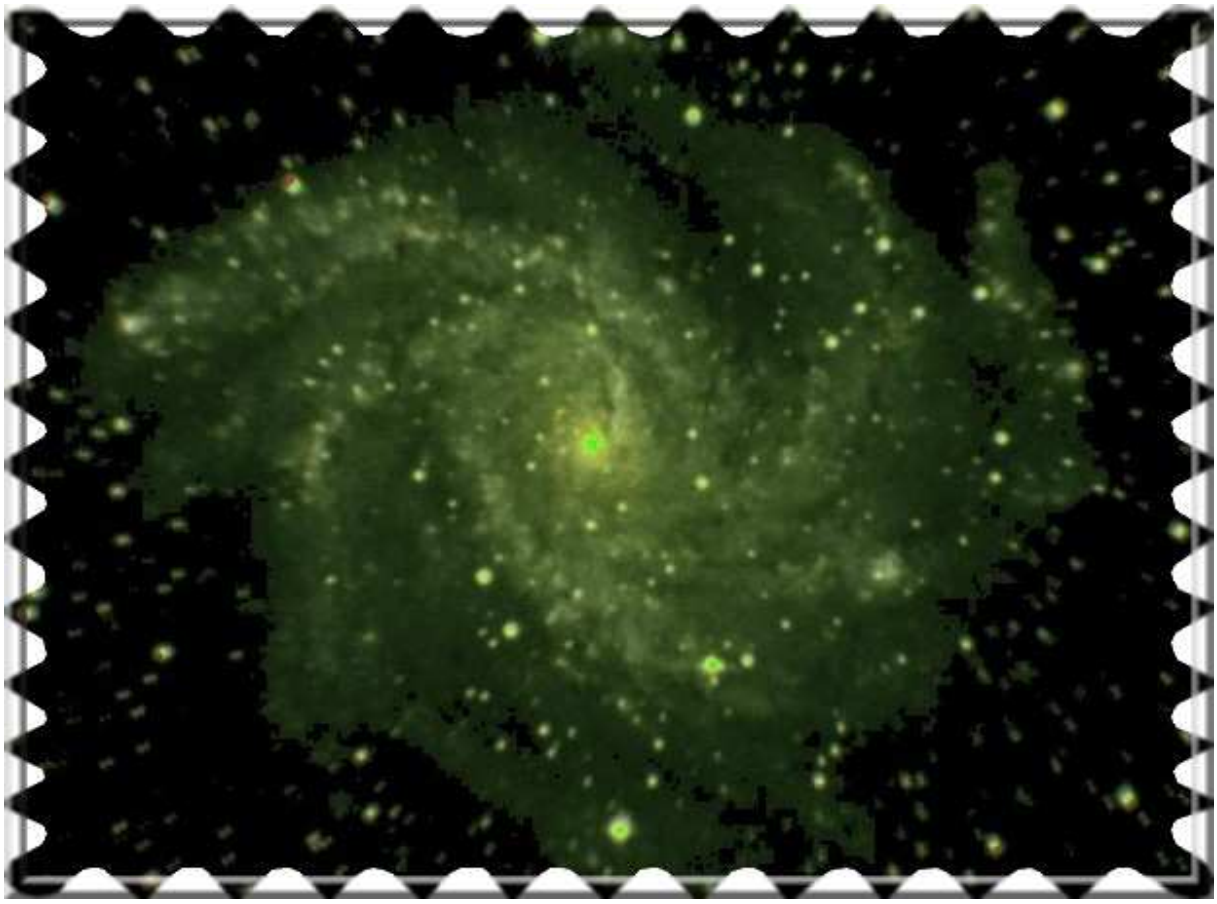
Stage à l'Observatoire de Strasbourg



Architecture d'accès à des ressources de Calcul et de Données

Tuteur du stage :
Mr Schaaff André

Rapport rédigé par :
Mr Pestel Cyril

Période du stage :
1 mars – 31 août

TABLE DES MATIERES

REMERCIEMENTS

Je remercie, pour le bon accueil, toutes les personnes que j'ai côtoyé durant la période de mon stage à l'Observatoire de Strasbourg. Principalement mon tuteur André Schaaff, pour tous ses bons conseils.

Je remercie Mme Génova, la directrice du CDS, et M. Hameury, le directeur de l'observatoire, pour leurs soutiens et pour m'avoir permis de joindre leur personnel durant ses six mois de stage.

Je remercie tout particulièrement les deux administrateurs de l'observatoire, Thomas Keller et Jean-Yves Hangouet, pour m'avoir toujours ouvert leur porte ainsi que la porte du local informatique d'où sont entreposés les machines du cluster.

INTRODUCTION

Ce rapport fait suite au stage de Master 2 professionnel de l'université Louis Paster à Strasbourg. Celui-ci a eu lieu du 1er mars au 31 août 2006 à l'Observatoire Astronomique de Strasbourg et plus précisément au Centre de Données Astronomiques de Strasbourg (CDS).

Je débuterai ce rapport par une présentation du centre de recherche dans lequel j'ai travaillé, puis je présenterai le sujet du projet en précisant les besoins et les objectifs. J'apporterai des précisions sur les notions abordées avant de détailler les outils testés. Puis nous étudierons un comparatif de ces utilitaires afin de mettre en évidence les raisons qui nous ont amené à installer l'outil choisi. Nous verrons par la suite ce qui a été nécessaire de réaliser pour obtenir un cluster pratique et fonctionnel. Et enfin, nous essayerons de faire ressortir de ce projet les évolutions possibles à apporter afin de le compléter, avant de finir sur un bilan de ce qui a été réalisé et sur la conclusion du stage.

1. PRESENTATION DU LABORATOIRE

1.1. STRUCTURE JURIDIQUE

L'Observatoire de Strasbourg est une Unité de Formation et de Recherche (UFR) de l'Université Louis Pasteur. Il est également une Unité Mixte de Recherche du CNRS et de l'Université Louis Pasteur (UMR 7550).

1.2. PRESENTATION GENERALE

L'Observatoire est situé sur le campus de l'Esplanade ; ses bâtiments font partie du campus historique de l'université de Strasbourg.

Enseignements dispensés :

– Master 2 « Analyse et Traitement des Données sur les Milieux Astronomiques »

– DEUG Sciences et autres DEUG (enseignements d'ouverture).

– Licence de Géosciences.

– Maîtrise de Géosciences, Maîtrise de Physique, Maîtrise de Sciences Naturelles.

– Préparation au CAPES et à l'Agrégation.

– Master Applications des Technologies Spatiales.

– Diffusion de la Culture

La partie publique de l'Observatoire, le Planétarium, est ouverte pour la vulgarisation de l'astronomie.

L'Observatoire se compose de quatre équipes de recherche :

1.2.1. HAUTES ENERGIES

L'équipe Astrophysique des Hautes énergies a pour thème l'étude des astres et sites de l'univers émetteurs de photons de haute énergie. Cette thématique générale recouvre des aspects variés, comme l'étude des astres compacts en fin d'évolution, la physique de leur activité, les phénomènes de haute énergie intéressant les étoiles jeunes ou le soleil, ou l'étude de ces phénomènes à l'échelle galactique. Ces recherches se sont largement appuyées sur les

données acquises par le satellite ROSAT et s'appuieront dans l'avenir sur celles des satellites X de nouvelle génération, tout spécialement XMM.

1.2.2. ETOILES ET SYSTEMES STELLAIRES

Les recherches menées par l'équipe « Etoiles et systèmes stellaires » recouvrent un domaine étendu, incluant les étoiles, les milieux interstellaires, la Galaxie et les galaxies proches. De l'étoile, objet individuel, l'intérêt s'est porté aux groupes d'étoiles, témoins de l'évolution stellaire mais aussi traceurs des grandes structures de la Voie Lactée.

1.2.3. GALAXIES

Les activités de l'équipe sont centrées sur les problèmes de la structure du Groupe Local, de ses populations stellaires et sur la dynamique gravitationnelle. De plus, l'équipe possède un savoir-faire sur les outils statistiques d'analyse de données et sur les méthodes inverses non paramétriques. Un des objectifs consiste à combiner les informations d'évolution des populations stellaires et celles de dynamique afin de reconstituer les événements déterminants liés aux processus de formation et d'évolution galactique.

1.2.4. LE CENTRE DE DONNEES (CDS)

L'activité de recherche du CDS s'est concentrée sur l'étude de la dynamique galactique et des populations d'étoiles binaires, sur une participation importante à la mission HIPPARCOS de l'Agence Spatiale Européenne, ainsi que sur le développement de méthodologies nouvelles applicables à l'analyse et au traitement de données astronomiques.

1.3. ORGANISATION DU PERSONNEL

Le CDS est un laboratoire de l'Institut National des Sciences de l'Univers (INSU), rattaché au CNRS. L'Observatoire de Strasbourg est un institut de l'Université Louis Pasteur. Le personnel permanent du CDS comprend 10 chercheurs, 6 ingénieurs de recherche, 6 ingénieurs d'étude, 3 techniciens, et plusieurs collaborateurs à contrat temporaire (projets européens, etc.) et des invités.

1.4. CENTRE DE DONNEES ASTRONOMIQUES DE STRASBOURG (CDS)

J'ai effectué mon stage au sein du Centre de Données astronomiques de Strasbourg (CDS) qui est un centre de données dédié à la collection et à la distribution dans le monde entier de données astronomiques.

Le CDS héberge la base de données *Simbad*, la base de référence mondiale pour l'identification d'objets astronomiques. Le but du CDS est de :

– rassembler toutes les informations utiles, concernant les objets astronomiques, disponibles sous forme informatisée : données d'observations produites par les observatoires du monde entier, au sol ou dans l'espace ;

– mettre en valeur ces données par des évaluations et des comparaisons critiques ;

– distribuer les résultats dans la communauté astronomique ;

– conduire des recherches utilisant ces données.

Le CDS joue, ou a joué, un rôle dans d'importantes missions astronomiques spatiales : contribuant aux catalogues d'étoiles guides, aidant à identifier les sources observées ou organisant l'accès aux archives, etc. Le CDS contribue au XMM Survey Science Center, sous la responsabilité de l'équipe « Hautes-Energies » de l'Observatoire de Strasbourg.

Le CDS a signé des accords d'échanges internationaux avec les organismes suivants :

– NASA,

– National Astronomical Observatory (Tokyo, Japon),

– l'Académie des Sciences de Russie,

– le réseau PPARC Starlink au Royaume-Uni,

– l'Observatoire de Beijing (Chine),

– l'Université de Porto Allegre au Brésil,

– l'Université de La Plata en Argentine,

– InterUniversity Center for Astronomy and Astrophysics (Inde).

Il est membre de la Fédération des Services d'Analyse de Données Astrophysiques et Géophysiques.

Le CDS coopère aussi avec l'Agence spatiale Européenne (transfert au CDS du service de catalogues du projet ESIS : le projet *VizieR*), et avec la NASA : il abrite en particulier une copie miroir du Système de Données Astrophysiques (ADS) et ADS abrite une copie miroir de *Simbad*. Le CDS contribue aussi au projet NASA AstroBrowse. Il abrite les copies miroirs Européennes des journaux de l'American Astronomical Society (AAS).

1.5. LES SERVICES DU CDS

1.5.1. SIMBAD

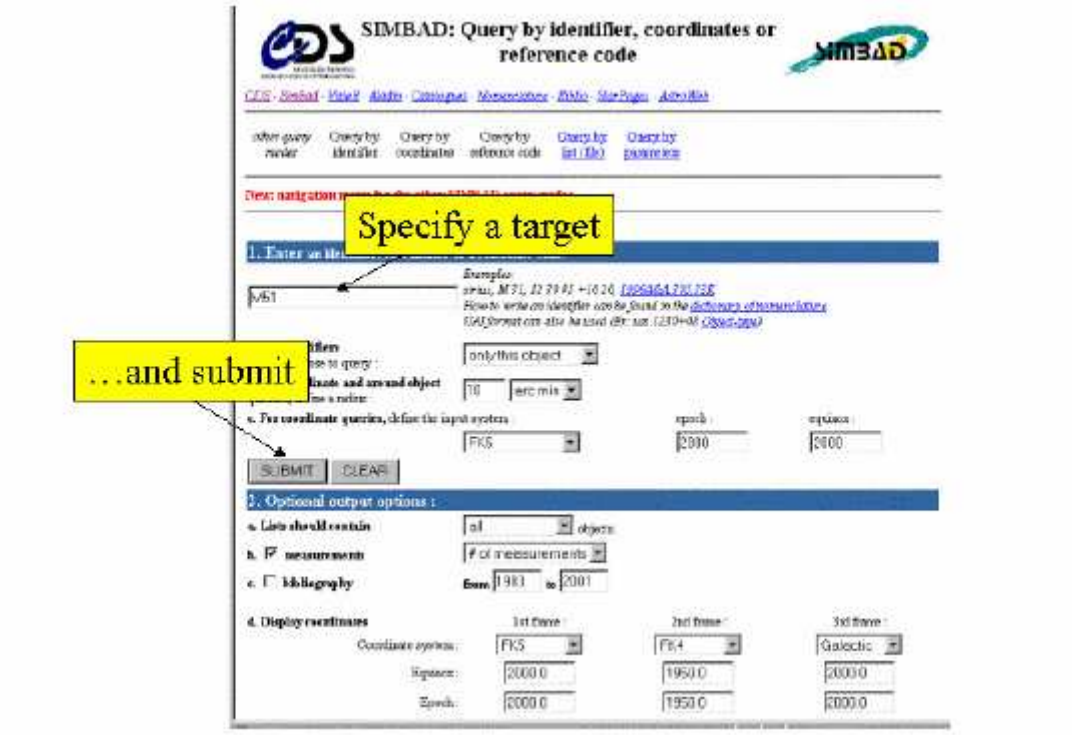


Figure. 1.1 - Page Web permettant d'effectuer une requête sur Simbad

Simbad est une base de données de référence pour les identifications et la bibliographie d'objets astronomiques. *Simbad* contient plus 7,5 millions d'identificateurs pour plus de 2,8 millions d'objets différents. Pour chaque objet figurent dans la base quelques mesures (position, magnitude dans différents domaines de longueurs d'ondes), ainsi que les références bibliographiques où l'objet est cité (plus de 110 000 articles sont concernés). L'utilisateur peut choisir le format du fichier où seront entreposés les résultats de la requête (***Fig. 1.1***). *Simbad* peut générer des fichiers HTML, XML ou XLS (fichiers Excel).

Cet ensemble de données résulte d'un long travail d'identification croisée entre de nombreux catalogues, listes d'objets et articles de journaux, entrepris au début des années 1980, et constamment développé et mis à jour depuis.

1.5.2. VIZIER

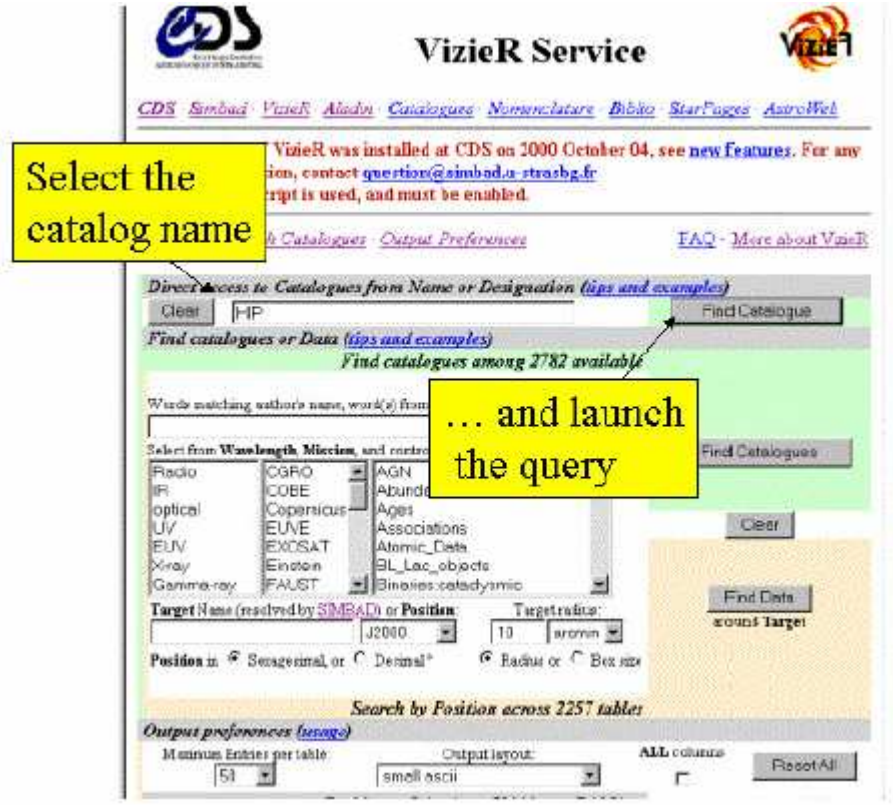


Figure. 1.2 – Page Web permettant d'effectuer une requête sur VizieR

VizieR est une base de données rassemblant plusieurs milliers de catalogues astronomiques sous un format homogène. Une description standardisée du contenu des catalogues permet leur inclusion dans un système de gestion de base de données (SGBD) relationnel. Un ensemble de liens, entre les tables de *VizieR*, et avec des services externes (bibliographiques, archives externes, serveurs d'images), permettent de naviguer entre les données des catalogues et d'autres données associées (***Fig. 1.2***). Il faut noter que les très grands catalogues (plus de 107 enregistrements) ne peuvent pas être gérés par un SGBD relationnel pour des raisons de performances, c'est pourquoi des outils spécifiques doivent être utilisés.

1.5.3. ALADIN

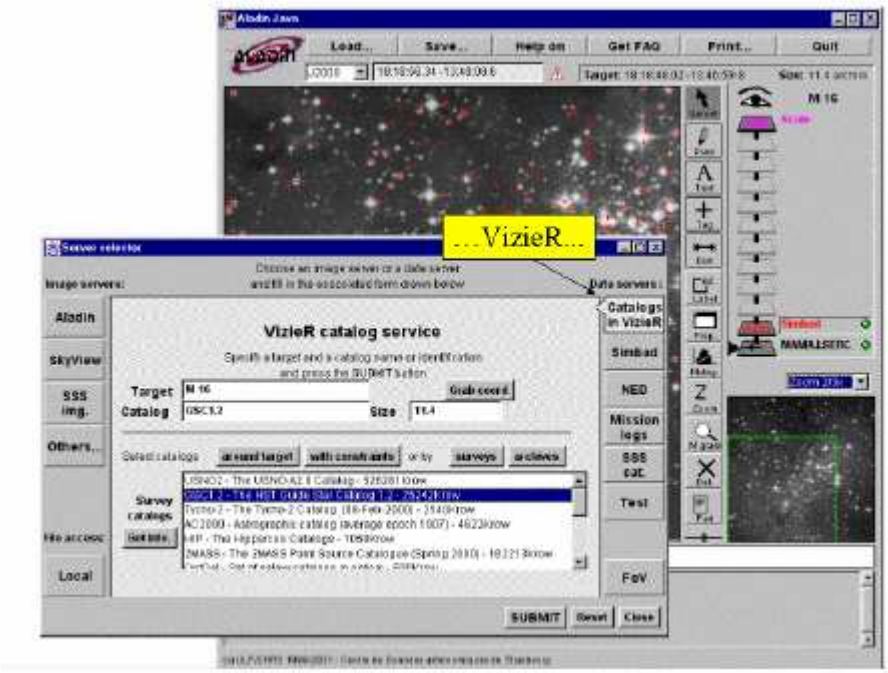


Figure. 1.3 – Exemple d'utilisation d'Aladin

Aladin est un atlas interactif du ciel permettant d'accéder simultanément à des images numérisées du ciel, ainsi qu'à des catalogues et bases de données astronomiques. Cet outil permet de superposer, sur des images du ciel optique, les objets présents dans *Simbad*, des sources de catalogues contenus dans *VizieR*, mais aussi d'autres données, locales ou situées sur des serveurs distants (archives, HST, . . .).

2. ANALYSE DE LA SITUATION

2.1. ANALYSE DE L'EXISTANT

L'observatoire de Strasbourg dispose de services qu'il met à disposition de l'ensemble de la communauté astronomique internationale. Cette dernière comprend des chercheurs en astronomie et astrophysique mais également des amateurs (les services étant publiques). Les principaux services sont proposés à travers un portail Web. Ils sont assez variés puisque le CDS proposent aussi bien des publications que des accès aux bases de données (VizieR et Simbad qui contiennent respectivement des catalogues astronomiques et des identificateurs pour des objets astronomiques) ou un atlas astronomique (Aladin qui permet à n'importe qui de pouvoir afficher chez lui des images astronomiques issues du serveur et de les manipuler (rotations, déplacements, zooms, fusions de plusieurs images, …)). Ce genre de traitement d'image nécessite beaucoup de puissance de calcul, et les opérations associées sont traitées au sein même de l'observatoire pour que seules les images résultantes soient transmisses sur les postes clients. Le service Aladin dispose d'un serveur Sun (quatri-processeurs) basé sur Solaris, un système d'exploitation Unix. Afin d'alléger la charge CPU de cette machine, le CDS a mis en place un cluster (voir chapitre *3.1. Définition*) de 5 machines. Ce dernier a été installé lors d'un précédent stage par Thomas Bucher (UTBM) en 2003. Il a conçu Ali, un répartiteur de charge adapté au besoin d'Aladin. Minimaliste mais efficace, il se distingue par sa souplesse et sa facilité d'installation et de maintenance. Les requêtes envoyées au service Aladin sont alors traitées soit par le serveur Sun soit sous-traitées au cluster. Celui-ci a permis également de renforcer le service VizieR dont une partie des catalogues est répartie sur le cluster.

2.2. ANALYSE DES BESOINS

Depuis la fin du stage de Thomas Bucher en 2003, et la naissance du cluster, de nombreux traitements de complexité variée sont ajoutés afin de répondre aux demandes permanentes des internautes et autres utilisateurs d'Aladin. Le cluster est opérationnel depuis décembre 2003, cependant le nombre et la complexité des traitements augmentent, ce qui favorise un

accroissement de la demande au niveau des utilisateurs. Cette augmentation des besoins, conjuguée au vieillissement du matériel impliquait une refonte du cluster. Dans un premier temps, nous avons décidé de spécialiser les nœuds (calcul ou catalogue) afin de réduire les interactions néfastes (un même nœud sollicité simultanément pour du calcul et des entrées/sorties massives liées à l'accès aux catalogues). Puis, nous avons également décidé de remplacer une partie des machines par des machines biprocesseurs plus puissantes et plus performantes. Dix nouvelles machines ont ainsi été commandées, 8 dédiées au calcul et 2 autres (avec 3,2To de disque dur) pour les catalogues. A terme, puisque les demandes d'accès aux services du CDS ne font qu'augmenter, la sous-traitance des tâches les plus longues pourraient être faîte à une grille du type Grid5000 *[1]* ou EGEE *[2]* (projet initialisé par la commission européenne) et celles demandant une réponse rapide seraient traitées sur le cluster de calcul.

2.3. OBJECTIFS A ATTEINDRE

Le premier objectif du projet était de tester et sélectionner les outils nécessaires à l'installation et à la configuration de l'ensemble des noeuds de la grappe. En effet, lorsqu'on possède un parc informatique, il est souhaitable de ne pas devoir installer une à une les machines. Il faut pour cela employer des utilitaires qui automatisent certaines phases afin de rendre l'installation moins interactive. Le but final étant de connecter un noeud à la grappe avec le moins d'intervention possible de notre part, l'installation de l'OS et des programmes utiles se faisant à l'identique des autres nœuds. Dans le cas du cluster, les machines sont semblables (matériels et logiciels) afin d'assurer une maintenance aisée ainsi qu'une automatisation poussée des phases d'installation.

Une fois l'installation des machines effectuée, on souhaite que le cluster soit capable d'effectuer les traitements envoyés, et cela le plus rapidement et le plus efficacement possible. De plus, certains programmes utilisent les librairies pour la programmation parallèle du type MPI *[3]* (Message Passing Interface) ou PVM *[4]* (Parallel Virtual Machine), ce qui induit l'installation de bibliothèques spécifiques. La grappe de calcul ainsi formée doit être en mesure de répondre à tous ces objectifs.

2.4. DEROULEMENT DU STAGE

Tout à commencer par un état de l'art visant à réaliser une recherche approfondie des éléments à prendre en compte pour mener à bien le projet. Le mois de mars m'a permis de me familiariser avec les solutions existantes, et surtout avec le sujet afin de prendre en compte les points importants. À la fin de ce mois, une réunion avec mon tuteur et une personne concernée par mon travail, a permis d'en retirer les orientations possibles.

Pendant le mois suivant, je me suis consacré à l'exploration de toutes les solutions retenues. Durant cette période, j'ai testé un système d'exploitation, trois grands utilitaires réunissant de nombreux programmes spécialisés dans l'administration de cluster. Ce mois fût fort intéressant car j'ai pu dialoguer avec Romaric David qui s'occupe du cluster situé au pôle API. De plus, j'ai eu l'occasion d'assister à une présentation d'un outil dédié au cluster appelé *Kerrighed [6]* qui permet de voir l'ensemble des ressources des machines comme un tout.

Durant les deux mois suivants, j'ai travaillé en parallèle sur deux projets. En plus de la mise en place du cluster, j'ai repris le projet que j'avais commencé l'année précédente à l'observatoire durant la même époque. Celui-ci correspondait à la construction d'une librairie graphique Java permettant la conception d'un *Workflow* (voir chapitre *7.4. Workflow*). J'ai organisé mon emploi du temps afin d'ajouter des fonctionnalités à la librairie et de continuer les tests pour installer le gestionnaire de clustering sélectionné. Il est à noter que ces améliorations étaient nécessaires en vue d'une présentation auprès du groupe *OVFrance* (Observatoire Virtuel de France) ainsi qu'en raison de l'introduction de la notion de *Workflow* dans la suite du stage.

A partir du mois de juillet, nous avions un cluster administrable de manière efficace et simple comme s'il s'agissait d'une seule et même machine. Il nous restait encore à installer les différents services que souhaitait proposer le CDS.

3. PRESENTATION DES TECHNOLOGIES ABORDEES

Pour commencer, il serait bon de définir ce qu'est un cluster. Il faut préciser tout d'abord que c'est une appellation anglaise qui exprime l'idée de grappe. On utilise régulièrement le terme grappe de serveurs ou ferme de calcul. Et pour cause, puisqu'il sert bien souvent à résoudre des tâches demandant une capacité de calcul extrêmement importante (HPC *[6]*, terme anglais signifiant High Performance Computing). On peut également parler de grappe de serveurs dans le cas d'un cluster composé de machines serveurs permettant de garantir en permanence l'accès à des services et cela même en cas de défaillance de l'un des serveurs.

3.1. LES TYPES DE CLUSTERS

Il existe trois type de cluster : clusters de calcul, clusters de haute disponibilité, clusters à répartition de la charge.

3.1.1. CLUSTERS DE CALCUL

Ce type de cluster est destiné à permettre la mise en commun de ressources de calcul. Il est utilisé pour résoudre des problèmes calculatoires en distribuant les parties indépendantes du calcul sur les différents noeuds.

3.1.2. CLUSTERS DE HAUTE DISPONIBILITE

Lorsque l'on met en place un cluster de ce type, on ne cherche pas en priorité la performance du calcul mais plutôt à assurer la disponibilité de la ressource, principalement en cas de panne de l'un des serveurs. La ressource peut aussi bien être un service (calculs numériques, applications, etc.) qu'un fichier. Ils sont basés sur des mécanismes de redondance et de détection des pannes.

3.1.3. CLUSTERS A REPARTITION DE LA CHARGE

Ces clusters sont typiquement utilisés pour les services web, mais ils peuvent tout aussi bien être utilisés dans d'autres domaines. Toutes les requêtes arrivent sur un même noeud du cluster, le seul visible de l'extérieur, mais la requête sera envoyée sur un autre. Le choix se fait en fonction de la charge de chacun des noeuds, de manière à traiter la requête le plus rapidement possible.

3.2. DU SUPERORDINATEUR AU CLUSTER

3.2.1. LES SUPERORDINATEURS

Le cluster a été inventé dans le but d'apporter une puissance de calcul équivalente aux superordinateurs que fabrique les géants de l'informatique comme par exemple IBM, HP et d'autres encore. Le site Top500 référence deux fois par an les 500 plus grands superordinateurs du monde entier. Et bien sûr, le césar des supercalculateurs est attribué à BlueGene/L d'IBM. Le Tera10 (classé 62ieme) de Bull (société française spécialisée dans l'informatique professionnelle) basé à Bruyères-le-Châtel peut se vanter d'être l'ordinateur le plus puissant de France. Il appartient à la Direction des Applications Militaires du CEA (Commissariat à l'énergie atomique) et compte pas moins de 1008 processeurs (à titre comparatif BlueGene comporte 65636 processeurs à double coeur) pour une puissance de calcul de 60 TFlops/s (280 TFlops/s pour celui d'IBM).

3.2.2. LES CLUSTERS

Les plus grands clusters sont en général reliés par des liaisons spécialisées à très hauts débits, on peut citer pour exemple la technologie Myrinet, créé par la société Mircom. Elle permet d'atteindre des débits de l'ordre de 2 Gbits/s et offre des temps de latence très faible. Malheureusement, cette technologie a un coût nettement supérieur à l'Ethernet. Il existe aussi la technologie InfiniBand qui fournit un débit de 10 Gbits/s dans chaque direction. L'InfiniBand utilise une technologie permettant à plusieurs périphériques d'accéder en même temps au réseau.

On constate qu'il existe également une catégorie de cluster appelé Beowulf qui à la particularité d'être fait pour le calcul HPC *[6]* mais qui est composé de machines bon marché. Évidemment l'ensemble est peu coûteux puisque le réseau (LAN) est constitué au mieux de câble Ethernet GigaBit. De plus, le système d'exploitation est de préférence basé sur une infrastructure Open Source et gratuite comme GNU Linux. Au final, l'objectif est de composer un cluster avec le plus de machines possibles afin d'obtenir les meilleures performances au plus bas prix.

Sur ce genre d'infrastructure il est nécessaire d'utiliser des librairies parallèles du genre MPI *[3]* ou PVM *[4]* afin d'optimiser les performances. Il est alors possible avec la programmation parallèle d'utiliser toute la puissance de calcul mise à disposition par le cluster. Cette bibliothèque est aussi appelée lors de tests effectués sur les grappes afin de mesurer ces performances. Linpack et Lapack sont les deux librairies écrites en Fortran les plus utilisées pour ces mesures.

3.3. LES GRILLES

Les techniques employées pour mettre en place des grilles ou des clusters ont les mêmes objectifs, notamment celui de mise en commun de ressources de calcul ou de ressources de stockage. Mais l'environnement et les hypothèses de travail dans les deux cas ne sont pas les mêmes.

Dans le cas des clusters, l'ensemble des noeuds se trouve dans la même organisation (une entreprise ou un centre de recherche par exemple). Cette proximité géographique permet un contrôle plus grand sur le matériel, sur les logiciels et sur la politique de sécurité. Cela permet d'avoir un environnement aussi homogène que l'on souhaite et de bénéficier d'un réseau de communication dédié entre les noeuds ayant un haut débit et une latence faible.

Par contre dans le cas des grilles, la répartition géographique des noeuds de la grille est complètement différente. Elle est beaucoup plus vaste que pour les clusters. Bien souvent, les noeuds sont situés dans des organisations différentes. On se trouve donc dans un environnement qui a peu de chance d'être homogène. L'homogénéité est perdue aussi bien au niveau matériel et logiciel, qu'au niveau des politiques de sécurité qui sont spécifiques à chaque organisation.

Une grille est en effet une infrastructure, c'est-à-dire un ensemble d'équipements techniques d'ordre matériels et logiciels. Cette infrastructure est qualifiée de virtuelle car les relations entre les entités qui la composent n'existent pas matériellement mais numériquement.

La grille souffre de deux défauts : la faiblesse de son niveau de sécurité, qui dissuade nombre d'entreprises d'imaginer des grilles qui vont au delà de leurs pare-feux ; et la lenteur des temps d'accès, incomparables à ceux d'un supercalculateur traditionnel : les requêtes qui cheminent sur une grille doivent emprunter le réseau avant d'être traitées. Certains types de calculs souffrent beaucoup de ces lenteurs, qui interdisent de rebondir en quelques fractions de secondes vers une autre opération.

4. PRESENTATION DES OUTILS DISPONIBLES

4.1 SOLUTIONS POSSIBLES

Il existe différentes solutions qui répondent à nos besoins. Cela peut consister en l'installation d'une distribution toute entière (GNU Linux car nous souhaitons limiter l'achat de licences). Cela permet d'avoir de facto un cluster homogène au niveau logiciel. Il en existe quelques unes dédiées à l'architecture d'une grappe, ce qui paraît le plus judicieux dans notre cas. Rocks est la plus connue des distributions GNU Linux spécifique au cluster. Il connaît un franc succès auprès des superordinateurs puisqu'il a enregistré depuis 5 ans pas moins de 600 d'entre eux sur le site TOP500 répertoriant les 500 plus puissants superordinateurs du monde (2 fois par an). Rocks est principalement conçu pour de gros cluster de calcul (une centaine de machines). Si on possède une petite grappe (une dizaine de noeuds) on peut opter pour CAOS qui comporte le gestionnaire de cluster Warewulf décrit plus loin. C'est une solution dédiée au clustering qui a été réalisée par une équipe d'experts. Ils mettent à jour autant que possible leur OS (dont une version pour l'architecture 64 bits) et la documentation associée.

Il est également possible d'installer un système d'exploitation classique (Debian, Fedora, CentOS, ...) et d'y associer des utilitaires regroupant un ensemble de scripts et de programmes facilitant la gestion d'un cluster. Ils sont soutenus par une grande communauté se chargeant de mettre à jour régulièrement les sources et de corriger les failles de sécurité. De plus on trouve plus facilement des programmes et de la documentation pour les OS populaires.

Ces deux possibilités comportent des utilitaires qui existent depuis très longtemps. Si on retrouve très souvent les mêmes, c'est parce qu'on peut installer chaque outil séparément, sans utiliser d'aide logicielle. C'est la solution qui prend à priori le plus de temps puisqu'il faut tout configurer à la main, mais elle a le mérite de permettre le contrôle le plus fin.

4.2. LES UTILITAIRES D'ADMINISTRATION

Afin de se constituer une grappe de serveurs de calculs, il faut ajouter des fonctionnalités à une distribution GNU Linux classique. En effet, nous avons besoin d'utilitaires de maintenance pour garder un ensemble de noeuds dans un état stable et cohérent. L'intérêt est de dissocier les tâches telles que les mises à jour système du nombre de noeuds en les propageant à partir d'un nœud référent vers l'ensemble du cluster. Par ailleurs, il est souhaitable de renforcer la sécurité et d'assurer une surveillance de l'ensemble des machines car le cluster constitue un système critique.

Le but des différents projets que nous abordons dans cette rubrique est de regrouper des utilitaires spécifiques au cluster et de développer une couche supplémentaire pour faciliter leur utilisation. Cela permet d'automatiser certaines tâches répétitives et fastidieuses. Ainsi le système, une fois mis en place, fournit un environnement adapté au déploiement de calcul à haute performance mais permet également une maintenance aisée de la grappe.

4.2.1 DEBIAN CLUSTER COMPONENTS

DCC est un projet lancé par le centre informatique de l'Institut Ruder Boskovic. Il fait partie d'un projet interne. Il est basé, comme son nom l'indique, sur le système d'exploitation libre Debian. Cette dernière a l'avantage de bénéficier de nombreuses extensions pour combler les trous de sécurité. Beaucoup d'administrateurs utilisent ce système car il facilite grandement la gestion d'un parc informatique. Le système de paquetage logiciel est l'un des plus performants car il possède de nombreux atouts comme la gestion des dépendances entre les paquetages ou bien encore l'apport de méta données extrêmement riche. De plus la Debian bénéficie d'une équipe réactive aux failles de sécurité. Celle-ci est chargée de mettre à disposition la correction d'un trou de sécurité dans les 48 heures. Ainsi, Debian garantit le maintien du système dans un état stable.

Il n'installe pas uniquement les utilitaires ci-dessous puisqu'il fourni également des commandes (scripts) qui propose ainsi une surcouche logicielle. Il automatise ainsi certaines tâches de manière astucieuse.

Voici la liste des outils qui compose DCC :

- SIS : Utilitaire nécessaire pour l'installation d'une image sur une machine distante
- Torque : Répartiteur de charge
- C3 : Utilitaire d'aide à la maintenance d'un cluster
- MPICH : Librairie MPI (programmation parallèle)
- LAM/MPI : Libraire MPI
- NSS/PAM/SLAPD : Serveur LDAP avec des extentions pour stocker les informations importantes (nom de comptes, groupes, …)
- Ganglia : Moniteur de surveillance d'un ensemble de machine
- Shorewall firewall : Pare-feu

4.2.2. OSCAR

OSCAR a été testé sur les distributions Fedora (Core 2 et 3), Mandriva (10.0 et 10.1) et Red Hat Entreprise (3 et 4). Il dispose des mêmes outils que DCC à quelques exceptions près. L'utilisation de Maui *[7]* pour l'ordonnancement des processus rend Torque (répartiteur de charge) plus flexible et améliore ses performances.

On constate qu'au contraire de DCC, OSCAR est déployé sur de nombreuses architectures à travers le monde. Il bénéficie d'une grande popularité et d'une communauté très active. A l'heure actuelle, 82 organismes sont référencés sur le site officiel d'OSCAR. Ce sont pour la plupart des universités ou des écoles. Cela concerne de petites grappes mais également des clusters d'une centaine de machines composées de divers matériels (AMD, Pentium, SMP, etc) et systèmes d'exploitation (RHL9, FC2, RHEL4, etc).

À l'instar de DCC, il installe des scripts permettant la maintenance du cluster. Ceux-ci ont été développés en Perl, et peuvent même recourir à une interface graphique plus « user friendly » qu'un fichier de configuration à éditer.

Ci-dessous, la liste des composants d'OSCAR :

- SIS : Utilitaire nécessaire pour l'installation d'une image sur une machine distante
- Torque : Répartiteur de charge
- Maui : Ordonnanceur évolué
- C3 : Utilitaire d'aide à la maintenance d'un cluster
- LAM/MPI : Librairie MPI (programmation parallèle)
- MPICH : Librairie MPI

- PVM : Bibliothèque de communication un réseau d'ordinateurs
- Ganglia : Moniteur de surveillance d'un ensemble de machines
- Switcher : Paquet permettant de manipuler son environnement
- OPIUM (OSCAR Password Installer and User Management) : Paquet aidant à la synchronisation des comptes
- Pfilter : Pare-feu évolué

4.2.3. WAREWULF

Warewulf a été conçu pour une grappe de serveurs Linux. Il met l'accent sur la facilité de redimensionner et d'administrer un cluster. Il permet la maintenance de noeuds esclaves à partir d'un seul point (appelé maître). Cet outil comprend des utilitaires pour la configuration de fichiers, la surveillance, et le contrôle des noeuds. Il est personnalisable en fonction des besoins pour un type de cluster. Qu'il y ait deux ou deux cents noeuds, Warewulf offre la même simplicité d'administration.

Warewulf se veut simple et flexible. La volonté de cet outil est de faciliter l'administration des noeuds d'un cluster de la même façon qu'elle peut être faite avec un seul ordinateur. Pour cela, il met a disposition les méthodes standards (PXE, DHCP, etc.).

Il utilise un système de fichiers (VNFS) commun à tous les noeuds qui se place dans la RAM, permettant ainsi d'économiser l'achat d'un disque dur. De plus, il n'y a pas d'installation à effectuer sur les noeuds de calcul. Ce qui est placé en mémoire (RAM) prend en général 50 Mo et est transféré par TFTP *[8]* au démarrage de la machine. Il peut exister également des points de montage à la manière des liens NFS provenant d'un disque dur.

Il est indépendant de la distribution bien qu'il ait été créé pour Red Hat. Il peut être également installé sur une Debian car un portage a été réalisé avec succès. La version du noyau GNU Linux n'influe pas sur ses performances.

Le désavantage de cet outil est qu'il ne bénéficie pas d'un soutien massif, il n'y a pas de communauté derrière ce projet mais c'est l'oeuvre d'une seule personne. Il ne cherche pas à regrouper les meilleurs outils disponibles sur le marché actuel mais plutôt à fournir sa propre solution, et c'est ce qui le rend unique.

4.3. HAUTE PERFORMANCE

Ces différents outils d'administration fournissent les librairies nécessaires à la programmation parallèle. Ils se chargent également de la répartition de la charge sur l'ensemble des machines du cluster. Par conséquent, on peut considérer que ce sont eux qui permettent d'optimiser les performances en exploitant au maximum les ressources fournies par le cluster. Nous verrons plus tard, que nous avons choisi de remplacer le répartiteur de charge d'origine par un autre déjà connu et utilisé à l'observatoire.

4.4. HAUTE DISPONIBILITE

Malgré tous les outils d'administrations que j'ai pu découvrir lors de mon étude, aucun ne cherchait à obtenir des performances en terme de disponibilité. Pour bénéficier d'une telle capacité il faut chercher du coté des utilitaires installés sur les serveurs HTTP Internet car ils sont tellement sollicités qu'ils ont besoin souvent eux aussi de plusieurs machines devant apparaître comme une seule.

4.4.1. HEARTBEAT

Heartbeat est issu du projet Linux-HA initié dès 1999 pour toutes les plateformes Linux mais aussi pour FreeBSD et Solaris. Linux-HA constitue un bon point de départ pour celui qui souhaite s'initier à la haute disponibilité car le site propose une foule de documents et d'aide. Il est sans doute l'un des plus vieux projets portant sur la haute disponibilité encore actif. Beaucoup l'on copié mais jamais égalé.

Il est conçu pour réagir rapidement en cas de problème car si une machine tombe en panne, une autre doit être en mesure de prendre la relève et proposer les mêmes services.

Linux-HA propose un mécanisme de prise de pouls afin de vérifier en permanence que les machines sont joignables. En effet, si on a deux machines, l'une va servir de serveur principal et va rendre un service, tandis que l'autre devra régulièrement tenter de joindre la première afin d'être sûr qu'elle n'est pas en panne. Lorsque le serveur principal se retrouve inaccessible, c'est le second qui se charge d'assurer la disponibilité du service. On peut

proposer autant de services que l'on souhaite, car chacun sera réveillé dans un ordre bien précis sur le serveur secondaire.

4.4.2. KIMBERLITE

Kimberlite a été conçu par des développeurs expérimentés dans le domaine du clustering. Il a été créé dans le but de servir en production pour des groupes commerciaux. Il fourni un moyen de garantir l'intégrité des données et propose un mécanisme permettant de détecter la perte d'une machine (*voir figure 2.1*). En utilisant la redondance matérielle, des disques durs partagés, une gestion de l'alimentation, et un mécanisme pour accroître la robustesse des communications du cluster et des pannes logicielles, Kimberlite se place en tête de cortège dans les solutions de Haute disponibilité pour les entreprises commerciales.

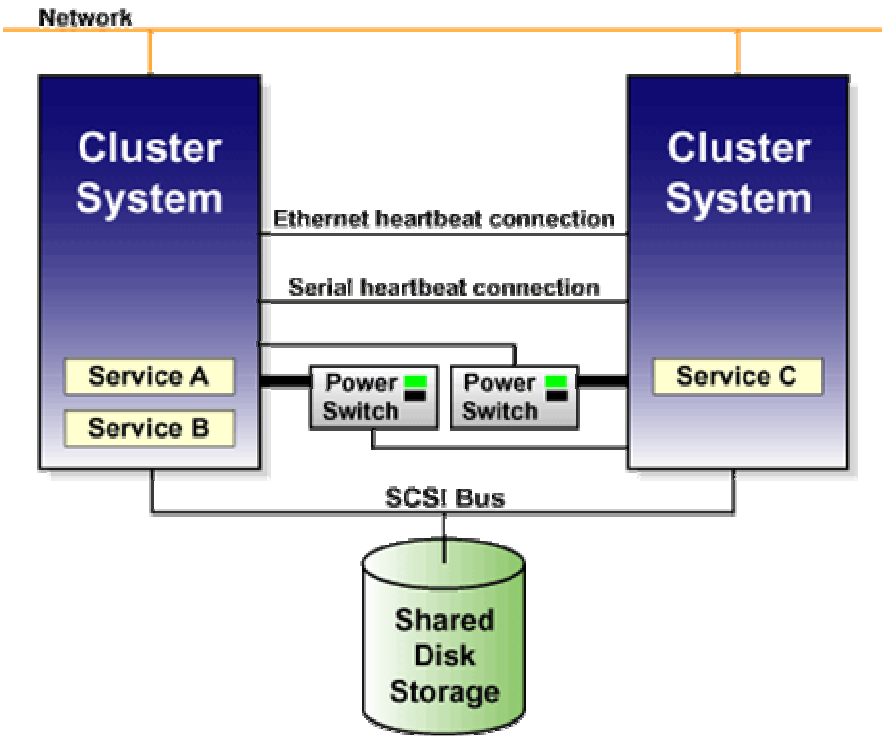


Figure 2.1 : Infrastructure de Kimberlite

Afin d'administrer en temps réel son cluster, Mission Critical Linux propose également Secure Service Technology TM (SST), qui permet aux ingénieurs et autres personnes habilitées de sécuriser le système d'accès, d'effectuer des diagnostiques à distance, et de corriger les problèmes existants. En utilisant SST et l'outil d'analyse de Mission Critical

Linux, cela assure que n'importe quels problèmes sont résolus rapidement et simplement, avec le minimum d'interruption de la grappe.

4.4.3. HA-OSCAR

L'utilitaire d'aide à l'administration de cluster OSCAR dispose de quelques extensions notamment une permettant de rendre le cluster disponible en permanence. HA-OSCAR a été réalisé dans le cadre d'un projet Open Source sous GNU Linux. Il constitue un paquet qui ne peut être installé que grâce à OSCAR. Par le mécanisme de redondance de service, il permet d'éliminer le point faible lié au point d'entrée unique.

Comme pour les précédents utilitaires il comprend un mécanisme de détection de la panne, et de réactivation automatique des services.

4.4.4. LVS

Linux Virtual Server est un vieux projet Open Source initié en mai 1998 par Wensong Zhang. C'est un répartiteur de charge qui propose en plus de la haute disponibilité pour la technologie de clustering. Il constitue donc une autre solution pour la haute disponibilité du cluster mais différente des précédentes, nous verrons par la suite qu'il n'est pas un concurrent direct de Heartbeat ou Kimberlite. Il est principalement utilisé par les serveurs Web pour assurer la disponibilité de leurs services. Ces derniers peuvent être assez variés puisqu'ils comprennent notamment les Web services, la messagerie électronique et la voix sur IP.

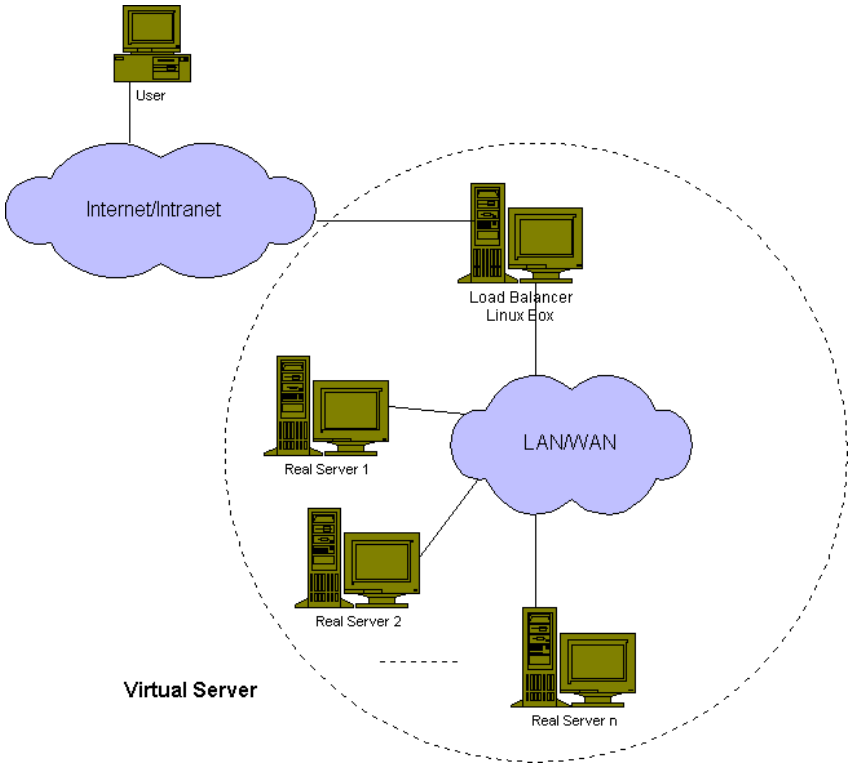


Figure 2.2 : Mécanisme du serveur virtuel de LVS

Il est constitué de deux composants que sont IPVS et KTCPVS. Le premier fait de la répartition de charge au niveau IP tandis que l'autre s'occupe de la couche applicative sur un réseau. Ils sont intégrés au noyau GNU Linux. Il y a un serveur principal qui constitue l'entrée du cluster et qui distribue les requêtes émises de l'extérieur sur ces multiples nœuds (serveurs réels). On peut enlever ou ajouter un nœud sans que cela perturbe le système. Plus il y a de nœuds, plus on est sûr de pouvoir répondre aux nombreuses demandes, c'est pour cela qu'on parle de haute disponibilité.

L'architecture de LVS (*voir figure 2.2*) implique qu'il y ait un point critique ou point faible (en anglais « single point of failure ») qui est le serveur qui sert à la répartition de la charge. Si il tombe en panne, le système ne peut plus fonctionner. Pour pallier au problème, il suffit de dupliquer la machine critique à l'aide d'un mécanisme tel que Heartbeat ou Kimberlite (*voir figure 2.3*) cité précédemment.

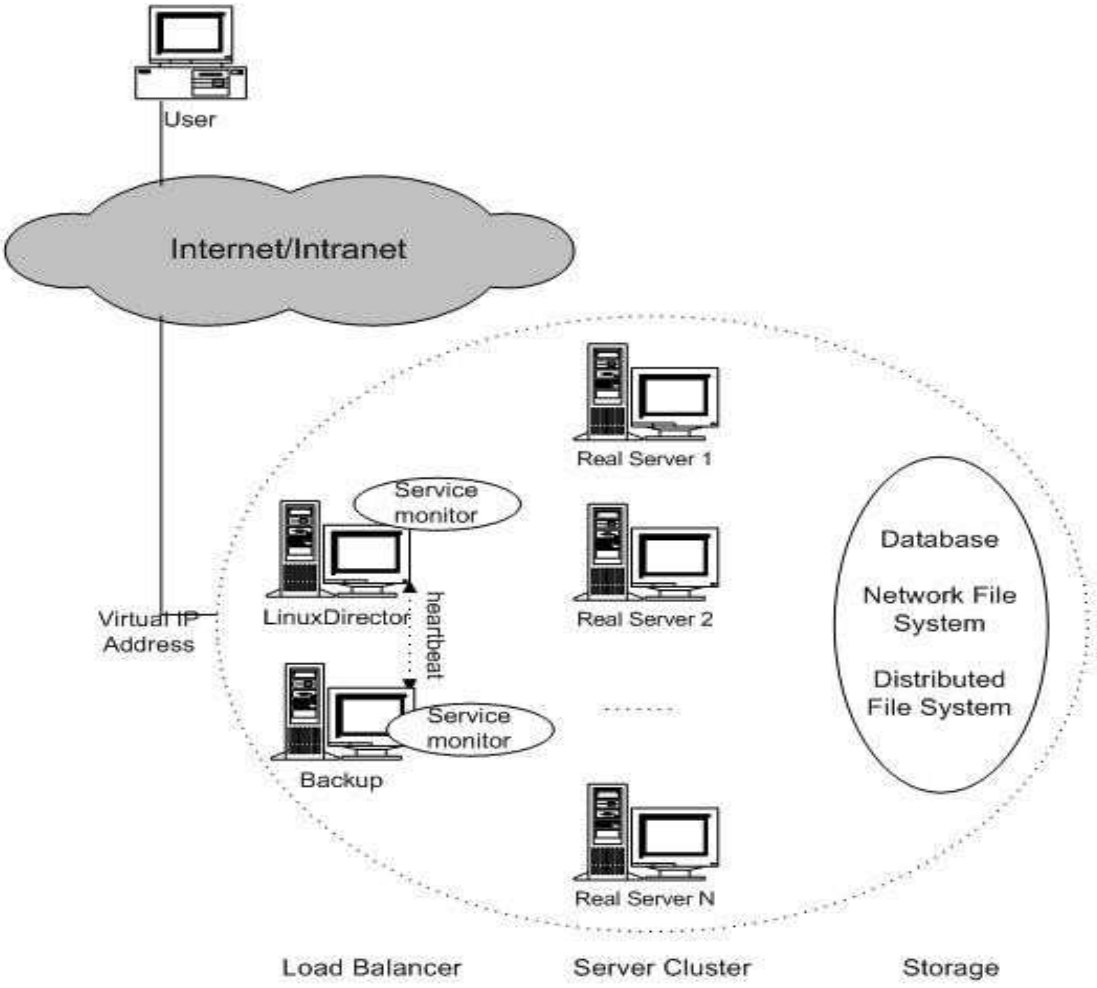


Figure 2.3 : Combinaison LVS - Heartbeat

5. ANALYSE COMPARATIVE DES UTILITAIRES

Au final, nous avons éliminé l'idée d'utiliser une distribution trop spécifique au clustering afin d'opter pour l'installation d'un système d'exploitation populaire et l'ajout d'un utilitaire d'aide à l'administration. Le choix de la distribution va dépendre de l'utilitaire installé, c'est pourquoi nous allons comparer les différentes solutions retenues.

5.1. CHOIX DU GESTIONNAIRE D'ADMINISTRATION

Différents critères sont à notre disposition pour évaluer et comparer les trois gestionnaires cités précédemment que sont DCC, OSCAR et Warewulf. Nous allons les confronter afin de faire ressortir le choix final qui s'est porté sur *Debian Cluster Componant*.

Il est à noter que la liste des critères n'est en aucun cas exhaustive. Elle comporte des points liés aux matériels dont nous disposons (bi-processeur 64 bits, boot PXE *[9]*). Il faut également tenir compte de nos objectifs (maintenance aisée, cluster sécurisé, haute disponibilité, performance de calcul). Certains items de cette liste donnent lieu à une notation délivrée par moi-même. J'ai voulu mettre en évidence que le choix doit être effectué en tenant compte de la facilité d'installation et de maintenance.

Tout au long de ce chapitre, nous détaillerons l'intérêt de chaque critère avant de donner le résultat des tests pour chacune des trois solutions retenues.

5.1.1. LE MATERIEL

Le matériel mis à disposition est à l'origine des premiers problèmes rencontrés car il existe encore peu de projet ayant testé leurs programmes sur une plateforme 64 bits. L'émergence de cette architecture 64 bits commencent à être prise en compte par les distributions Gnu Linux avec des versions qui font preuve de stabilité. Cependant, du coté des logiciels, ils en existent encore de nombreux qui n'ont pas franchis le cap du 64 bits. Autrement dit, au mieux nous réussirons à exécuter le programme mais il utilisera les librairies 32 bits.

DCC comporte des logiciels portés sur la version amd64 de la distribution Debian et des scripts évolués écrits en Perl ou Shell. Cependant, les versions des programmes disponibles avec la Sarge amd64 ne sont pas compatibles, c'est pourquoi j'ai dû installer d'autres versions trouvées sur des sites non-officiel. En ce qui concerne les scripts, ils ne posent aucun problème en utilisant les librairies 64 bits de la machine.

OSCAR quant à lui, propose ses sources sous forme de paquet RPM testé sur les distributions Fedora (Core 2 et 3), Mandriva (10.0 et 10.1) et Red Hat Enterprise Linux (3 et 4). Le souci est que la dernière version stable n'est pas compatible avec l'architecture 64 bits. Il est nécessaire d'installer la version bêta pour faire fonctionner OSCAR sur le cluster. J'ai remarqué néanmoins des anomalies lors de l'installation de nœuds. N'ayant pu tester que la version bêta, je ne sais pas si ces problèmes seront encore présents dans la prochaine mouture qui devrait prévoir une installation pour une machine 64 bits.

J'ai pu tester Warewulf sur la Fedora Core 3 (version x86_64) après avoir recompilé les sources du paquet RPM. Il faut passer un peu de temps pour créer le script permettant de construire l'image des nœuds. Mis à part ce petit inconvénient, l'installation prévus pour l'architecture 64 bits fonctionne convenablement.

Il est à noter que l'installation des machines par le réseau via le Boot PXE *[9]* est disponible avec toutes ces solutions. En effet, elles utilisent un mécanisme permettant d'installer une machine dès le démarrage de celle-ci sans aucune aide extérieure et sans qu'il n'y ait au préalable de système d'exploitation sur ce nœud.

5.1.2. FACILITE DE MAINTENANCE

Afin de garantir une simplicité dans la maintenance du cluster, j'ai tenté de vérifier s'il était possible d'effectuer certaines tâches dont nous aurons besoin ultérieurement après l'installation complète. Il faut tout d'abord qu'il y ait la possibilité d'installer l'ensemble des nœuds de manière identique. Puis éventuellement, il peut être nécessaire d'installer des nœuds composés d'éléments différents (matériel et logiciel). Ensuite, nous devons être en mesure d'installer à tout moment un programme soit sur l'ensemble des nœuds du cluster, sur un seul nœud ou sur un ensemble de noeuds. Il y a également des actions facultatives que nous pourrions utiliser comme l'arrêt et le démarrage de l'ensemble du cluster ou d'une partie simplement. Enfin, en cas de panne d'une des machines, il faut remédier rapidement au

problème, et quoi de mieux que d'en être averti par un système de surveillance de toute la grappe.

DCC et OSCAR étant très similaires au niveau logiciel, leurs avantages sont pratiquement les mêmes. Ils sont tous deux capables de répondre à l'ensemble des points qui ont été abordés au paragraphe précédent puisqu'ils exploitent les capacités de la suite SIS *[10]* combiné à l'outil C3 *[11]*.

Concernant Warewulf, il faut admettre que du point de vue maintenance il est assez en retard par rapport aux autres solutions. Lors de mes tests, je n'ai pas trouvé le moyen d'installer un logiciel sur un nœud en particulier ni la façon d'arrêter la totalité des machines de la grappe.

5.1.3. SECURITE

Warewulf, contrairement à ses deux autres concurrents, ne met pas en œuvre de dispositif de sécurité pour donner des garanties sur l'intégrité du cluster. Il n'a pas été réalisé dans cette optique, c'est pourquoi il faut ajouter les règles de filtrage soi-même avec les outils dont dispose la distribution hôte.

DCC prévoit non seulement le filtrage classique sous Linux mais également la possibilité aux machines du réseau privé de communiquer avec l'extérieur à l'aide la technologie NAT. Il utilise l'utilitaire *Shorewall* pour ajouter les règles. Ce dernier configure Netfilter *[12]* à l'aide de l'outil Iptables via des fichiers de configuration simplifiant ainsi la configuration du pare-feu Linux. A l'instar de DCC, OSCAR propose le même service via cette fois-ci Pfilter. Il se configure aussi simplement que *Shorewall* permettant ainsi un contrôle efficace.

5.1.4. DISPONIBILITE

Aucune des solutions testées ne mettent en œuvre de quoi rendre le cluster accessible en permanence. En cas de panne du nœud principale, tous les services ne sont plus disponibles et nous n'avons aucun moyen de réactiver la tête du cluster à distance. Par contre en cas de crash d'une des machines de travail, nous serons rapidement informé grâce au moniteur *Ganglia* disponible dans DCC et OSCAR. Warewulf comporte également une interface graphique et

des commandes permettant de surveiller l'état des nœuds de la grappe. Il y a des indicateurs de la charge CPU, de l'espace disque utilisé et libre, ainsi que d'autres items éléments utiles.

5.1.5. PERFORMANCE

Puisque nous disposons d'un ensemble de machines, nous avons la possibilité d'exécuter des programmes simultanément sur différents noeuds. Pour cela, il existe des librairies qui permettent la programmation parallèle comme LAM/MPI, MPICH et PVM *[4]*. Cela permet d'accroître la vitesse de traitement et les performances du même coup. OSCAR comprend l'ensemble de ces bibliothèques laissant ainsi le choix au programmeur. Par contre, DCC n'installe pas PVM *[4]* mais les deux librairies MPI *[3]*. Quant à Warewulf, il ne comporte aucun outil destiné à la programmation parallèle mais propose tout de même une page Internet décrivant comment installer MPICH sur le cluster.

Le gain de performance peut provenir de la répartition des traitements sur le cluster. Pour cela DCC fait confiance à Torque qui est une surcouche d'OpenPBS. Il se charge de répartir la charge sur l'ensemble des nœuds de manière à ne pas encombrer une machine en particulier. Ainsi l'ensemble de la grappe dispose d'une puissance de calcul qui doit être exploitée à son maximum. C'est pourquoi OSCAR possède parmi ses rangs, en plus de Torque, Maui *[7]* qui est un ordonnanceur évolué. Par contre, Warewulf ne dispose pas de ce genre d'outil pour augmenter les performances.

5.2. CONCLUSION

Warewulf est celui qui satisfait le moins à nos attentes car il ne comporte que le moyen d'installer et de surveiller une grappe de machines. De plus, c'est une solution qui offre peu de documentation et est soutenu que par une très petite communauté.

OSCAR semble être une solution sérieuse car il possède la plupart des outils dont nous avons besoin. Il reste simple et pratique, permettant à n'importe qui d'administrer un cluster de machines à l'aide d'une interface graphique. De nombreux organismes se font référencer sur le site Internet d'OSCAR (107 clusters enregistrés) montrant ainsi tout l'engouement porté à cette utilitaire. Cependant, lors des tests, j'ai relevé quelques problèmes gênants comme

l'arrêt du processus d'installation. Cela peut être dû à la version bêta qui était la seule à vouloir s'installer sur les machines.

DCC est selon moi l'utilitaire le plus fiable que j'ai eu à tester. Il nous permet d'atteindre une partie des objectifs que nous nous sommes fixés. Il est fiable, flexible et assez bien documenté, ce qui le place devant OSCAR. Avec cette solution, nous avons une aide à l'administration du cluster que nous pouvons personnaliser comme nous le voulons. Tout ce qui nous manque c'est un système résistant aux défaillances du nœud principal.

6. MISE EN PLACE DU CLUSTER

Afin d'administrer le cluster de machines, nous avons finalement installé l'utilitaire DCC. Pour cela, un système d'exploitation basé sous Debian GNU Linux est nécessaire, c'est pourquoi nous avons choisi la version appelée Sarge (3.1). Il existe un CD d'installation de celle-ci pour l'architecture 64 bits sur un site non officiel en attendant la version de l'équipe Debian. Le portage AMD64 est entièrement en 64 bits mais permet également l'exécution de binaire 32 bits moyennant l'ajout du paquet ia32-libs contenant les bibliothèques.

6.1. GESTIONNAIRE D'ADMINISTRATION

DCC a été choisi pour permettre la gestion du parc des 8 machines réservées aux traitements des calculs. Il s'installe via un ensemble de paquets Debian contenant soit des scripts d'aides à l'administration soit des liens servant à installer les utilitaires qui composent DCC (SystemImager, C3, Shorewall, …).

6.1.1 LES PAQUETS INSTALLES

Certains paquets ont été modifiés afin de les adapter à nos besoins. Pour cela, j'ai dû récupérer les sources puis apporter les modifications nécessaires pour finalement le reconstruire. Voici, la liste complète des paquets DCC :

- ***debconf-dcc*** : Ce paquet doit être installé avant tous les autres car il a pour rôle l'édition des fichiers de configuration contenant des informations très importantes. L'un de ces fichiers contient notamment l'adresse IP et MAC de la machine locale (le serveur), les préfixes et suffixes donnés au nom des machines et d'autres informations nécessaire lors de l'installation d'un nouveau nœud.
- ***dcc-front*** : Le paquet ***dcc-front*** est chargé d'installer tous les utilitaires que propose DCC sur le point d'entrée du cluster qui se trouve être la machine sur laquelle s'effectue l'administration. Étant donnée que l'utilisation du répartiteur de charge Ali (conçu au sein même de l'observatoire) fonctionne en continu depuis

trois ans et rempli convenablement ses fonctions, l'observatoire a décidé de le conserver et de l'introduire dans la liste des programmes à installer. Par conséquent, nous n'avons pas besoin de TORQUE qui remplissait les mêmes fonctions pour DCC. J'ai pris l'initiative de l'enlever des paquets ***dcc-front*** et ***dcc-node***.

- ***dcc-node*** : Le paquet ***dcc-node*** est réservé aux machines qui sont installées en tant que noeud de travail. Elles doivent disposer de certains utilitaires afin de nous permettre de les administrer à partir de la tête du cluster.
- ***dcc-utils*** : Le paquet a subi quelques petits changements permettant d'utiliser la nouvelle version de *SystemImager* car celle disponible comporte des bugs dus à l'architecture 64 bits. Puisqu'il contient les scripts d'administration il a fallût les modifier afin notamment d'enlever le lien vers *qmgr* (la commande de lancement des requêtes de Torque). De plus, j'y ai ajouté des modifications afin de résoudre un problème rencontré lors des tests, concernant le matériel (copie du fichier « modules » du répertoire « /etc » lors de la création d'une image système afin de pallier les problèmes dus au manque du pilote de la carte SCSI).
- ***shorewall-dcc*** : Ce paquet est présent pour garantir la sécurité du cluster en installant ***shorewall***, le pare-feu GNU Linux. Ce dernier permet une configuration aisée et fine des ports et des règles NAT *[13]*. ***Shorewall-dcc*** comporte un script qui est exécuté après son installation afin d'obtenir une configuration optimale du pare-feu. J'ai alors ajouté la règle permettant d'ouvrir le port SSH *[14]* afin de pouvoir accéder à la tête du cluster depuis l'extérieur.

6.1.2 PRINCIPE DE FONCTIONNEMENT

L'installation d'un cluster à l'aide de DCC commence par l'ajout des utilitaires SystemImager Suite et C3 *[11]*. À eux deux, ils permettent l'ajout d'un nœud sur la grappe existante. DCC nous permet d'accélérer le processus à l'aide de scripts Perl ou Shell tandis que SIS *[10]* et C3 *[11]* se chargent de l'installation en elle-même. Cela nécessite la présence d'une carte réseau compatible PXE *[9]* et un serveur TFTP *[8]* sur le nœud principal.

SystemImager Suite est employé pour l'installation d'une distribution GNU Linux lorsqu'on dispose de nombreux ordinateurs. L'environnement peut être extrêmement varié, car

il est utilisable pour des machines bien spécifiques comportant des bases de données, des services Internet ou des traitements complexes à exécuter, mais également de simples ordinateurs de bureau.

Le principe est relativement simple puisqu'il est basé, comme son nom l'indique, sur un système d'image. Il suffit d'être en possession de l'image d'un système d'exploitation GNU Linux personnalisé (nommé « golden client ») qu'on souhaite installer sur un poste client appelé communément nœud client. Puis SIS *[10]* installe automatiquement, sur autant de nœuds qu'on souhaite, l'image de référence.

C'est lors de cette première phase que DCC intervient pour accélérer le processus. L'un de ces scripts permets de créer l'image d'un système d'exploitation de notre choix sur le nœud maître à partir duquel elle sera envoyée sur les nœuds clients. Si on utilisait uniquement SIS *[10]*, il faudrait installer entièrement une distribution sur l'un des nœuds client puis la télécharger sur le nœud maître pour qu'elle serve d'image de référence lors de l'installation des autres nœuds du cluster. On économise ainsi du temps grâce à DCC dès cette première phase.

SystemImager (composant essentiel de la suite SIS *[10]*) utilise System Configurator qui lui permet d'être compatible avec la plupart des distributions GNU Linux. Ce dernier permet la configuration du réseau, du chargeur de démarrage et des périphériques matériels d'un ordinateur à l'aide d'une unique API (interface de programmation). System Configurator reconnaît la distribution installée (grâce à une technique appelée « footprints ») et associe alors cette machine à un type de configuration. Ainsi, un grand nombre de systèmes d'exploitations sont reconnus afin de permettre une compatibilité assez large.

L'image de référence (ou « golden client ») est un système d'exploitation personnalisé dont on peut à tout moment modifier les fichiers de configuration, installer ou enlever un programme et même changer le noyau Gnu Linux présent initialement. À l'aide d'une seule commande, la nouvelle image est déployée sur tous les nœuds concernés et afin de rendre le système le plus efficace possible, les mises à jour sont effectuées suite à la synchronisation de l'image de référence avec celle présente sur la machine cible via la commande *rsync*, permettant ainsi de transférer uniquement les modifications apportées.

6.3. PERFORMANCE

Nous avons choisi d'ajouter tous les utilitaires dont nous avons besoin sous forme de paquet Debian afin de garder la même simplicité d'installation que DCC. En effet, nous souhaitons garder le répartiteur de charge Ali au sein du nouveau cluster, ce qui a donné lieu à la création du paquet ***ali-dcc***.

Le paquet ***ali-dcc*** est comme son nom l'indique, celui qui installe le répartiteur de charge Ali et le configure de manière optimale. Le paquet originel est un tarball qui n'est absolument pas adapté au passage vers un paquet Debian, m'obligeant alors à modifier la procédure d'installation d'Ali.

Le paquet Ali ne propose pas de script pour l'initialisation du service au démarrage de la machine pour la distribution Debian mais uniquement Mandrake et Suse. Le problème est que cela ne fonctionne pas de la même manière selon l'OS où l'on se trouve. J'ai donc créé mes propres scripts de démarrage pour Ali afin de pallier à ce manque et de pouvoir les installer grâce au paquet Debian créé.

Le programme Ali, développé sur une architecture 32bits, n'affiche pas d'erreur à la compilation sur nos machines 64 bits mais les commandes générées ne fonctionnent pas correctement. Deux problèmes sont apparus, tout d'abord il y a une incompatibilité dans les messages envoyés car leurs structures sont basées sur les tailles des types de variables (entier, long, flottant, ...) qui diffère selon l'architecture locale (32 ou 64 bits). De plus, le programme 32 bits ne se lance pas au sein d'un environnement 64 bits. C'est pourquoi deux solutions s'offrent à nous, soit nous gardons le programme tel quel en le faisant s'exécuter avec les librairies 32 bits, soit nous modifions le code d'Ali afin qu'il fonctionne correctement sur nos ordinateurs. La première solution a été préférée car c'est celle qui semble la plus rapide et la plus fiable.

6.4. DISPONIBILITE

Le CDS (Centre de Données astronomique de Strasbourg) propose des services accessibles à la communauté internationale via leur portail Internet. De nombreux chercheurs comptent sur cette aide pour avancer dans leur projet. Un arrêt, même de quelques heures,

ralentirait potentiellement beaucoup de personne dans leur recherche, c'est pourquoi il faut surveiller et maintenir ces services en permanence. Nous devons donc trouver le moyen de rendre le cluster actif à tout moment. Le projet Linux-HA, qui existe depuis 1998, propose une solution appelée Heartbeat que nous avons donc testée puis mise en place sur le cluster.

6.4.1. PRINCIPE HEARTBEAT

Heartbeat permet d'activer ou de désactiver des services sur un ordinateur. Dans le cas où le noeud maître principal est actif tous les services tournent sur celui-ci. Puis dès qu'il tombe en panne, ces mêmes services sont démarrés sur la machine secondaire. Heartbeat possède la notion de ressource qui peut correspondre à une adresse IP, un serveur HTTP ou d'autres services plus complexes.

Lorsqu'on a deux machines qui doivent proposer le même service il est nécessaire que ce soit transparent et pour cela il faut commencer par obtenir une adresse IP commune aux deux. Puis comme nous souhaitons utiliser un serveur DHCP et HTTP, ils doivent être installés sur les deux noeuds de manière similaire. Et ils seront actifs ou non selon l'état des serveurs.

Afin de déterminer à quel moment activer les services sur le serveur secondaire, Heartbeat émet des paquets UDP en broadcast (voir en *annexe 2*) sur les deux sorties réseaux disponibles et éventuellement sur une connexion série. Lorsqu'il ne perçoit plus l'autre machine (la principale) sur aucune des interfaces, il décide de lancer ses propres services. Lors du lancement du noeud maître principal, deux solutions s'offrent à nous, soit rien ne change et c'est uniquement lorsqu'on redémarre le service Heartbeat que le noeud secondaire se met en mode passif tandis que l'autre relance les services, soit Heartbeat force l'activation des services sur l'hôte principal.

6.4.2. CREATION DU PAQUET HEARTBEAT-DCC

Ce paquet a été réalisé pour l'installation de Heartbeat (projet Linux-HA) et pour sa configuration au sein du cluster. Il doit donc être ajouté sur les serveurs primaire et secondaire. J'ai construit un script BASH permettant de constituer les fichiers de configuration de Linux-HA à travers une petite interface textuelle. Il est appelé après

l'installation de ce paquet et est disponible à tout instant dans le répertoire « /usr/bin ». Il contient des valeurs par défaut qui s'appliquent tout particulièrement à notre situation ce qui permet de ne rien faire lors de sa première exécution. S'il constate que des fichiers ont déjà été créés, il change ses valeurs par défaut et les affiche comme exemple. De cette manière si la valeur par défaut nous semble correcte, il nous faut simplement valider la ligne.

DCC utilise le pare-feu *shorewall* pour restreindre les connexions uniquement sur les ports connus. Au départ, ils se limitaient aux ports SSH *[14]* et HTTP (après la modification dans le paquet ***shorewall-dcc***, voir chapitre *6.1.1 Les Paquets installés*). À cause du principe utilisé par Linux-HA qui consiste à envoyer des messages dont il attend une réponse pour déterminer si l'autre serveur est actif, il m'a fallut ajouter l'ouverture des ports 8 (ICMP) pour les ping et 694 (UDP) pour la prise de pouls.

7. INTEGRATION DES SERVICES

7.1. ALI

Le rôle principal du serveur Aladin est de reconstruire à la volée des images pour des régions du ciel, processus qui suit les étapes suivantes :

- analyse de la requête,
- récupération des images recouvrant la région demandée,
- décompression éventuelle de ces images,
- extraction des portions concernées,
- prétraitements éventuels (rééchantillonnage, ajustement de la dynamique, etc.)
- concaténation des différentes portions pour créer l'image finale,
- post-traitements éventuels (déconvolution, extraction d'objets, combinaison avec d'autres images),
- codage de l'image finale dans le format requis,
- compression de l'image finale si nécessaire.

Avec le nombre croissant de requêtes qu'Aladin doit traiter quotidiennement, la machine serveur se trouve souvent surchargée, et les utilisateurs sont alors confrontés à des temps d' attente qui peuvent être parfois relativement longs (selon les traitements demandés). L'équipe Aladin a donc décidé d'améliorer les performances du serveur en installant un cluster de machines. C'est dans se cadre qu'Ali a été créé afin de répartir la charge en dispatchant les requêtes sur les nœuds de travail (*voir la figure 3.1.1*).

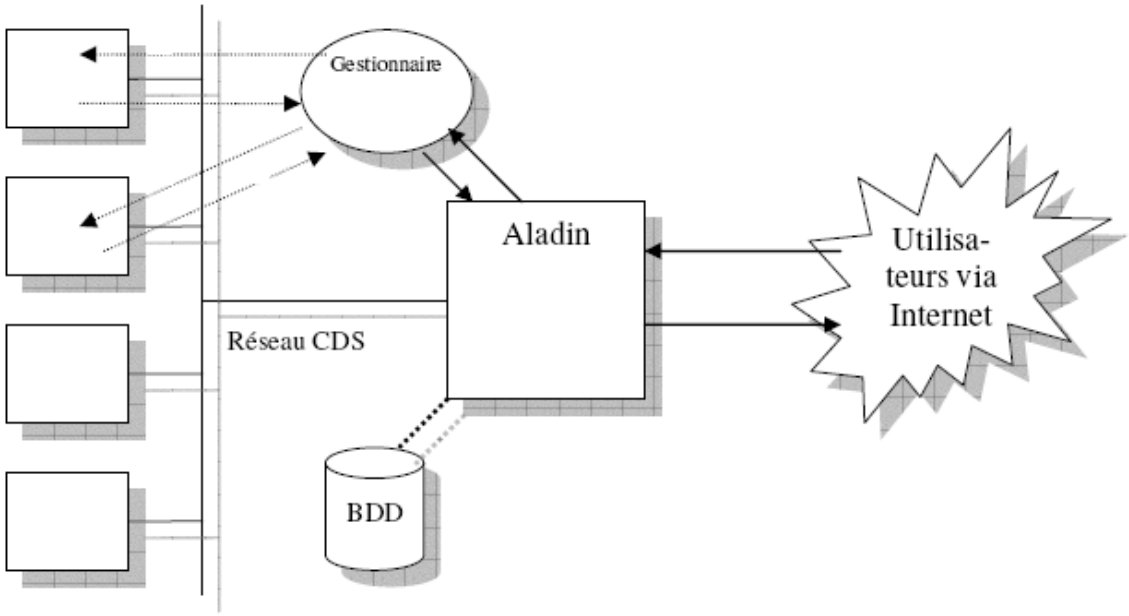


Figure 3.1.1 : Schéma global de l'interaction Aladin - Gestionnaire Ali

Typiquement, une tâche pouvant être accélérée via un cluster est le mosaïquage. Le ciel est découpé en une multitude d'images ; ce sont ces images que contient la base de données d'Aladin. Lorsqu'un utilisateur demande une région du ciel autour d'un point précis, alors le serveur récupère les quatre images qui « entourent » ce point. Chaque image sera décompressée individuellement (quatre traitements pouvant donc être faits en parallèle), puis elles seront « ressoudées » et l'image résultante sera de nouveau compressée. Dans ce cas, on tire parti du fait que quatre traitements peuvent être faits en parallèle, sur quatre machines différentes, alors que sur Aladin ils seront soit séquentiels, soit concurrents en terme de ressources.

Le répartiteur de charge Ali est adapté à nos besoins car il est extrêmement réactif (bonne *overhead*) comparé à d'autres répartiteurs plus connus comme par exemple OpenPBS *[15]*. En effet, ce dernier est un gestionnaire de batch (traitement par lot ou différé), et n'assure aucunement les traitements instantanés nécessaire à l'interactivité du service Aladin. De plus, la solution d'environnement partagé répartissant la charge sur un ensemble de machines comme OpenSSI *[16]*, ou OpenMosix *[17]* demande également beaucoup de temps avant la migration effective du traitement. C'est pour ces raisons que nous avons décidé d'introduire Ali sur le nouveau cluster.

7.1.1 PRINCIPE

Ali est composé de trois éléments localisés sur le cluster selon la nature du nœud. Le nœud serveur est celui qui reçoit la requête provenant de l'extérieur du cluster. Il est chargé d'analyser celle-ci, puis d'organiser dans le bon ordre les traitements et sous traitements de la requête. C'est le travail du serveur Ali (*voir figure 3.1.2*), appelé communément Sali. Il est situé sur la tête du cluster en compagnie du moniteur Ali, appelé Mali. Ce dernier doit être en mesure de connaître l'ensemble des nœuds sur lesquels il y a un Wali ou « worker Ali ». Il répond également aux demandes venues du serveur pour lui indiquer sur quelles machines il peut envoyer le traitement.

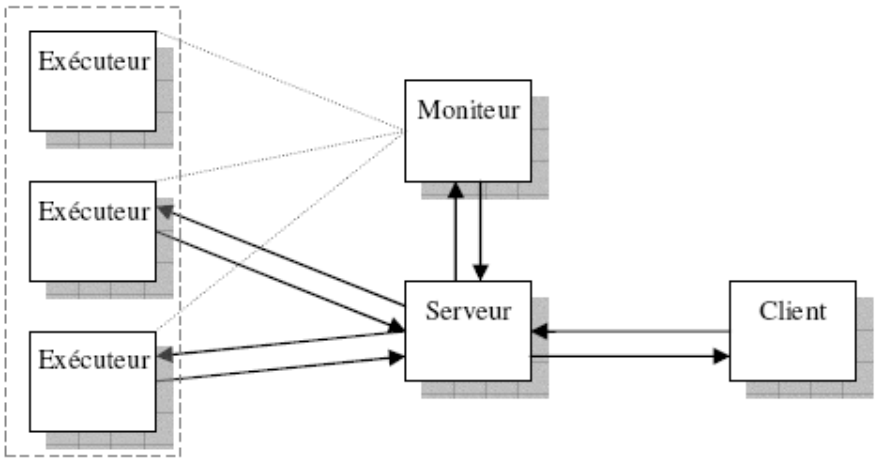


Figure 3.1.2 : Les Composants d'Ali

Le moniteur connaît uniquement les items que proposent les nœuds de travail. Ces items représentent un traitement qui peut être exécuté sur la machine qui le détient. Pour connaître, l'ensemble des items, le moniteur émet une requête vers les « worker Ali » se trouvant sur les nœuds. Ensuite, le serveur questionne le moniteur pour avoir un premier ensemble de machines à qui il peut demander l'exécution d'un calcul, puis il fait son choix en fonction de bien d'autres paramètres comme la charge CPU.

7.1.2 INSTALLATION SUR LE CLUSTER

Après la mise en place de l'utilitaire Ali (ali-dcc), il faut le configurer afin de pouvoir profiter pleinement de ses compétences en matière de répartition de charge. La configuration est assez simple car elle est faite à travers un ensemble de fichiers.

Afin de vérifier le bon fonctionnement d'Ali, il est possible d'utiliser l'interface Web (*voir figure 3.1.3*) qui a été installée sur une des machines du CDS. Elle dialogue avec le moniteur Ali situé sur une machine donnée. Nous pouvons y voir des statistiques sur la charge de chacune des machines, ainsi que les items qu'elles possèdent.

Workers monitored by *cocat1*

Name	Load	Cpuscore	Cpumoy	Cpumin	Cpumax	Netscore	CPU	Suitable	Running	Availlable items
cocat2.u-strasbg.fr		15	14	14	1050226	2633	1	3345491	4	apm cocat2 denis denis2 find2m find2psc3 gsc gsc1.1 gsc1.2 gsc1.3 gsc2.2 pmm1 pmm2 sdss3 u-strasbg:cocat2 ucac1 ucac2 usnob1
cocat3.u-strasbg.fr		27	26	16	540549	2604	1	355194	0	cocat3 denis2 denis3 find2m gsc1.3 gsc2.2 pmm2 sdss3 u-strasbg:cocat3 ucac2 usnob1
cocat4.u-strasbg.fr		518	15	15	1026668	2689	1	1067082	0	cocat4 denis3 find2m find2psc3 gsc1.1 gsc1.2 gsc1.3 gsc2.2 pmm1 pmm2 u-strasbg:cocat4 ucac1 ucac2 usnob1
cocat5.u-strasbg.fr		16	15	7	958242	2673	1	1239914	0	cocat5 denis2 denis3 find2m gsc1.1 gsc1.2 gsc1.3 gsc2.2 pmm1 pmm2 u-strasbg:cocat5 ucac1 ucac2 usnob1
wali2.u-strasbg.fr		16	14	14	197938	2441	1	824648	23	ali-rgb deconvol find2m u-strasbg:wali2 wali2
wali3.u-strasbg.fr		22	21	21	532386	2402	1	17318	0	sdss3 u-strasbg:wali3 wali3
wali4.u-strasbg.fr		13	13	13	66	2861	4	3887	0	ali-rgb sextractor u-strasbg:wali4 wali4
wali5.u-strasbg.fr		23	18	2	97892	2477	4	101477	0	ali-rgb gsc2.2 nomad1 sextractor u-strasbg:wali5 usnob1 wali5

Figure 3.1.3 : Interface Web pour le moniteur Ali

7.2. ALADIN

Aladin est un service du CDS, qui peut être appelé à l'aide d'une archive Java (appel soit en ligne de commande, soit par l'interface graphique). Il existe une technique pour fusionner 3 images de la même partie du ciel, mais à des longueurs d'onde différentes, en une seule image colorée (*voir figure 3.2.1*). Cette colorisation permet, d'une part d'avoir des images visuellement plus belles (destinées au grand public), mais aussi de permettre aux astronomes de visualiser simplement les différences entres les longueurs d'ondes (chacune représentée par une composante RGB). Dans l'exemple ci-dessous, on peut aisément voir que la Nébuleuse du Crabe est dominée par des gaz rayonnants dans le rouge (émission de rayons infrarouge), rayonne un peu moins dans le bleu et beaucoup moins dans le vert (Rayons X).

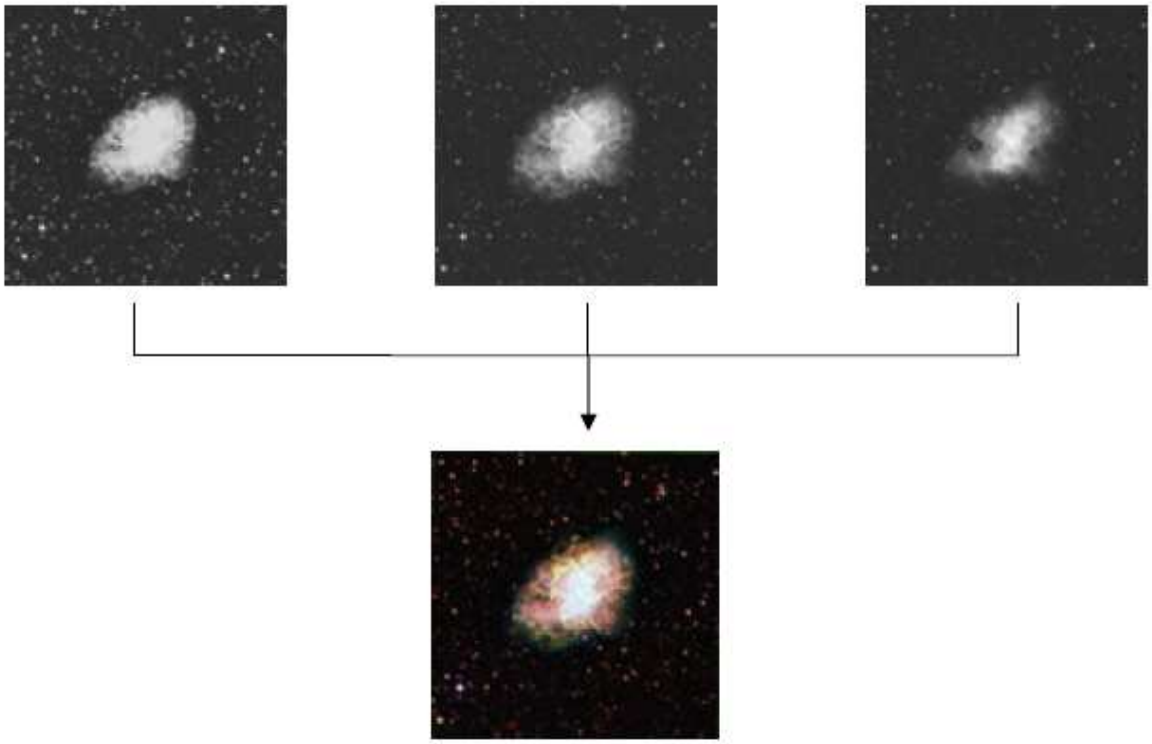


Figure 3.2.1 : Composition RGB

Concrètement pour permettre l'utilisation de cette outil, il faut mettre les fichiers Aladin.jar et ali-rgb dans le répertoire « /root/bin/aladin » des nœuds de travail. Ali le repère automatiquement et crée l'item correspondant. Par la suite, il suffit d'installer le programme *Xvfb* pour émuler un serveur X car les nœuds de travail ne comportent pas de gestionnaire graphique. *Xvfb* nécessite l'ajout d'un script dans les répertoires adéquates afin qu'il s'exécute au démarrage de la machine (voir *annexe 3*). Il est également nécessaire d'ajouter l'utilitaire *cjpeg* (compression en JPEG) pour que tout cela fonctionne.

Il existe un autre outil utilisé régulièrement par les chercheurs astronomes de l'observatoire, nécessitant une image FITS en entrée et qui génère un catalogue d'objet astronomique. Sur l'image suivante, nous retrouvons un objet (voir *figure 3.2.2*) grâce à aux coordonnées indiquées dans le fichier résultant (voir *figure 3.2.3*).

Nous avons remarqué un gain de temps considérable lors de l'exécution de ces deux outils sur le nouveau cluster. Nous estimons à 30% le temps de gagné par rapport à l'ancienne grappe de machines. Cela correspond à nos espérances car nous recherchons en priorité le

maintiens des performances même avec des charges élevées (centaines ou milliers de requêtes simultanées).

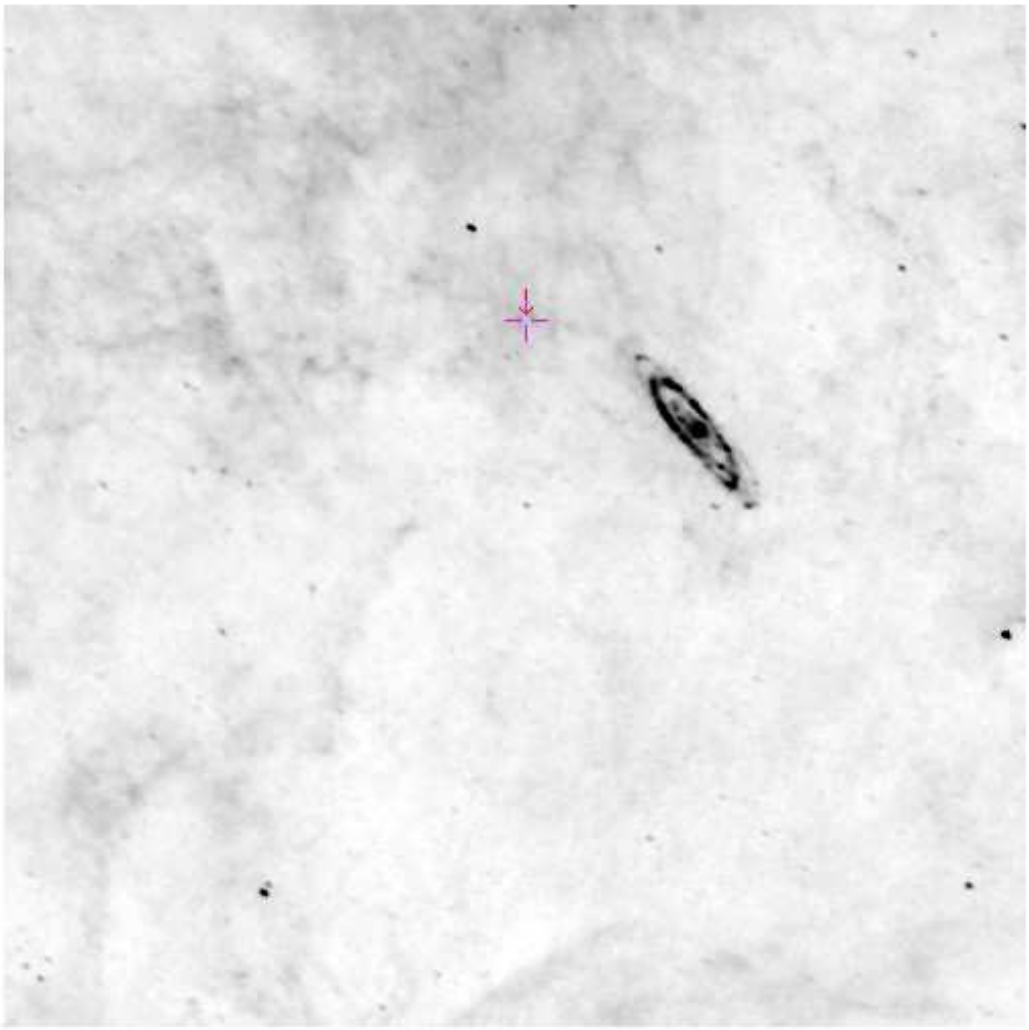


Figure 3.2.2 : Localisation d'un objet grâce à ces coordonnées

```
#   1 NUMBER          Running object number
#   2 MAG_ISO         Isophotal magnitude                          [mag]
#   3 MAGERR_ISO      RMS error for isophotal magnitude            [mag]
#   4 X_IMAGE         Object position along x                      [pixel]
#   5 Y_IMAGE         Object position along y                      [pixel]
#   6 ALPHA_J2000     Right ascension of barycenter (J2000)        [deg]
#   7 DELTA_J2000     Declination of barycenter (J2000)            [deg]
#   8 A_WORLD         Profile RMS along major axis (world units)   [deg]
#   9 B_WORLD         Profile RMS along minor axis (world units)   [deg]
#  10 THETA_WORLD     Position angle (CCW/world-x)                 [deg]
#  11 FLAGS           Extraction flags
#  12 CLASS_STAR      S/G classifier output
     1  -7.9761   0.0055    256.173    345.802  13.4897848 +42.6645817    1.294875
     2  -7.4532   0.0030    325.355    298.883  11.1841173 +41.4682230   0.3076304
     3  -6.5831   0.0091    488.532    204.895   6.0498366 +38.9019189   0.3732468
     4  -5.8055   0.0134    122.464     69.307  17.5974007 +35.6951906   0.2220612
     5  -6.4558   0.0113    112.304    108.888  17.9680208 +36.6660735   0.5141431
     6  -6.8623   0.0089     42.796    110.897  20.1177235 +36.6171153   0.4007519
     7  -4.7349   0.0271    447.537     30.087   7.7202747 +34.6509259   0.1418599
     8  -4.8464   0.0244    141.073     26.322  16.9774792 +34.6486083   0.1856625
     9  -3.6233   0.0439    149.390      6.947  16.7058481 +34.1767346   0.1032758
    10  -4.6290   0.0270    116.059     24.841  17.7291956 +34.5885929   0.1536183
    11  -4.6401   0.0267    123.899     11.771  17.4758084 +34.2737689    0.121874
    12  -4.3480   0.0300    358.904     31.671  10.3866784 +34.7882701   0.1371353
    13  -3.9992   0.0372    228.023      5.477  14.3442881 +34.1783551   0.1473174
    14  -4.5015   0.0294    264.743     23.606  13.2389150 +34.6290196   0.2331842
    15  -4.9884   0.0231    248.059     16.326  13.7429189 +34.4490879    0.209314
    16  -5.6699   0.0162    303.545      8.196  12.0739439 +34.2393553   0.2182117
```

Figure 3.2.3 : Catalogue d'objets astronomiques

7.3. AÏDA

AÏDA (Astronomical Image processIng Distribution Architecture) est un projet initié à l'observatoire permettant l'accès à des traitements d'images. Toutes sortes de techniques ont été implémentées dans des outils de traitement d'images applicables dans le domaine de l'astronomie. AÏDA a la volonté d'uniformiser l'accès à ces outils pour leur donner une meilleure visibilité et permettre leur utilisation par un plus grand nombre de personnes.

7.3.1 PRINCIPE

Pour lancer un outil via Aïda, il suffit d'envoyer une requête HTTP sur un CGI qui va se charger d'exécuter un programme (*voir figure 3.3.1*). Il envoie par la suite la localisation du résultat obtenu sous la forme d'une URL. Les éléments résultants sont situés sur le serveur qui héberge AÏDA. L'emploi du protocole HTTP rend l'accès à ces outils assez aisé.

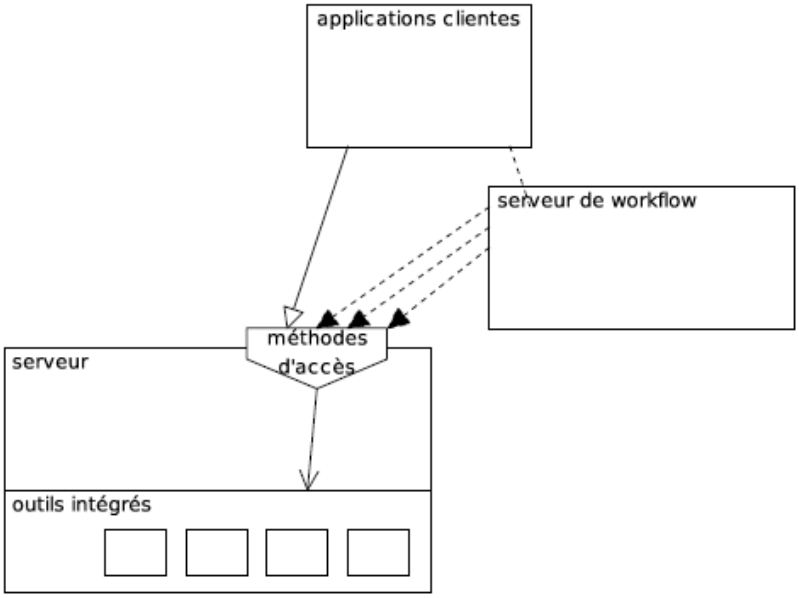


Figure 3.3.1 : Vue d'ensemble de l'architecture Aïda

De nombreux outils sont ainsi disponibles au travers d'AÏDA, dont deux qui nous intéressent grandement. Le premier permet d'obtenir la carte de segmentation d'images multibandes grâce à une approche utilisant le modèle Markovien (*voir figure 3.3.2*). La segmentation permet d'identifier des zones, selon le rayonnement des objets sur plusieurs bandes (différentes longueurs d'onde), comme le montre les images ci-dessous.

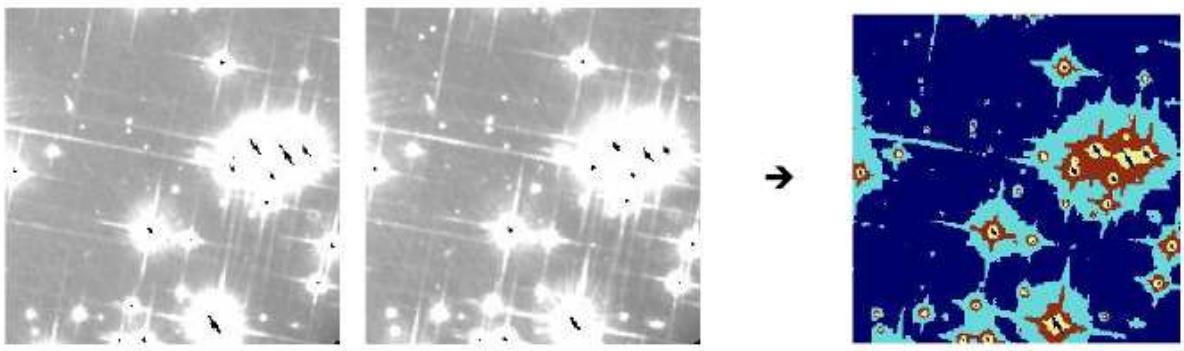


Figure 3.3.2 : Segmentation de deux images de Hubble prises à 606 et 814 nm

Le second outil permet de visualiser une image dans l'espace de couleur TSL (Teinte – Saturation – Lumière). À l'instar de la coloration RGB (voir chapitre *7.2 Aladin*), il est

possible d'effectuer la même opération mais avec une méthode différente. L'outil génère une image colorée au format JPEG (voir *figure 3.3.3*).

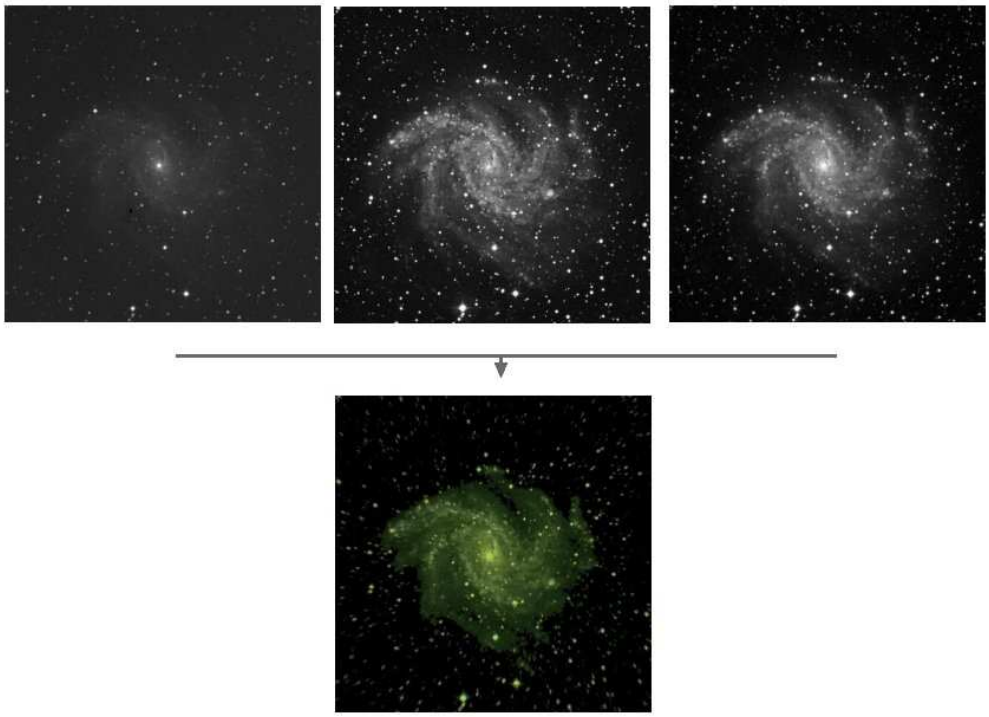


Figure 3.3.3 : Coloration dans l'espace de couleur TSL

7.3.2 INSTALLATION SUR LE CLUSTER

L'installation de cet outil se réalise de manière particulière puisqu'il faut tout d'abord récupérer la totalité d'un répertoire contenant ce qu'il y a à installer sur la machine cible. Il est nécessaire d'avoir au préalable un serveur HTTP (configurer avec le module suexec) afin que l'installeur puisse déposer le CGI dans le répertoire prévu à cet effet. Il faut également créer un utilisateur « aida » car toute l'installation est faite dans le répertoire personnel de celui-ci.

Il ne m'a pas été possible de créer un paquet Debian pour l'installation d'AÏDA du fait de sa complexité à être mis en place. Il m'a fallut donc suivre scrupuleusement la procédure d'installation lors de son ajout sur l'image déployée sur les nœuds de travail.

Afin de faire fonctionner AÏDA sur le cluster, j'ai dû créer un programme PERL pour chacun des outils AÏDA dont nous avions besoin. Ce programme se charge d'émettre la requête HTTP au CGI avec les bons paramètres passés en argument. Pour le moment, il existe donc deux programmes PERL (voir *annexe 4*) localisés sur les nœuds du cluster permettant d'effectuer une segmentation d'images multibandes et la visualisation d'une image dans l'espace de couleur TSL (Teint - Saturation - Lumière). Si le besoin s'en faisait sentir, il serait intéressant de concevoir un programme PERL qui donnerait la possibilité d'effectuer n'importe quelle opération AÏDA.

7.4. WORKFLOW

On appelle « WorkFlow » (traduisez littéralement "flux de travail") la modélisation et la gestion informatique de l'ensemble des tâches à accomplir et des différents acteurs impliqués dans la réalisation d'un processus métier (aussi appelé processus opérationnel). Le terme de Workflow pourrait donc être traduit en français par Gestion électronique des processus métiers. De façon plus pratique le WorkFlow décrit le circuit de validation, les tâches à accomplir entre les différents acteurs d'un processus, les délais, les modes de validation et fournit à chacun des acteurs les informations nécessaires pour la réalisation de sa tâche.

JLow est le projet sur lequel j'ai travaillé lors de mon précédent stage en IUP3 (2005). Il est composé de deux librairies de classes Java permettant de concevoir une application cliente et serveur de Workflow. Elles sont utilisées dans le projet Aïda pour la constitution d'un logiciel qui donne la possibilité de réaliser un enchaînement d'outils Aïda (*voir la figure 3.4*).

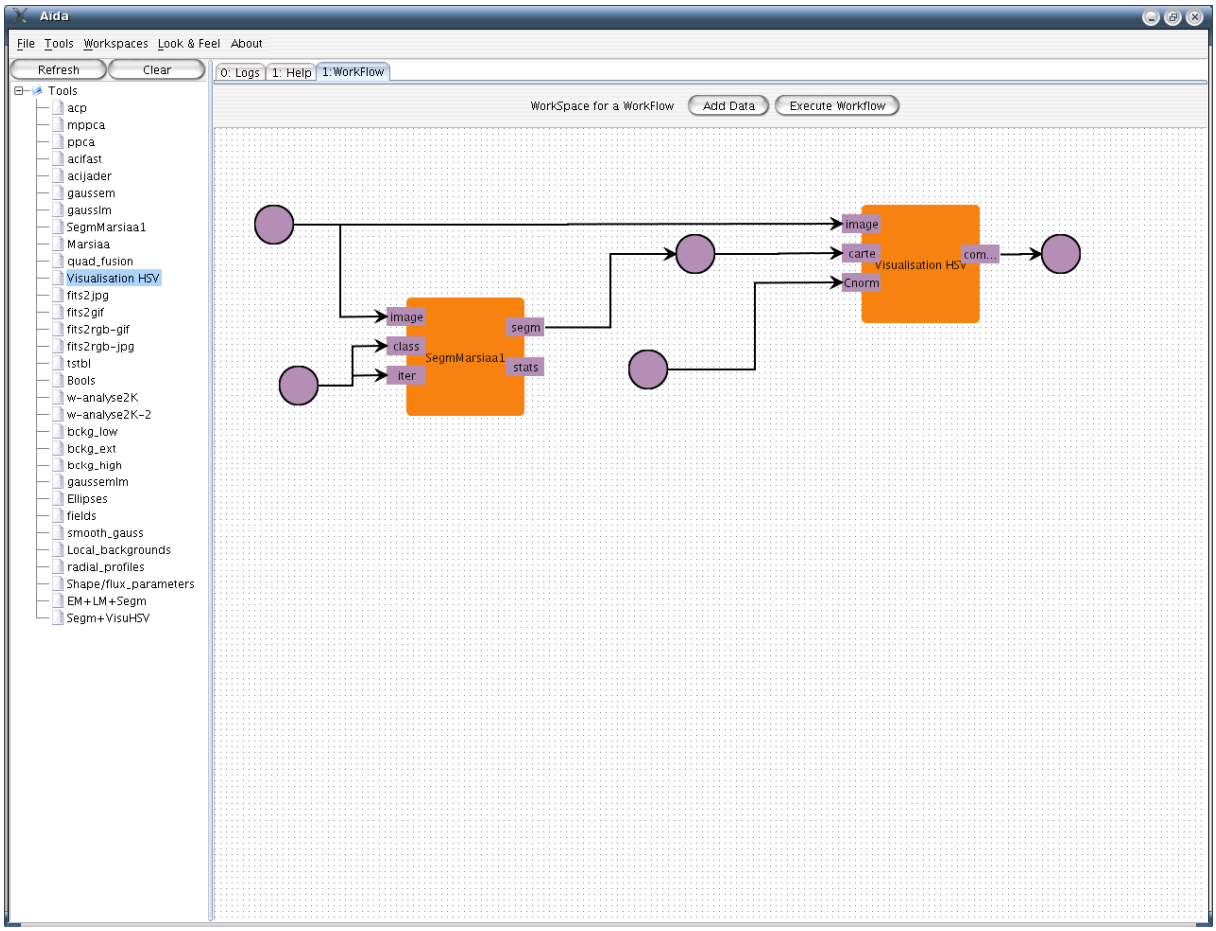


Figure 3.4 : Enchaînement des boites "SegmentationMarsiaa" et "Visualisation HSV"

Il est possible de commander l'exécution d'un traitement à partir de n'importe quelle machine cliente déclarée auprès *d'Ali*. Nous avons donc décidé de créer un serveur *Jlow* sur une machine connue d'Ali, permettant ainsi l'appel au serveur Ali mis en place sur le cluster. Le serveur lit le fichier Ali situé dans un répertoire prédéfini. Il doit peut contenir des valeurs qui serviront par défaut (voir *annexe 5*). Dès la sollicitation d'un outil, un environnement d'exécution est créé afin d'y placer les fichiers nécessaire (fichier Ali reconstitué, et éventuellement un fichier Fits, une carte de segmentation, …).

Cela permet d'exécuter un *Workflow* constitué de plusieurs traitements Ali sur le cluster. Nous pouvons alors lancer l'exécution d'un grand nombre de calculs en parallèle sur le cluster en même temps et sans avoir à écrire un script nous-même.

8. BILAN

Nous avons atteint nos objectifs, concernant l'installation d'une grappe de 8 machines administrables rapidement et simplement. Le cluster sera toujours dans un état stable lorsque l'on souhaitera y ajouter un nœud supplémentaire ou en supprimer un. Il n'est ni sensible à la panne d'une machine ni à celle du nœud serveur grâce à la redondance de celui-ci. Par contre, nous n'avons pas abordé la notion de grille de calcul, car il n'a pas été prouvé que nous en ayons besoin pour le moment. Une étude est en cours afin de déterminer quels traitements pourraient être exécutés par la grille, mais en attendant, le cluster nous suffit.

La simplicité de DCC permet à des personnes non initiées de le prendre en main rapidement. J'ai dû tout de même pour cela, effectuer quelques modifications et améliorations notamment pour obtenir un cluster résistant aux pannes. De plus, une amélioration significative des performances a été relevée grâce à l'utilisation de ce cluster. Je pense qu'il sera intéressant de voir comment il se comporte lorsque l'accès sera public.

L'utilisation conjointe d'Ali (le répartiteur de charge) et d'AÏDA a permis d'ajouter de nouveaux traitements possibles. En effet, AÏDA dispose d'un ensemble d'outils de calcul qui n'étaient accessibles uniquement via l'interface Java ou CGI. L'exécution se déroulait sur une seule machine dédiée. Grâce à Ali, son utilisation s'en retrouve simplifiée en éditant un fichier de configuration à soumettre au serveur Ali. De plus, l'exécution peut maintenant être localisée sur n'importe quel nœud du cluster.

Une application capable d'émettre une demande à Ali verrait sa panoplie d'outils augmentée. Il apparaît donc une notion de couche sur le cluster puisque l'ajout d'un nouveau traitement dans AÏDA peut très facilement impliquer l'ajout de celui-ci à Ali. Ce qui veut dire que l'application cliente qui utilise Ali se voit en perpétuelle évolution. À ce jour, il n'existe pas réellement d'application cliente disposant d'une interface graphique pour Ali, mais ce pourrait être un projet intéressant.

9. EVOLUTIONS POSSIBLES

9.1. LES GRILLES DE CALCULS

Les calculs que traite le cluster ne prennent pas plus d'une minute, temps de transfert du fichier résultant compris. On peut donc considérer que l'envoi de ce genre de traitements sur une grille ne permettrait pas d'accroître les performances. L'emploi d'une grille serait intéressant si l'on disposait de calculs plus complexes nécessitant davantage de ressources et de temps. C'est pourquoi, des tests sont en cours, afin de déterminer quels pourrait être les traitements candidats à une exécution sur la grille. A compte sur Grid5000 a été créé afin d'effectuer des tests.

Suite à ces tests, nous serons en mesure de savoir si le recours à une grille est intéressant pour l'observatoire. Dans ce cas, nous pourrions imaginer une répartition des traitements soit sur le cluster soit sur la grille. Ali pourrait servir d'intermédiaire entre la grille et l'utilisateur comme entre ce dernier et Aida.

9.2. VOSPACE ET VOSTORE

VOSpace et VOStore sont deux éléments importants dans certains projets liés à l'astronomie comme AstroGrid MySpace (projet britannique visant à construire une infrastructure complète pour l'observatoire virtuel) et MyDB développé au John Hopkins University (États-Unis). VOStore est un espace de stockage « à plat » de fichiers (niveau le plus bas), tandis que VOSpace contient les métadonnées sur les fichiers du VOStore.

Pour le moment, lorsqu'on exécute un traitement via Ali, celui-ci se charge de rapatrier le résultat sur la machine cliente. Cela peut durer parfois autant de temps que pour le calcul lui-même puisque le temps moyen du calcul est inférieur à la minute. Nous pourrions alors mettre en place un système d'espace utilisateur qui permettrait à chacun de retrouver son résultat non pas chez lui, mais sur cet espace. Nous gagnerions alors sûrement du temps en regroupant l'espace de travail et l'espace de stockage.

9.3. UNE INTERFACE GRAPHIQUE

Si l'on souhaitait continuer ce projet, je pense qu'il serait intéressant de réaliser une interface graphique qui centraliserait et automatiserait l'ensemble des commandes disponibles à l'image de ce que propose OSCAR.

9.4. LVS ET HEARTBEAT

Si nous constatons dans les prochains mois une augmentation de la demande, mettant en danger la tête du cluster dû à une charge trop importante, nous devrions envisager d'utiliser le système de serveur virtuel que propose le projet LVS (voir figure 2.3 du chapitre 4.4.4 LVS). Au lieu d'avoir un seul serveur s'occupant de dispatcher les traitements sur les nœuds de travail, il y en aurait deux ou plus. Heartbeat, quant à lui, permettrait de dupliquer le point critique de la machine qui est chargé de répartir les demandes extérieures sur les « serveurs réels ».

CONCLUSION

Ce stage a permis de remplacer le cluster existant constitué de machines obsolètes par un nouvel ensemble de machines plus puissantes. Des performances plus importantes des services proposés par le CDS (Centre de Données astronomiques de Strasbourg) ont été ainsi obtenues.

Le nouveau cluster améliore les performances, la disponibilité mais permet également d'ajouter plus facilement de nouvelles fonctionnalités (nouveaux outils de traitements d'images par exemple). Par ailleurs, nous avons intégré dans la nouvelle architecture un outil d'exécution de chaînes de traitements (Workflow) afin de répondre à de nouveaux besoins plus complexes.

La prochaine évolution du projet consistera à ajouter la notion de grille en permettant certains des traitements sur celle-ci. Une réflexion se poursuit actuellement afin d'identifier les traitements pouvant bénéficier de l'utilisation de Grilles et d'intégrer cela dans l'architecture. J'apporterai les évolutions nécessaires après mon stage, durant le mois de septembre.

Concernant les éléments VOStore et VOSpace cités dans le chapitre *évolutions possibles,* il m'a été confié la mission de tester et implémenter la structure nécessaire à l'observatoire dès le mois d'octobre. J'en profite donc pour remercier Mme Françoise Génova et M André Schaaff pour la confiance qu'ils me portent et pour leur soutien face à mon entrée dans la vie active.

GLOSSAIRE

[1] Grid5000, https://www.grid5000.fr/mediawiki/index.php/Grid5000:Home

Grid5000 est un projet qui a pour but de construire une grille expérimentale rassemblant neuf sites à travers toutes la France, soit au total *5000* processeurs.

[2] EGEE, http://public.eu-egee.org/

EGEE est le projet mené par *27* pays et ayant comme objectif commun de fonder une infrastructure en employant la technologie *Grille*, disponible en permanence aux scientifiques qui le veulent.

[3] MPI, http://www-unix.mcs.anl.gov/mpi/

MPI est une librairie de fonction permettant d'exploiter des ordinateurs distants ou multiprocesseurs par passage de messages. Il est devenu de facto un standard de communication pour des nœuds exécutant des programmes parallèles sur des systèmes à mémoire distribuée.

[4] PVM, http://www.csm.ornl.gov/pvm/pvm_home.html

PVM est une bibliothèque de communication pour un groupe d'ordinateurs relié par un réseau distant ou local. Il permet à un réseau d'ordinateurs d'apparaître comme un seul ordinateur, appelé machine virtuelle.

[5] Kerrighed, http://kerrighed.org/

Kerrighed est un système à image unique (*SSI*, *Single System Image*) permettant de voir un ensemble de machine (multi-processeur) comme un tout. Les *SSI* offre une transparence dans la distribution des ressources, une utilisation de la totalité de la puissance fournie par l'ensemble des machines (répartition de la charge), une excellente disponibilité (tolérance aux pannes), et une configuration automatique lors de l'ajout ou la suppression d'une machines.

[6] HPC,

HPC vient de l'anglais *High Performance Computing*, ce qu'on pourrait traduire par calcul à haute performance. Ce terme couvre l'ensemble de l'utilisation d'outils informatiques nécessaires pour effectuer des calculs mathématiques importants.

[7] Maui, http://www.clusterresources.com/pages/products/maui-cluster-scheduler.php

Maui constitue un excellent ordonnanceur de traitement pour un cluster de machine.

[8] TFTP,

TFTP signifie en anglais Trivial File Transfert Protocol, ce qu'on pourrait traduire par protocole simplifié de transfert de fichiers. Il fonctionne en UDP au contraire du FTP qui utilise le TCP (plus fiable).

[9] PXE, http://syslinux.zytor.com/pxe.php

PXE est le sigle de *Pre-boot eXecution Environnemnt*. Il permet de démarrage d'un OS situé sur une machine serveur distante ou d'installer de manière automatique une distribution à partir d'un serveur distant.

[10] SIS, http://wiki.sisuite.org

System Imager Suite est un projet permettant le déploiement d'une image système sur un ensemble de machine. Le projet se compose des outils suivants : SystemImager, SystemInstaller, SystemConfigurator et RescureCD.

[11] C3, http://www.csm.ornl.gov/torc/C3/index.html

Cluster Command and Control est une suite qui implémente de nombreux outils permettant d'améliorer la gestion d'un parc de machine.

[12] Netfilter,

Netfilter est un module qui fournit à Linux les fonctions de pare-feu, de partage de connexions Internet (*NAT [13]*) et d'historisation du trafic réseau. *Iptables* est la commande qui permet de configurer *Netfilter* en mode utilisateur.

[13] NAT,

NAT est l'acronyme de *Network Address Translation* ou en français Traduction d'adresses réseau. C'est un mécanisme qui permet de faire correspondre une adresse IP interne non unique vers un ensemble d'adresses externes uniques.

[14] SSH,

SSH signifie Secure Shell et est à la fois un programme permettant l'execution à distance de commandes SHELL en mode sécurisé et un protocole de communication sécurisé.

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

OpenPBS (*Portable Batch System*) est un gestionnaire de batch (traitement par lot ou différé). Il a été développé par la NASA au milieu des années 90.

[16] OpenSSI, http://openssi.org/cgi-bin/view?page=openssi.html

OpenSSI est un projet qui à pour but d'offrir des capacités de haute disponibilité aux environnements Linux, en partageant les ressources, et en contrôlant la charge CPU des nœuds.

[17] OpenMosix, http://openmosix.sourceforge.net/

OpenMosix est une solution SSI pour un cluster. Il réparti la charge sur l'ensemble des machines en optimisant l'utilisation des ressources disponibles. Il se différencie des autres SSI en visant les performances avant tout.

ANNEXES

ANNEXE 1 : GUIDE ADMINISTRATEUR

GUIDE ADMINISTRATEUR

Ce guide à pour but d'aider la prise en main du cluster en décrivant un ensemble d'action usuelle qu'un administrateur peut avoir à effectuer sur celui-ci. Pour plus de précision, je vous recommande d'aller regarder sur le site de DCC (*http://www.irb.hr/en/cir/projects/internal/dcc/*).

Installation des nœuds

Installation d'un noeud existant

Dans le cas où vous souhaitez réinstaller un noeud qui faisait déjà parti du cluster et qui est donc reconnu par le noeud principale, il vous faut simplement changer le fichier que la machine récupère grâce au protocole PXE au démarrage de la machine. Pour ce faire, j'ai créé un script SHELL qui prend en paramètre le nom de la machine à réinstaller afin de changer le bon fichier. Cette commande s'appelle *install-node* et en voici un exemple d'utilisation :

EX. `$ install-node wali11`

En exécutant cette commande vous indiquez qu'au prochain démarrage de la machine qui porte le nom *wali11*, elle devra se réinstaller avec l'image correspondante.

Il ne vous reste plus qu'à redémarrer la machine si elle est en cours de fonctionnement à l'aide de la commande suivante :

```
$ cexec name_headnode:num_node reboot
EX. $ cexec wali1:1 reboot
```

Ou :

```
$ cshutdown  name_headnode:num_node -r t 0
EX. $ cshutdown wali1:1 -r t 0
```

Installation d'un nouveau noeud

Concernant l'installation d'un noeud qui n'est pas connu par la tête du cluster, il est nécessaire auparavant de l'enregistrer auprès de ce dernier. Il subira automatiquement une installation. Pour cela, vous pouvez employer la commande *dcc_discovernode*. Sans paramètre elle va chercher les éléments dont elle a besoin dans le fichier de configuration « /etc/dcc/config » et elle se chargera de réserver une adresse IP non utilisée pour le nouveau noeud. Et enfin, elle installera la machine avec l'image qui est spécifié dans le fichier de configuration (voir */etc/dcc/config*).

Il est préférable d'exécuter la commande *cpushimage* après chaque installation afin de mettre à jour les autres noeuds du cluster. De plus, il est plutôt recommandé d'utiliser plutôt le script *update-node* si le répartiteur de charge Ali est en place, afin de régler certains problèmes.

EX. `$ cpushimage node` ou `$ update-node node`

Modification des nœuds

Installation d'un logiciel sur l'image

Afin d'installer un nouveau programme sur les machines du cluster il faut l'ajouter tout d'abord à l'image qui est située sur le noeud serveur. Pour cela, il existe une commande qui permet d'arriver dans un environnement chrooté à partir de l'image désirée.

EX. `$ dcc_editimage node`

```
$ CHROOT [node@wali1: /] apt-get install vim
$ CHROOT [node@wali1: /] exit
```

Une fois chrooté dans l'image, vous vous retrouvez dans un SHELL classique et il vous suffit de faire votre installation. Il est à noter que la commande APT fonctionne parfaitement à l'intérieur de l'environnement.

Note :

Sachez que l'image que vous avez à modifier est dans le répertoire « /var/lib/systemimager/images ».

Après l'installation du logiciel terminée, vous avez simplement à sortir de l'environnement dans lequel vous étiez en tapant la commande *exit* et il ne vous reste plus qu'à mettre à jour les noeuds à l'aide de la commande *cpushimage*. Comme pour l'installation d'une nouvelle machine il est préférable d'utiliser le script *update-node* dans le cas où Ali est installé.

Dans certain cas, il est nécessaire de réinstaller entièrement les machines, dans ce cas reporter vous à l'intitulé *Installation d'un noeud existant*.

Modification du disque des noeuds

La modification du partitionnement du disque des noeuds ne pose absolument aucun problème et est totalement indépendant autant de l'image que du contenu des machines. Il vous faut modifier le fichier de configuration « /etc/dcc/disktable » qui contient les partitions actuelles. Puis, afin que les modifications soient prises en compte, il est nécessaire d'appeler la commande *mksidisk* et *mkautoinstallscript* qui auront pour effet de changer le script lancé à l'installation de la machine.

```
$ mksidisk --A --name node_name --file /etc/dcc/disktable
$ mkautoinstallscript --image node_name force --ip-assignment
dhcp --post-install reboot
```

En effet, c'est à l'installation de la machine que les partitions sont faites, c'est pourquoi il vous faudra la réinstaller (voir *Installation d'un noeud existant*).

Suppression

Suppression d'une machine

Dans le cas de la suppression d'une machine dans le cluster, il faut lancer la commande *mksimachine* afin de mettre à jour la base de données maintenue sur le noeud server. Puis, le script *cpushimage* ou *update-node* permet aux noeuds de prendre connaissances de la suppression d'une machine.

```
ex. $ mksimachine -D --name wali12
    $ update-node
```

Suppression d'une image

Afin de retirer une image sur le noeud principale, il est conseillé d'utiliser *mksiimage*.

```
EX. $ mksiimage -D --name node_test
```

Mais il n'est pas possible d'enlever une image qui est utilisée par des machines du cluster. Si vous avez besoin d'enlever le lien entre une image et une machine exécuter la commande *mksimachine* de la manière suivante :

```
$ mksimachine -U --name image_name --image other_image_name
```

Les fichiers de configuration

/etc/dcc/config

/etc/dec/disktable

/etc/dec/modules

/etc/dec/packages.list

/etc/dec/sources.list

ANNEXE 2 : FICHIER DE CONFIGURATION DE HEARTBEAT

```
#######################################################
# ha.cf : configuration file of Linux-HA
#######################################################


logfacility daemon          # Log to syslog as facility "daemon"
node wali1 wali2            # List our cluster members (IP public)
keepalive 1                 # Send one heartbeat each second
deadtime 10                 # Declare nodes dead after 10 seconds
bcast eth0 eth1             # Broadcast heartbeats on eth0 and eth1
interfaces
ping 130.79.129.254         # Ping our router to monitor ethernet
connectivity
auto_failback no            # Don't fail back to wali1 automatically
respawn hacluster /usr/lib/heartbeat/ipfail  # Failover on network
failures
```

ANNEXE 3 : SCRIPT DE DEMARAGE POUR LE PROGRAMME *XVFB*

```
#! /bin/sh
set -e

# /etc/init.d/xvfb: for item rgb of Ali

xvfb_command=/usr/bin/X11/Xvfb

case "$1" in
        start)
           echo -n "Starting Xvfb..."
           start-stop-daemon --start --quiet -b --exec $xvfb_command
-- ":2" -screen 4 1024x768x8 2>/dev/null
           echo "OK"
           ;;
     stop)
           echo -n "Stopping Xvfb..."
           start-stop-daemon    --stop    --quiet    --oknodo    --exec
$xvfb_command
           echo "OK"
           ;;
     reload|force-reload)
           echo -n "Reloading Xvfb..."
           start-stop-daemon --stop --signal 1 --quiet --oknodo --
exec $xvfb_command
           echo "OK"
           ;;
     restart)
           echo -n "Restarting Xvfb..."
           start-stop-daemon --stop --quiet --oknodo --retry 30 --
exec $xvfb_command
           start-stop-daemon --start --quiet -b --exec $xvfb_command
-- ":2" -screen 4 1024x768x8 2>/dev/null
                echo "OK"
```

```
        ;;
    *)
        echo "Usage: $0 {start|stop|reload|force-reload|restart}"
            exit 1
esac
exit 0
```

ANNEXE 4 : PROGRAMME PERL FAISANT LA JONCTION ENTRE AÏDA ET ALI

```
#!/usr/bin/perl
use strict;
use warnings;
use HTTP::Request::Common;
use LWP::UserAgent;

use Getopt::Std;
use Pod::Usage;

my %Opts;
my($ImgSegm,$ClassCount, $IteCount);

ParseArgs();
CallToCgi();

sub ParseArgs
{

getopt('', \%Opts);

$Opts{H} and
        pod2usage(VERBOSE=>1);

$Opts{M} and
        pod2usage(VERBOSE=>2);

($ImgSegm, $ClassCount, $IteCount) = @ARGV;
    $IteCount or
         pod2usage(VERBOSE=>0);

}

sub CallToCgi
{
```

```
my $ua=LWP::UserAgent->new();
$ua->timeout(4200);

my %rest = ( "tool.id"=>52); # segmentation using marsiaa basic
descriptor

my @Imgs = split /;\s*/, $ImgSegm;
my $ImgCount = @Imgs;

my $i=0;
for $i (0 .. ($ImgCount-1)) {
     $rest{"input.1.".($i+1)} = $Imgs[$i];
}
$rest{"input.2"} = $ClassCount; # number of classes
$rest{"input.3"}=$IteCount; # maximum number of iteration
$rest{"output_format"}="fits"; # get the segmentation map result as
mime(image/fits)

my $res=$ua->request(
        POST "http://localhost/~aida/cgi-bin/useTool.cgi",
     [ %rest ]
);
my $ofile="segm1_perl.fits";
open FITS, ">$ofile" or die "cannot write result";
print FITS $res->content();
close FITS;
}
```

ANNEXE 5 : FICHIER *ALI* SERVANT DE REFERENCE POUR LE SERVEUR *JLOW*

1. FICHIER ALI ORIGINEL

```
# ------------------------------------------------------------------
-------------- #

%HEADER SEGM1 by aida
%FIFILES
%FOFILES segm1_perl.fits
%SEPARATOR ' '

#
%ID 1
%CMD /root/bin/test/segm1
'http://localhost/~aida/input.sam/1;http://localhost/~aida/input.sam
/2;http://localhost/~aida/input.sam/3;http://localhost/~aida/input.s
am/4' 2 2
%IFILES
%OFILES
%LFILES segm1_perl.fits
%DEPS
%COST 50
%REQUIRE wali11
%TIMEOUT 1000000
%RETRIES 0

# Fin du fichier de requete.
# ------------------------------------------------------------------
-------------- #
```

2. FICHIER ALI MODIFIE POUR LE SERVEUR JLOW

```
# ------------------------------------------------------------------
-------------- #

%HEADER SEGM1 by aida
```

```
%FIFILES
%FOFILES segm1_perl.fits
%SEPARATOR ' '

#
%ID 1
%CMD /root/bin/test/segm1
Images='http://localhost/~aida/input.sam/1;http://localhost/~aida/in
put.sam/2;http://localhost/~aida/input.sam/3;http://localhost/~aida/
input.sam/4' Classes=2 Iterations=2
%IFILES
%OFILES
%LFILES segm1_perl.fits
%DEPS
%COST 50
%REQUIRE wali11
%TIMEOUT 1000000
%RETRIES 0

# Fin du fichier de requete.
# ------------------------------------------------------------------
--------------- #
```