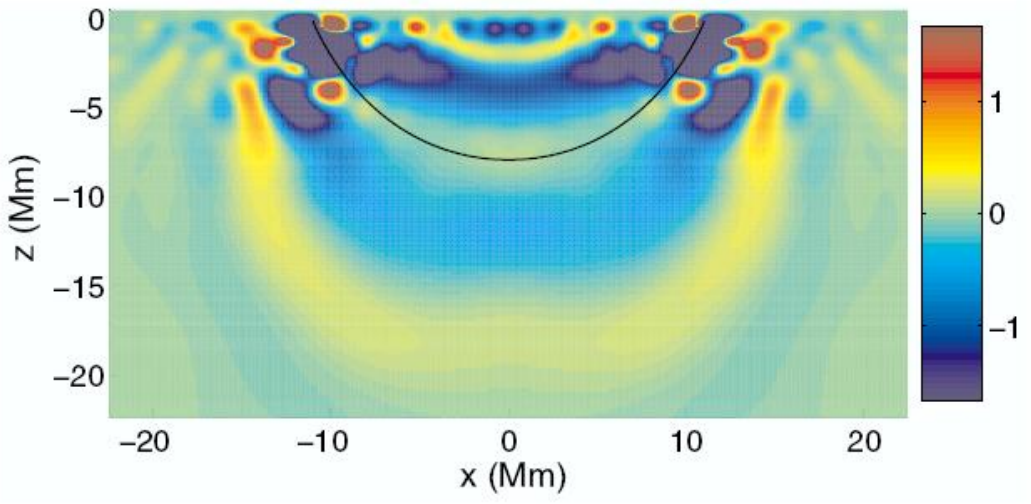

Figure 2 – Vue en coupe d'un noyau de sensibilité pour la vitesse du son.

La Figure 2 montre une fonction de noyau obtenue par Birch et al. [9] pour la sensibilité du temps de parcours à une perturbation locale $\delta q_\alpha = \delta c^2/c^3$, où c est la vitesse du son. Le noyau est non nul au voisinage du rayon acoustique qui connecte les deux points d'observation (courbe noire). La complexité du noyau est due au fait que les fréquences des ondes sont seulement comprises entre 2 et 5 mHz (le noyau serait confiné le long du rayon acoustique si toutes les fréquences étaient présentes). Le lobe central du noyau de sensibilité s'appelle la première zone de Fresnel.

La connaissance des noyaux de sensibilité est essentielle en héliosismologie temps-distance. Un code de calcul a été développé par A. Birch[3] dans le cas des perturbations de la vitesse du son et des écoulements de plasma.

Le but de mon travail est de rendre ce code accessible à la communauté scientifique dans le cadre du réseau d'héliosismologie Européen HELAS. Ces calculs devront exploiter la puissance de calcul qui leur sera réservée à l'Institut Max Planck (MPS).

[3] http://www.cora.nwra.com/~aaronb/

CHAPITRE II

Expression des besoins et analyse des choix

« Bien définir est bon pour se faire comprendre ; ne pas définir est indispensable pour discourir en paix. »

Rémy de Gourmont (1858-1915).

La solution à la problématique exprimée dans le chapitre précédent sera nécessairement basée sur un système informatique[4]. Afin de mener à bien la mission de le concevoir, en maîtrisant les ressources et les délais impartis, il est impératif de réduire le risque de malentendus concernant les objectifs fixés. Pour cela, il faut dans un premier temps clarifier la terminologie et définir précisément les exigences.

II.1 Cycle de vie d'un système informatique

Pour rappel, la définition d'un projet dans le contexte des systèmes informatiques, se résume à un cycle d'activités techniques, de la définition à la mise en œuvre en passant par la conception et la réalisation. Ces activités peuvent être menées à différents moments dans le cycle de vie d'un système logiciel. Ce cycle est composé de quatre phases majeures (Figure 3): définition des besoins, développement, exploitation, et maintenance.

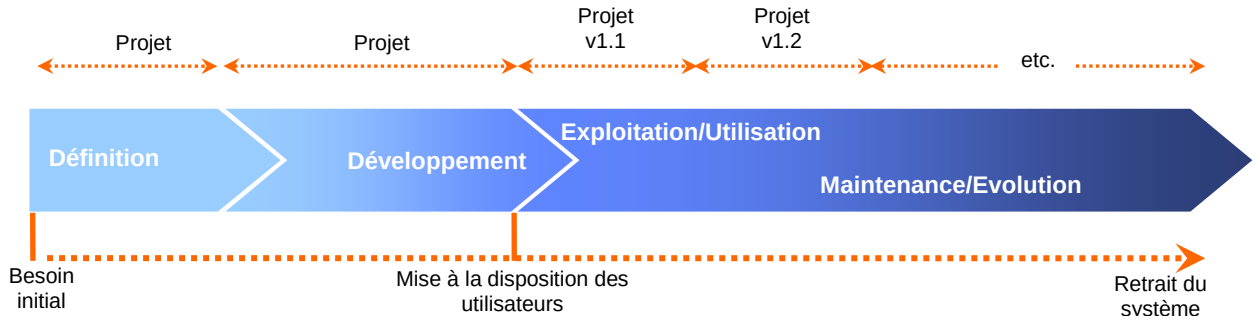

Figure 3 – Cycle de vie d'une solution logicielle.

[4] Avec éventuellement une composante matérielle et logicielle.

II.2 Analyse des besoins

Comme précisé précédemment, ce travail devra permettre d'initier une approche collaborative dans le cadre du projet HELAS. À cet effet, nous considérons, dans un premier temps, un code de calcul de noyaux de sensibilité pour l'héliosismologie temps-distance, qu'on devra rendre accessible à la communauté scientifique.

Dans un premier temps, il s'agira de concevoir un système applicatif qui simplifiera l'utilisation et l'exploitation du code de calcul des noyaux de sensibilité.

A cet effet, une interface simple devra faire abstraction de la complexité à paramétrer ledit code. Les calculs étant potentiellement long le système devra permettre de tirer profit de la puissance de calcul offerte par les machines disponibles, ou en cours d'acquisition au niveau du MPS.

Nous pouvons résumer la fonction principale de notre solution informatique comme suit :

FP Tout utilisateur doit pouvoir accéder instantanément à une interface simple afin de lancer le code de calcul sur plusieurs machines reliées entre elles.

Dans un second temps, il sera question de créer une base de données des noyaux de sensibilité :

FS Le programme de calcul devra interagir avec la base de données (récupérer et insérer des données).

En toute rigueur, cette fonctionnalité doit être prise en compte lors de la phase d'expression des besoins, et ce, même si son implémentation n'intervient qu'ultérieurement.

Comme le montre la Figure 4, et rien qu'en énumérant les fonctionnalités on arrive à distinguer les éléments qui vont intervenir d'une façon ou d'une autre dans le fonctionnement global de notre système.

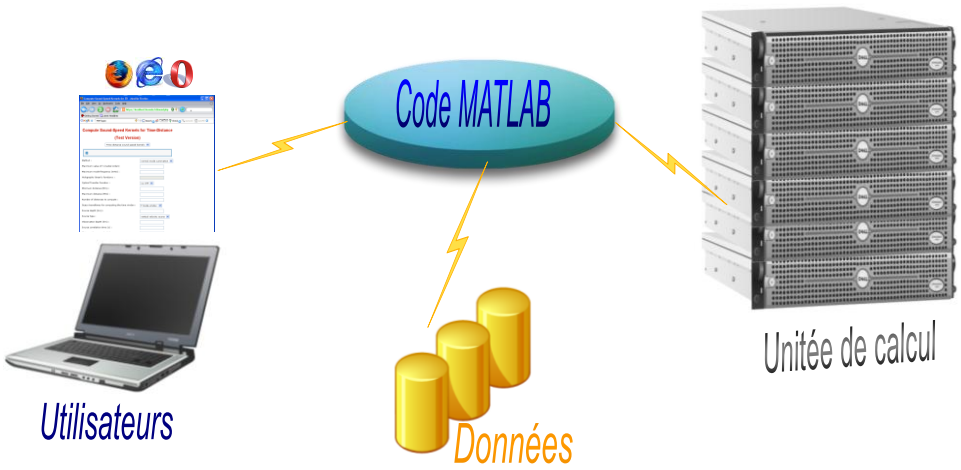

Figure 4 – Schéma simplifié des besoins.

II.3 Spécifications du système et contraintes de conception

Les spécifications d'un système informatique doivent prendre en compte les interactions entre lui et son environnement. Les spécifications les plus importantes de notre système sont celles liées à son utilisation (Figure 5) et à son implémentation, elles se présentent comme suit :

SP1 Interface : l'utilisation du système doit se faire via une interface simple, et commune au plus grand nombre d'utilisateurs sans « *add-ons* » logiciels.

SP2 Distribution de charge : les calculs devront tirer profit des ressources disponibles.

SP3 Sécurité et authentification : l'utilisation des ressources ne pourra se faire que lors que l'accès est autorisé.

SP4 Portabilité : la solution développée devra être indépendante du système d'exploitation.

SP5 Simplicité d'implémentation : afin qu'il puisse être distribué, le système devra être le plus simple possible.

L'estimation des besoins consiste à faire une projection de l'analyse sur les contraintes. Dans notre cas les contraintes se présentent comme suit :

C1 Le code de calcul avec son mode de fonctionnement et ses contraintes (nécessite un fichier avec les paramètres d'entrée pour s'exécuter, il est peu documenté, ...).

C2 Les calculs sont très longs[5].

C3 Les connaissances du concepteur du système.

C4 La limitation des licences MATLAB (une seule pourra être dédiée au projet).

C5 Le budget initial[6] : 1000 euros.

C6 La durée initiale : 6 mois.

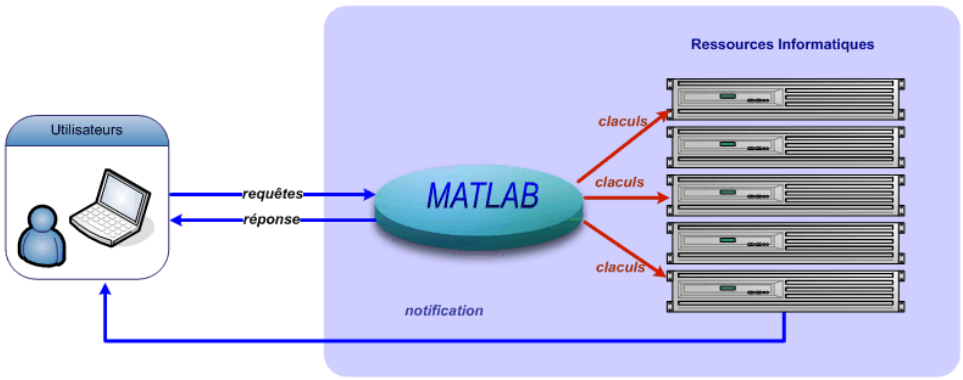

Figure 5 – Concept fonctionnel simplifié du futur système.

[5] Plusieurs jours semblent être le minimum pour des calculs classiques.

[6] Matériel et logiciel.

II.4 Choix de conception

Au vu des besoins précédemment spécifiés, le système devra offrir un large degré d'ouverture et d'évolutivité et ce pour chaque élément qui le constitue. Ceci ne pourra se faire qu'en disposant d'un bon niveau de modularité.

Une solution serait de concevoir un système réparti, c'est-à-dire un système dont les ressources[7] ne sont pas centralisées, le but étant de permettre à des utilisateurs de manipuler (calcul), leurs données (stockage) sans contrainte sur les localisations respectives des éléments. Par ailleurs, ce type de système permet au concepteur de diviser les problématiques.

Un tel concept trouve sa concrétisation dans l'approche à couches (Figure 6), et qui dans notre cas se présente de la façon suivante :

1) **Un client :** dans le cas présent, il devra être léger et surtout facile à déployer, typiquement un navigateur Web (SP1).
2) **Un serveur applicatif :** segmenté lui-même en couches de : présentation, logique de l'application, et gestion des ressources.
3) **Une base de données :** fournit les données au serveur applicatif.
4) **Une unité de calcul** (*cluster*[8])**:** constituée de plusieurs ordinateurs interconnectés.

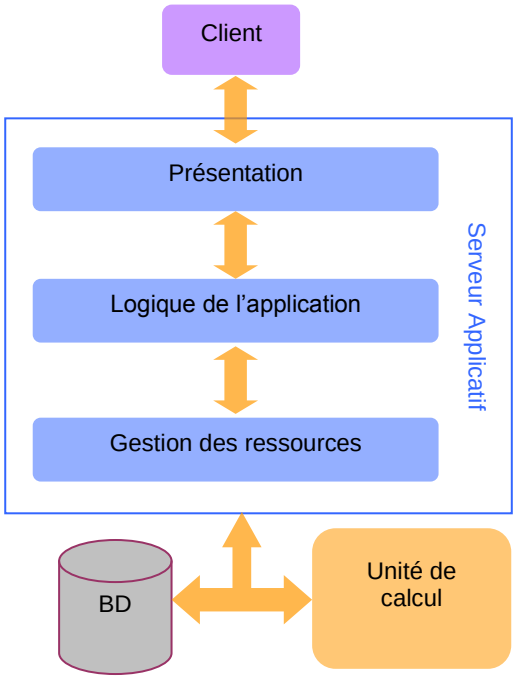

Figure 6 – Différentes couches d'un système applicatif.

[7] Ressources au sens très large : utilisateurs, stockage, bases de données, puissance de calcul, etc.

[8] Grappe ou ferme de calcul.

L'élément principal de notre système est un serveur applicatif, que nous allons présenter dans le détail.

II.4.1 Conception d'un serveur applicatif

Par définition, un système applicatif est un serveur hébergeant les applications destinées à être utilisées dans un réseau distribué. En comparaison au serveur de fichiers qui abrite les données destinées à être téléchargées et traitées par le client, le serveur d'application(s) assume une partie du traitement. Ainsi, le serveur d'application(s) permet à des utilisateurs connectés en réseau[9] d'accéder à tout ou une partie d'un logiciel applicatif (programme, interface graphique, données, etc.), à partir d'un exemplaire unique situé sur une machine informatique.

Comme on l'a précédemment présenté, la conception d'un serveur applicatif est généralement organisée en trois couches (*layers*) :

1) **La couche présentation :** qui construit l'Interface[10] Homme-Machine (IHM). En général, cette couche correspond à la partie de l'application visible, et interactive avec les utilisateurs.

2) **La couche logique d'application :** Elle effectue les traitements applicatifs et constitue de plus le tampon entre la présentation et les données. Ces traitements utilisent un ou plusieurs programmes qui implémentent les opérations demandées par le client à travers la couche de présentation. Ces programmes représentent souvent les services fournis par le serveur d'application.

3) **La couche gestion des ressources :** Elle est chargée des échanges entre le serveur applicatif et le ou les serveur(s) de données. En effet, un serveur d'application accède et/ou gère un ensemble de données qui peuvent résider dans une base de données, un système de fichiers ou tout autre gisement de données.

II.4.2 Architecture du serveur applicatif

Les trois couches présentées précédemment sont des constructions conceptuelles, qui séparent logiquement les fonctionnalités du serveur applicatif. Dans une approche plus concrète, ces couches sont matérialisées par des *tier*[11]. Nous pouvons distinguer 3 types d'architectures qui reposent sur ce concept :

---

[9] Mondial et/ou local.

[10] Peut être textuelle (CLI) ou graphique (GUI).

[11] Terme anglais, ne pas confondre avec le mot français "tier".

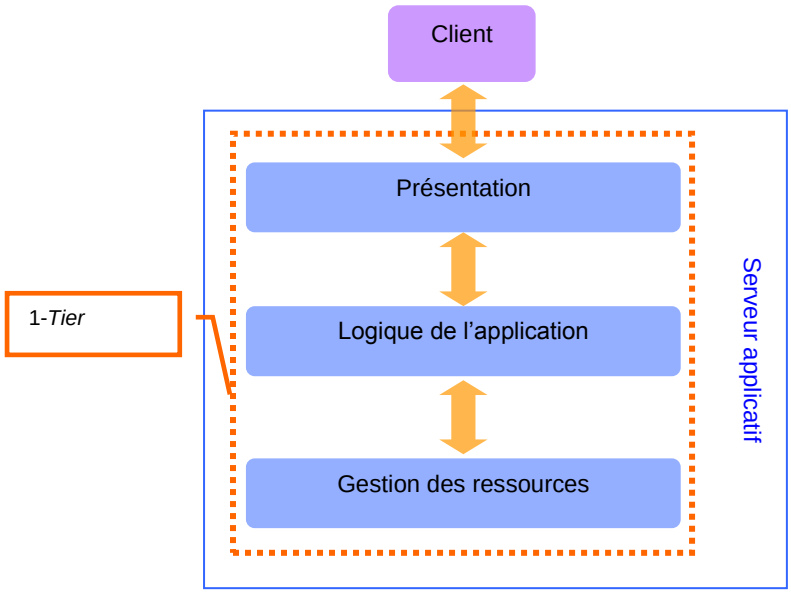

Figure 7 – Architecture 1-*tier*.

1) **L'architecture 1-*tier*** (dite monolithique)**:** ce type d'architecture (Figure 7) est des plus anciens, il remonte à l'âge d'or des *mainframes*[12]. Dans ce type de configuration, la présentation est conçue dans le *mainframe* et visualisée sur le terminal passif. Cette approche est de type centralisée est ne répond pas à nos besoins.

2) **L'architecture 2-*tier* :** ce type d'architecture (Figure 8) est apparu au même moment que les ordinateurs personnels (*Personal Computer* - PC). Cette construction est très populaire pour les architectures client/serveur et constitue actuellement le minimum quand on veut travailler avec les bases de données.

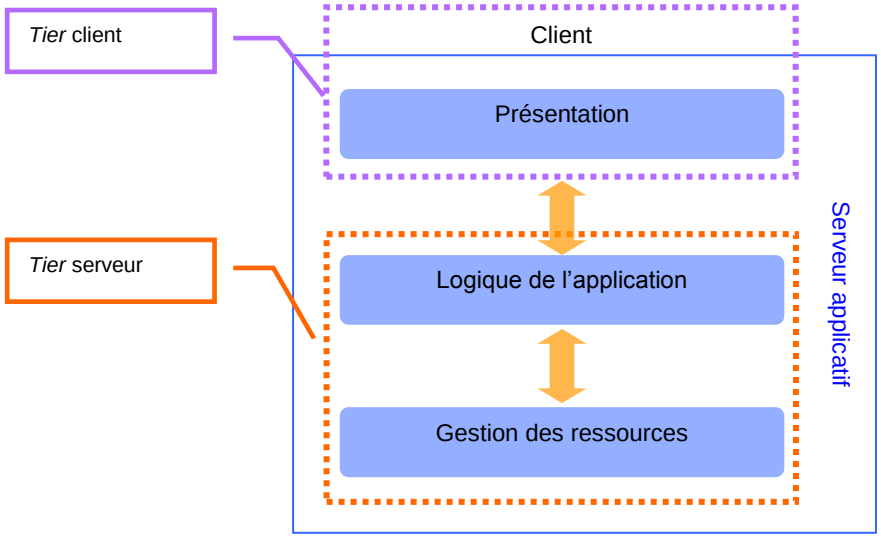

Figure 8 – Architecture 2-*tier*.

[12] Ordinateur central de grande puissance.

Dans cette approche, le client est responsable de la couche de présentation, et le serveur prend en charge la couche logique de l'application et la gestion des ressources.

Les avantages principaux de cette architecture sont :

- Le développement d'une architecture 2-*tier* peut être réalisé rapidement, dans la majorité des cas, bien sûr cela dépend aussi de la complexité du projet.
- Du moment que les couches logique de l'application et gestion des ressources sont au même niveau, la communication entre elles, est très efficace.

Les inconvénients sont :

- Le fait que les couches logiques d'application et la gestion des ressources soient au même niveau, nécessiterait une machine plus puissante pour les héberger, ce qui implique un coût matériel nécessaire plus élevé (C5).
- Dans la mesure où le logiciel client est généralement spécifique au serveur, cette architecture doit faire face aux problèmes de contrôle des évolutions (non conforme à SP5), ce qui implique que toute modification au niveau serveur peut avoir un impact sur la logique de fonctionnement de l'application client, ce qui nécessiterait des changements dans cette application, et donc sur tous les postes sur lesquels elle serait installée.

3) **L'architecture 3-*tier* :** Dans cette architecture (Figure 9), la couche de présentation réside généralement au niveau client, tout comme dans l'architecture précédente. La logique de l'application et la gestion des ressources quant à elles, se trouvent respectivement dans le *tier* métier (*middle-tier*) et le *tier* de gestion de base de données (*back-end*).

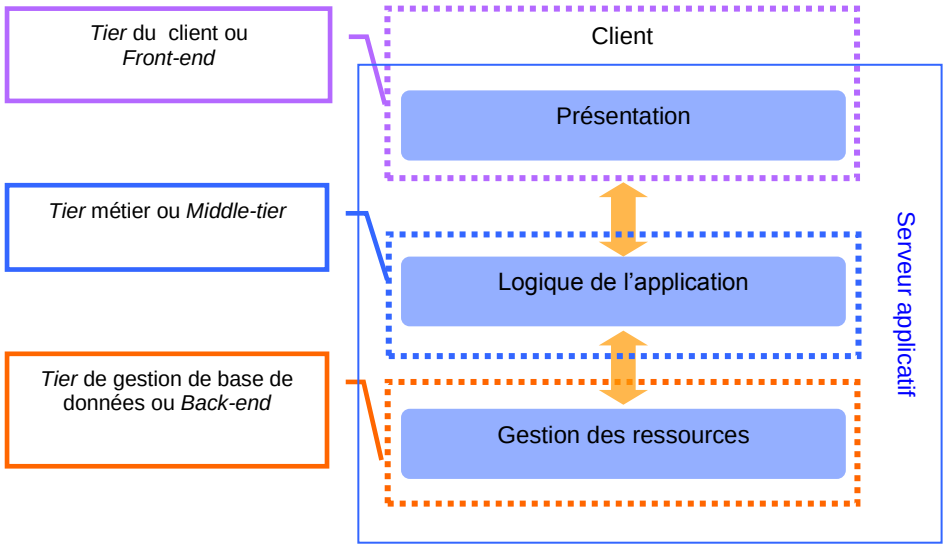

Figure 9 – Architecture 3-*tier*.

Les avantages de l'architecture 3-*tier* sont principalement:

- D'un point de vue développement, la séparation qui existe entre le client, le serveur et le système de gestion de base de données (SGBD) permet une spécialisation des développeurs sur chaque *tier* de l'architecture.
- Plus de flexibilité dans la portabilité du *tier* serveur ce qui permet d'envisager une des modifications au gré des évolutions.

L'inconvénient principal est :

- L'expertise de développement à acquérir, qui est plus longue que dans le cadre d'une architecture 2-*tier*.

Il existe une variante (Figure 10) très intéressante de l'architecture 3-*tier,* basée sur le concept de client léger (*thin-client*). Ce dernier, repose sur le fait que la présentation n'est plus créée dans le client, mais par un serveur d'application Web[13] dédié, ce qui allège le client (conforme à SP1).

L'implémentation de cette fonctionnalité peut se faire en utilisant diverses technologies que sont les : CGI (*Common Gateway Interface*), JAVA Servelets, JSP (*JavaServer Pages*), ASP.NET (*Active Server Pages)* ou PHP (*Hypertext Preprocessor*) pour créer dynamiquement des pages HTML (*Hyper Text Markup Language*) consultables par le navigateur.

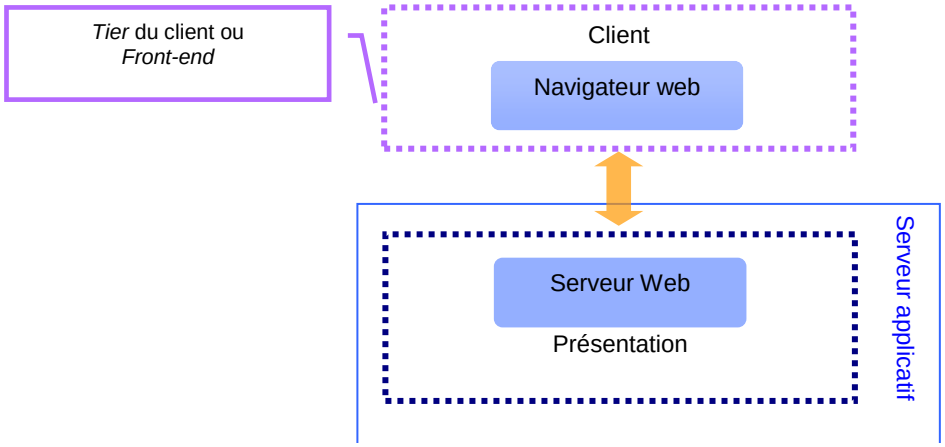

Figure 10 – Architecture 3-*tier* avec client léger (partie présentation et client).

Pour la suite de mes développements, j'ai décidé autant que possible de suivre le concept d'une architecture 3-*tier* avec client léger. Cette dernière répond le mieux aux spécifications de notre système et surtout nous offre la possibilité de séparer les problématiques, autant au niveau de l'analyse, que celui de la programmation.

[13] *World Wide Web :* méthode d'exploitation de l'Internet, par l'usage de l'hypertexte.

II.5 Choix technologiques

Maintenant que le choix de conception et d'architecture ont été effectués, il reste à sélectionner les entités technologiques qui vont nous permettre de concevoir le système :

II.5.1 Serveur Web

La première des démarches du développement d'une application dont Internet est un des éléments clés, consiste à mettre en place un serveur frontal ou serveur Web. Communément, on désigne par serveur Web un ordinateur tenant le rôle de serveur informatique sur lequel fonctionne un logiciel dénommé serveur HTTP[14]. Le terme serveur Web, peut aussi désigner le serveur logiciel lui-même.

D'un point de vu fonctionnel, un serveur Web est donc un logiciel capable d'interpréter les requêtes arrivant sur le port associé[15] au protocole de communication HTTP, et de fournir une réponse avec ce même protocole.

L'intérêt d'un tel système dans notre cas est qu'il permet à des clients d'accéder à des pages Web, ou plutôt à des fichiers qui sont dans la plupart du temps au format HTML à partir d'un simple navigateur (aussi appelé *browser*) installé sur leur ordinateur, ce qui dans notre cas répondra à la spécification SP1.

Le rôle clé joué par cet élément dans notre infrastructure future nécessite le passage par une phase de sélection.

Si on s'intéresse de pres au marché des serveurs Internet, on constate qu'il est caractérisé par un grand nombre de produits concurrents. Netcraft[16] et E-Soft[17] qui analysent ce marché, en dénombrent plusieurs dizaines. Toutefois, et toujours d'après les mêmes sources deux serveur Web; *Apache HTTP Server* [13] et *Internet Information Services* (IIS) [15]; se détachent du lot, avec respectivement 67.20 % et 20.51 % de part de marché[18], et une évolution constante d'Apache (Figure 11).

---

[14] *Hypertext Transfer Protocol :* littéralement « protocole de transfert hypertexte » qui se trouve être un protocole de communication client-serveur développé pour le Web.

[15] Généralement le protocole HTTP utilise le port 80.

[16] http://www.netcraft.com/

[17] http://www.securityspace.com/s_survey/data/index.html

[18] Estimation faite au mois de mars 2006.

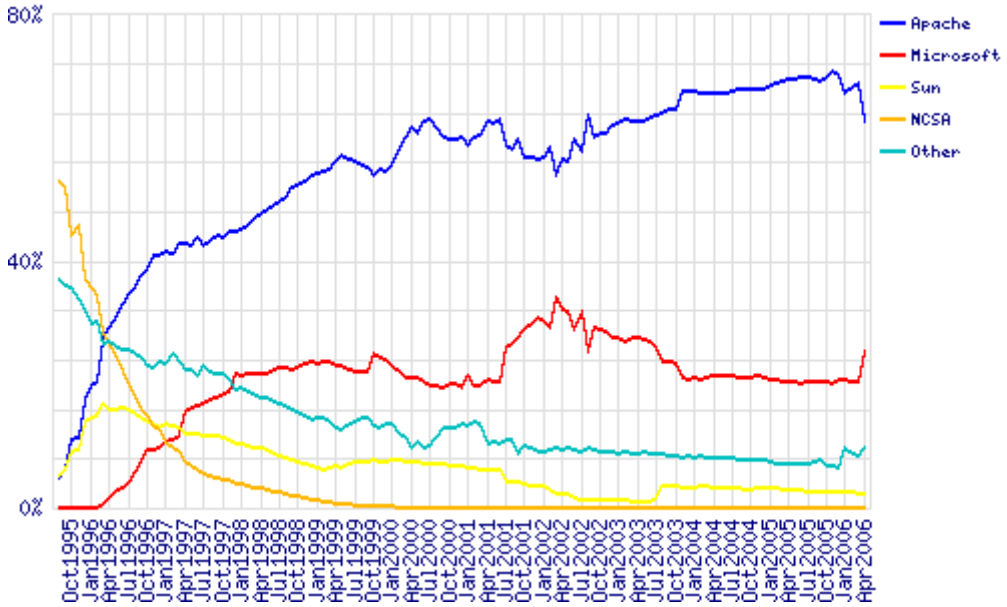

Figure 11 – Evolution des parts de marché d'Apache HTTP et d'IIS depuis octobre 1995.
(Source : Netcraft, mai 2006).

Pour une utilisation professionnelle, un serveur Web doit assurer une certaine qualité de service (QoS), tout en coûtant le moins cher possible. Concernant la QoS, et d'après NetCraft, Apache devance nettement IIS. Quand à l'estimation du coût, une étude poussé avec un outil d'analyse crédible sont indispensables, pour cela un rapport d'IDC[19] évalue justement le TCO[20] d'Apache et celui d'IIS tout inclus (implémentation, support, exploitation, coûts matériels, logiciels et humains). Le résultat est encore une fois en faveur d'Apache, qui en ressort premier, avec un TCO 6 % inférieur à celui de l'IIS.

En résumé, Apache est gratuit, multiplateforme (Tableau 1) et très fiable. Son seul inconvénient reste sa complexité à paramétrer et à administrer, difficulté pondérée par le nombre de sites et tutoriels qui lui sont consacrés. Tous ces éléments nous incitent à choisir Apache comme serveur frontal de notre solution.

[19] http://www.idc.com/

[20] *Total Cost of Ownership*, littéralement, le coût total d'appropriation. Notion introduite au cours des années 90 pour faire prendre conscience aux responsables d'une entreprise que le coût de fonctionnement d'un système informatique dépassait le prix du matériel et du logiciel.

	Systèmes d'exploitation	Sécurité			Contenu dynamique[21]			Licence	Coût
		Basic[22]	DAA[23]	HTTPS	JSP/Servelet	CGI	PHP		
Apache HTTP Server	Windows, Mac OS X, Linux, BSD, Solaris.	Oui	Oui	Oui	Non[24]	Oui	Oui	licence Apache	Gratuit
Microsoft Internet Information Services (IIS)	Windows	Oui	Oui	Oui	Non	Oui	Oui	propriétaire	Prix du système d'exploitation

Tableau 1 – Comparatif Apache HTTP Server et Microsoft IIS.

[21] Cette colonne indique si le serveur lui-même intègre cette option.

[22] *Basic authentication scheme.*

[23] *Digest Access Authentication.*

[24] Cette option est déléguée au Serveur Tomcat qui permet d'exécuter des servlets et des pages serveur Java (JSP).

II.5.2 Système de gestion de base de données

Les bases de données occupent aujourd'hui une place de plus en plus importante dans les systèmes informatiques. Leur système de gestion (SGBD) est un logiciel permettant de manager plusieurs bases de données réparties sur une ou plusieurs machines.

Pour agir sur les données, on utilise un langage d'interrogation. Le standard actuel est le SQL (*Structured Query Language*, ou Langage d'Interrogation des Données), dans ses déclinaisons SQL 92 et SQL 99 et SQL 2003**.**

La plupart des serveurs de bases de données actuels incluent un modèle relationnel : cela permet de créer simplement des relations entre différents champs de différentes tables. Le SGBD devient alors un SGBDR (Système de Gestion de Bases de Données Relationnels). Parmi les principaux serveurs de bases de données, on compte Oracle[25] et SQL Server[26] pour les solutions propriétaires les plus connues, PostgreSQL[27] et MySQL[28], pour les solutions ouvertes.

L'intérêt étant dans les données elles-mêmes, et non le SGBD. Il serait bon de signaler que dans la première phase de ce travail, les données se limiteront aux simples informations nécessaires à l'identification des utilisateurs. Toutefois, en prévision d'une phase d'extension des types de données à gérer (paramètres d'entrées, et surtout résultats), ainsi qu'une possible migration vers Oracle[29], il est primordial de faire le bon choix.

Après avoir fait la synthèse de plusieurs études et comparatifs de SGBDR [16],[17],[18],[19],[20],[21] notamment PostgreSQL et MySQL, force est de constater que la gratuité de ces dernières n'enlève rien au fait qu'elles soient des solutions abouties. Leur couverture fonctionnelle et leur standardisation sont très proches des outils payants. Surtout en ce qui concerne PostgreSQL. MySQL est encore un peu en retrait dans le domaine de la conformité aux standards.

Du point de vue de l'installation et de l'administration, ces deux systèmes disposent d'outils graphiques pour l'administration des bases de données[30], ce qui est un avantage certain pour des personnes sans grandes expériences de ce genre d'applications.

---

[25] http://www.oracle.com

[26] http://www.microsoft.com/sql/default.mspx

[27] http://www.postgresql.org/

[28] http://www.mysql.com/

[29] SGBDR prévu pour les données de la mission SDO.

[30] pgAdmin pour postgreSQL et MySQL Administrator pour son concurrent.

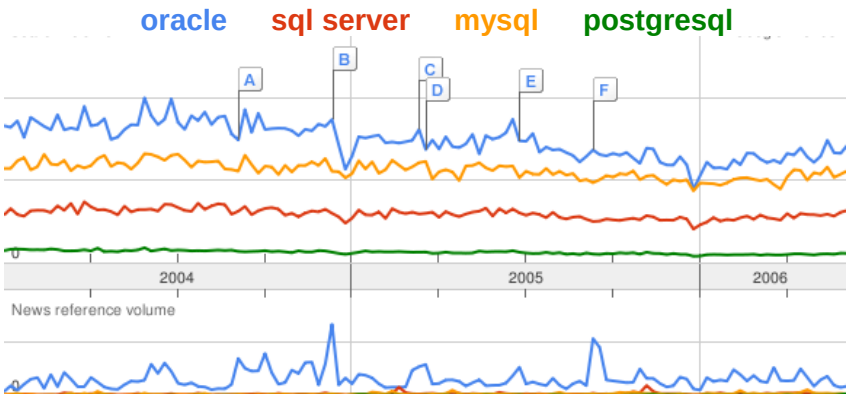

Figure 12 – Volume des recherches des différents SGBDR sur Google.

Source : Google Trends, juin 2006.

A mon avis, le choix entre ces deux systèmes sera plus déterminé par l'application que l'on désire réaliser qu'autre chose. Dans notre cas, la couverture fonctionnelle qu'offre MySQL me semble satisfaisante, au vu des manipulations qu'on aura à faire sur les données, du moins dans un premier temps. De plus, la rapidité de déploiement de cette solution et la présence d'une grande communauté d'utilisateurs (Figure 12) sont des facteurs déterminants au vu de certains contraintes et impératifs (C3, C6).

Notons aussi, la disponibilité d'un outil de migration[31] performant, pour l'éventuel passage de MySQL à Oracle [22].

Toutefois, il serait nécessaire de faire une étude un peu plus poussée avant la validation définitive de ce choix, et surtout avant le passage aux autres phases du projet (FS).

II.5.3 Système de calcul distribué

Comme précisé dans les spécifications de notre système (SP2), les opérations de calcul devront exploiter la puissance des machines actuellement disponibles. Le but étant que notre système puisse exploiter les futures machines dans les meilleurs temps, au risque d'être dépassé par l'évolution technologique (Figure 13).

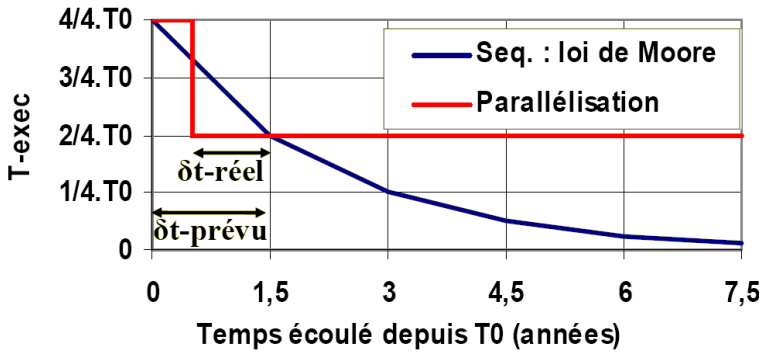

Figure 13 – Temps de mise en œuvre de solutions de calcul parallèle.

[31] Oracle Migration Workbench

La solution réside en partie dans le fait de regrouper virtuellement[32] plusieurs machines (nœuds) en réseau, ce concept est nommé *cluster* ou grappe. Par la suite, il faudra songer à adapter les algorithmes de calculs à cette architecture, ce qui pour le moment n'est pas le cas du code que nous utiliserons (C1).

Pour les besoins du cluster et pour éviter toute interférence, les machines devront être reliées par un réseau spécifique. Seules les machines qui appartiennent au cluster pourront émettre des messages sur ce réseau. Précisons aussi, que le regroupement de plusieurs machines sur un réseau ne suffit pas à la constitution d'un véritable *cluster*. Un système de gestion des ressources est indispensable afin d'avoir les meilleures performances et de permettre à plusieurs utilisateurs de lancer des tâches simultanément. On peut aussi rajouter un système de surveillance de l'ensemble du *cluster* (matériel et logiciel).

Concernant, le logiciel assurant la gestion de l'ensemble des machines, il est connu sous le nom de *Job Management System* (JMS), littéralement Système de Gestion des Tâches. Ce dernier, a pour but de centraliser le contrôle et la gestion de toutes les ressources disponibles; et permet à plusieurs utilisateurs de lancer des tâches simultanément.

Comme le montre la Figure 14 un système de gestion des taches est constitué généralement de trois éléments :

- **le serveur**, chargé d'interfacer les requêtes des utilisateurs avec les autres éléments,
- **l'ordonnanceur,** chargé de gérer les ressources du cluster,
- **le gestionnaire de ressource (mom),** chargé de l'exécution d'un job sur le nœud sur lequel il se trouve, et communique les informations sur l'état du nœud.

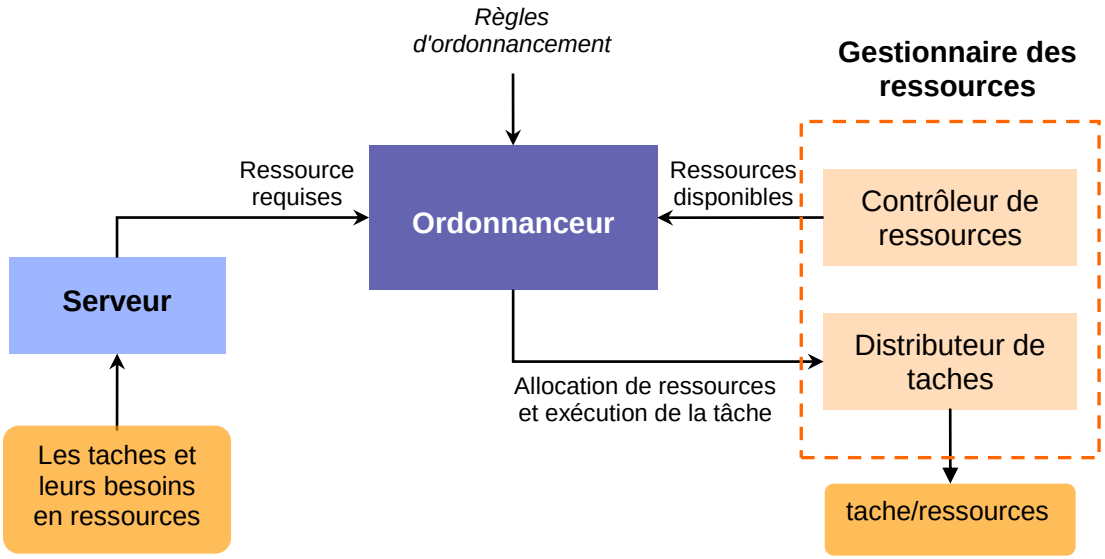

Figure 14 – Schéma de principe d'un système de gestion des taches.

[32] Le système aura une seule adresse réseau pour l'ensemble des machines du cluster.

Tout comme pour les serveurs Web, on retrouve un très grand choix de JMS, certains sont publics, d'autres payants, les plus connus sont : LSF (*Load Sharing System*) [23], Condor [24], SGE (*Sun Grid Engine*)[25], TORQUE (*Tera-scale Open-source Resource and QUEue manager*) [28], PBS (*Portable Batch System*) avec ses variantes OpenPBS [26] et PBS Pro [26].

Concernant le choix de JMS, nous allons le restreindre aux systèmes ouverts, pour être en conformité avec nos besoins. De plus, le prix de la licence d'utilisation des solutions JMS payantes est proportionnel au nombre de nœuds du cluster, ce qui introduit un coût supplémentaire à chaque évolution matérielle. Par ailleurs, selon de récentes études [29],[30],[31], les solutions libres (Condor, SGE, TORQUE et OpenPBS) sont aussi performantes que les solutions propriétaires (LSF et OpenPBS).

La sélection entre les différentes solutions gratuites est relativement difficile à faire, du fait qu'elles sont quasiment identiques au niveau des fonctionnalités et des options. Toutefois, il semble que TORQUE [32] se démarque du lot, dans le sens où il est basé sur la dernière version libre d'OpenPBS tout en rajoutant un certain nombre de fonctionnalités (scalabilité[33], tolérance aux pannes, ...). De plus, cette solution est maintenue par une communauté de grands utilisateurs (NCSA, OSC, the U.S. Dept of Energy, PNNL, Univ of Buffalo, TeraGrid,...), ainsi qu'un nombre croissant d'utilisateurs, ce qui garantit quelque peu sa pérennité.

Tout comme avec les bases de données, la conception du système de *cluster* dédié au calcul scientifique distribué, est très fortement influencée par la ou les applications qu'il devra faire fonctionner. Actuellement le code de calcul de noyau de sensibilité n'a aucunement été adapté à ce type de système (C1).

II.6 Problématique

L'une des spécificités de notre système est que sa partie applicative devra intégrer un code de calcul développé sous MATLAB.

II.6.1 MATLAB

MATLAB [33], abréviation de (*Matrix Laboratory*), est un applicatif dédié au calcul scientifique, développé par MathWorks. MATLAB est considéré comme un langage de programmation au même titre que C, C++, FORTRAN. Toutefois, il est plus performant dans le sens où il permet la résolution de nombreux problèmes en beaucoup moins de temps

[33] La capacité d'un système, à évoluer en puissance, le plus souvent par ajout ou remplacement de composants. Il s'agit d'une extension du concept de « portabilité ».

qu'il n'en faudrait pour les formuler en langage compilé. De plus, il intègre de nombreuses boîtes à outils « *toolbox* » dans de nombreux domaines (traitement du signal, traitement d'image, optimisation, contrôle ...).

Dans notre cas, MATLAB n'a pas que des qualités. En effet, les programmes MATLAB sont écrits en langage script, ce qui nécessite la présence de l'applicatif pour pouvoir interpréter le code.

A ce niveau, se pose la question de comment doit-on faire pour relier les différents éléments qui constituent notre système (Serveur Web, MATLAB, Bases de données, JMS) ?

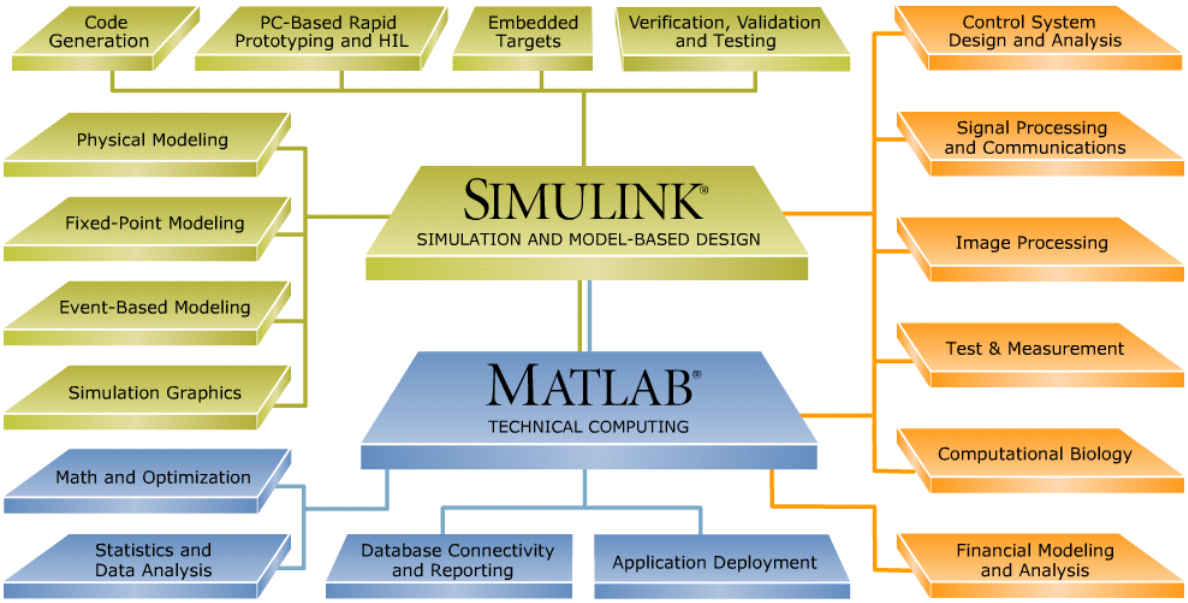

Figure 15 – Schéma des différents *toolbox* de MATLAB.

II.6.2 Intégration de MATLAB au Système

Apres étude des technologies MATLAB [33], nous pouvons dire que pour obtenir une solution quasi fonctionnelle (Figure 16) qui puisse intégrer MATLAB et l'interfacer avec tous nos choix technologiques (cf. II.5), il nous faudra :

1) MATLAB W*ebserver Toolbox* [34], pour connecter le serveur Web (Apache) à MATLAB.

2) MATLAB *Database Toolbox* [35], pour permettre à MATLAB d'établir une connexion aux bases de données compatibles ODBC[34]/JBDC[35].

[34] *Open Database Connectivity.*

[35] *Java DataBase Connectivity.*

3) MATLAB *Distributed Computing Toolbox* [36], et *Distributed Computing Engine* [37], sont nécessaires pour pouvoir lancer des taches de calcul sur les différents nœuds du *cluster*. Concernant le JMS, la solution de distribution MATLAB intègre un *job manager* assez simple.

Cette approche n'est pas conforme aux spécifications de notre système. En effet, utiliser des outils MATLAB est long à mettre en œuvre et difficile à distribuer (problème de licence). De plus, cette solution nécessite un investissement initial non négligeable, qui devra évoluer avec les rajouts des nœuds de calcul. Par ailleurs, la pérennité des outils MATLAB est très influencée par le marché[36].

[36] MATLAB *Webserver Toolbox* a disparu de la dernière version de MATLAB (2006b).

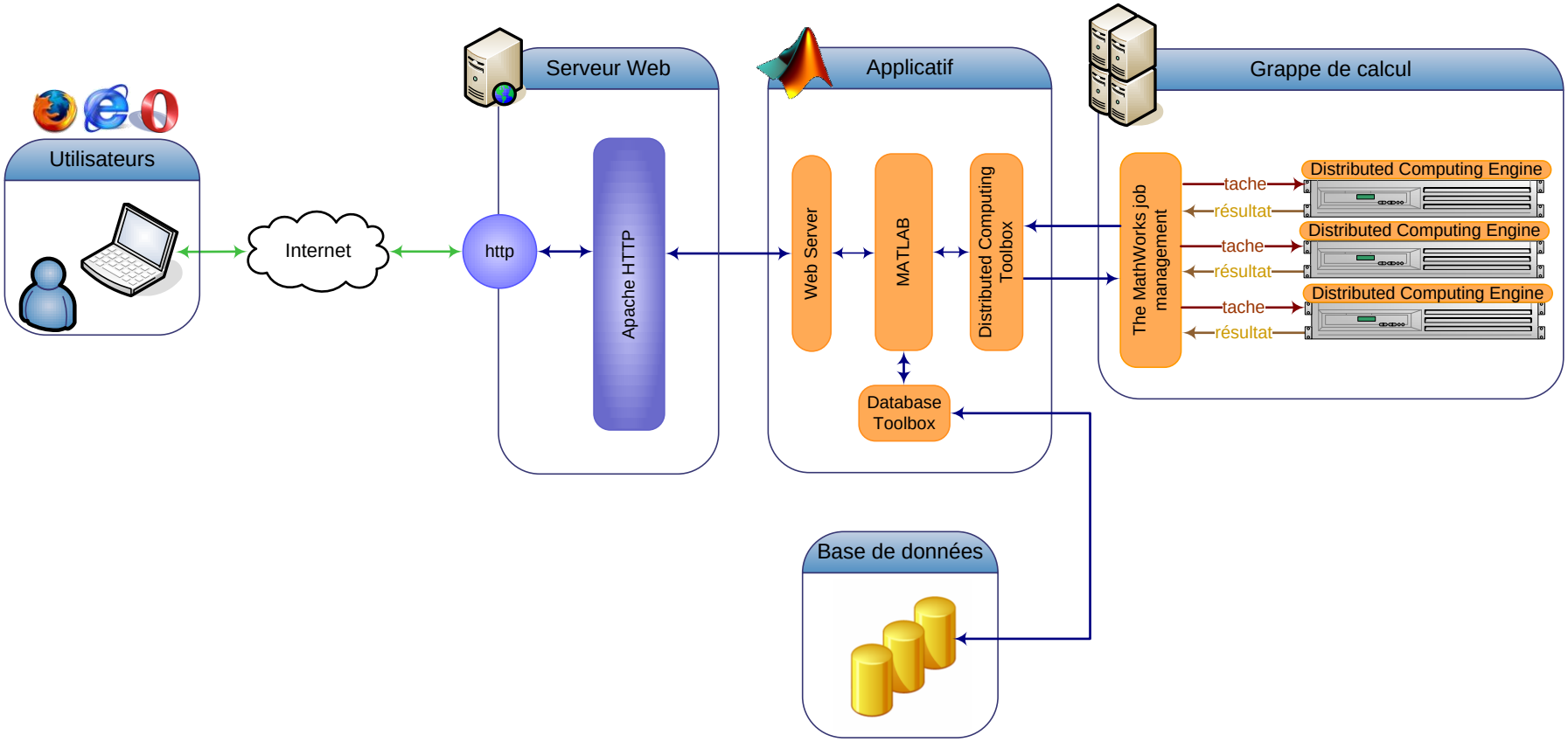

Figure 16 – Schéma de notre système dans le cas d'une solution propriétaire MATLAB.

CHAPITRE III
Implémentation et mise en œuvre

« La simplicité est la sophistication suprême »

Leonard de Vinci (1452-1519).

Le système tel qu'il a été représenté dans le chapitre précédent, ne répond pas à nos exigences. À cet effet, nous avons dû penser à une autre façon de faire, et qui puisse aboutir à une solution plus conforme avec nos spécifications.

III.1 Quel langage choisir ?

Comme souligné dans la partie concernant l'architecture du serveur applicatif (cf. II.4.2), on choisira l'approche avec client léger. Cette dernière, repose sur le fait que la présentation n'est plus créée dans le client, mais par un serveur d'application Web intégrant au moins l'une des technologies suivantes :

1) ASP.NET (*Active Server Pages)* : standard propriétaire (Microsoft) qui permet de développer des applications Web interactives, c'est-à-dire dont le contenu est dynamique. L'inconvénient majeur d'ASP est qu'il n'est disponible qu'avec le serveur Web de Microsoft (*IIS*) et donc ne s'exécute que sous le système d'exploitation Windows.
2) CGI (*Common Gateway Interface*) : est une norme qui définit un mécanisme permettant au serveur HTTP de transmettre les informations d'une requête à des programmes externes. On peut écrire des langages scripts dans différents langages. L'inconvénient de ces scripts est qu'ils sont très gourmands en ressources systèmes. Chaque script utilise un processus différent, ce qui demande beaucoup de mémoire et d'utilisation processeur.
3) JSP (*JavaServer Pages*) : est une technologie JAVA directement concurrente d'ASP, de plus elle est indépendante de la plateforme. Les commandes JSP sont placées dans le code HTML. Leur mode de fonctionnement est relativement lourd, il se déroule en 4 temps : La requête est reçue par le serveur, la page demandée est traduite en servlets, puis compilée et exécutée pour être à la fin transmise au client.

4) JAVA *Servlets* : sont un peu l'inverse des JSP, dans le sens où le code HTML est intégré dans du code JAVA. Leur mode de fonctionnement se passe en deux temps : la requête est reçue par le serveur puis la *servlet* est exécutée et la page HTML est transmise au client.

5) PHP (Hypertext Preprocessor) est un langage interprété (un langage de script) exécuté du côté du serveur. La syntaxe du langage provient de celle du langage C. Ses principaux atouts sont : sa gratuité, son intégration au sein de nombreux serveurs web (Apache, Microsoft IIS, ...), la simplicité d'écriture de scripts, et la simplicité d'interfaçage avec des bases de données.

Une analyse sommaire de ces différents concepts, nous permet de réduire notre choix et de retenir les solutions : JAVA (Servlet/JSP) et PHP.

III.2 Les solutions d'interfaçage pour MATLAB ?

Si on regarde de prêt la documentation[37] de MATLAB concernant les appels externes, on remarque que seul FORTRAN et C à travers ce qu'on appelle la *Engine Library* peuvent faire appel à MATLAB.

L'approche qui rejoint notre choix de langage de programmation consiste à utiliser du code JAVA et du C [39] ; cette technologie est connue sous le nom de *Java Native Interface* (JNI). Toutefois, cette façon de faire place toujours MATLAB au centre de notre système, et nous rend entièrement dépendant de ce logiciel et surtout de ses contraintes. A cela, on peut rajouter le fait que le code devra être recompilé et parfois réaménagé au gré des systèmes d'exploitation sur lesquels il devra être porté (SP4).

III.3 Mon approche

Pendant que je m'intéressais à la façon de contourner les contraintes de la solution précédente, un autre souci est apparu et qui concerne cette fois le nombre insuffisant de licences MATLAB[38](C4). A ce moment-là, il fallait soit acheter une nouvelle licence, soit trouver une façon de contourner MATLAB.

III.3.1 La compilation du code MATLAB

Cette opération est rendue possible grâce à un outil dénommé MATLAB Compiler [38]. Ce dernier permet notamment de convertir un code script MATLAB en un programme

[37] http://www.mathworks.com/access/helpdesk/help/techdoc/matlab_external/index.html

[38] Le MPS dispose de quatre licences MATLAB, en d'utilisation la majorité du temps.

exécutable (Figure 17), facilement distribuable (SP4). Dans notre cas le code ainsi généré sera placé dans un répertoire commun et visible par toutes les machines dédiées au calcul.

Les questions qui se posent maintenant sont : comment doit-on faire pour passer les paramètres de calcul (C1) saisis via un navigateur Web (SP1) audit code ? Suite à cela, comment faire pour exécuter programme sur des machines distantes (SP2)?

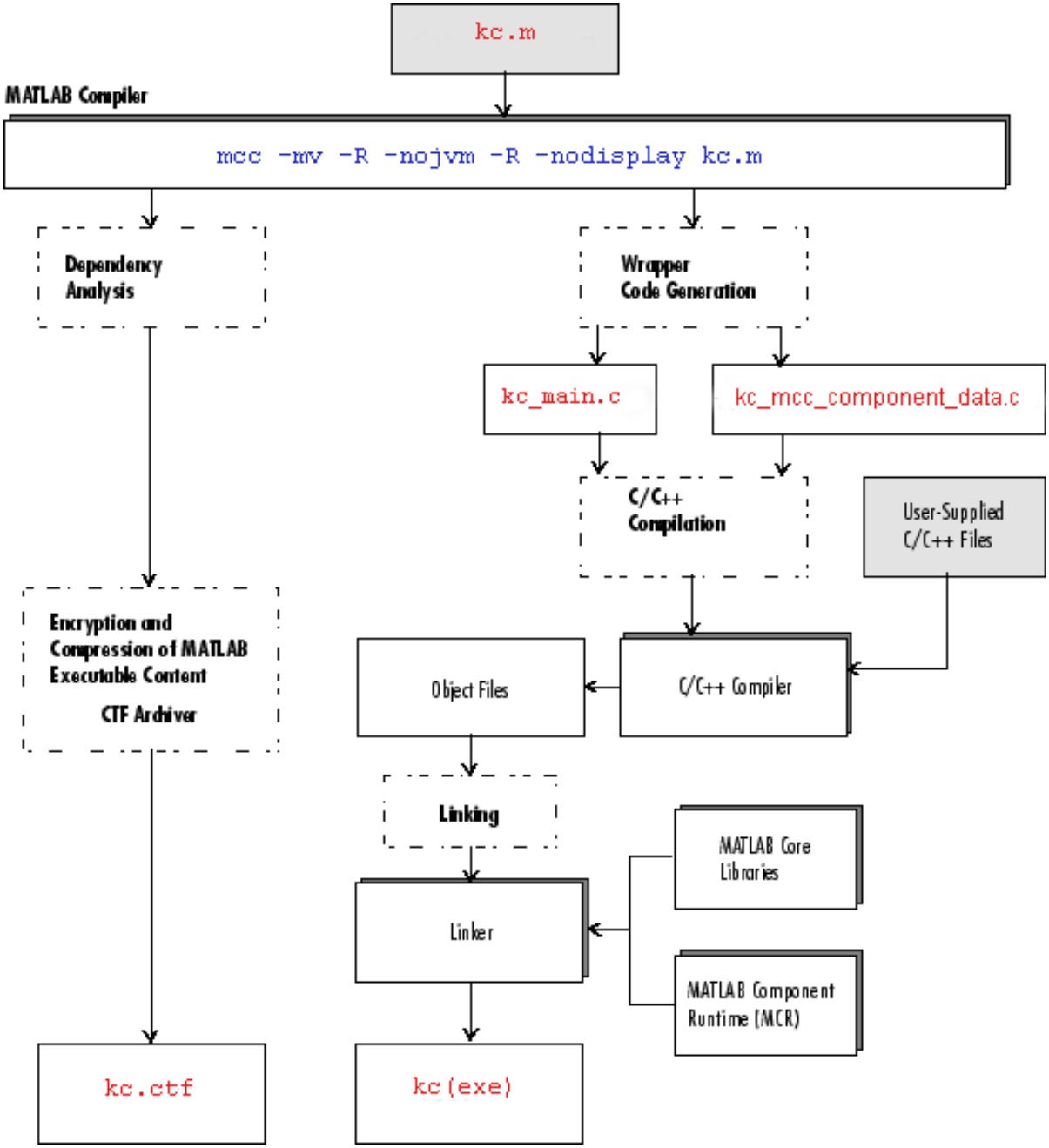

Figure 17 – Etapes de conversion du code de calcul des noyaux de sensibilité.

III.3.2 Les réponses de PHP

Les pages contenant les formulaires ont été développées en HTML. Leur organisation est présentée dans la

Figure **18**. Une partie authentification précède l'accès aux formulaires. Afin de réduire le poids total des pages, nous avons crée une feuille de style CSS (*Cascading Style Sheets*).

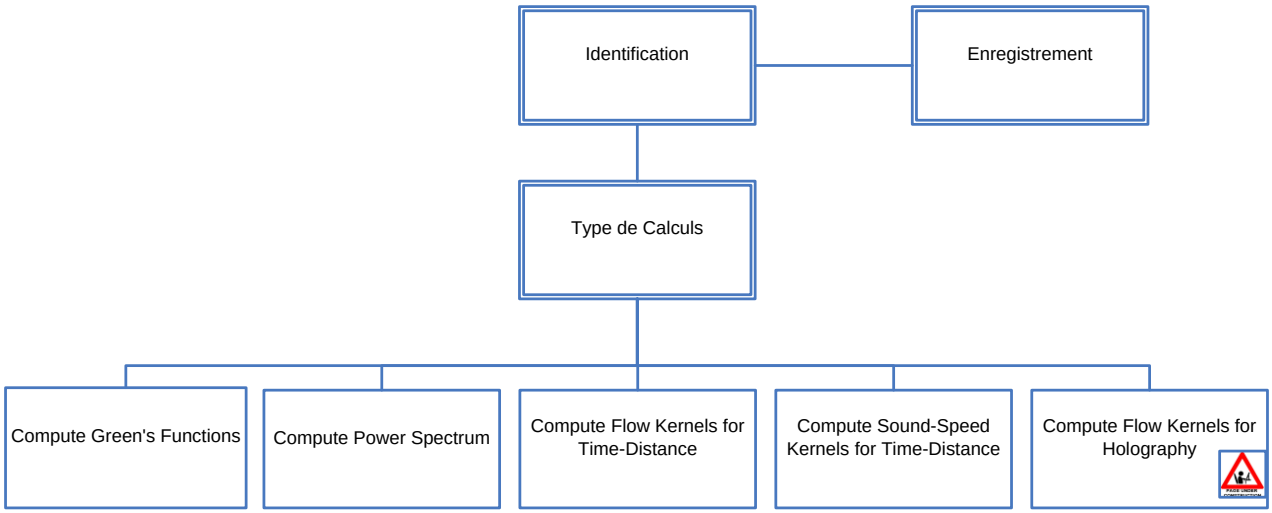

Figure 18 – Arborescence des pages web

Un code PHP [40] est intégré dans le fichier HTML, il se chargera notamment de récupérer les champs des différents formulaires et de les stocker dans un fichier texte. Ce dernier contiendra toutes les informations nécessaires à l'exécution du code de calcul.

La procédure d'envois du fichier, et d'exécution du code, se fait grâce à des fonctions PHP qui exploitent le protocole sécurisé SSH (Figure 19).

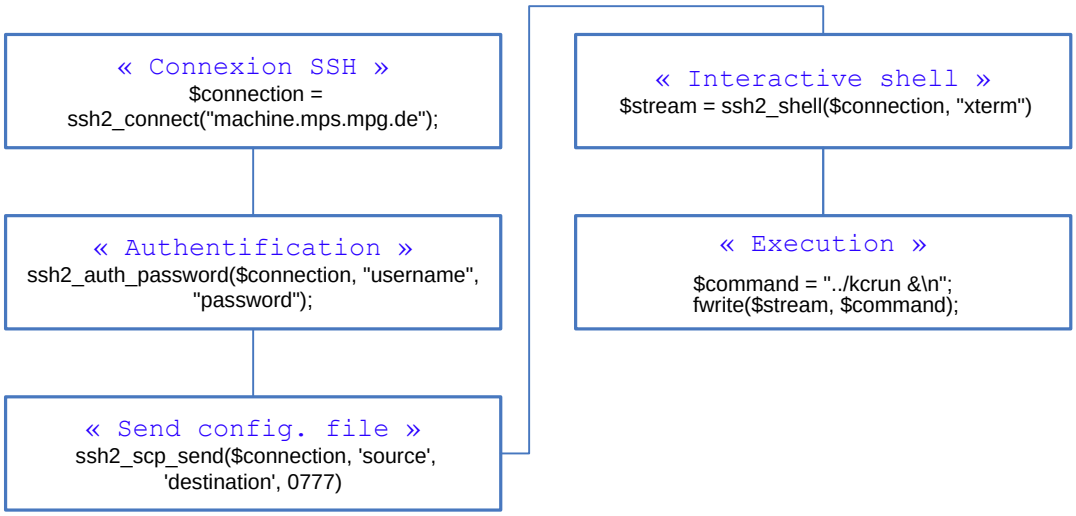

Figure 19 – Processus nécessaires à l'exécution du code de calcul sur machine distante

III.4 Synthèse

L'état actuel du système représenté par la Figure 20, est conforme à la plupart de nos exigences :

1) Interface : un simple navigateur Web avec JavaScript activé.
2) Distribution de charge : est effectué explicitement lors de la connexion SSH.
3) Sécurité et authentification : dans la version fonctionnelle,[39] les utilisateurs seront identifiés avant d'accéder aux pages hébergeant les formulaires de calcul.
4) Portabilité : la solution est multiplateforme, elle nécessite juste la présence des différentes technologies (Serveur Web, PHP, et certains modules qui permettent d'exploiter les fonctions SSH de PHP[40]).

Les fichiers nécessaires aux lancements des calculs, ont été validés par le concepteur du code (A. Birch). Les résultats obtenus en utilisant l'interface semblent identiques à ceux obtenus directement à partir de MATLAB.

Avec la récente acquisition de machines dédiées aux calculs, il sera question d'implémenter le JMS sélectionné (cf. II.5.3) et de confirmer qu'il s'intègre bien au système (Figure 21). C'est-à-dire que les tâches seront distribuées en fonction des ressources disponibles (de façon implicite).

[39] Cette fonctionnalité existe mais non encore implémenté dans la version de test.

[40] http://cz2.php.net/manual/fr/ref.ssh2.php

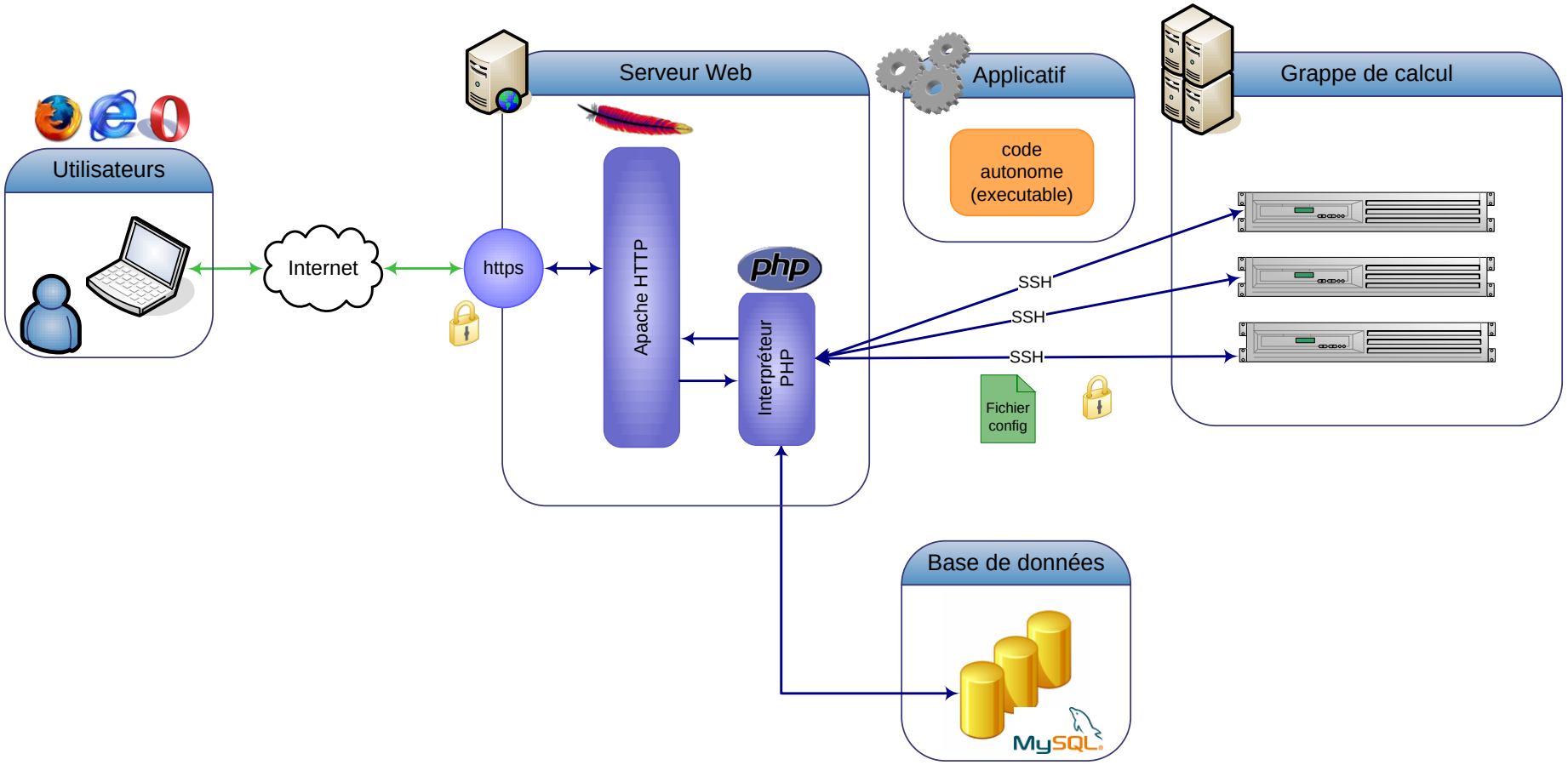

Figure 20 – Schéma du système actuel.

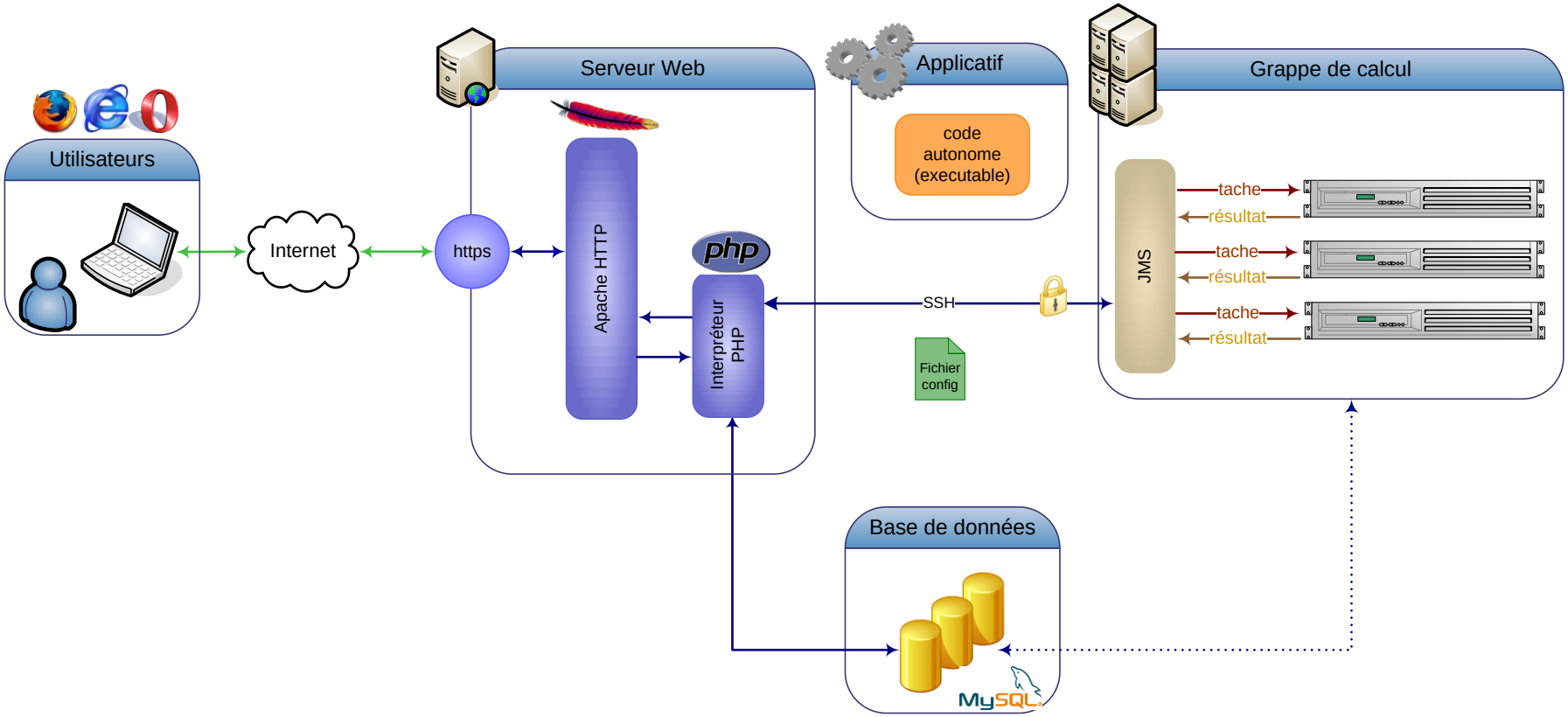

Figure 21 – Schéma du système après intégration du JMS.

Conclusion

Au terme de ce travail, nous pouvons dire que l'objectif, qui consistait à mettre en place un mécanisme rendant accessible des codes de calcul de noyaux de sensibilité via une interface Web, a été atteint.

La concrétisation de ce travail, à était rendu possible grâce à une approche systémique appliquée à la conception d'outils informatique. Dans notre démarche, nous avons défini les besoins ainsi que les contraintes, qui allaient conditionner le cycle de vie de notre système. Cette approche nous a aussi permis d'identifier d'autres problèmes non prévus initialement, et pour lesquels nous avons proposé des solutions. Certes, certains choix techniques de ce système sont encore à définir en détail, mais le principe général devrait toutefois en être retenu et amélioré.

Par ailleurs, la diversité des technologies impliquées dans ce projet et le temps imparti à le réaliser, ont modéré m'a curiosité d'aller dans les détails de certains points. À cet effet, j'espère avoir l'occasion de continuer à travailler sur ce projet et revoir toutes les étapes avec plus de détails. Ceci sera un préalable nécessaire avant le passage à la seconde étape, qui consiste à développer et à implémenter une base de données pour notamment les noyaux de sensibilité.

Bibliographie

[1] R. B. Leighton, R. W. Noyes and R. W. Simon, Astrophys. J. 135, 474 (1962).

[2] R. K. Ulrich, Astrophys. J. 162, 993 (1970).

[3] F.-L. Deubner, Astron. Astrophys. 44, 371 (1975).

[4] R. F. Stein and A. Nordlund, Astrophys. J. 546, 585 (2001).

[5] J. Christensen-Dalsgaard, Rev. Mod. Phys. 74, 1073 (2002).

[6] B. Mosser, Structure Interne et Sismologie, III Sismologie, Ecole Doctorale d'Astrophysique d'Ile de France, (2001-2002).

[7] T. L. Duvall, S. M. Jefferies, J. W. Harvey and M. A. Pomerantz, Nature 362, 430 (1993).

[8] T. L. Duvall, A. G. Kosovichev, P. H. Scherrer, R. S. Bogart, R. I. Bush, C. de Forest, J. T. Hoeksema, J. Schou, J. L. R. Saba, T. D. Tarbell, A. M. Title, C. J. Wolfson and P. N. Milford, Solar Phys. 170, 63 (1997).

[9] A. C. Birch, A. G. Kosovichev and T. L. Duvall, Astrophys. J. 608, 580 (2004).

[10] L. Gizon and A. C. Birch, Astrophys. J. 571, 966 (2002).

[11] L. Gizon and A. C. Birch, Living Rev. Solar Phys. 6, 2 (2005).

[12] http://preprints.ians.uni-stuttgart.de/downloads/2003/2003-005.pdf

[13] Apache Software Foundation The Apache HTTP Server Project http://httpd.apache.org/

[14] Apache Software Foundation Apache Tomcat http://tomcat.apache.org/

[15] Microsoft Internet Information Services http://www.microsoft.com/WindowsServer2003/iis/default.mspx

[16] F. Niño, H. Le Grand, N. Foures, Bases de Données et Interfaces Web , Revision: 1.1 MEDIAS-FRANCE,

[17] Comparison of the Enterprise Functionalities of Open Source Database Management Systems Daniel Fallmann, Helmut Fallmann, Andreas Pramböck, Horst Reiterer, Martin Schumacher, Thomas Steinmaurer, Roland Wagner, April 26, 2005

[18] Lewis R Cunningham, Oracle 10g vs PostgreSQL 8 vs MySQL 5, 2005 http://www.suite101.com/print_article.cfm/oracle/115560

[19] Mohammed-Lotfi Mattou, Digitalis Consulting Sàrl 2003
http://www.digitalisconsulting.com/doc/tech/OpenSource-SGBDR-6-2003.pdf

[20] F. Bordage, Quatre SGBDR open source pour Windows, Decision Informatique, (2006).
http://www.globalis-ms.com/download/DMR200311170570024.pdf

[21] Fermi National Accelerator Laboratory, Comparison of Oracle, MySQL and PostgreSQL DBMS, 2005
http://www-css.fnal.gov/dsg/external/freeware/mysql-vs-pgsql.html

[22] Oracle Corp., Oracle Migration Workbench,
http://www.oracle.com/technology/tech/migration/workbench/index.html

[23] LSF, http://www.platform.com/Products/Platform.LSF.Family/home.htm

[24] Condor, http://www.cs.wisc.edu/condor/

[25] SGE, http://gridengine.sunsource.net/

[26] OpenPBS, http://www.openpbs.org

[27] PBSPro, http://www.pbspro.com/

[28] Torque, http://supercluster.org/torque

[29] O. Hassaine: «Issues in Selecting a Job Management Systems (JMS)», Proc SUPerG, Tokyo, April 2001.

[30] T. El-Ghazawi, K. Gaj, N. Alexandridis, B. Schott, A. V.Staicu, J. R. Radzikowski, N. Nguyen, S. A. Suboh: « Conceptual Comparative Study of Job Management Systems », Technical Report, (2001).

[31] T. El-Ghazawi, K. Gaj, N. Alexandridis, F. Vroman, N. Nguyen, J. R. Radzikowski, P. Samipagdi, S. A. Suboh: « A performance study of job management systems », (2002).

[32] E. Imamagis, B. Radis, D. Dobrenis, « Job Management Systems Analysis », (2004)

[33] The MathWorks, Inc. MATLAB. (2006),
http://www.mathworks.com/access/helpdesk/help/techdoc/

[34] The MathWorks, Inc. MATLAB Web Server. (2006),
http://www.mathworks.com/access/helpdesk/help/toolbox/webserver/

[35] The MathWorks, Inc. MATLAB Database Toolbox. (2006),
http://www.mathworks.com/access/helpdesk/help/toolbox/database/

[36] The MathWorks, Inc. Distributed Computing Toolbox. (2006),
http://www.mathworks.fr/access/helpdesk/help/toolbox/distcomp/

[37] The MathWorks, Inc. MATLAB Distributed Computing Engine. (2006), http://www.mathworks.fr/access/helpdesk/help/toolbox/mdce/

[38] The MathWorks, Inc. MATLAB Compiler. (2006), http://www.mathworks.com/access/helpdesk/help/toolbox/compiler/

[39] A. Klimke, « How to access Matlab from Java », IANS Technical Report. (2003).

[40] PHP Reference Manaual http://www.php.net/manual/en/

[41] MySQL 5.1 Reference Manual http://dev.mysql.com/doc/refman/5.1/en/