Une nouvelle architecture cluster pour la recherche scientifique : Condor, PROOF et Xrootd

Hao Ni

CONSERVATOIRE NATIONAL DES ARTS ET METIERS

CENTRE REGIONAL ASSOCIE DE Lyon

_____

MEMOIRE

présenté en vue d'obtenir

le DIPLOME d'INGENIEUR CNAM

SPECIALITE : Informatique

OPTION : Systèmes d'information (ISI)

par

Hao, Ni

_____

Une nouvelle architecture cluster pour la recherche scientifique : Condor, PROOF et Xrootd

Soutenu le 25 juin 2010

_____

JURY

PRESIDENT : M. Christophe Picouleau (Cnam Paris)

MEMBRES : M. Bertrand David (Cnam Lyon)

M. Claude Genier (Cnam Lyon)

M. Neng Xu

M. Wen Guan

Remerciements

Je tiens tout d'abord à remercier Monsieur Genier, l'encadrant de mon mémoire qui m'a guidé et encouragé pendant toute la durée de la rédaction de ce mémoire

Je remercie vivement Madame Sau Lan Wu, Professeur à l'université Wisconsin, qui m'a permis de réaliser mon projet dans son Groupe Wisconsin en m'accordant toute sa confiance.

Je voudrais remercier également M. Neng Xu, le responsable système d'information du groupe, qui m'a prodigué de nombreux conseils et a suivi régulièrement l'évolution de mon travail.

Je voudrais aussi remercier M. Wen Guan et l'équipe PROOF pour leur collaboration active au déroulement de ce projet, leurs explications techniques et leurs précieux conseils.

Enfin, un grand merci à ma femme qui m'a aidé et soutenu pendant toutes ces années. Sans elle je ne serais jamais arrivé jusqu'ici.

Ni Hao

Table des matières

Remerciements 2

Introduction 7

Première partie 9

Chapitre 1 présentation du CERN 10

1.1 L'histoire du CERN et ses dates clés 10

1.2 La mission du CERN 11

1.3 Présentation LHC 12

1.4 Les expériences du LHC 12

1.5 Conclusion 14

Chapitre 2 Le Wisconsin groupe 15

2.1 La mission du groupe 15

2.2 L'organisation du groupe 15

2.3 Conclusion 16

Chapitre 3 Système d'information du groupe 17

3.1 Charge de travail 17

3.2 Infrastructure du système d'information 18

3.3 Architecture du système d'information 21

3.4 Conclusion 22

Chapitre 4 La motivation du projet 23

4.1 Extensibilité 23

4.2 Évolutivité 23

4.3 Fiabilité 23

4.4 Rentabilité 24

4.5 Visibilité 24

4.6 Conclusion 24

Conclusion de la première partie 25

Deuxième Partie 27

Chapitre 5 Condor 28

5.1 Caractéristiques 28

5.2 Le fonctionnement du système Condor 29

5.3 Les Outils d'administration 30

5.4 Conclusion 31

Chapitre 6 Xrootd 32

6.1 Caractéristiques 32
6.2 Equilibre de charge dynamique 33
6.3 Fonctionnement de Xrootd et son extensibilité 34
6.4 Conclusion 35
Chapitre 7 ROOT et PROOF 36
7.1 Présentation de ROOT 36
7.2 Présentation de PROOF 37
7.3 Conclusion 38
Chapitre 8 Grid 39
8.1 Introduction aux grilles 39
8.2 Les domaines d'application des grilles informatiques 40
8.3 L'architecture Grid 40
8.4 Globus Toolkit et Visual Data Toolkit 42
8.5 Conclusion 44
Conclusion de la deuxième partie 46
Troisième Partie 47
Chapitre 9 Ingénierie des besoins 48
9.1 Les approches de l'ingénierie des besoins 49
9.2 Les parties prenantes 49
9.3 Les types de besoins 50
9.4 La démarche utilisée dans le cadre du projet 51
9.5 Les contraintes potentielles 52
9.6 La découverte des besoins 52
9.7 Analyse des besoins 54
9.8 Spécification des besoins 56
9.9 Validation des besoins 60
9.10 Conclusion 61
Chapitre 10 L'infrastructure matérielle du système existant 62
10.1 Le réseau 62
10.2 Le stockage 64
10.3 Les serveurs 67
10.4 Conclusion 70
Chapitre 11 Le clustering 71
11.1 Les différentes formes de scalabilité 71
11.2 Les différentes formes de clustering 73

11.3 Conclusion 74
Chapitre 12 Le flux de données 76
12.1 Le flux de données interne 76
12.2 Le flux de données externe 78
12.3 Conclusion 80
Chapitre 13 La production 81
13.1 La production : Grid 81
13.2 La production : Condor 83
13.3 La production : PROOF 86
13.4 La base de données 89
13.5 Conclusion 90
Conclusion de la troisième partie 91
Quatrième partie 93
Chapitre 14 La stratégie 94
14.1 La démarche 94
14.2 Etat des lieux 94
14.3 Les moyens 95
14.4 Les risques 97
14.5 Les objectifs du système d'information 98
14.6 Le plan de pilotage 100
14.7 Conclusion 101
Chapitre 15 Le découpage du projet et la méthode adoptée 102
15.1 Le découpage du projet 102
15.2 L'adoption de la méthode « agile » 104
15.3 Les bonnes pratiques 105
15.4Conclusion 108
Chapitre 16 La réalisation 109
16.1 Le sous-système « Architecture du système » 109
16.2 Le sous-système « La production » 114
16.4 Le sous-système « Infrastructure du système » 118
16.5 Le sous-système « Gestion de la base de données » 119
16.6 Conclusion 122
Conclusion de la quatrième partie 123
Le bilan 124
Conclusion 126

Table des figures.....127
Liste des tableaux 128
Bibliographie 129
Webgraphie.....131
Index 132
Résumé et mot clé.....133

Introduction

Selon une étude de l'INSEE, en 2004, 31% des ménages avaient accès à internet, c'est à dire cinq fois plus qu'en 1999. En 2004, plus de 45% foyers disposaient d'au moins un micro-ordinateur, trois fois plus qu'en 1996. Il est évident que l'informatique fait aujourd'hui partie de notre vie quotidienne.

En effet, le premier ordinateur fit son apparition dans les années 40. Inventé pour des besoins en calcul, il fut plus tard utilisé pour corréler des informations. Cependant sa fonctionnalité initiale est largement utilisée dans le monde de la recherche scientifique. Il est vrai que la performance du SI[1] est primordiale pour les chercheurs, les institutions ou des entreprises.

Le cabinet d'étude Gartner a publié une étude expliquant que, contrairement aux idées reçues, les « plantages » imprévus sont majoritairement liés à des erreurs applicatives (40%), et à des erreurs humaines (40%) plutôt qu'à des pannes matérielles (20%). Ceci démontre l'importance de la conception de l'architecture du SI et de l'organisation de la performance.

Le Wisconsin groupe fut un des premiers groupes des États-Unis à intégrer l'expérience d'ATLAS[2] au CERN. Ce groupe dispose d'une centaine de serveurs pour répondre aux besoins en puissance de calculs et de simulations. Il n'est pas difficile à comprendre que la performance de son système d'information est un facteur clé de succès.

Dans les chapitres suivants, je vais présenter un projet de mise en place d'une nouvelle architecture du système d'information de ce groupe, visant à en accroître la performance et la fiabilité. Pour bien distinguer les différentes phases de notre réalisation, ce document est structuré en quatre grandes parties. Dans la première partie, nous allons expliquer le contexte et la motivation du projet. Nous allons évoquer les principales technologies utilisées dans notre système d'information au cours de la seconde partie. Dans la troisième, seront étudiés les besoins des utilisateurs et le système existant. En dernière partie, nous présenterons les différentes réalisations effectuées pour mener à bien notre projet.

[1] SI : système d'information

[2] ATLAS : A Toroidal LHC ApparatuS. C'est une collaboration mondiale réunissant plus de 2100 scientifiques et ingénieurs de 167 instituts et de 37 pays et régions.

Première partie

Le contexte et la motivation du projet

Chaque projet a sa spécificité. Comprendre le contexte et la motivation d'un projet sont des aspects très importants pour tracer sa feuille de route. Alors dans cette partie j'aborderai le contexte du projet ce qui permettra de découvrir l'ensemble des conditions et des circonstances qui englobent ce projet. Je décrirai, dans un deuxième temps, la motivation du projet. Cette description du projet permettra ainsi de réunir l'ensemble des raisons qui expliquent le besoin de changement.

Pour expliquer le contexte et la motivation du projet, seront présentés le CERN puis le Wisconsin groupe et son organisation. Je décrirai ensuite le SI actuel du groupe et puis je finirai par une illustration de la motivation du projet.

Chapitre 1 présentation du CERN

Le CERN[1], Organisation européenne pour la recherche nucléaire se situe à quelques kilomètres de Genève, sur la frontière franco-suisse. C'est l'un des plus grands et des plus prestigieux laboratoires scientifiques du monde. Il est également le plus grand centre de physique des particules du monde.

Le CERN compte vingt États membres européens. De nombreux pays non européens participent néanmoins à ses activités. D'ailleurs, le Wisconsin groupe vient des États-Unis, donc non européen, exerce des activités de recherche au sein du CERN.

En 2004, le CERN a soufflé ses 50 bougies. Aujourd'hui, il accueille environ 8000 scientifiques visiteurs, soit la moitié des physiciens des particules du monde. Ils viennent au CERN pour mener des recherches. Il est vrai que nous ne pouvons ignorer l'importance du CERN dans la recherche de physique de particules. Avec ses 50 ans d'existence, le CERN a construit plusieurs accélérateurs et collisionneurs pour exercer des expériences dans ce domaine. Dans ce mémoire, l'histoire du CERN ainsi que sa mission seront décrites. Ensuite sera présentée sa plus récente réalisation : le LHC[2] (Large Hadron Collider).

1.1 L'histoire du CERN et ses dates clés

L'idée de créer un laboratoire scientifique européen a été lancée par un Français Louis de Broglie, le Prix Nobel de physique de 1929, lors de la Conférence Européenne de la Culture à Lausanne en 1949. En mai 1954, les premiers travaux pour la construction du laboratoire et de son accélérateur ont commencé. Le 29 septembre 1954, la convention du CERN est ratifiée par 12 États européens : le CERN est officiellement créé et se nomme maintenant Organisation Européenne pour la Recherche Nucléaire.

En 1957, le Synchrocyclotron à proton de 600 MeV[3], le premier accélérateur du CERN, est mis en service. Il fournit des faisceaux aux premières expériences de physique des particules et de physique nucléaire du laboratoire. Il a servi jusqu'en 1990, après 33 ans de services.
Deux ans plus tard, le 24 novembre 1959, le PS[4] accélère ses premiers protons. Avec une énergie de faisceau de 28 GeV[5], le PS se met au service du programme de physique des particules du

[1] Le CERN, Organisation européenne pour la recherche nucléaire, est le plus éminent laboratoire de recherche en physique des particules du monde. Il a son siège à Genève.

[2] LHC : Large Hadron Collider (en français : Le Grand collisionneur de hadrons)

[3] MeV : Méga Électronvolt. En physique, l'électronvolt (symbole eV) est une unité de mesure d'énergie.

[4] PS : Synchrotron à proton

[5] GeV : Giga Électronvolt.

CERN ; Aujourd'hui encore, il continue à fournir des faisceaux de particules pour les expériences.

En 1971, le CERN a réalisé le premier collisionneur de proton à proton du monde. Plus tard, en 1976, est mis en place un second collisionneur SPS [1]. C'est une machine d'une circonférence de 7 kilomètres, implantée à 40 mètres sous terre. Avec ce collisionneur, deux physiciens, Carlo Rubbia et Simon van der Meer, ont pu découvrir des particules de champ W et Z. Cette découverte a permis à ces deux physiciens d'obtenir le prix Nobel en 1984. Ce collisionneur a servi de précurseur pour le Large Electron Positron (LEP) et sert également au Large Hardon collider (LHC). Cette construction a fourni un précieux savoir faire pour les projets ultérieurs de collisionneurs.

Le Grand collisionneur électron-positron (LEP) a été mis en service en juillet 1989. Avec ses 27 kilomètres de circonférence, le LEP reste aujourd'hui le plus grand accélérateur d'électrons et de positrons au monde. Le LEP a arrêté son service le 2 novembre 2000. Il laisse sa place au LHC.

Tout récemment, en septembre 2008, le LHC fournit ses services. Les expériences du LHC vont tenter de répondre à plusieurs questions, par exemple : comment les particules acquièrent-elles leur masse ? Comment la matière a-t-elle évolué au tout début de l'existence de l'Univers ?

Ces grandes œuvres que sont la construction de ces accélérateurs et collisionneurs pour la recherche fondamentale ne représentent pas les seuls domaines de compétence du CERN. Par exemple, le Web que nous utilisons tous les jours, est né au CERN en 1990.

1.2 La mission du CERN

La Convention instituant le CERN en 1954 définit clairement quatre missions principales de l'Organisation:

- Recherche : Chercher des réponses aux questions concernant l'Univers,
- Technologie : Faire reculer les limites de la technologie,
- Collaboration : Rassembler les nations au travers de la science,
- Éducation : Former les scientifiques de demain.

Le CERN suit toujours cette ligne : chaque année des milliers de chercheurs et de scientifiques y viennent pour la recherche, des centaines d'étudiants pour faire un stage et réaliser leurs thèses. Avec le démarrage de LHC, le CERN va renforcer la coopération internationale dans la recherche avec d'autres laboratoires et instituts.

[1] SPS : Super Proton Synchrotron

1.3 Présentation LHC

Le LHC a démarré en Septembre 2008. Il a été construit pour aider les scientifiques à répondre à certaines questions essentielles de la physique des particules qui restent sans réponse. En revanche, le projet de construction du LHC a été adopté en décembre 1994 par le conseil du CERN. Au départ, en raison des contraintes budgétaires, le LHC devrait être réalisé en deux étapes. Cependant, grâce à des contributions du Japon, des États-Unis, de l'Inde et d'autres Etats non-membres, le conseil a accepté en 1995 que le projet soit mené à bien en une seule étape.

Entre 1996 et 1998, quatre expériences – ALICE[1], ATLAS, CMS[2], LHCb[3] – ont été officiellement approuvées et les travaux de construction ont débuté sur quatre sites. Depuis, deux expériences plus petites sont venues s'ajouter au projet : TOTEM[4], installée à côté de CMS, et LHCf[5], près d'ATLAS.

1.4 Les expériences du LHC

Le LHC est installé sous terre. A 100 mètres en moyenne avec une circonférence de 27 kilomètres. Dans le LHC, il y a six expériences : ALICE, ATLAS, CMS, LHCb, LHCf et TOTEM. Les expériences ALICE, ATLAS, CMS et LHCb sont installées dans quatre énormes cavernes souterraines construites autour des quatre points de collision des faisceaux du LHC. L'expérience LHCf se trouve près de l'expérience ATLAS et l'expérience TOTEM est placée près du point d'interaction de CMS.

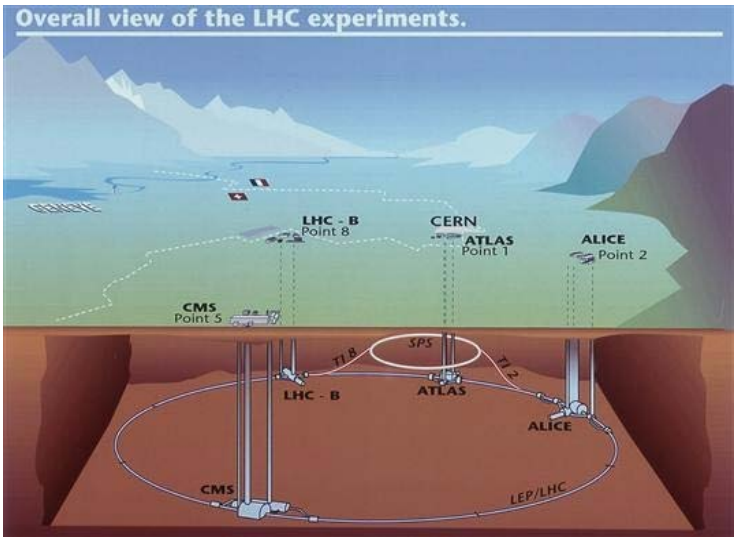

Figure 1 : Les expériences de LHC

[1] ALICE : A Large Ion Collider experiment

[2] CMS : Solénoïde compact à muons

[3] LHCb : Large Hardon Collider beauty

[4] TOTEM : TOTal Elastic and diffractive Cross section Measurement

[5] LHCf : Large Hardon Collider forward

La figure 1 montre que le LHC est installé sous la terre, non loin de Genève, tant sur le territoire suisse que sur le territoire français. Les détecteurs (points de collision des particules) sont installés dans différents points et sont utilisés pour diverses expériences. Le tableau 1 montre les caractéristiques telles que la taille, le poids, le coût et le domaine de recherche pour chaque expérience. Nous pouvons constater que parfois leurs caractéristiques sont très variées:

Tableau 1: Les caractéristiques des expériences du LHC

Expérience	Domaine de recherche	Taille	Poids	Coût	Situation
ALICE	Détecteur spécialisé dans l'analyse des collisions d'ions plomb	Longueur :26m Largeur :16m Hauteur :16m	10,000T	115 MCHF	St Genis-Pouilly (France)
ATLAS	Détecteur polyvalent conçu pour couvrir les aspects les plus divers de la physique au LHC, de la recherche du boson de Higgs à celle de la supersymétrie (SUSY) en passant par la quête de dimensions supplémentaires	Longueur :46m Largeur :25m Hauteur :25m	7,000T	540 MCHF	Meyrin (Suisse)
CMS	Détecteur qui poursuit les mêmes objectifs de physique qu'ATLAS, mais avec une conception et des solutions techniques différentes.	Longueur :21m Largeur :15m Hauteur :15m	12,500T	500 MCHF	Cessy (France)
LHCb	Consacré à l'étude de la légère asymétrie entre matière et antimatière, par l'observation des mésons B (Particule contenant le quark b).	Longueur :21m Largeur :13m Hauteur :10m	5600T	75 MCHF	Ferney-Voltaire (France)
LHCf	Expérience destinée à mesurer les particules émises selon un angle très petit par rapport à la direction du faisceau lors des collisions proton-proton dans le LHC	Longueur :30cm Largeur :10cm Hauteur :10cm	40Kg	--	Meyrin (Suisse)
TOTEM	Conçu pour détecter les particules produites au plus près des faisceaux du LHC	Longueur :440m Largeur :5m Hauteur :5m	20 T	6,5 MCHF	Cessy (France)

Le collisionneur LHC a démarré début septembre 2008. Cependant, suite à une défaillance d'une connexion électrique, il a été arrêté à partir du 19 septembre 2008. Il a été remis en marche le 30 mars 2010 et fonctionne aujourd'hui.

1.5 Conclusion

Depuis la création du CERN en 1954, plusieurs collisionneurs ont été réalisés pour répondre aux besoins de ses activités. La plus récente réalisation, le LHC, nous permet de répondre à certaines questions essentielles de la physique des particules. Il faut souligner que de nombreuses institutions et universités provenant des quatre coins du monde, participent aux expériences du LHC. Le Wisconsin groupe, dont nous allons parler largement par la suite, est un groupe de recherche universitaire qui vient des États-Unis.

Chapitre 2 Le Wisconsin groupe

Le Wisconsin groupe est un des groupes qui effectuent leurs recherches dans les expériences du LHC. Ce groupe est impliqué fortement dans plusieurs expériences physiques sur les particules dans le monde. Le Wisconsin groupe a commencé sa participation à l'expérience ATLAS du LHC en Septembre 2003 et a rejoint deux ans plus tard l'expérience BaBar du SLAC[1]. SLAC est implanté à l'université de Stanford aux Etats-Unis. Ce centre est attaché à l'U.S Department Of Energy (le ministère de l'énergie) dont les domaines de recherches sont assez similaires à ceux du CERN.

2.1 La mission du groupe

Le nom exact du groupe Wisconsin est Wisconsin-Madison high energy physics research group (groupe de recherche physique pour la haute énergie), et leur budget provient principalement du ministère de l'énergie des Etats-Unis.

Depuis presque vingt ans le groupe est très actif dans la recherche du fameux boson de Higgs. C'est une particule élémentaire « proposée » en 1964 par Peter Higgs, qui n'a cependant pas encore pu être observée de nos jours. Jusqu'à présent, tous les résultats obtenus au cours des expériences de recherche physique sur les particules sont conformes au Modèle Standard[2]. Dans la modèle standard, le boson de Higgs est l'élément responsable donnant une masse non nulle aux autres particules. Pour cela, la recherche du boson de Higgs occupe une place très importante dans la mission du groupe Wisconsin. Le groupe s'intéresse également à la « SUSY » [3]et à l'exotique.

2.2 L'organisation du groupe

Madame Sau Lan Wu dirige ce groupe depuis une vingtaine d'années et est également professeur de physique à l'université du Wisconsin-Madison. Le groupe est composé d'environ vingt-cinq personnes permanentes qui travaillent au CERN, plus quelques personnes basées aux Etats-Unis.

L'ensemble des activités du groupe se partage en 4 parties, et chaque partie est assurée par une section : CS, SUSY, Higgs et Exotic.

---

[1] SLAC : Stanford Linear Accelerator Center

[2] Modèle Standard : Ensemble de théories intégrant toutes les connaissances actuelles sur les particules et les forces fondamentales.

[3] SUSY : SUSY signifie Supersymétrie.

- La section CS : CS, « Computer science » s'occupe, comme son nom l'indique, du système d'information du groupe Wisconsin. Cinq personnes travaillent dans cette section, se chargeant d'administrer les systèmes et le réseau, de développer des applications et de dessiner une architecture afin d'accroître les besoins des utilisateurs.
- La section Higgs : cette section est constituée sept personnes dédiées aux recherches sur le boson de Higgs.
- La section SUSY : Six personnes environ y travaillent, s'intéressant plus particulièrement à la SUSY. La SUSY est une symétrie supposée de la physique des particules pour expliquer une relation forte entre les particules demi-spin (les Fermions) et les particules de spin entier (les Bosons).
- La section Exotic : cette section se charge d'étudier et observer des particules exotiques. Dans le monde physique des particules, ce type de particule a déjà été observé dans différentes expériences, mais son existence est toujours controversée dans ce monde.

Ces trois sections, Higgs, SUSY et Exotic constituent le cœur du métier du groupe, ayant chacune son domaine de recherche précis. Par ailleurs la section CS exerce sa fonction de support pour assurer le besoin en calculs en amont demandé par les autres trois sections.

2.3 Conclusion

Le Wisconsin groupe pratique sa recherche dans la physique des particules depuis une vingtaine d'années. Avant avril 2008 ce groupe était basé sur deux sites et deux expériences, le LHC pour le CERN et le Babar pour le SLAC. Cependant depuis que l'arrêt des besoins en données de l'expérience BaBar, le Wisconsin groupe concentre sa force dans l'expérience du LHC au CERN. Grâce au LHC le groupe pourrait avoir l'occasion d'observer et d'étudier le Boson de Higgs, la SUSY et les Exotiques.

La section CS exerce sa fonction de support, son but étant d'assurer le bon fonctionnement du SI [1] du groupe. Il faut noter que la demande sur la performance des calculs, la demande sur la sécurisation des données et la gestion des travaux demandent encore énormément d'efforts de la section CS pour améliorer leur SI et de mieux répondre aux besoins des utilisateurs.

[1] SI : système d'information

Chapitre 3 Système d'information du groupe

Aujourd'hui dans beaucoup de recherches scientifiques, les nouvelles technologies sont de plus en plus mises à contribution. Notons par exemple, la technologie informatique qui permet de raccourcir le temps de calcul afin d'atteindre plus rapidement le but de la recherche. D'ailleurs, c'est pour cette raison que fut co-inventé le premier ordinateur par Monsieur John Von Neumann, mathématicien et physicien.

Aujourd'hui économiser le temps de recherche n'est plus seulement le rêve de Monsieur John Von Neumann mais celui de tous : Pour le Wisconsin groupe, l'efficacité de leur système d'information joue un rôle essentiel dans leur projet de recherche. Avant d'aborder et analyser en détail leur système d'information il faut que nous ayons une vue globale du système.

3.1 Charge de travail

Lorsque le LHC démarrera, quelques 15 Péta-octets (15 millions de Giga-octets) de données seront produites chaque année. Pour donner un ordre de grandeur, ceci représente une pile de CD haute de 20 kilomètres. Pour traiter ce nombre incroyable de données, le CERN a mis en place une infrastructure virtuelle – Grille informatique (en anglais Grid) et le Wisconsin groupe se situe dans le Tier 3 (le quatrième niveau). Le Tier 0 (le premier niveau) correspond au CERN, le Tier 1 (le deuxième niveau) correspond aux 11 laboratoires de recherches nationaux tels que BNL (Brookhaven National Laboratory) pour les Etats-Unis, CCIN 2P3 (Centre de Calcul de l'institut National de Physique Nucléaire et de Physique des Particules) de Lyon pour la France, etc. Le Tier 2 correspond souvent aux centres de recherche régionaux et il existe 160 centres dits Tier 2 à travers le monde. Enfin, le Tier 3 correspond aux groupes de recherche universitaires (département universitaire), comme le Wisconsin groupe.

Les plusieurs centaines d'instituts, d'universités et des laboratoires se sont regroupés pour traiter les données produites par l'expérience du LHC, car le CERN seul n'arrive pas à tous les traiter : au total environ 100,000 processeurs seront nécessaires pour « digérer » ces données. Les données capturées par les capteurs du LHC sont des données brutes (RAW[1]) : elles ne peuvent donc pas être directement utilisées pour l'analyse et les publications scientifiques. Pour arriver à la publication il est nécessaire de passer quelques étapes supplémentaires. Tout d'abord, il faut convertir ces données brutes en information physique exploitable et les stocker dans les fichiers ESD[2]. Par la suite nous pourrons compacter et filtrer les données de type ESD et obtiendrons

[1] RAW : les données brutes

[2] ESD : Event Summary Data

ainsi un autre type de données AOD[1]. Enfin, une autre étape est nécessaire pour convertir les données AOD en données DPD[2] et CBNT[3]. Les données DPD et CBNT peuvent être lues par la librairie ROOT[4] afin de produire des tableaux ou des graphiques pour servir à des analyses.

Bien entendu le Wisconsin groupe ne va pas participer à toutes ces étapes de conversion : la production des données brutes (RAW) se font au CERN, et la conversion des données RAW à ESD se font sur les Tier 1 et Tier 0 (CERN). Comme le Wisconsin groupe vient des Etats-Unis il est donc attaché à un des 11 laboratoires de Tier 1 : le laboratoire BNL. Il s'occupera d'une partie de la transformation des données AOD en DPD et CBNT. Même si cela ne représente qu'une petite partie des données à traiter, on estime que le groupe va supporter une charge d'au minimum 400 Go des données à convertir par jour. De plus, on doit compter les travaux émis par les chercheurs du groupe qui peuvent atteindre 100 Go par jour. Au total le système d'information du groupe devrait supporter une charge de travail à traiter d'environ 500 Go.

3.2 Infrastructure du système d'information

Le dictionnaire « Larousse », définit « une infrastructure » comme étant l'ensemble des travaux relatifs aux fondations d'un ouvrage par exemple voie ferrée, route, etc. Dans le domaine informatique, le mot infrastructure signifie l'ensemble des éléments de type matériel et des logiciels composant le système d'informations d'une entreprise ou d'une organisation.

Pour la partie logiciel, le système d'informations du groupe est basé sur des technologies comme le Grid[5], Condor[6], ROOT, PROOF [7]et Xrootd[8]. Il utilise la technologie Grid pour partager sa puissance de calcul, le Condor pour gérer des travaux de hautes densités, la librairie ROOT pour l'analyse de données, le Xrootd pour l'accès aux données, etc. Dans ce chapitre, la description de ces éléments n'est pas exhaustive car ils sont très complexes et mériteraient quelques chapitres de plus pour aller plus en profondeur. C'est la raison pour laquelle la deuxième partie de mon mémoire va être consacrée à la description détaillée de ces technologies.

[1] AOD : Analysis Object Data

[2] DPD : Derived Physics Dada

[3] CBNT : « ComBined NTuple », est utilisée les sorties de la reconstruction

[4] ROOT : est une librairie orientée objet pour utiliser dans les analyses des données dans le domaine physique nuclaire

[5] Grid : grille informatique

[6] Condor : est un système spécialisé dans gestion de travail de haute densité

[7] PROOF : Parallel ROOT Facility

[8] Xrootd : eXtended ROOT daemon

Pour la partie matériel, le nombre des serveurs du groupe est très important en comparaison avec les autres instituts de type Tier 3. En général, les instituts de type Tier 3 ont en moyenne 10 à 20 serveurs pour réaliser leurs travaux, alors que le Wisconsin groupe a environ deux cents serveurs en production sur deux sites – côté CERN et côté Université du Wisconsin-Madison aux Etats-Unis.

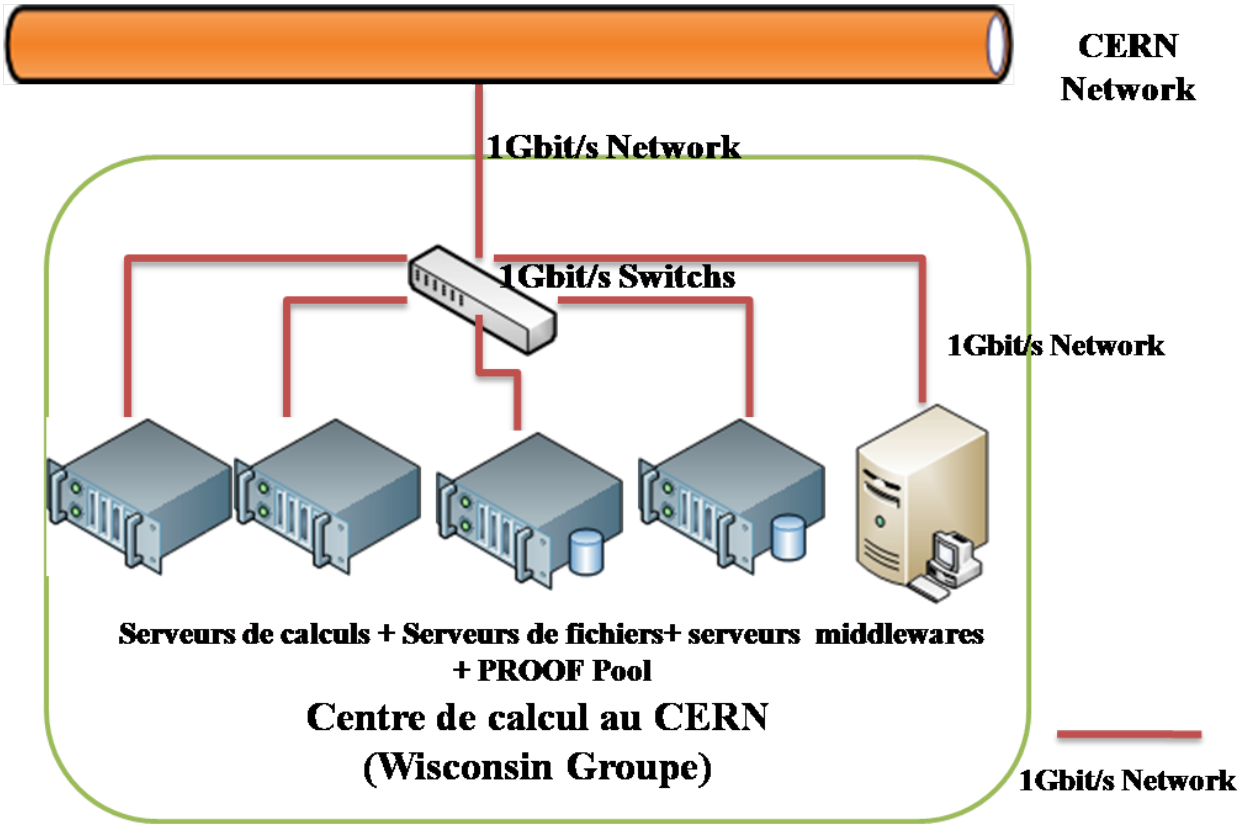

Figure 2 : Infrastructure du système d'information du groupe (côté CERN)

Sur le site du CERN, le Wisconsin groupe possède un local qui permet d'installer une partie de leurs serveurs sur place. Actuellement il y a environ 130 serveurs (plus de 600 CPU) en production. Nous pouvons classifier ces serveurs en trois types différents : serveur de données, serveur de calculs et serveur middleware.

Au CERN, le groupe a 18 serveurs de fichiers qui permettent de sauvegarder les données provenant du LHC. Ils ont tous 8 CPU de 2,0 GHz, 16 Go de mémoire vive et 24 disques de 500 Go montés en RAID[1].

Le groupe a aussi 106 serveurs de calculs, ces serveurs étant très performants. Ce sont des serveurs avec des multiprocesseurs, des mémoires assez importantes mais qui n'ont par contre qu'un seul petit disque dur sur chaque serveur. Lorsqu'un job sur ce type de serveur est terminé,

[1] RAID : qui signifie « matrice redondante de disques indépendants » en anglais « Redundant Array of Independent Disks ». Il permet de répartir des données sur plusieurs disques durs afin d'améliorer soit la tolérance aux pannes, soit la sécurité.

il renvoie le résultat à un serveur de données, et supprime par la suite le job ainsi que le résultat sur son local.

Après avoir présenté les serveurs de fichiers et les serveurs de calculs, il reste à présenter les serveurs middleware. Nous pouvons les retrouver sous plusieurs noms comme « serveur Xrootd » ou encore « serveur Condor ». La mise en place des serveurs Xrootd a pour but de gérer l'accès aux données et des serveurs Condor pour gérer des travaux. Concernant le réseau, le groupe n'a obtenu qu'une ligne de 1 Gbit/s par seconde pour ces nombres serveurs.

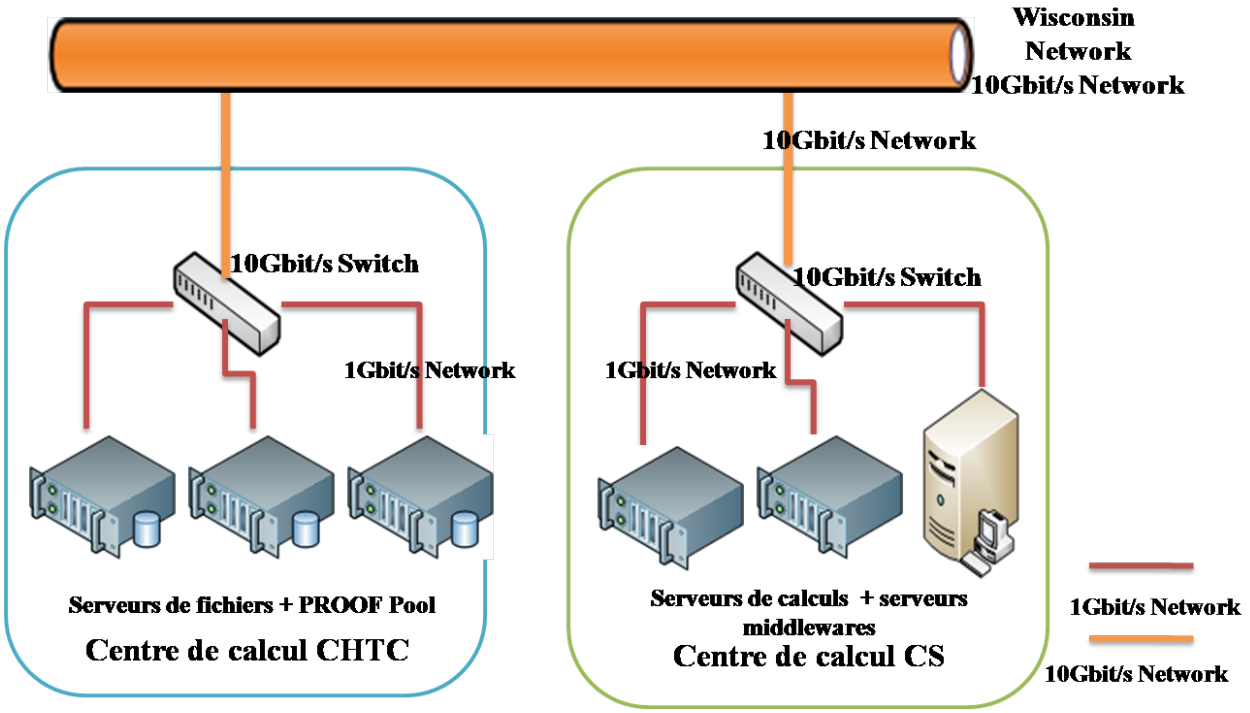

Figure 3 : Infrastructure du système d'information du groupe (côté Université du Wisconsin)

Une autre partie de leurs serveurs est installée sur le network de l'université qui sont connectés sur le réseau de celle-ci. La bande passante du réseau universitaire est de 10 Gbit/s. Chaque serveur a une carte réseau de 1 Gbit/s reliée par des switch de 10 Gbit/s (voir Figure 3).

Les serveurs du groupe sont répartis sur deux locaux différents du centre de calcul de l'université. Dans le local CHTC (The Center for High Throughput Computing), 50 serveurs de fichiers appartiennent au groupe. Ils ont une configuration hardware identique : 8 processeurs de 2.66 GHz, 16 Go de mémoire vive et 8 disques de 750 Go monté en RAID. Par ailleurs, 40 serveurs sont installés dans le local CS (Computer Sciences), dont une quarantaine de serveurs de calculs et le restant pour les serveurs de middleware.

Le groupe a aussi mis en place un « PROOF Pool » sur chaque site dédié à la transformation des fichiers DPD et CBNT en fichiers graphiques ou en tableaux pour la publication ou la rédaction de thèses. Chaque « PROOF Pool » contient deux serveurs et a la

même configuration que des serveurs de fichiers de chaque site. Ces serveurs stockent une grande partie des fichiers DPD et CBNT, et une petite partie des fichiers DPD et CBNT qui sont encore stockés sur les serveurs de fichiers.

Sur les deux sites, il y a plus de 200 serveurs en production dont environ 100 serveurs sur chaque site. Nous pouvons observer qu'il y a une grande différence de vitesse de bande passante entre deux sites. En fait, les installations de ces deux sites présentent aussi une légère différence.

3.3 Architecture du système d'information

Un système d'information permet de traiter, d'acheminer et de sécuriser des données. Un système bien conçu peut faciliter la prise de décision et obtenir des avantages concurrentiels importants et durables. Une architecture du système d'information décrit sa structuration en termes de composants et d'organisation de ses fonctions.

Les 200 serveurs du groupe sont répartis sur deux sites. L'architecture de leur système d'information se divise en deux parties : celle située sur le site du CERN et celle implantée à l'université du Wisconsin-Madison. Nous pouvons constater que leur architecture est similaire dans les grandes lignes tout en présentant toutefois des différences. Par la suite, l'ensemble de l'architecture du système d'information du groupe va être présenté et nous évoquerons également les disparités entre ces deux sites.

Dans la section précédente, j'ai déjà décrit brièvement l'architecture du système d'information du groupe. Si nous regardons séparément le système d'information de ces deux sites, nous pouvons noter que chaque site est formé par une architecture indépendante, mais cependant très similaire.

Les deux architectures sont toutes les deux construites en quatre blocs :

- Le bloc serveurs middleware qui permet d'assurer l'intermédiaire entre les applications et le transport des données par les réseaux ; par exemple le serveur Condor master, server Xrootd, etc;
- Le bloc serveurs de calculs (nous pouvons aussi les nommer serveurs applicatifs ou serveurs d'applications) qui permet d'exécuter les travaux demandés par des clients ;
- Le bloc serveurs de fichiers qui permet de mettre des fichiers à disposition à travers le réseau ;
- Le bloc « PROOF Pool » qui utilise la bibliothèque « ROOT » pour une transformation des fichiers DPD en fichiers graphiques ou tableaux. Pour ma part, je classerai donc ce bloc dans le bloc serveurs de calculs.

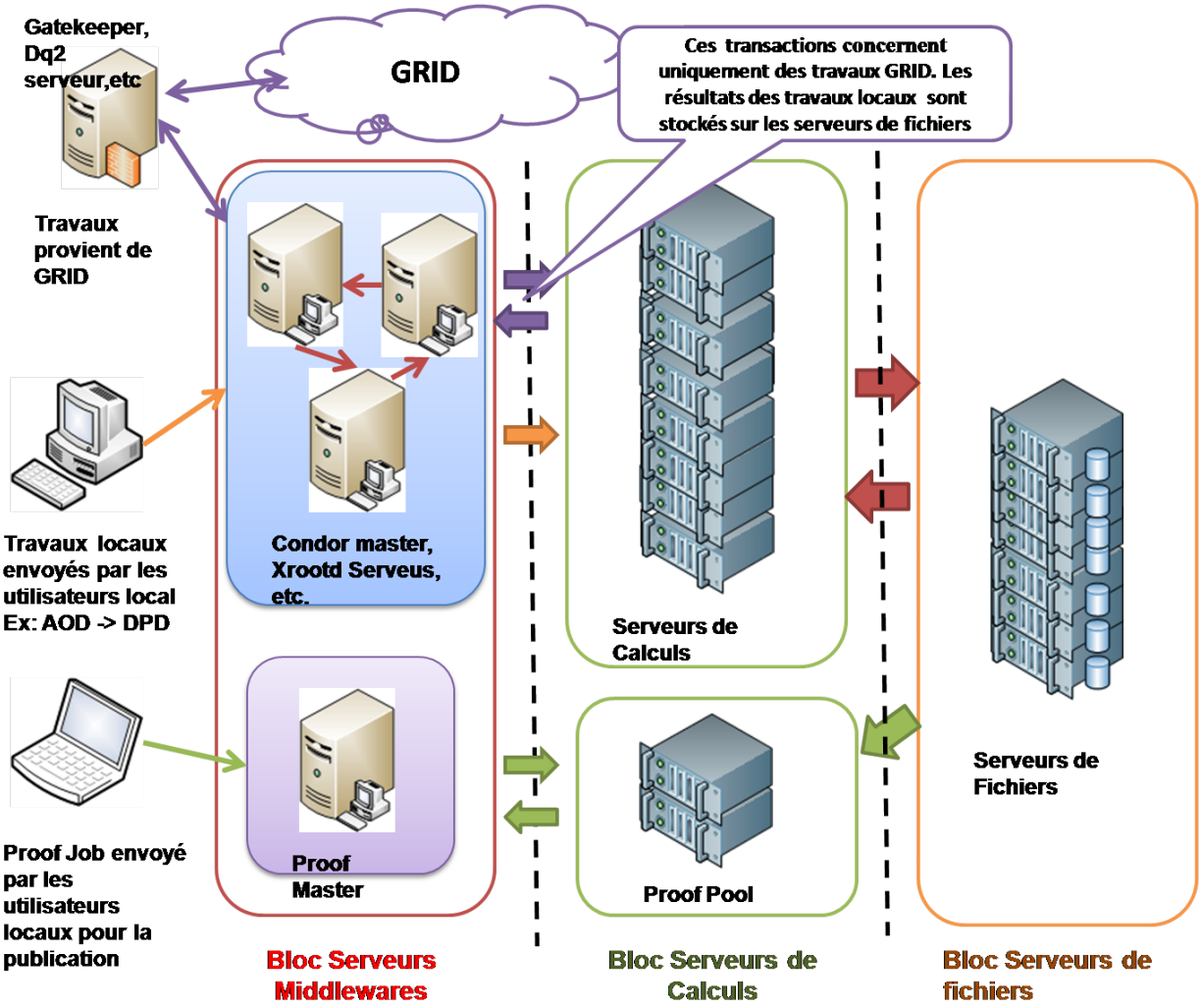

Figure 4 : Architecture du système d'information du groupe (côté Université du Wisconsin)

La figure 4 est un court résumé de l'architecture du SI de l'université du Wisconsin-Madison. Au CERN, son architecture est très similaire ; par contre il faut noter qu'il ne reçoit pas les travaux provenant du Grid. Sur cette figure, nous pouvons constater que l'ensemble du SI reçoit trois types de jobs : les jobs provenant de Grid, les travaux locaux et les PROOF job. Nous pouvons également constater que les frontières entre chaque bloc sont claires et chaque bloc a son propre rôle bien défini et des applications associées.

3.4 Conclusion

Dans ce chapitre, nous avons pris connaissance de l'infrastructure et de l'architecture du SI du groupe. Les travaux qui doivent être traités par le système ont aussi été abordés. Il n'est pas très difficile de constater que le Wisconsin groupe a un nombre de CPU assez conséquent par rapport aux autres Tier 3 groupes. Lorsque le LHC commencera à produire des données, il y aura une quantité de données immenses à traiter par le SI. La performance du système d'information influencera donc fortement sur l'efficacité du groupe.

Chapitre 4 La motivation du projet

Dans les chapitres précédents, j'ai présenté le CERN, le Wisconsin groupe, son système d'information et sa charge de travail. A travers ces points, nous avons désormais une vision globale du contexte du projet. Le métier du groupe est la recherche physique fondamentale. Aujourd'hui, pour effectuer ce type de recherche, il est indispensable de se faire aider par des nouvelles technologies pour réduire le temps de calcul afin d'accélérer le cycle de recherche. Pour un groupe de chercheurs scientifiques comme le Wisconsin groupe, la qualité et la performance de son système d'information ont donc des effets directs sur son activité. De plus avec le redémarrage du LHC, une augmentation des données à traiter est inévitable. Ce sont ces facteurs internes et externes qui incitent le groupe à revoir son système d'information à un niveau plus haut pour faire face aux difficultés qu'il rencontrera prochainement. Hormis la qualité et la performance que le groupe veut obtenir de son SI, les points suivants sont aussi les points clés de la motivation du projet.

4.1 Extensibilité

L'extensibilité, en Anglais « Scalability » (voir Webgraphie Scalability), exprime la capacité d'un système à faire face à des charges d'utilisation variables, la consommation de ressources étant la plus linéaire possible. Ce terme est aussi utilisé pour désigner les systèmes dont nous pouvons étendre les capacités par l'ajout de modules ou par des mises à jour.

Au redémarrage du LHC, la charge des données à traiter va beaucoup augmenter et garder une souplesse de son système d'information compte beaucoup pour le groupe. En effet, une architecture plus souple permet d'intégrer des nouveaux matériels sans difficulté, et ce rajout de matériels permet d'accroître sa capacité de calcul malgré une charge de travail accrue.

4.2 Évolutivité

Si le terme « extensibilité » concerne plutôt l'architecture matérielle, le terme « évolutivité » concerne l'architecture logicielle. Le groupe veut rendre son SI plus modulaire, car une architecture modulaire permet d'intégrer ou de développer de nouvelles fonctions selon les besoins des utilisateurs sans engendrer de coûts importants.

4.3 Fiabilité

La Fiabilité se mesure par une probabilité : c'est la probabilité qu'un système informatique (matériel et/ou logiciel) fonctionne sans tomber en panne pendant un certain temps.

Le groupe veut fiabiliser son système d'information : nous travaillerons donc sur les trois points suivants : la disponibilité, la robustesse et la tolérance aux pannes.

- La disponibilité désigne le ratio de temps pendant lequel il est en état de fonctionner correctement sur une période de temps donnée. Pour la calculer, il suffit de prendre MTBF [1] et le diviser par la somme de MTBF et MTTR [2] : $\text{Disponibilité} = \frac{\text{MTBF}}{(\text{MTBF} + \text{MTTR})}$
- La robustesse d'un système désigne sa capacité à ne pas « planter » et « perdre ou corrompre » des données ou des messages lorsqu'il est soumis à des sollicitations inhabituelles ;
- La tolérance aux pannes désigne l'aptitude d'un système à fonctionner malgré ses défaillances éventuelles.

4.4 Rentabilité

La rentabilité désigne le rapport entre le résultat obtenu et les ressources employées pour l'obtenir. Le groupe voudrait que leur SI soit plus rentable à travers ce projet. Nous pouvons alors se poser la question suivante : le Wisconsin groupe étant un groupe de recherche universitaire sans but lucratif, comment peut-on utiliser ce mot pour un tel groupe ? En effet le mot rentabilité est tout à fait relatif, le groupe veut mieux gérer son budget d'informatique pour l'achat de matériel afin d'optimiser ses dépenses d'informatique.

4.5 Visibilité

La gestion des fichiers est toujours un problème préoccupant du système. Avec des milliers et des milliers de fichiers comment peut-on les gérer et les visualiser ? Jusqu'à présent le groupe n'a pas encore trouvé de solution satisfaisante. A travers ce projet une solution doit être proposée pour répondre à ce besoin.

4.6 Conclusion

Pour mieux préparer l'immense travail à venir pour son système d'information, le groupe me confie la tâche, en tant que chef de projet, d'améliorer la qualité de son système d'information. Pour cela, je vais focaliser mes efforts vers la construction d'un système performant, fiable et extensible tout en respectant les contraintes budgétaires.

[1] MTBF : est le temps moyen écoulé entre deux pannes, bien entendu y compris le temps de réparation. (Mean Time Between Failures)

[2] MTTR : est le temps mis pour réparer un système en panne. (Mean Time To Repair)

Conclusion de la première partie

Dans cette partie nous avons fait connaissance avec le CERN et le Wisconsin groupe. J'ai présenté également leurs activités et missions. Cela fait déjà plus de 10 ans que le Wisconsin groupe participe à l'expérience d'ATLAS au CERN. Pour mieux poursuivre leur activité, l'utilisation de nouvelles technologies est indispensable pour le groupe. Le système d'information du groupe est construit sur une base de 200 serveurs et des technologies de pointes : Condor, Grid, PROOF et Xrootd.

Le démarrage du LHC au CERN provoquera une hausse significative des données à traiter par son système d'information. Pour mieux préparer cette réalité, le groupe décide de lancer un projet qui permet d'améliorer son système d'information. Ce projet prévoit de travailler sur les points suivants : la performance, l'évolutivité, l'extensibilité, la fiabilité et la rentabilité.

Le Wisconsin groupe est l'un des 159 instituts travaillant sur l'expérience ATLAS. Bien que les groupes travaillent ensemble sur des projets en commun afin de faire avancer leur recherche, on peut noter une certaine tendance à la concurrence. Le groupe proposant le meilleur projet prendra probablement une longueur d'avance sur les autres groupes dans son domaine de recherche.

Néanmoins les technologies déployées dans ce projet sont assez récentes et peu utilisées par le grand public. De plus, elles sont assez complexes. Pour cette raison, ces technologies vont être étudiées et analysées dans la partie suivante afin de mieux les comprendre. Quelles sont les relations entre ces différentes technologies ? Et quelles sont leurs principales fonctionnalités ?

Deuxième Partie

Les technologies et leurs principales fonctionnalités

Comme mentionné dans la partie précédente, l'architecture du système d'information du groupe Wisconsin est relativement complexe : Les technologies employées notamment dans son système mais aussi par la répartition de clusters sur deux sites (à l'université du Wisconsin-Madison et au CERN) et l'hétérogénéité de la bande passante entre les deux sites.

Ce sont néanmoins des technologies très intéressantes. Pour le moment elles sont utilisées plutôt dans le domaine de la recherche scientifique mais on peut espérer les voir utilisées un jour dans d'autres domaines que le domaine de la recherche. Par exemple on pourrait employer le Grid pour partager la puissance informatique et la capacité de mémoire sur internet afin d'accroitre la capacité de traitement. Et on peut imaginer que cette technologie pourrait être utilisée dans les entreprises qui ont de temps en temps des pics de travail : ces entreprises pourraient utiliser le Grid pour partager leurs ressources sans avoir besoin d'acquérir de nouveaux matériels pour accroitre leur capacité de calcul.

Il est vrai que chaque technologie utilisée dans ce projet a ses propres particularités. Pour mieux comprendre, il est indispensable de présenter ces technologies avec leurs fonctionnalités et les articulations qui les lient entre elles. Dans les chapitres suivants, les technologies telles que Condor, Grid, ROOT, PROOF et Xrootd vont être présentées.

Chapitre 5 Condor

Condor est un système logiciel de traitement par lot spécialisé pour l'exécution d'applications de calcul intensif. Il a été développé par une équipe de l'université du Wisconsin-Madison. Ce projet a commencé en 1988 sous la direction du professeur Livny. Ce système peut être utilisé dans plusieurs OS[1] notamment les plus courants comme Microsoft Windows, Mac OS X, Linux, Unix, AIX, FreeBSD et Solaris.

Le but de ce projet est de développer un système visant à satisfaire les besoins en calcul en termes de volume plutôt qu'en termes de vitesse. On qualifie ce type de système logiciel comme « High-Throughput Computing » (HTC[2]) et il est différent d'un autre type de système logiciel « High-Performance Computing » (HPC[3]).

Le HPC qui se traduit en français par Calcul Haute Performance, se compose souvent d'un ou plusieurs supercalculateurs interconnectés pour atteindre les plus hautes performances possibles. Cependant l'investissement d'un supercalculateur est très coûteux, et sa durée d'exploitation et de production est courte.

C'est la raison pour laquelle, le système Condor est particulièrement intéressant, car il consiste à utiliser du temps d'inactivé des machines interconnectées pour effectuer des calculs distribués. Le gros avantage de ce type de système est le coût d'exploitation faible et la simplification d'accès aux ressources.

5.1 Caractéristiques

Comme mentionné plus haut, le projet Condor a commencé à la fin des années 80 et a toujours été maintenu jusqu'à ce jour. Aujourd'hui il est utilisé par des centaines d'institutions attachées à l'industrie, l'Université et l'administration. Condor est un système très intéressant qui présente de réels intérêts pour ses utilisateurs : il suffit de prendre connaissance de ses caractéristiques pour s'en convaincre :

Les caractéristiques de système Condor peuvent donc être résumées comme suit :

- C'est un système de traitement par lot ;
- Il gère la propriété d'une machine privée ;
- Il permet de migrer des processus d'une machine à d'autre ;

[1] OS : Operating system

[2] HTC : est un système capable de fournir une puissance de calcul convenable sur une longue durée

[3] HPC : est un système capable de fournir une puissance de calcul performante sur un court laps de temps

- Il permet de surveiller l'état de charge des machines et soumet des tâches sur des machines non utilisées ;
- Les machines participantes peuvent avoir le choix si des espaces de fichiers partagés et des espaces d'utilisateur commun sont nécessaires ou pas.

Après avoir pris connaissance des caractéristiques du système Condor, une illustration du fonctionnement de ce système sur un réseau va être présentée dans la section suivante.

5.2 Le fonctionnement du système Condor

Le système Condor contient 5 démons appelés « Collector », « Negotiator », « Master », « Schedd » et « Startd ». Nous pouvons trouver trois types de machine dans l'architecture d'un parc Condor qui sont le « Gestionnaire central », l'« Exécutant » et le « Soumissionnaire ».

Le démon « Master » est un service de base pour le Condor et il doit être démarré sur tous les types de machine. Une machine qui tourne un démon « Startd » est une machine exécutante. Ce type de machine publie ses spécifications et sa disponibilité au « Gestionnaire central », et exécute les travaux sur demande.

Lorsqu'un démon « Schedd » fonctionne sur une machine, cette machine est un « Soumissionnaire ». Le rôle de ce type de machine est de présenter au « Gestionnaire central » les tâches à accomplir et de soumettre les travaux aux « Exécutants ».

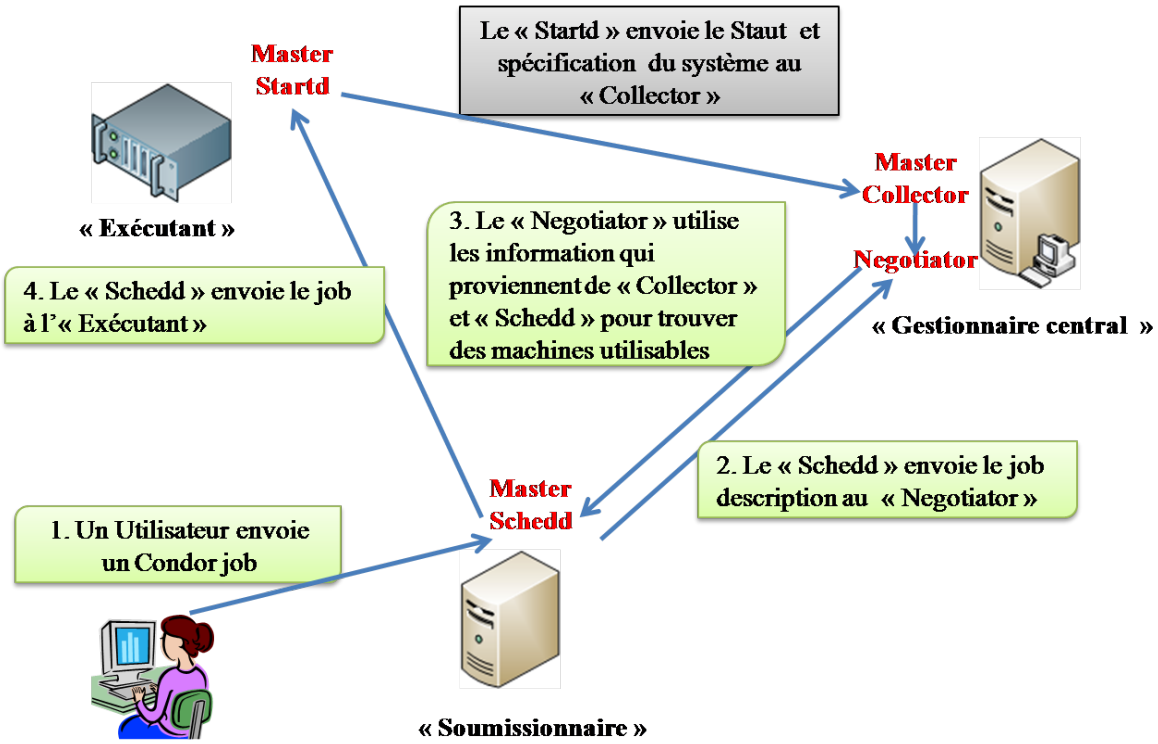

Figure 5 : Illustration d'organisation du système Condor sur un réseau de clusters

Sur une machine de type « Gestionnaire central », nous trouvons le démon « Collector » et « Negotiator » qui tournent sur la machine. Le démon « Collector » permet de collecter des

informations sur des machines exécutantes. Le démon « Negotiator » lui, permet de mettre en relation les demandes et les disponibilités afin d'organiser les exécutions des processus.

La figure 5 illustre le fonctionnement du système Condor dans la pratique. Pour commencer, il faut savoir qu'une machine de type « Exécutant » peut présenter plusieurs états, par exemple :

- Attribuée : un processus Condor est attribué et va démarrer ;
- Libre : il n'y a aucun activité sur cette machine ;
- Préemption : un processus Condor sauvegarde son état et va quitter le système ;
- Suspendue : un processus est dans un état STOP temporairement ;
- Utilisée : un processus Condor est en exécution ;

Lorsqu'une machine « Exécutante » change d'état, il envoie les informations au « Gestionnaire central » qui les collecte. Quand un « Soumissionnaire » reçoit un Condor job envoyé par un utilisateur, il contacte le « Gestionnaire central » en lui soumettant le job description. Par la suite, le « Gestionnaire central » combine le job description et l'information formée par le « Collector » pour trouver des machines utilisables et informer le « Soumissionnaire ». Pour la dernière étape le « Soumissionnaire » envoie le Condor job à la machine concernée pour effectuer l'exécution.

C'est un exemple d'organisation du système Condor. Il est à noter par ailleurs qu'un serveur peut être à la fois « Soumissionnaire », « Exécutant » et « Gestionnaire central » . Tout dépend de la manière dont l'administrateur du système configure les machines.

5.3 Les Outils d'administration

Le système Condor fournit environ trente commandes pour la gestion de son environnement. Dans cette trentaine de commandes nous pouvons trouver par exemple les commandes suivantes :

- Condor_hold : elle permet d'enlever les travaux dans une file d'attente et les mettre de côté ;
- Condor_prio : cette commande permet de changer la priorité des travaux en attente de leur tour d'exécution ;
- Condor_q : commande affichant les informations concernant la file d'attente des travaux ;
- Condor_submit : Cette commande envoie des travaux dans la file d'attente des travaux Condor ;
- Condor_status : affiche les statuts d'environnement Condor ;

- Condor_stats : permet d'afficher les informations historiques d'environnement du système Condor.

Dans les paragraphes précédents, j'ai présenté quelques commandes fournies par le système Condor. Ces commandes sont indispensables pour la gestion d'un système Condor. Bien entendu, il existe d'autres commandes (dont la liste est fort longue) très utiles pour gérer cet environnement.

5.4 Conclusion

Le Condor est un système logiciel développé dans la même université que le Wisconsin groupe. Ce système spécialisé dans la gestion concurrentielle de traitement et la gestion de ressource d'un système permet de traiter un nombre très important de tâches. Dans le Wisconsin groupe, nous utilisons le Condor pour gérer les travaux émis par les utilisateurs du groupe. Par ailleurs, le Condor est également utilisé par l'intergiciel (middleware) Grid, qui fera l'objet du chapitre 8.

Les caractéristiques spécifiques de Condor ont permis à ce système de s'intégrer au projet Grid. Dans le chapitre suivant une autre technologie utilisée dans notre système va être présentée : Il s'agit de « Xrootd ». Dans notre système, le « Xrootd » et le « Condor » s'utilisent mutuellement pour la réalisation des travaux émis par les utilisateurs locaux. Cependant, les deux font parties du projet Grid que nous utilisons pour partager notre puissance de calculs dans l'infrastructure de Grid.

Chapitre 6 Xrootd

Le volume de données généré par l'expérience physique expérimentale est de plus en plus important. Aujourd'hui, les données peuvent atteindre une taille entre cinq à dix Péta-octets par an. Face à cette taille de données, bien répartir des données dans un système de fichiers distribué et satisfaire le besoin d'un accès aux données rapide sont devenus des besoins majors pour gérer un système fichier distribué.

La venue de « Xrootd » doit répondre à ces besoins. Le projet de développement « Xrootd » a commencé début 2003 à l'occasion de l'abandon du format « Objectivity » au profit d'un autre format « ROOT ». Ce projet est réalisé par SLAC pour l'expérience BaBar. Il appartient à un service qui permet d'effectuer un accès aux données à haute performance et de répartir dynamiquement des fichiers dans un système de fichiers distribué.

6.1 Caractéristiques

À travers les paragraphes précédents, nous avons évoqué la naissance du projet « Xrootd ». Dans cette section, la présentation sur les caractéristiques de « Xrootd » nous permet de comprendre quels sont les avantages à utiliser le « Xrootd » en particulier dans un système comme celui de groupe Wisconsin.

Tout d'abord, nous pouvons citer sa compatibilité car le « Xrootd » est compatible avec des fichiers format « ROOT ». De plus il fait désormais partie des distributions standard de ROOT. En revanche le « Xrootd » n'offre pas uniquement ses services au format ROOT car bien au contraire il permet d'ouvrir et d'accéder à n'importe quel type de fichiers en dehors de l'infrastructure ROOT.

Nous pouvons aussi citer sa flexibilité, car il permet d'être compatible avec les autres services fichiers. Le Xrootd a par exemple un composant qui s'appelle « Xrootdfs ». C'est un système de fichiers POSIX désigné pour une architecture clusters Xrootd. Le « Xrootdfs » peut fonctionner avec des composants de Grid tels que GridFTP [1] pour effectuer des opérations entrée-sortie.

Il est évident qu'un seul serveur fichier ne pourra pas satisfaire le besoin en volume de données. Imposé par cette contrainte et le nombre d'accès aux données du côté client, s'orienter vers une architecture distribuée est devenu presque une obligation. Le Xrootd est un service qui permet de mettre en place rapidement un système de fichiers distribué car un système de fichiers construit sur la base de Xrootd est facile à maintenir et le rajout ou de retrait d'un serveur est très

[1] GridFTP : est un protocole de transfert de données performant et sécurisé

simple : il suffit de changer quelques lignes de sa configuration et redémarrer le démon Xrootd. La mise en place d'une architecture des serveurs Xrootd pourra donc se faire rapidement.

Dans un environnement serveurs fichiers distribués, plus le nombre de serveurs fichiers est important plus les problèmes potentiels liés à l'hardware et au réseau sont nombreux. L'équilibre de charge dynamique du « Xrootd » nous permet de continuer de travailler normalement lorsqu'un serveur de fichiers trombe en panne et cette perte est, de plus, totalement transparente pour les utilisateurs.

Les tailles de la requête et de la réponse transmises par Xrootd sont extrêmement petites. Ceci permet au Xrootd de fournir une très bonne performance par rapport au NFS[1]. Nous pouvons observer une différence jusqu'à soixante-quinze pourcent d'utilisation de CPU de moins et une rapidité trois fois plus élevée au niveau transaction réseau entre les deux systèmes.

6.2 Equilibre de charge dynamique

Les données produites par chaque collision de LHC sont traitées et stockées dans un répertoire, appelé aussi « DataSet »[2]. Un DataSet peut contenir plusieurs centaines de fichiers. Lorsqu'un nouveau DataSet arrive sur notre site, le service Xrootd se charge de créer un répertoire sur chaque serveur de données et de dispatcher ces données de la façon la plus équilibrée possible sur chacun des serveurs.

Cet équilibrage de charge est très bénéfique pour l'ensemble du système. Tout d'abord cette distribution des données bien équilibrée permet une meilleure répartition de charge du travail pour les serveurs de données et une meilleure utilisation de la ressource, comme le processeur, la mémoire, le disque et le réseau.

Par ailleurs, cet équilibrage de charge représente un autre avantage : il permet de relever le niveau de robustesse du système. Par exemple, la perte d'un serveur de fichiers qui entraînerait la perte d'une partie des données. Grâce à cette démarche, plus le nombre de serveurs fichiers est important plus la perte des fichiers un DataSet est limitée. La figure 6 illustre comment le Xrootd répartit les données provenant de l'extérieurs.

[1] NFS : Network File system.

[2] DataSet : est une collection de données.

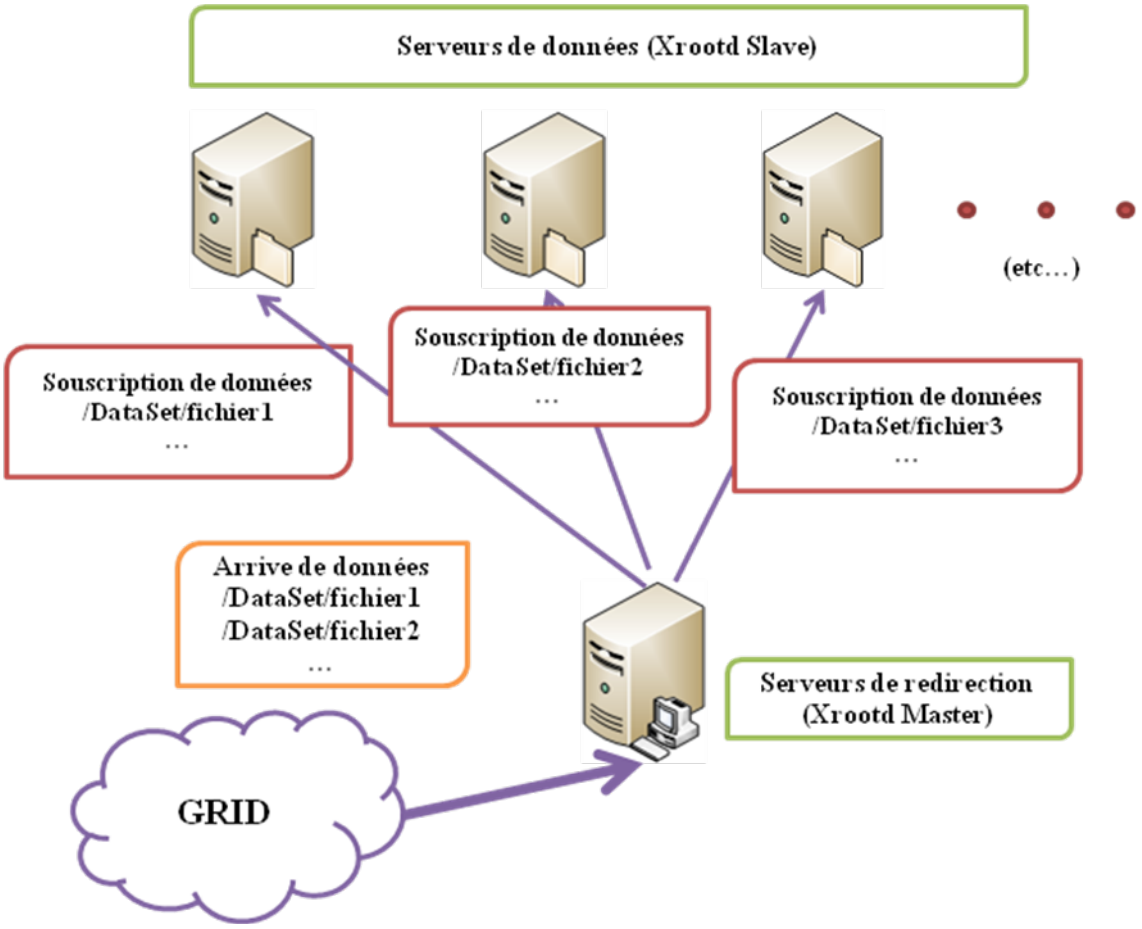

Figure 6 : La répartition de charge dynamique Xrootd

6.3 Fonctionnement de Xrootd et son extensibilité

Dans la section précédente, nous avons appris que la mise en place d'un système distribué Xrootd représente des avantages pour l'ensemble du système. Dans cette section nous allons évoquer le fonctionnement du Xrootd et son extensibilité.

La figure 7 illustre la mise en place d'une architecture Xrootd. Dans le cadre du groupe Wisconsin, nous avons une architecture à deux niveaux : Les serveurs « Xrootd Slave » sont reliés directement au « Xrootd Master ». Il est envisageable aussi de construire une architecture Xrootd à plusieurs niveaux comme celle représentée dans la figure 7.

Une architecture Xrootd se construit sur une base d'arbre de 64 nœuds, c'est-à-dire qu'un serveur Xrootd de niveau supérieur (Master ou Sub-Master) peut attacher un maximum de 64 serveurs Xrootd du niveau inférieur. Si ce nombre dépassait plus de 64 nœuds, il faudrait alors ajouter un autre serveur Xrootd master ou Sub-Master. Par exemple, dans une architecture de 3 niveaux, le nombre de serveurs esclaves peut atteindre 4096, alors qu'une architecture de 4 niveaux peut comporter 262144 serveurs fichiers.

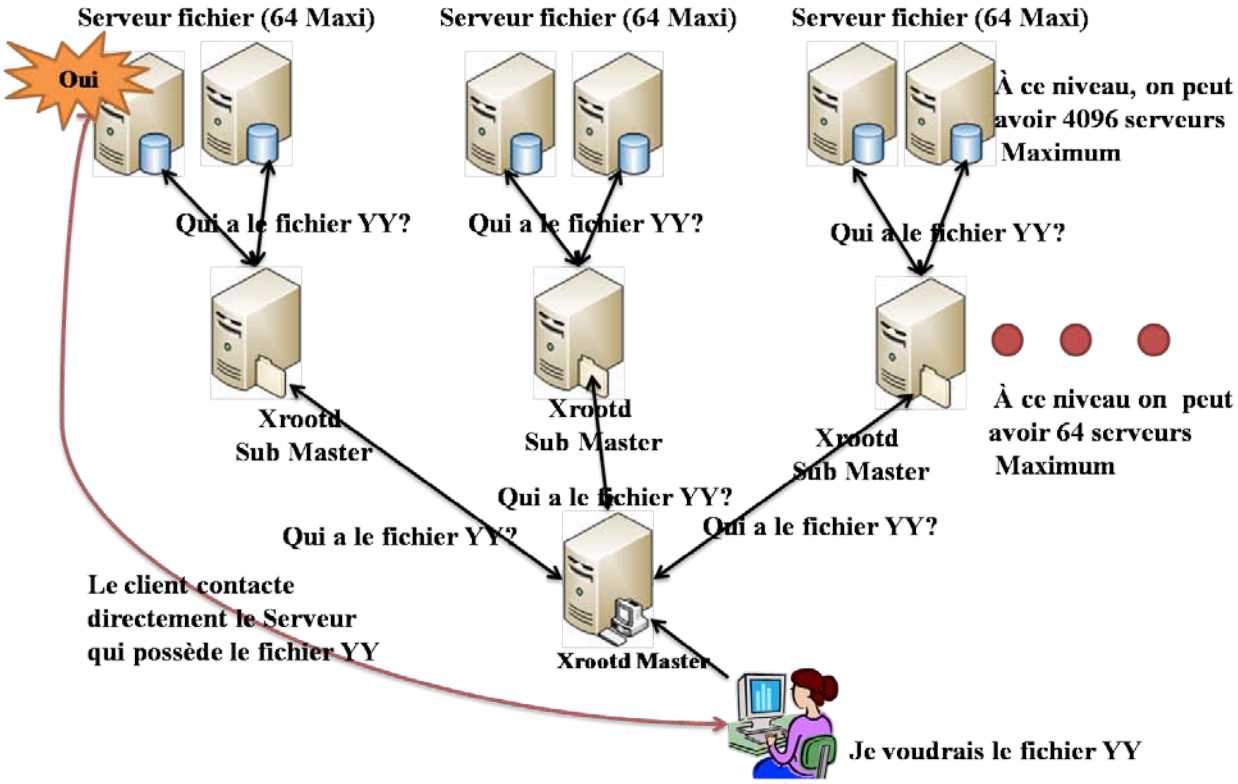

Figure 7 : Xrootd architecture

Alors comment le serveur Xrootd peut retrouver un fichier qui est noyé dans l'un des serveurs Srootd ? C'est très simple ! Lorsqu'un utilisateur demande d'accéder à un fichier, le client émet une requête au Xrootd Master, le Xrootd master reçoit la demande et envoie cette demande à son niveau inférieur soit aux « Xrootd sub-master » soit aux « Xrootd slave » en demandant « Qui a le fichier YY ? ». Cette demande va être envoyée d'un niveau à l'autre (si on a plusieurs niveaux), jusqu'au moment où un serveur répond oui « j'ai ». Enfin le serveur établit une connexion avec le client, et lui délivre les fichiers qu'il a demandés.

6.4 Conclusion

A travers ce chapitre, nous avons présenté les caractéristiques de Xrootd, son fonctionnement et son excellente extensibilité. Le Xrootd présente de nombreux atouts pour être intégré dans notre système. Parmi son extensibilité, sa robustesse et sa compatibilité avec le Grid, il peut aussi s'associer avec les autres systèmes de stockage de grand volume (ex : CASTOR [1]ou HPSS [2]). C'est la raison pour laquelle, aujourd'hui le service Xrootd est intégré dans la distribution de « Framework ROOT ».

[1] CASTOR: Cern Advanced STORage manager.

[2] HPSS : High Performance Storage System

Chapitre 7 ROOT et PROOF

Les données produites par l'expérience LHC doivent passer plusieurs étapes pour arriver à la publication. Pour cela nous utilisons différentes technologies pour assurer chaque étape de la transformation des données. Dans ce chapitre, on va s'intéresser au Framework ROOT et son extension, outil permettant aux physiciens d'effectuer des analyses finales.

7.1 Présentation de ROOT

Le ROOT est un groupe de logiciels orientés objet qui permet d'effectuer des analyses afin de résoudre des problèmes dans le domaine de la physique de haute-énergie. Le développement du Framework ROOT a commencé en 1995 dans le cadre de l'expérience NA49. Les données produites peuvent atteindre 10 Terabytes par jour. C'est ce contexte qui a permis ce projet de développement d'un nouveau groupe de logiciels pouvant analyser et interpréter toutes ces données.

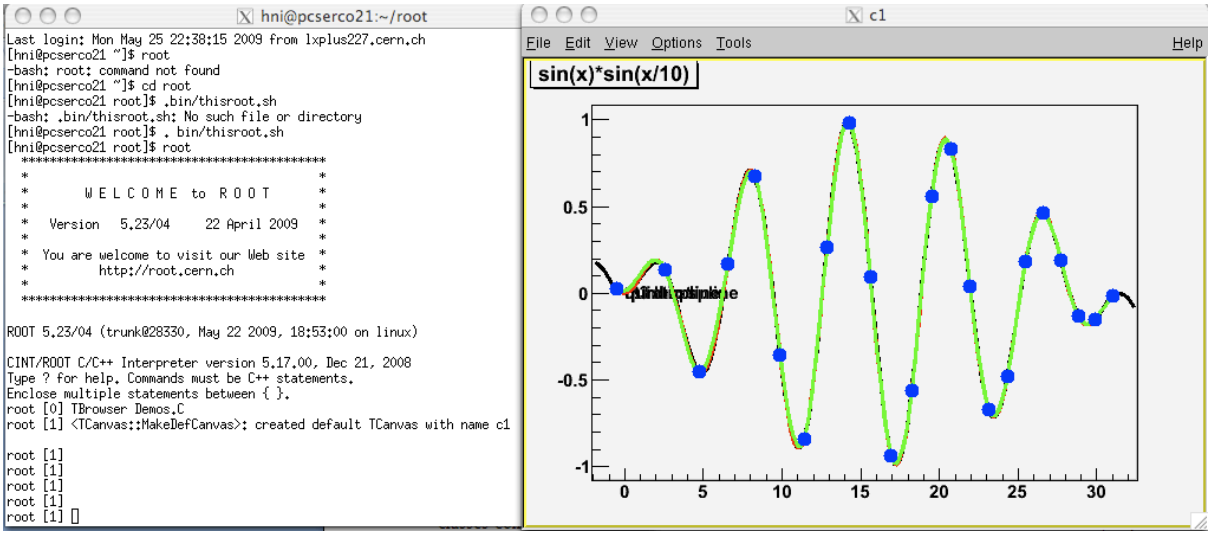

Figure 8 : Exemple une interface ROOT et son interface graphique

Aujourd'hui, Ce projet a franchi sa quatorzième année de fonctionnement. On trouve environ 1200 classes dans ce groupe de logiciels et ces 1200 classes construisent les 19 modules de ROOT. C'est un outil d'analyses, de présentations et de gestion des données. Comme par exemple, le module « Mathematical » qui offre des classes mathématiques et des fonctions statistiques nécessaires pour des chercheurs effectuant des analyses et des calculs. Par ailleurs le module « Algèbre Linaire » est capable d'aider les chercheurs à résoudre des équations extrêmement complexes.

Parmi les modules « Mathematical » et « Algèbre Linaire », le Framework ROOT intègre un interpréteur C++. Il peut aussi exécuter des codes C++. Le module « Graphique » permet non seulement aux utilisateurs d'avoir une interface utilisateur graphique mais de plus il permet

d'interpréter des données en fichier graphique. Ces fichiers peuvent être sous forme de 2D, 3D ou de tableau.

Le ROOT contient d'autres modules tout aussi intéressants. L'extension représentant le plus d'intérêts pour notre système s'appelle « PROOF ». Elle permet d'effectuer des calculs en mode parallèle.

7.2 Présentation de PROOF

Le grand défi pour notre système est de chercher un meilleur équilibre entre la performance, la robustesse, la fiabilité, l'extensibilité et l'évolutivité. Le redémarrage du LHC nous ramènera une immense quantité de données à analyser. La venue du PROOF, qui est une extension de la bibliothèque ROOT, utilise le parallélisme pour répondre aux besoins de performance.

Le PROOF peut être utilisé sous deux formes d'architecture : la première est basée sur une architecture Client Serveur à trois niveaux. Lorsqu'un client soumet des travaux au système, ces travaux vont être gérés par le « PROOF Master », middleware permettant de faire la liaison entre le client et le serveur calculs. Il analyse les demandes du client et distribue les travaux correspondants à des serveurs calculs PROOF. Le PROOF master collecte les résultats envoyés par les serveurs calculs PROOF. Une fois les travaux finis, le PROOF master rassemble les résultats et fournit un résultat final.

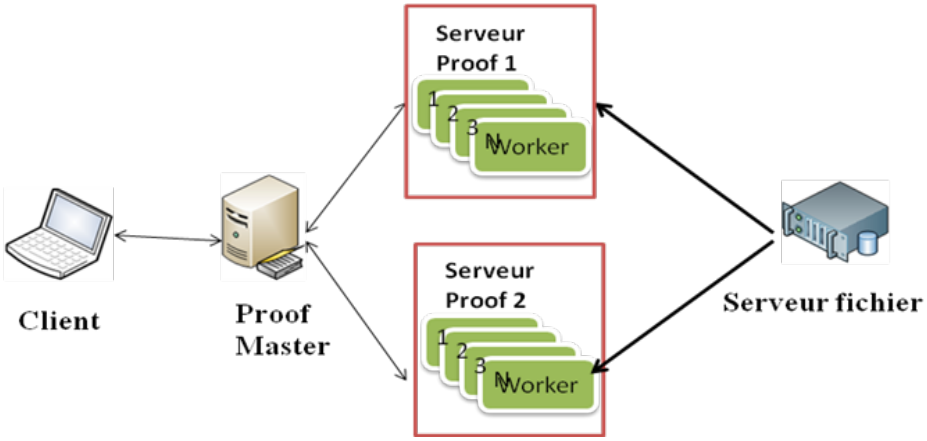

Figure 9 : PROOF Architecture

La seconde architecture, appelée PROOF-Lite, est, comme son nom l'indique, un PROOF allégé. La différence entre PROOF et PROOF-Lite se situe dans le fait que le PROOF-lite est dédié principalement à un PC de bureau ou à un portable multi-core. Le PROOF master est, lui, intégré au côté client, le client contrôlant tout.

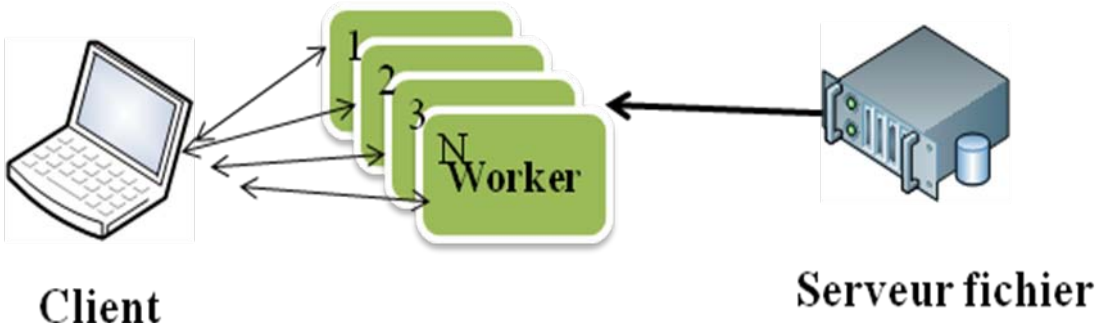

Figure 10 : PROOF-Lite

Le PROOF permet d'effectuer des calculs parallèles. L'idée de cette démarche est d'utiliser un minimum de temps pour effectuer les travaux demandés par le client. Pour économiser le temps d'exécution, le PROOF dispose d'un outil qui s'appelle générateur de package. Cet outil permet de découper l'ensemble des travaux émis par le client en paquets, et ces paquets peuvent être fractionnés jusqu'à un « Event [1]». Il se charge ensuite d'équilibrer la charge de travail dynamiquement entre différents « Workers [2]» associés, assurant ainsi que les travaux sont terminés en même temps.

Dans notre système, l'utilisation de PROOF est couplée avec le « Xrootd ». Cette association permet aux serveurs calculs de récupérer des données provenant des serveurs fichiers afin de les analyser.

7.3 Conclusion

Le Framework ROOT fournit de nombreux modules pour aider nos chercheurs. Il est devenu aujourd'hui l'un des principaux outils pour effectuer les analyses au sein de l'expérience ATLAS. Son extension « PROOF » permet de réaliser des calculs en mode parallèle, ceci représentant un grand d'intérêt pour réduire le temps d'exécution des tâches. Sa conception est assez différente de celle de « Condor ». Le concept du « Condor » consiste à exécuter un maximum de travaux possible dans un axe de temps donné, alors que le concept du « PROOF » est d'utiliser le minimum de temps possible pour réaliser les travaux ordonnés par l'utilisateur.

[1] Event : en français « événement », Ensemble des résultats obtenus pendant la collision entre deux protons.

[2] Worker : ici un « Worker » est un processeur.

Chapitre 8 Grid

Grid en français « grille informatique » est un intergiciel qui permet de construire une infrastructure virtuelle pour offrir la possibilité de partager des ressources informatiques (par exemple la capacité de calcul ou le stockage en réseau). Au départ le mot Grid était utilisé pour décrire un réseau électrique permettant de délivrer l'électricité depuis le producteur d'électricité jusqu'aux consommateurs finaux. Une grille informatique fonctionne comme un réseau de distribution électrique qui fournit des ressources nécessaires à chaque utilisateur à travers une interface simplifiée et une prise vers la ressource partagée.

8.1 Introduction aux grilles

Le Grid est une technologie qui repose sur une infrastructure distribuée. En effet la grille informatique et la grille électrique ne sont pas les seules technologies qui reposent sur ce type d'infrastructure. Il est possible de citer de nombreuses autres technologies comme par exemple le réseau ferroviaire et le réseau téléphonique. Utiliser ce type d'infrastructure permet de baisser l'ensemble du coût de la construction du réseau et une meilleure exploitation de la ressource.

L'origine du « Grid Computing » est assez floue. Par contre le terme de « Grid Computing » fait son apparition au milieu des années 1990. En 1998, Ian Foster (Professeur à l'Université de Chicago) et Carl Kesselman (Professeur à l'Université de Californie du Sud) ont proposé le paradigme de la grille d'informatique. Et plus tard, dans un article nommé « What is the Grid? A Three Point Checklist » de monsieur Ian Foster (2002) une définition de la grille informatique a été formulée. Les trois points définis sont :

- Les ressources associées ne sont pas administrées de manière centralisée ;
- Les méthodes utilisées sont standardisées ;
- La qualité de service des ressources n'est pas assurée.

Début 2000, on considère la grille informatique comme la technologie la plus appropriée pour l'avenir car elle offre aux utilisateurs des ressources de stockage de données quasi inépuisables. Grâce aux évolutions des technologies, le réseau informatique a subit une forte progression depuis ces dernières années. Des réseaux à très haut débit sur des distances longues sont possibles, et ceci permet aux utilisateurs d'accéder à un vaste ensemble de ressources informatiques distribuées de façon transparente.

La parution de la grille informatique fit aussi apparaître le terme VO (ex : Virtual Organisation). Les Virtual Organisation sont les entités (Département d'Entreprise, Entreprise, Instituts, Laboratoires, Petits Groupes de Chercheurs, etc.) réparties sur différents sites

géographiques. Ils disposent d'une ressource informatique importante et mettent leurs ressources informatiques à la disposition de chaque membre de la VO.

8.2 Les domaines d'application des grilles informatiques

Quand nous abordons les domaines d'application des grilles informatiques, nous pensons naturellement à l'utilisation la grille informatique dans la recherche expérimentale. C'est vrai qu'il existe de nombreux projets Grid dans ce domaine. Par exemple, au CERN, on estime que 70,000 CPUs seraient nécessaires pour traiter les données produites par l'accélérateur de particules LHC. Le projet LCG [1] espère répondre à ce besoin : son objectif est de construire une infrastructure de calculs planétaire pour simuler et traiter les données provenant du LHC.

Hormis le projet LHC, nous trouvons aussi un projet comme BIRN[2] constitué par un partenariat de 14 universités ou hôpitaux pour effectuer des recherches dans le domaine médical en particulier pour les maladies du cerveau. Un autre exemple le projet NEESgrid financé par la « National Science Foundation » se concentre sur l'utilisation de la grille informatique pour le développement de nouveaux modèles de simulations numériques des tremblements de terre.

Le Grid apparaît dans des nombreux projets de recherche expérimentale, mais dans le domaine industriel nous trouvons de plus en plus d'entreprises utilisatrices de la grille informatique. Comme par exemple le fabriquant d'engins de chantier Caterpillar qui a travaillé plusieurs années avec la NASA [3] pour développer un environnement virtuel afin de supporter le projet de développement d'un prototype virtuel. L'équipementier Canadian Magna travaille également avec le géant informatique IBM, en utilisant le Grid pour effectuer des analyses sur les données provenant des crash tests de véhicules.

8.3 L'architecture Grid

Dans les deux sections précédentes, nous avons eu une vue d'ensemble sur le Grid. Le Grid permet de partager la puissance de calcul et la capacité de stockage. Il existe de nombreux projets Grid à travers le monde ; pour la plupart ce sont des projets de recherche scientifique, néanmoins certaines grosses entreprises commencent à s'intéresser à ce type de technologie dans le cadre de leur développement. Pour mieux comprendre le Grid, nous allons nous intéresser à son architecture.

[1] LCG : LHC Computing Grid.

[2] BIRN : Biomedical Informatics Research Network.

[3] NASA : National Center for Supercomputing Applications

Une version d'architecture protocolaire a été proposée par M. Ian Foster, un des pères du Grid Computing. Cette architecture se construit à partir de cinq couches : Fabrique, Connectivité, Ressources, Collective et Applications.

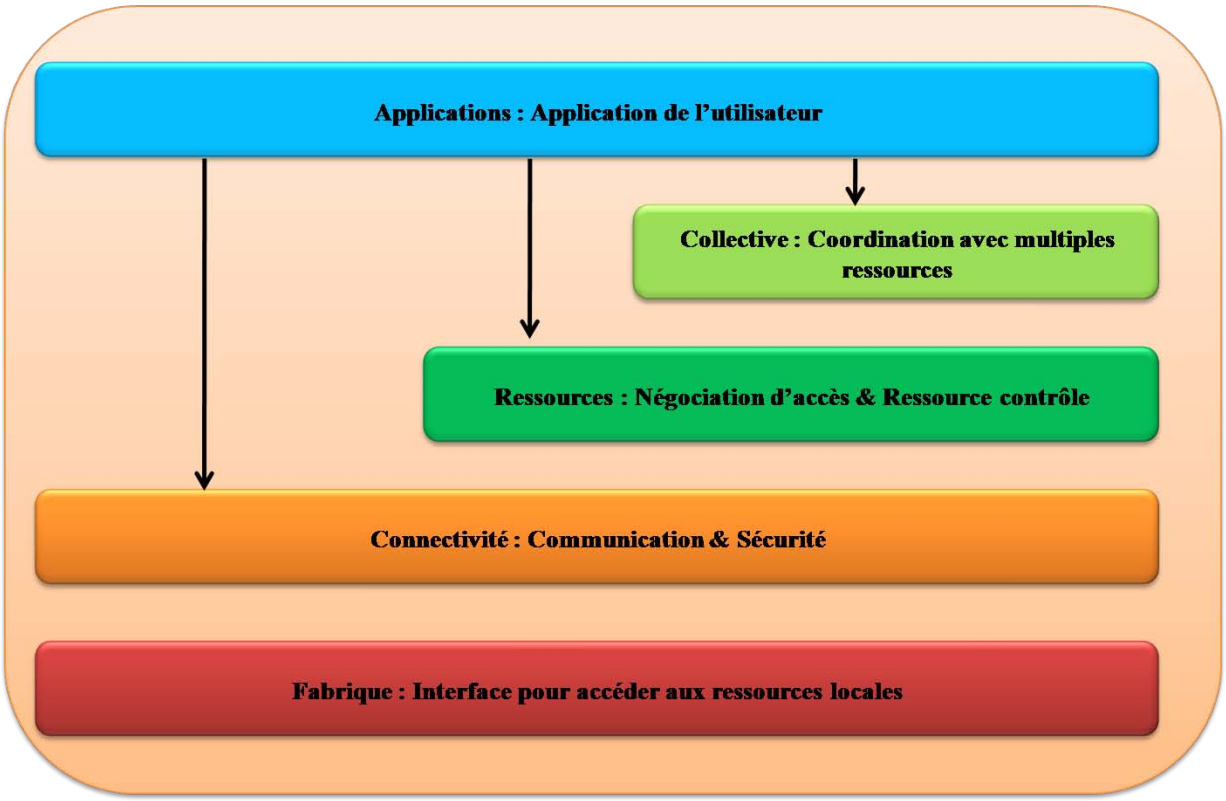

Figure 11 : Architecture Protocolaire de Grid Computing

La couche Fabrique contient des protocoles et interfaces qui permettent de fournir un accès à des ressources partagées. Ces ressources peuvent être la puissance de calculs ou le système de stockage. Dans le cadre du groupe Wisconsin, nous utilisons le Condor pour gérer l'accès aux CPUs et Xrootd pour l'accès aux stockages.

La coche « Connectivité » définit des protocoles de communication et sécurité nécessaires pour assurer des transactions au sein du Grid. Cette couche utilise des protocoles biens connus, comme par exemple : UDP, TCP/IP, pour la gestion de la communication. Tandis que les problèmes de sécurité sont souvent plus complexes, le Grid préfère utiliser des protocoles standard et existants. Ces protocoles construisent un module de Grid et on le nome Grid Sécurité Infrastructure (GSI). Le GSI fournit des services tels que l'authentification, l'autorisation et la protection de message.

La couche « Ressources » constituée par des protocoles permettant des échanges s'effectuant sur une ressource spécifique. Les protocoles définis dans cette couche sont :

- Grid Ressource Allocation Management (GRAM) : affectation à distance, réservation, monitoring, et contrôle de la ressource ;

- GridFTP (une extension de FTP) : il fournit une très bonne performance pour accéder à des données et les transférer;
- Grid Ressource Information Service (GRIS) : il permet de fournir les informations sur une ressource spécifique (un cluster).

Ces protocoles sont basés sur le module GSI de la couche connectivité et utilisent le standard protocole IP (Internet Protocole) pour la communication.

La couche « Collective » contient des protocoles et des services qui ne sont plus associés avec une seule ressource spécifique mais qui captent plutôt les interactions entre les collections de ressources. Grâce à ces composants construits sur la couche « Ressource » et « Connectivité », nous pouvons découvrir, collecter, allouer des ressources partagées. Par exemple les « Services des Répertoires » permettent à un participant VO de découvrir l'existence ou la propriété d'une ressource d'un VO.

La dernière couche « Applications » comporte des applications d'utilisateur exécutées dans un environnement de VO. Ces applications sont construites sur les protocoles et les services qui sont définis par des couches inférieures. Jusqu'à présent, cette couche est la couche recevant le moins de restrictions par une architecture du Grid.

En effet, chaque couche fournit un lot de services et de protocoles qui permettent d'identifier et d'accéder à une ressource Grid en respectant l'ensemble des règles. Les règles sont définies par les utilisateurs et l'administrateur.

8.4 Globus Toolkit et Visual Data Toolkit

Le Globus Toolkit est la référence pour la mise en place d'une architecture Grid. C'est un projet réalisé aux Etats-Unis par des équipes de recherches qui sont s'installées à ANL (Argonne National Labs), Université de Chicago, Université du Sud Californie et NCSA (National Center for Supercomputing Applications). Le Globus Toolkit est développé en langage C et en Java ; c'est une open source qui permet à de nombreux développeurs dans le monde de contribuer à ce projet.

Le Globus Toolkit inclut des outils et une bibliothèque pour résoudre les problèmes suivants :

- La sécurité : ce module est soutenu par le module de sécurité GSI. Elle permet d'authentifier les utilisateurs (membres) de la grille et d'assurer l'intégrité et la confidentialité des données ;
- Data Management : ce module est soutenu par le GridFTP et RFT (Reliable File Transfer). Le GridFTP permet des transferts de fichier d'une machine A vers une

machine B, mais aussi d'une machine A vers une machine B depuis une machine C. Le RFT utilise les mêmes principes de base que le GridFTP, par contre il travaille sur une couche plus haute. Il utilise les messages SOAP à travers le HTTP pour assurer la fiabilité de transfert des « gros » fichiers ;

- Exécution Management : ce module est soutenu par l'outil GRAM. Il permet de contrôler, localiser, exécuter et annuler une tâche sur la grille. Le Gatekeeper, l'un des composants d'outil GRAM est chargé de faire appliquer les règles d'accès. Il s'appuie sur GSI, il accepte les connexions, authentifie l'utilisateur à distance et transforme le certificat à distance en une identification d'utilisateur local. Une fois authentifié, le client peut demander des services particuliers, comme par exemple contrôler une liste d'attente de travaux via le « Jobmanager ».
- Information Service : ce module est soutenu par le Grid Information Service (GIS). Le GIS fournit des capacités de localisation des ressources sur la base des caractéristiques requises par les jobs (OS, CPU, mémoire, etc.).

Le Globus Tooolkit est construit avec des modules essentiels qui permettent de réaliser une infrastructure Grid. Pour cette raison, il est devenu un module de base fondamental pour réaliser des projets ou outils. Le Virtual Data Toolkit (VDT) est un outil qui utilise le Globus Toolkit comme un module de base pour mettre en place une infrastructure Grid.

Le VDT est un ensemble de logiciels Grid qui est plus facile à installer et configurer. Il est le fruit d'une collaboration des participants d'expérience CMS aux Etats-Unis. Le but de ce projet est de construire un outil Grid simple et facilite à utiliser, à déployer et à maintenir. L'idée de base est d'utiliser une simple ligne de commande pour accéder au Grid ressources ou fournir nos ressources aux autres.

Parmi le Globus Toolkit, nous trouvons également un gestionnaire de package « Pacman ». Il gère des installations et la mise à jour du kit VDT. On y trouve aussi Condor et Gondor-G[1]. Le Condor-G se situe plutôt dans la couche « Collective », et la mise en place du Condor-G aide les utilisateurs à soumettre des travaux dans un environnement Grid via une ressource distante. En revanche, le système Condor se place dans la couche « Ressources », car son rôle est de fixer la priorité des travaux, le monitoring et le management des ressources, le monitoring des travaux et la règle de planification.

[1] Condor-G : est le mariage de technologie entre le Condor et le Globus. Il permet de organiser et dispatcher des tâches dans un environnement GRID.

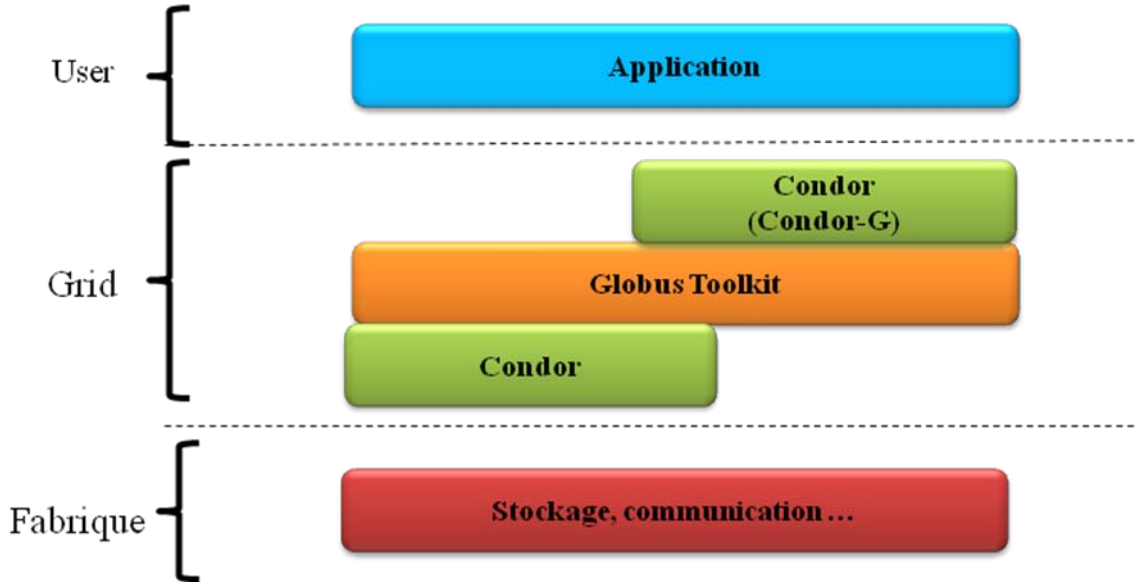

Figure 12 : Le système Condor dans un environnement Grid

Le SRM (Storage Resource Manager) est un intergiciel qui gère l'allocation de l'espace et le management des fichiers dans un espace de stockage partagé qui se trouve dans un environnement Grid. Le SRM ne s'occupe pas du transfert des fichiers : il est utilisé pour négocier et réserver un espace disque. Le SRM permet d'associer le GridFTP pour transférer des fichiers.

Il sera envisageable aussi d'évoquer le LFC [1](LCG File Catalog) qui est développé par le CERN. Le LFC est un catalogue qui contient une liste de la localisation des fichiers dans le Grid. Ce catalogue est sécurisé et les données peuvent être stockées dans une base de données MySQL ou Oracle. Le VDT contient des centaines de modules et chaque module cible une fonctionnalité spécifique et selon les besoins il est possible d'activer ou désactiver certains modules.

8.5 Conclusion

Dans ce chapitre nous avons pris connaissance de la grille informatique, son architecture, ses domaines d'application et les outils Grid qui permettent de mettre en place une infrastructure de grille d'informatique.

La grille d'informatique est utilisée dans nombreux domaines. Elle représente des bonnes perspectives pour l'avenir. L'application d'une infrastructure Grid permet une organisation réduisant les dépenses opérationnelles, l'exploitation maximale de l'investissement et de la ressource informatique existante, de créer une infrastructure informatique flexible et extensible afin d'accélérer le cycle du développement d'un produit.

La mise en place d'une infrastructure Grid présente de nombreux d'avantages. Cependant, ce type de technologie est réservé pour le moment aux grosses entreprises ou instituts de

[1] LFC : est un catalogue fichier qui permet de tracer la localisation d'un fichier sur un VO.

recherche scientifique. Le Wisconsin groupe s'implique dans la recherche de la physique des hautes énergies dans le cadre de l'expérience ATLAS. Le groupe partage ses ressources informatiques dans une infrastructure grille informatique qui est mise en place par le CERN et ses collaborateurs. La mise en place d'une infrastructure Grid est bénéfique pour l'ensemble des projets de recherche mais elle n'est pas sans conséquence pour le groupe, sur laquelle nous allons revenir dans la partie suivante.

Conclusion de la deuxième partie

Nous avons évoqué les principales technologies employées par le système d'information du groupe Wisconsin. Le déploiement du système Condor permet de gérer et planifier des travaux à mettre dans la liste d'attente d'exécution. Le groupe utilise ce système pour gérer de gros volumes de travaux émis par les chercheurs du groupe. Par contre le Condor peut être aussi utilisé dans le cadre Grid pour collecter les informations nécessaires à l'utilisation du système.

L'utilisation de Xrootd permet d'avoir une meilleure compatibilité avec le fichier format ROOT. Il offre un accès rapide aux fichiers, un équilibrage de charge de données dynamique et une excellente extensibilité pour répondre aux problèmes liés à l'augmentation constante des données produites par le LHC. Le Xrootd peut aussi être intégré dans une architecture Grid. Grâce à son excellente compatibilité, il nous permet d'accéder à la plupart des systèmes de stockage.

Puis nous avons présenté le Framework ROOT qui contient de nombreux modules comme par exemple le module « Mathematical », « Algèbre Linaire » et « Graphique » permettant d'aider les chercheurs à résoudre des calculs compliqués et représenter les résultats sous forme graphique. Le PROOF, une extension de ROOT, offre la possibilité de traiter les fichiers ROOT en mode parallèle. Utiliser ce mode augmente sensiblement la performance de traitement.

Dans le dernier chapitre de cette partie, nous avons parlé du Grid. Pour traiter le nombre considérable de données produites par le LHC, le CERN a mis en place une infrastructure de grille d'informatique. Le Wisconsin groupe est une VO (Virtuelle Organisation) qui fait partie de cette infrastructure. Dans ce chapitre, nous avons aussi fait connaissance avec l'architecture protocolaire, les technologies employées et l'outil de Grid. L'utilisation de Grid représente plusieurs avantages : elle est plus économique, elle fournit une ressource illimitée et elle permet de réduire le cycle de développement.

Les technologies utilisées dans le système d'information du groupe sont très intéressantes. Il ne suffit cependant pas d'intégrer uniquement ces technologies dans un système d'information pour bénéficier de ses atouts. En effet, il nécessaire de mettre en place une bonne architecture. Pour cela, il faut connaître les besoins des utilisateurs et comprendre le système existant.

Troisième Partie

Ingénierie des besoins et Etude du système existant

Rappelons que, dans des deux parties précédentes, nous avons évoqué l'environnement et la motivation du projet. Nous avons également pris connaissance de l'architecture actuelle du système d'information du groupe. Les principales technologies utilisées par le système sont aussi présentées. Ces deux parties nous ont donc permis d'avoir une vue globale du projet. Néanmoins, une perception globale n'est pas suffisante pour réaliser ce projet. Pour revoir l'architecture du système d'information, nous devons tout d'abord comprendre quels sont les besoins des utilisateurs et du système.

Pour cette raison, dans cette partie, nous allons effectuer une analyse des besoins. Cette analyse nous permettra de comprendre ce que le système doit fournir comme services et informations attendus par leurs utilisateurs. Cette étape étant primordiale pour le projet, nous l'évoquerons dans le chapitre suivant.

Nous étudierons par la suite plus en détail l'architecture et l'infrastructure du système actuel. Nous allons aussi étudier les autres solutions possibles du marché pour faire une comparaison avec la solution existante. En effet, dans ce projet, il ne s'agit pas de construire une nouvelle architecture mais de remanier le système existant. Comprendre le système actuel et les solutions alternatives ont donc autant de valeur que l'ingénierie des besoins. L'ingénierie des besoins et l'étude du système existant nous permettront de découvrir les contraintes environnementales, de déterminer la stratégie et d'élaborer la conception d'une nouvelle architecture pour répondre aux exigences imposées par le système et les attentes des utilisateurs.

Chapitre 9 Ingénierie des besoins

Selon une étude de « The Standish Group »[1], seuls 16.2% des projets informatiques respectent les coûts et les délais prévus initialement. De plus, environ 31% des projets sont abandonnés pendant leur période de réalisation. Cette situation peut être expliquée par le manque de compréhension des besoins des utilisateurs, le manque d'implication des utilisateurs, les cahiers des charges incomplets ou des attentes irréalistes.

Pour construire un système d'information, il est nécessaire de bien comprendre les besoins des utilisateurs. Une mauvaise compréhension des besoins peut être fatale. Un exemple édifiant est celui de la refonte du système d'information du SAM londonien (gestion des ambulances d'urgence) qui, à cause d'une mauvaise compréhension des besoins, a entraîné plusieurs décès. Il faut également noter que corriger les erreurs dues à la mauvaise compréhension des besoins demandent un effort considérable.

Une mauvaise compréhension des besoins peut également entraîner un dépassement de délai prévu initialement et un surcoût pour un projet. Différentes études montrent que le coût supplémentaire pour corriger les erreurs commises dans la phase de l'ingénierie des besoins est disproportionné par rapport aux erreurs détectées dans chaque phase de développement. La table suivante sert à montrer cet effet. Exemple : si résoudre un problème pendant la phase d'expression des besoins coûte 1 €, alors pour résoudre le même problème mais dans la phase de livraison coûtera entre 68 à 110 €.

Tableau 2 : Le coût relatif pour la correction d'un défaut dû à l'ingénierie des besoins

Phase du projet	**Le coût pour la correction des erreurs**
Expression des besoins	1 x
Conception	2-3 x
Codage	5-10 x
Test unitaires	8-20 x
Livraison	68-110 x

En regardant le tableau ci-dessus, il n'est pas difficile de comprendre l'importance de l'ingénierie des besoins dans un projet informatique. Pour effectuer la recherche des besoins, il existe des approches bien différentes. Dans la section suivante, nous allons présenter les approches et la méthode utilisées dans ce projet.

[1] On peut la trouver notamment sur : http://www.projectsmart.co.uk/docs/chaos-report.pdf

9.1 Les approches de l'ingénierie des besoins

Selon Madame Colette Rolland professeur à l'Université de Paris 1 Sorbonne, « l'ingénierie des besoins est l'activité qui transforme une idée floue en une spécification précise de besoins, souhaits, exigences exprimés par une communauté d'usagers et donc définit la relation existante entre un système et son environnement » [6].

En effet, l'ingénierie des besoins est une tâche bien plus complexe que l'on pourrait penser. Les questions telles que « Quelles sont les parties prenantes? », « Quelles sont les techniques et outils que l'on pourrait utiliser pour faire ressortir les besoins ? », sont très souvent posées dans la phase de l'ingénierie des besoins. Il existe certaines approches pour nous aider à formuler les besoins.

La grande majorité des démarches de l'ingénierie des besoins est basée sur deux concepts : le but et le scénario. L'approche basée sur le concept « But » cherche à comprendre quels sont les objectifs à atteindre pour le futur système. Dans la phase d'identification du « But », les questions à poser sont « Quoi ? », « Comment ? » et « Pourquoi ? ». Pour poser ces questions nous utilisons le langage naturel.

En revanche, l'approche basée sur le « Scénario » cherche à comprendre comment le système doit procéder. Cette approche permet de découvrir les besoins pour des situations prévus à l'avance. Un scénario est une description d'une transaction entre l'utilisateur et le système. Il est écrit en langage naturel comprenant une ou plusieurs actions et il est caractérisé par deux états : initial et final. Nous utilisons le diagramme de séquence d'UML [1] (Langage de Modélisation Unifié) pour montrer les interactions dans le cadre d'un scénario d'un diagramme des cas d'utilisation.

Enfin, il existe aussi un procédé qui couple ces deux approches en cherchant un bon équilibre entre l'approche par le « But » et le « Scénario ». La section à suivre présentera les approches permettant de réaliser l'ingénierie des besoins. Le choix d'une approche peut être influencé par le type de projet et les besoins des parties prenantes.

9.2 Les parties prenantes

L'identification des parties prenantes est une étape préliminaire pour l'ingénierie des besoins. Parce que pour construire un système solide, nous devons comprendre quels sont les exigences. Et pour comprendre les exigences, nous devons déjà connaître les besoins des parties prenantes.

[1] UML : en anglais Unified Modeling Language, est un langage graphique de modélisation des données et des traitements

Alors qu'est ce qu'une partie prenante? En 1984, dans un livre « Strategic Management: A stakeholder approach » de M. Freeman R. Edward, une définition à la « partie prenante » a été avancée [9]. Selon lui :

" Dans une organisation, une partie prenante est un individu ou une organisation qui peut avoir une influence ou peut être influencé par la réalisation des objectifs de l'organisation **"**

Aujourd'hui dans le cadre du système d'information, quand nous parlons de partie prenante, nous pensons aussitôt aux utilisateurs du système. Cependant les utilisateurs ne sont pas les seuls qui font partie de la partie prenante. Pour identifier les parties prenantes, nous devons s'appuyer sur certaines de leurs caractéristiques. Et il est possible de les rechercher en posant les questions suivantes :

- qui sont les bénéficiaires de notre système car ils utilisent le système et un changement du système peut avoir un impact sur leurs activités ;
- qui sont les individus ou groupes impliqués dans le cycle de la conception, du développement et de la maintenance du système ;
- qui finance le projet ;
- qui peut apporter des effets négatifs à notre système.

Les parties prenantes ainsi identifiées, nous pouvons commencer à entrer dans les différentes phases d'ingénierie des besoins.

9.3 Les types de besoins

Avant d'évoquer les démarches adoptées dans ce projet, nous allons regarder d'abord la typologie des besoins. Lorsque les critères de recherche ou les niveaux de détail sont différents alors les types de besoins peuvent être bien différents

Par exemple, le système peut être envisagé selon deux critères : les besoins des utilisateurs et le système lui-même. Les besoins utilisateurs sont exprimés en langage naturel ou graphique représentant les services que le système fournit ainsi que ses contraintes opérationnelles. En revanche, les « besoins système » décrivent les fonctionnalités attendues du système. Ce sont souvent des documents structurés détaillant les fonctions dudit système.

Il est possible également de classifier les besoins par leur nature. Comme par exemple les besoins pérennes ou les besoins volatiles. Les besoins pérennes sont liés à l'activité principale du client et les besoins volatiles peuvent changer pendant le développement ou pendant l'exploitation du système.

Enfin il aussi possible de classer les besoins par les besoins parties prenantes, comme par exemple des besoins fonctionnels et non-fonctionnels. Ces deux types de besoins sont très

fréquemment utilisés dans la phase de l'ingénierie des besoins. Les besoins fonctionnels décrivent les fonctionnalités du système. Par exemple les utilisateurs peuvent retrouver le plus rapidement possible un fichier dans l'ensemble des serveurs fichier.

Par contre, les besoins non-fonctionnels concernent les besoins tels que la performance, la fiabilité, la robustesse, etc. Ce type de besoin décrit plutôt sous quelle contrainte le système doit travailler. Il faut noter que ces besoins sont tout aussi importants que les besoins fonctionnels. .

9.4 La démarche utilisée dans le cadre du projet

Dans les sections précédentes, nous avons eu l'occasion de prendre connaissance des thèmes et des approches utilisées dans la phase de l'ingénierie des besoins. Dans cette section nous allons présenter la démarche utilisée dans ce projet. Nous avons adopté une démarche en quatre phases pour la mise en chantier de l'ingénierie des besoins. Ces phases incluent la phase de découverte des besoins, l'analyse des besoins, la spécification des besoins et la validation des besoins.

Les paragraphes suivants vont détailler ces quatre phases :

- **La découverte des besoins :** elle est la première étape que nous avons adoptée dans ce projet. Cette phase consiste à essayer de découvrir les besoins. Pour cela, nous devons interviewer les parties prenantes, effectuer une étude préliminaire du système actuel et examiner les documents existants. Ces actions nous permettent de définir le domaine d'application et les services que le système doit assurer.
- **L'analyse des besoins :** cette deuxième phase consiste à rassembler les informations sur les systèmes souhaités et les informations sur le système existant. Ensuite nous les regroupons et nous les classons dans des classes cohérentes. Nous définissons ensuite le niveau de priorité pour chaque besoin. Si nous détectons qu'il y a des conflits entre les besoins exprimés par les parties prenantes, nous devons négocier avec certaines d'entre elles pour essayer de trouver un compromis. Enfin nous pouvons documenter les besoins ; ils pourront ainsi servir d'inputs à la phase suivante du cycle de vie.
- **La spécification des besoins :** cette phase consiste à détailler les besoins que nous avons élaborés dans la phase précédente. Pour cela, un diagramme des cas d'utilisation, des diagrammes de séquence ou des scénarii textuels peuvent être employés pour définir les besoins fonctionnels. En revanche pour les besoins non-fonctionnels, nous pouvons élaborer sous la forme de formules ou de textes des critères définis avec les parties prenantes.

- **La validation des besoins :** cette dernière phase consiste à s'assurer que les spécifications développées dans la phase précédente sont bien celles que les parties prenantes souhaitent. Cette phase n'est pas très difficile à réaliser en revanche elle est très importante, car le coût lié aux erreurs faites à cette étape du cycle de vie de développement peut être colossal.

Ces quatre phases sont utilisées pour élaborer les besoins dans ce projet. Il reste cependant quelques contraintes à régler dans la phase de l'ingénierie des besoins. La section suivante va décrire les contraintes que nous avons rencontrées dans cette phase.

9.5 Les contraintes potentielles

Dans la phase de l'ingénierie des besoins, nous devons faire très attention aux problèmes ou contraintes potentiels, car mal gérer ces problèmes et ces contraintes pourraient impacter la qualité de ce projet. Ces contraintes peuvent venir de différents domaines. Par exemple :

- **Technologie** : de nos jours, les technologies évoluent plus vite que par le passé. En conséquence, ce que les utilisateurs attentent du système évolue bien plus rapidement qu'auparavant. Leur besoins ne sont donc pas stables ;
- **Expression des besoins** : les parties prenantes ont souvent du mal à exprimer leurs besoins ou encore elles ne savent pas ce qu'elles veulent vraiment.
- **Langage** : les besoins exprimés par les parties prenantes sont souvent dans un langage particulier. Des mots en jargon sont souvent employées ;
- **Conflit** : dans un même projet, les différentes parties prenantes peuvent avoir des demandes contradictoires ;
- **Environnement** : des facteurs organisationnels et politiques peuvent influencer les spécifications.

Les points évoqués dans les paragraphes précédents montrent les problèmes et les contraintes que nous pouvons rencontrer dans la phase d'ingénierie des besoins. Pour régler ces problèmes, nous devons nous appuyer sur la qualité d'interview, d'analyse et de spécification pour élaborer les besoins. Et finalement ces besoins doivent être validés par les parties prenantes.

9.6 La découverte des besoins

Dans cette phase de découverte des besoins, nous avons multiplié des réunions et des rencontres avec les parties prenantes pour essayer de comprendre les besoins de chacun. En même temps nous avons effectué une étude sur le système existant pour mieux comprendre les thèmes exprimés par les parties prenantes.

Cette phase est découpée en trois étapes. La première étape consiste à rechercher les parties prenantes, la seconde à procéder à une analyse du système existant et enfin la troisième à « capturer » les besoins.

Les parties prenantes sont, quant à elles, classées en trois types :

- **Administrateur du système** : Ses principales missions sont le support, le développement, la proposition sur l'achat des matériels, la configuration et l'administration du système ;
- **Le Responsable du groupe** : le chef du groupe est le professeur Wu. Elle n'est pas l'utilisateur direct de notre système. Elle dirige les chercheurs mais aussi le budget du groupe. C'est elle qui a le vote véto sur les achats des matériels ;
- **Les utilisateurs du système** : Les utilisateurs internes sont des chercheurs du groupe qui permettent d'utiliser toutes les fonctionnalités du système. En revanche les utilisateurs peuvent venir de l'extérieur : ce sont des chercheurs qui utilisent notre système à travers le Grid.

Concernant la phase d'étude du système existant, il est possible de le résumer en trois grandes parties ou trois sous-systèmes. Chaque sous-système représente une fonctionnalité particulière dont :

- **Le Condor** : un « Batch système » qui permet aux chercheurs de traiter les fichiers selon leurs besoins ;
- **Le PROOF** : un système « Parallèle processing » qui permet aux chercheurs de transformer les fichiers « ROOT » en fichier graphique;
- **Le Grid** : pour la partie « Virtuelle Organisation » qui permet aux utilisateurs du groupe ou les utilisateurs externes d'utiliser la ressource informatique du groupe.

Parmi les principales fonctionnalités présentées ci-dessus, nous pouvons aussi évoquer la base de données qui enregistre les informations concernant les Datasets et aussi le système « Xrootd » pour le transfert de données. Ils sont utilisés pour coopérer avec le système « Condor » et « Grid ». Après d'avoir déterminé les groupes des parties prenantes et pris connaissance du système dans sa globalité, nous avons commencé à organiser les réunions et les interviews pour découvrir des besoins. Ils sont résumés et recensés dans le tableau suivant :

Tableau 3 : Les besoins capturés

Acteur	Les besoins
Administrateur	1. Il estime que les deux serveurs « PROOF » ne sont pas assez utilisés et ils ne veulent plus gaspiller cette ressource; 2. Pour le moment, il n'est pas possible de contrôler les travaux « PROOF » est-ce ce possible d'apporter une solution ?
Responsable du groupe	1. Le budget est serré, les investissements sur les matériels doivent être rationalisés. 2. Au niveau du personnel, actuellement deux informaticiens travaillent en permanence dans le groupe pour réaliser ce projet. Nous ne devons pas embaucher de nouvelles personnes.
Les utilisateurs	1. Lorsqu'un utilisateur fait exécuter un travail PROOF, le système met environ 20 minutes pour fournir le résultat, c'est trop lent ; 2. Ils demandent d'améliorer la base de données existante, pour que la mise à jour de la base de données soit automatique ; 3. Lorsqu'ils envoient des travaux « PROOF », parfois le système prend beaucoup de temps pour commencer à exécuter les travaux, pouvons-nous réduire ce temps d'attente ? 4. Passage de réseaux10 Gbit/s pour notre parc informatique.

9.7 Analyse des besoins

Dans la section précédente, le tableau présente les informations récoltées pendant la phase de découverte des besoins. Dans la phase d'analyse des besoins, nous les classons dans des classes cohérentes et par la suite nous les examinons afin de fixer leur niveau de priorité.

En examinant les besoins exprimés par les parties prenantes et le projet que nous sommes en train de mener, nous privilégions de différencier les besoins en « besoins fonctionnels » et « besoins non-fonctionnels ». Le tableau suivant est un résumé de cette analyse :

Tableau 4 : L'analyse des besoins

Acteur	Code*	Résumé « les besoins »	Type de besoins	Catégorie	Priorité
Admin**	A1	Besoin de réutiliser des serveurs mal exploités	Non-fonctionnel	Performance « PROOF système »	Haute
Admin	A2	Besoin de gérer des travaux « PROOF »	Fonctionnel	« PROOF système »	Haute
Responsable du groupe	C1	Achat des matériels doit être rationnel	Non-fonctionnel	Le coût	Médium
Responsable du groupe	C2	Besoins de contrôler le coût du développement	Non-fonctionnel	Le coût	Médium
Utilisateur	U1	Besoins de réduire le temps d'exécution des travaux « PROOF »	Non-fonctionnel	Performance « PROOF système »	Haute
Utilisateur	U2	Besoin de revoir la base de données pour qu'elle réponde aux besoins des utilisateurs (mise à jour)	Fonctionnel	« Base de données »	Haute
Utilisateur	U3	Besoin de réduire le temps d'attente entre le moment d'envoi et le moment où le système commence à exécuter les travaux	Non-fonctionnel	Performance « PROOF système »	Haute
Utilisateur	U4	Changement pour les équipements réseaux	Non-fonctionnel	Performance Pour tout le système	Médium

* Ce code correspond au mariage entre la lettre initiale du nom des acteurs et les numéros qui sont inscrits dans la colonne « Résumé » du tableau 3

** Admin = Administrateur

En étudiant les huit besoins recensés du tableau ci-dessus, seuls deux besoins sont fonctionnels. Nous classons les deux besoins fonctionnels en haute priorité, car la réalisation de ces besoins peut influer la performance de recherche du groupe.

Les priorités pour les besoins non-fonctionnels sont, elles, mitigées. Nous avons classé tout ce qui concerne « Le coût » en priorité medium. En effet ces besoins n'étant pas vraiment des besoins pour notre projet, nous pouvons les considérer plutôt comme des contraintes ou des règles que nous devons respecter tout au long du projet.

Par ailleurs, les besoins des utilisateurs qui demandent de changer les matériels réseaux pour pouvoir passer de 1 Gbit/s à 10 Gbit/s, ont été classés temporairement en priorité médium. Pour cette demande, des études plus poussées permettant de savoir si les besoins en matériels de ce type sont réels vont être nécessaires. De plus, la responsable du groupe nous demande de rationaliser les achats de matériels. C'est la raison pour laquelle nous préférons la classer en priorité médium pour le moment, et au fur et à mesure de l'avancement du projet nous pourrons décider de la validité de cette demande.

Pour le reste des besoins non-fonctionnels, nous les classons en haute priorité car ces besoins concernent la performance du système. En revanche, nous pouvons observer qu'il y a des problèmes avec les besoins exprimés par les utilisateurs. Ils sont en effet souvent flous ou imprécis, comme par exemple « Les utilisateurs ont besoins de réduire le temps d'attente pour exécuter les travaux PROOF » : lorsque nous demandons aux utilisateurs de nous donner le temps idéal pour définir un indice de référence, alors les utilisateurs ne savent plus quoi nous répondre. Cette situation est conforme à la synthèse que nous avons faite dans les sections précédentes, « les utilisateurs ne savent pas ce qu'ils veulent vraiment » Pour ce type de problème, nous avons décidé d'effectuer des tests plus poussés pour obtenir des informations du système plus précises et enfin de pouvoir négocier avec les utilisateurs pour mieux définir leurs besoins.

9.8 Spécification des besoins

La phase d'étude de la spécification des besoins se base sur une bonne compréhension des besoins exprimés des parties prenantes et les interactions prévisibles pour le futur système. En effet, ce projet est basé sur un système existant, par conséquent, pour dérouler la phase d'ingénierie des besoins, nous devons l'analyser et le tester pour tenter de mieux comprendre ce système existant. Dans cette section, nous n'allons pas décrire exhaustivement la phase d'analyse du système existant, car nous allons la détailler davantage dans le chapitre suivant. En revanche, certains chiffres employés dans cette section proviennent de la phase d'analyse du système existant.

Rappelons que, dans les sections précédentes, nous avons découpé notre système en trois sous systèmes : le système « Condor », « PROOF » et « Grid ». Et dans notre système, le « Xrootd » est utilisé par ces trois sous systèmes pour gérer les accès aux fichiers. Enfin, une base

de données permet d'aider les utilisateurs à trouver rapidement les informations sur les fichiers qui permettent effectuer les travaux.

Pour que cette phase d'analyse soit plus précise, nous découpons notre système en trois parties. Chaque partie correspond un sous-système. Se focaliser sur les sous-systèmes nous permettra d'avoir une meilleure visibilité de la correspondance entre les besoins et les sous-système.

En regardant les besoins recensés dans la section précédente, deux besoins fonctionnels ont été détectés : le premier pour la gestion de l'administration des travaux « PROOF » et le deuxième pour la gestion de mise à jour de la base de données. Consulter la base de données nous permet de retrouver des informations essentielles pour soumettre des travaux PROOF et Condor.

Dans la phase de spécification des besoins, pour effectuer des analyses approfondies sur les besoins fonctionnels, nous utilisons le diagramme de « Cas d'utilisation » et « Diagramme de séquence ». L'utilisation du Cas d'utilisation nous permet d'identifier des acteurs et les interactions entre les acteurs et le système. Le diagramme de séquence peut être utilisé pour enrichir le cas d'utilisation en se basant sur une représentation graphique des interactions entre les acteurs et le système.

La figure 13 est un diagramme de cas d'utilisation pour le sous-système « PROOF ». Nous avons choisi ce sous-système car le besoin fonctionnel « Gérer des PROOF job » réclamé par les parties prenantes est présenté dans ce sous-système. Les deux autres fonctionnalités font déjà partie des fonctionnalités du système existant. Nous constatons que parmi les trois fonctionnalités du sous-système « PROOF », deux sont utilisées par les utilisateurs et une seule par les administrateurs.

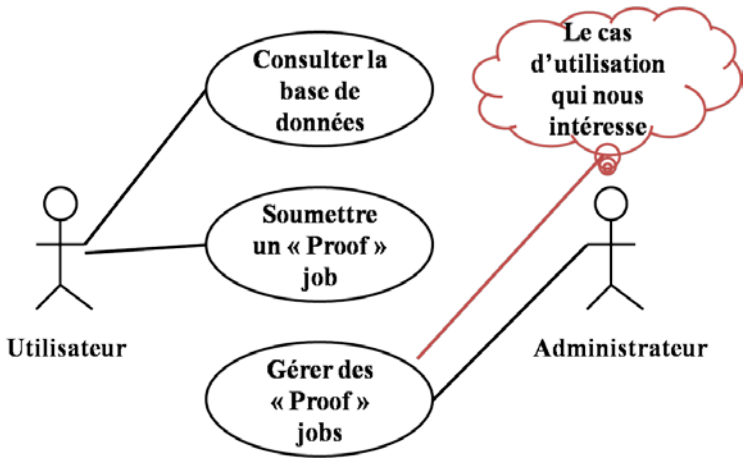

Figure 13 : Cas d'utilisation sous-système « PROOF »

Elaborer un diagramme des cas d'utilisation permet de clarifier le service qui doit être rendu par le système. Il exprime les interactions entre les acteurs et le système et apporte une

valeur ajoutée à l'acteur concerné. Une fois, le cas d'utilisation enfin analysé, il sera possible de passer à un niveau plus bas pour détailler davantage les cas d'utilisation qui nous intéressent.

La figure 14 est un diagramme de séquence pour le cas d'utilisation « Gérer des travaux PROOF ». Ce diagramme nous permet de comprendre que, pour gérer les travaux « PROOF », il faut tout d'abord obtenir la liste de travaux en cours et seulement ensuite, il est possible de modifier la priorité des travaux en cours.

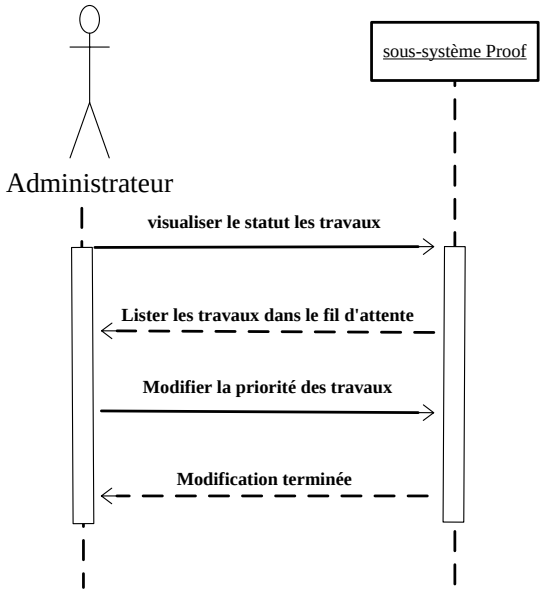

Figure 14 : Diagramme de séquence « Gérer des travaux PROOF »

La figure 15 représente un autre besoin fonctionnel, celui des « mises à jour de la base de données ». Comme nous pouvons le constater dans ce diagramme nous avons identifié deux acteurs « Utilisateur » et « Nouveaux fichiers ». L'« Utilisateur », représente les physiciens du groupe et l'acteur, les « Nouveaux fichiers ». Ce sont des systèmes informatiques extérieurs qui servent à alimenter notre système avec les fichiers demandés par les « Utilisateurs ». Notre système doit pouvoir mettre à jour automatiquement les informations concernant les fichiers transférés, comme par exemple leur localisation ou encore leur taille.

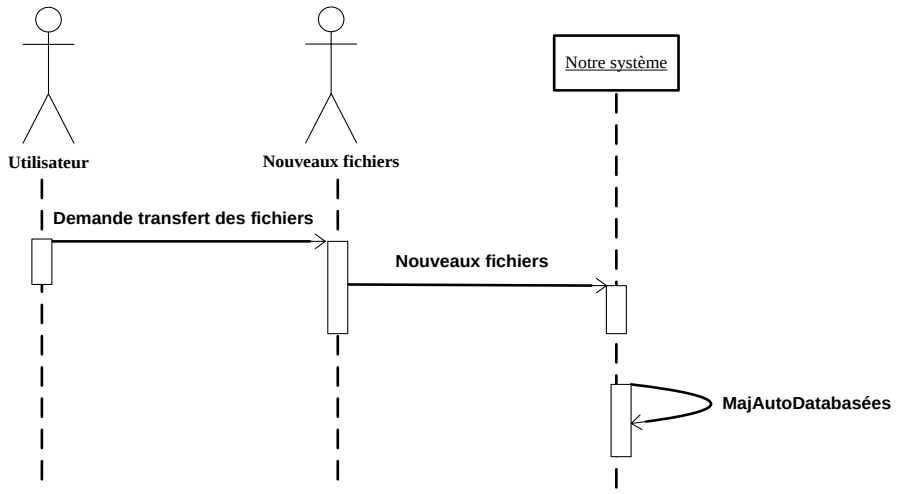

Figure 15 : Diagramme de séquence « mettre à jour la base de données »

Une fois que nous avons spécifié les besoins fonctionnels, nous avons commencé à étudier en détail les besoins non-fonctionnels. Le tableau 5 nous montre quels sont les cibles, les exigences et les cas d'utilisation correspondants de ces besoins non-fonctionnels.

Parmi les besoins recensés pendent la phase de découverte des besoins, nous pouvons observer que les parties prenantes n'ont pas exprimé par exemple la disponibilité, la robustesse et l'extensibilité du système. En revanche nous allons poser ces questions tout au long du projet, car la réalisation du projet ne doit pas nuire à la partie non-fonctionnelle déjà acquise de notre système. Et si nécessaire, nous allons l'insérer dans la liste des besoins.

Tableau 5 : Spécification des besoins non-fonctionnels

Code	Description des besoins	Cas d'utilisations	Catégorie	Cible	Priorité	Exigence
A1	Besoin de réutiliser des serveurs mal exploités	Cas d'utilisation « sous-système PROOF»	Performance	Admin	Haute	limiter le temps d'exécution des travaux au-dessous de 3 minutes
C1	Achat des matériels doit être rationnel (budget de 400,000$ par an)	Pour tout le système	Le coût	Responsable du groupe	Médium	budget 400,000$ par an
C2	Besoin de contrôler le coût du développement	Tout	Le coût	Responsable du groupe	Médium	100,000 $ de budget pour l'administration et le développement du SI
U1	Réduire le temps d'exécution des travaux « PROOF »	Cas d'utilisation « sous-système PROOF»	Performance	Utilisateur	Haute	limiter le temps d'exécution des travaux au-dessous de 3 minutes
U3	Réduire le temps d'attente entre le moment d'envoi et le moment où le système commence à exécuter les travaux	Cas d'utilisation « PROOF système »	Performance	Utilisateur	Haute	Assurer que les travaux commencent à exécuter en moins de 3 secondes
U4	Changement pour les équipements réseaux	Tout	Performance (achat des matériels)	Utilisateur	Médium	Ce besoin a un conflit avec le besoin C1 : nous devons réexaminer la politique d'achat des matériels

*Admin = Administrateur

9.9 Validation des besoins

Lorsque nous avons finalisé la phase de spécification des besoins, nous avons organisé à nouveau une réunion avec les parties prenantes pour valider notre recherche. Ces résultats nous serviront pour déterminer les objectifs que nous devons atteindre à la fin du projet.

Il faut souligner que cette étape n'est pas une simple validation des besoins exprimés par les parties prenantes, mais sert également à régler les conflits entre différentes parties, car les besoins que nous avons détectés doivent être approuvés par l'ensemble des parties prenantes.

9.10 Conclusion

Au début de ce chapitre, nous avons présenté les approches et une démarche à suivre dans la phase de l'ingénierie des besoins. Une mauvaise compréhension des besoins des parties prenantes peut entrainer un surcoût pour notre projet informatique. Pour cette raison, nous avons besoin de déterminer une bonne démarche à suivre pour assurer la qualité des besoins recueillis.

Par la suite, nous avons évoqué la démarche que nous avons poursuivie pendant la phase de l'ingénierie des besoins. Cette démarche est construite en cinq étapes (la recherche des parties prenantes, la découverte des besoins, l'analyse, la spécification et la validation des besoins). Les besoins recueillis nous serviront pour les phases suivantes du projet, ces phases incluant la recherche de la stratégie, la conception et la réalisation.

Chapitre 10 L'infrastructure matérielle du système existant

Dans cette section, nous allons principalement étudier en détail les composants matériels de notre système et la mise en place de ces composants. Cette étude est très importante pour la suite de l'analyse car l'infrastructure est un élément clé pour réaliser une bonne interopérabilité entre les éléments des systèmes d'information. Dans cette phase, nous allons nous focaliser sur la mise en place du réseau, du clustering et du stockage.

10.1 Le réseau

Depuis un peu moins d'une vingtaine d'années, les systèmes d'information sont passés de systèmes centraux à un système distribué. Un système distribué est une collection hétérogène d'ordinateurs reliés par un réseau. Par conséquent, les applications informatiques suivent cette évolution pour tirer partie des réseaux. Le Grid est un très bon exemple : les ressources sont réparties sur plusieurs pays, sites, organisations, réseaux et se situent à différents endroits géographiques. L'utilisation intensive des réseaux nous demande désormais de construire un réseau de qualité afin d'assurer une meilleure performance et la disponibilité de notre service.

Grâce aux avancées rapides de nouvelles technologies, la bande passante disponible des réseaux augmente aujourd'hui très rapidement. En revanche, les besoins en communication sont aussi en forte croissance et ces surplus de trafics ne peuvent pas toujours se gérer par la montée en puissance de la capacité des réseaux. Dans ce cas, il est nécessaire d'effectuer des optimisations de la mise en place de l'architecture ou de l'infrastructure pour réduire cet écart. Néanmoins un réseau performant reste un des plus importants facteurs pour assurer la qualité du système.

Un réseau de qualité peut être mesuré par trois critères : la bande passante, la latence et la disponibilité. La bande passante correspond à la quantité d'informations qu'un réseau est capable de transmettre durant un intervalle de temps donné. Il est possible de la mesurer en Mo/s ou en Go/s. la bande passante « utile » (visualisée par l'utilisateur) peut être différente de celle délivrée par le fournisseur.

La latence correspond au délai nécessaire à la propagation d'un signal. Il est possible d'utiliser la commande « ping », une commande réseau de base, pour obtenir des informations et en particulier le temps de réponse d'une machine.

Enfin, la disponibilité est une mesure de performance qui correspond à la période pendant laquelle le réseau est accessible et utilisable. Elle est exprimée classiquement en pourcentage et il

est possible de l'obtenir en divisant la durée opérationnelle d'un réseau par la durée totale d'accessibilité souhaitée.

L'ensemble des serveurs du groupe sont répartis sur les deux sites du groupe et les serveurs sont tous équipés d'une carte réseau de 1 Gbit/s. Il faut souligner que sur le site « CERN », les serveurs sont interconnectés par les switchs de 1 Gbit/s. En revanche, sur le site « Wisconsin », les switchs sont de 10 Gbit/s, car sur ce site, le groupe partage ces ressources pour traiter les travaux Grid.

Pendant cette phase d'analyse, nous avons effectué deux tests sur la latence et la bande de passante « utile » pour nous donner une idée de la performance de notre réseau. Pour les tests de la latence, nous avons utilisé la commande « ping ». Et pour les tests de la bande passante, nous avons utilisé un outil qui s'appelle « iperf ». Cet outil de mesure permet de mesurer la performance d'un réseau. Iperf doit être lancé sur deux machines. La première machine doit lancer la commande « iperf -s » pour être en « mode serveur » et la deuxième machine en mode client. Avec la commande « Iperf -c nom_du_serveur » (il est possible de rajouter une autre option) sur le côté client, ceci nous permettant de tester la performance de notre réseau. Le graphique suivant est une illustration de l'utilisation de l'outil « iperf ».

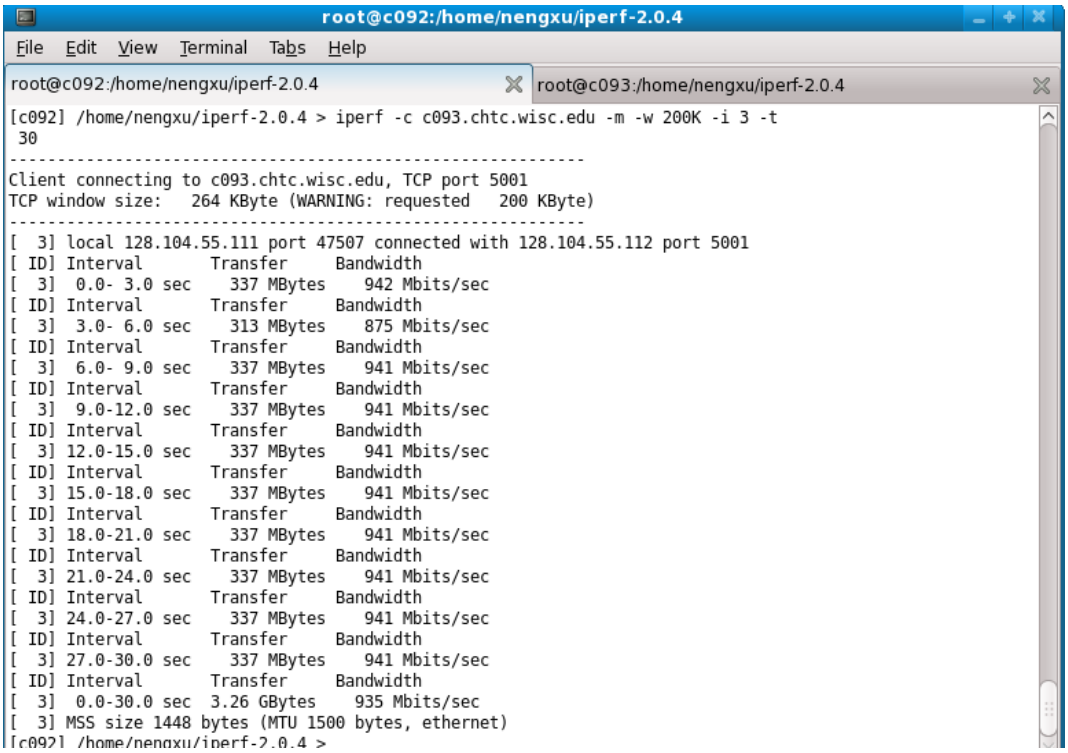

Figure 16 : illustration de l'utilisation de "iperf"

Hormis les tests effectués avec les outils tels que « ping » et « iperf », nous avons également effectué des tests de bande passante en utilisant « Xrootd ». L'utilisation de « iperf » sert à découvrir la bande passante « utile » de notre réseau, alors que le test utilisant « Xtootd »

sert à découvrir la vitesse de transfert des fichiers à travers le réseau local dans un contexte le plus proche possible de la réalité.

Tableau 6 : Test performance du réseau

Outil de Test	Critère	Mesure (en moyenne)
Ping	La latence	0.208ms
Iperf	Bande passante (utile)	116.82 Mo/s
Xrootd	Bande passante (réel)	107.28 Mo/s

Les chiffres du tableau ci-dessus sont des résultats que nous avons recueillis pendant la phase d'analyse. Ces chiffres sont issus d'une suite de tests. Nous pouvons constater que le résultat de la latence est assez bon : seulement 0.208ms. En revanche il y a une dégradation de bande passante d'environ 10Mo/s pour les tests de la bande passante avec l'outil « Xrootd» et « iperf ». Cette différence est sans doute liée à l'utilisation de « Xrootd », car lorsque nous avons besoin d'un fichier distant, le serveur « Xrootd master » doit tout d'abord retrouver le fichier dans quel « Xrootd slave » il se situe puis les serveurs « Xrootd slave » établissent la connexion avec le serveur distant pour commencer à transférer les fichiers.

Le test précédant nous montre que la performance du réseau est l'un des facteurs clé pour construire un système distribué performant. Pour qu'un réseau soit performant nous pouvons nous orienter vers une augmentation de la capacité de notre réseau. En revanche cette solution n'est pas le seul recours car il existe d'autres solutions comme la compression des données ou la réduction de la transaction des données.

A heure actuelle, le passage du réseau de 1 Gbit/s au réseau de 10 Gbit/s est un investissement très coûteux. Un des devis de nos fournisseurs nous le confirmant, la rénovation de l'ensemble des sites pour les équipements réseaux coûterait plus de 230,000$ c'est à dire plus de la moitié du budget informatique du groupe. Un tel budget nous oblige à réfléchir davantage avant toute opération.

10.2 Le stockage

Le Wisconsin groupe dispose en tout de 70 serveurs fichiers, dont 18 au CERN et 48 au Wisconsin. Les 20 serveurs fichiers au CERN sont tous équipés de 8 CPU de 2,66 GHz, 16 Go de mémoire et 24 disques de 500 Go montés en RAID. Concernant les serveurs fichiers au Wisconsin, la configuration du serveur est de 8 CPU de 2,0 GHz, 16 Go de mémoire 8 disques de 750 Go et ils sont montés en RAID.

La technologie RAID est née à la fin des années 1980. Elle permet de stocker des données sur de multiples disques durs classiques. Elle apporte aux systèmes un haut niveau de

performance, une robustesse de tolérance de pannes et des capacités de stockage multi-téraoctets. Il existe deux types de RAID : le RAID logiciel et le RAID matériel. Le RAID logiciel est assuré par un sous-système du système exploitation. Ce type d'implantation est plus économique et elle est facile à administrer. En revanche il entraîne une charge supplémentaire pour des ressources système.

Dans notre cas, nous utilisons le RAID de matériel. Utiliser le RAID matériel représente plusieurs avantages comme par exemple la détection de défauts du disque et le remplacement des disques à chaud. Il permet également de réduire sensiblement la charge du système. Sur les serveurs fichiers du groupe Wisconsin, les contrôleurs RAID de la marque 3Ware sont installés. Ce type de contrôleurs nous offre un autre avantage car on lui intègre une batterie de secours ce qui permet de maintenir la cohérence de leur caches.

Les implantations d'une architecture RAID peuvent se faire à plusieurs niveaux, et chaque niveau correspond à un mode d'utilisation en fonction de critères tels que le coût, les performances, ou la fiabilité des données et du système. Le tableau suivant sert à présenter les différents niveaux de RAID.

En analysant le tableau 6, nous pouvons noter que le RAID 5 présente des atouts pour être utilisé sur les serveurs importants ou critiques. C'est pour cette raison que les serveurs fichiers du groupe utilisent tous la technologie RAID 5. Par contre la mise en place de la grappe de disques entre les serveurs fichiers de 24 disques et celui de 8 disques est légèrement différente. Cette différence peut se traduire par le nombre de disques en mode hot-swap. En effet le disque hot-swap est un disque réserve. On l'installe initialement avec les autre disques de production, habituellement qui ne contient aucune donnée. C'est au moment où l'un des disques de la grappe de production tombe en panne que nous pouvons l'utiliser pour reconstruire le contenu distance. Alors pour les serveurs fichiers de 8 disques, nous avons mis en place un disque hot-swap et deux pour les serveurs de 24 disques.

Tableau 7 : Les différents niveaux de RAID

Niveau	Description	Minimum de disques	Avantages	Inconvénients
RAID0	Appelé aussi « Stripping », ou « entrelacement de disques ». Les données sont réparties sur l'ensemble des disques de la grappe	N ≥ 2	Offre une bonne performance, l'accès aux données se faisant simultanément sur plusieurs disques. La capacité de stockage est préservée.	La perte d'un seul disque fait perdre l'ensemble des données
RAID1	Appelé aussi « mirroring ». Chaque disque de la grappe contient exactement les mêmes données	N ≥ 2	Il apporte une grande tolérance aux pannes	Il n'y a pas de gain de performance. Le coût est plus élevé. La capacité de stockage n'est pas préservée.
RAID5	Il combine l'utilisation du « Stripping » et celle du « parity checking » (contrôle de parité).	N ≥ 3	Il permet d'allier fiabilité (en cas de défaillance d'un disque. Grâce à la parité nous pouvons reconstruire les éléments perdus) et performances à un coût réduit.	la capacité de stockage utile réelle est de n-1

Pendant cette phase d'analyse, nous avons également effectué des tests pour mesurer les performances entre différents types de configuration. Le but de ces tests est de comparer les deux types de configuration dans son l'ensemble. Pour effectuer les tests, nous utilisons un programme qui s'appelle « Bonnie++ ». C'est un outil de « benchmark » pour effectuer des tests de performance des disques et le système fichiers. Le tableau suivant est une présentation du résultat de nos tests et ce test est basé sur une moyenne pondérée de 10 tests.

Tableau 8 : Test single disque, 8 disques et 24 disques en RAID 5

Nombre de disques	Lecture séquentielle (MB/s)	Ecriture séquentielle (MB/s)	Accès aléatoires (/s)	Création séquentielle (/s)	Création aléatoire (/s)
1*	78.62	77.21	309.80	843	698
8	139.71	127.23	1486.20	3259	2203
24	158.80	135.36	2381.37	3411	2728

*Ce test est effectué sur le disque système qui n'est pas monté en RAID 5

Le tableau ci-dessus nous montre que l'augmentation du nombre de disques d'une grappe RAID 5 permet de réduire le temps de lecture ou d'écriture mais le gain est assez limité. Le test nous montre aussi que la lecture sur une grappe de disques en RAID 5 est plus rapide que l'écriture. Cette différence peut être due au calcul de la parité. On peut également constater que plus le nombre de disques augmente, meilleure est la performance d'accès aux données. En revanche, plus le nombre de disque d'une grappe RAID 5 augmente plus le temps de reconstruction d'un disque défaillant augmente car la reconstruction des éléments manquants nécessite de lire tous les éléments de tous les autres disques.

Nous pouvons également noter deux autres points se rapportant à la performance dans ce tableau. 1er point : en augmentant le nombre de disques dans une grappe en RAID5, il est possible d'améliorer la performance d'accès aux données. 2ème point : au niveau de la gestion des fichiers, le RAID 5 gère nettement mieux en mode séquentiel qu'en mode aléatoire.

Le stockage tout comme le réseau, est un élément principal pour construire l'infrastructure des matériels de notre système. Leur fiabilité, capacité, disponibilité et performance sont des points essentiels pour assurer le bon fonctionnement de notre système. À l'heure actuelle, notre système a une capacité de stockage proche de 500 To ; en revanche les données stockées ont atteint environ 80 pour cent de la capacité de son stockage. Avec ce constat, nous devons prendre en compte ce problème pendant les prochaines phases telles que l'élaboration de la stratégie et la conception.

10.3 Les serveurs

Dans les deux sections précédentes, nous avons étudié la mise en place du réseau et le stockage d'infrastructure des matériels du groupe. Le Wisconsin groupe a une ressource informatique très importante en comparaison avec les autres instituts de type Tier 3. Le groupe dispose d'un parc informatique très impressionnant.

Le but de cette section est de faire un état des lieux de nos serveurs et de fournir une vue aérienne de l'ensemble des serveurs répartis sur les deux sites de production. Cette étude pourrait également servir pour définir la politique d'achat des serveurs. A travers le tableau 9, on constate que Wisconsin a un nombre important de serveurs stockage. De plus, chaque serveur est équipé de 8 processeurs. Pour les 40 serveurs de calculs, deux serveurs sont dédiés aux travaux « PROOF » et les autres sont utilisés pour des travaux provenant de Grid et des travaux « Condor ». Les serveurs « Middleware » sont des serveurs middleware qui permettent d'orchestrer des travaux soumis par les utilisateurs internes ou externes. Ils permettent aussi d'assurer le bon fonctionnement entre différents services. En comparaison avec le tableau 10, nous pouvons constater que sur le site Wisconsin il y a beaucoup plus de serveurs middleware que sur le site du CERN. La raison est simple : Le site Wisconsin partage ses ressources informatiques pour les travaux Grid, et pour que ce partage soit réalisable nous avons besoin de mettre en place plusieurs services qui permettent le dialogue entre les différents composants d'une application répartie.

Tableau 9 : Présentation des serveurs du site Wisconsin

Serveur (Nœud)	Nombre de serveurs	CPU par serveur	Mémoire par serveur	Capacité de stockage par serveur	Lieu
Stockage (CHTC)	50	8*2.66GHz	16 GB	8*750Go (RAID 5) + 160 Go (système)	Wisconsin
Calcul (CS)	40	4*2.80GHz	8 GB	80 Go (système)	Wisconsin
Middleware	12	8*2.66GHz	16 GB	8*750Go (RAID 5) + 160 Go (système)	Wisconsin
Nombre total de CPU		656			

Nous pouvons noter que les serveurs calcul au Wisconsin sont assez homogènes ; en revanche ce n'est pas le cas du côté du CERN. La configuration entre chaque nœud des serveurs calcul est très différente.

Tableau 10 : Présentation des serveurs du site du CERN

Serveur (Nœud)	Nombre de serveurs	CPU par serveur	Mémoire par serveur	Capacité de stockage par serveur	Lieu
Stockage (PCUWDATA)	20	8*2.0GHz	16 GB	24*500Go (RAID 5) + 160 Go (système)	CERN
Calcul (PCUW)	35	2*2.80GHz	2 GB	80 Go (système)	CERN
Calcul (PCUWTWIN)	30	8*2.33GHz	16 GB	250 Go (système)	CERN
Calcul (PCUWSUN)	40	4*2.20GHz	4 GB	80 Go (système)	CERN
Middleware	4	2*2.80GHz	2 GB	160 Go (système)	CERN
Nombre total de CPU		638			

Par rapport au tableau 9, ici, nous n'avons plus que 4 serveurs middleware pour gérer les services « Condor », « PROOF », « Xrootd » et « une base de données production locale ». Sur ce site nous disposons de beaucoup plus de moyens de calcul (beaucoup plus de CPU). Sur ce site le groupe a également mis deux serveurs pour exécuter exclusivement des travaux « PROOF ». Ces deux serveurs sont souvent libres car les travaux « PROOF » soumis par les utilisateurs sont ponctuels. Cette situation est totalement contraire à celle des serveurs exécutant les travaux « Condor ».

Bien utiliser les CPU pourrait aider nos chercheurs à avancer convenablement dans leurs recherches, et à exploiter au maximum le potentiel de calcul nous permettant de faire des économies sur les achats des matériels. Pour cette raison, pendant cette phase d'analyse, nous avons surveillé régulièrement l'utilisation des CPU de chaque type de serveur. Le tableau suivant permet de résumer les informations récoltées pendant cette phase.

Tableau 11 : Taux d'utilisation des CPU

Type de serveur	Taux d'utilisation de CPU en moyenne
Middleware	5%
Calcul	98%
Stockage	10%

Ce tableau nous montre que les serveurs « Middleware » sont les moins sollicités par les traitements. Les serveurs calculs sont les serveurs les plus occupés entre eux, car ce sont eux qui traitent l'ensemble des travaux demandés par les utilisateurs. En revanche, on constate que les CPU des serveurs stockage sont mal exploités. En effet, chaque serveur stockage équipe 8 processeurs de haute performance et si on compte l'ensemble des CPU de ces serveurs, ils occupent presque la moitié des CPU de l'ensemble du parc informatique, mais on exploite uniquement 10% de leur puissance. Une réorganisation de l'exploitation de cette précieuse ressource nous permettra de regagner en puissance de calcul pour notre système.

10.4 Conclusion

Dans ce chapitre, nous avons parcouru l'infrastructure matérielle de notre système et nous avons étudié le réseau, le stockage et les serveurs. L'augmentation de la bande passante pourrait nous apporter un gain de performance, mais cette opération coûtera très cher. Nous avons aussi détecté que dans un future proche on pourrait manquer d'espace de stockage pour nos données. Augmenter cette capacité pourrait donc être l'une des hautes priorités pour la provision d'achat de matériels. Enfin, du côté des serveurs, nous constatons que certains serveurs sont moins bien exploités, l'utilisation de leurs CPU étant en moyenne de 10%. Pendant la phase d'élaboration de la stratégie et de la conception, nous devrons réfléchir à la manière dont nous pourrons utiliser ces ressources un peu oubliées.

Chapitre 11 Le clustering

En informatique, le terme cluster peut désigner une grappe de plusieurs serveurs (deux au minimum) qui est capable de supporter des niveaux de charge très élevés tout en offrant un haut niveau de disponibilité. Aujourd'hui, les clusters jouent un rôle de plus en plus important dans les systèmes d'information. D'une part les applications d'aujourd'hui sont de plus en plus gourmandes en ressource (qu'un seul serveur peut difficilement assumer à lui seul), d'autre part les contraintes de disponibilité nous conduisent à mettre en redondance les serveurs pour éviter l'arrêt complet de service ou de production en cas de panne matérielle. Dans ce chapitre, nous allons étudier les formes de scalabilité et de clustering, et cette étude nous permettra d'évaluer la force et la faiblesse de notre système d'information.

11.1 Les différentes formes de scalabilité

La scalabilité d'un système décrit la capacité de ce système à faire face à des charges de traitements augmentant de manière importante. Ces charges peuvent être dues à une augmentation du nombre de connections ou une croissance importante du volume des travaux à traiter.

Il existe deux approches fondamentales différentes pour mettre en œuvre la scalabilité :

- La scalabilité verticale qui est la possibilité d'augmenter la puissance de traitement d'un composant en lui allouant davantage de ressources physiques, comme par exemple des processeurs, de la mémoire ou des disques ;
- La scalabilité horizontale qui est la possibilité d'augmenter la puissance de traitement en lui ajoutant des serveurs d'un type donné.

Il existe une troisième solution alternative dite « scalabilité diagonale ». En effet, elle utilise conjointement la scalabilité verticale et la scalabilité horizontale pour augmenter la puissance de traitement.

Nous avons présenté les trois types de scalabilité, et nous pouvons observer que les frontières entre différentes formes de scalabilité ne sont pas toujours bien délimitées. Malgré cette difficulté, il est possible de résumer les architectures utilisant la scalabilité verticale par l'utilisation simultanée de plusieurs processeurs pour une même application. Les architectures utilisent la scalabilité horizontale par l'utilisation simultanée de plusieurs serveurs pour une même application. Enfin, pour la scalabilité diagonale, nous pouvons la résumer par l'utilisation de plusieurs instances de systèmes d'exploitation qui utilisent chacune plusieurs CPU pour effectuer les traitements demandés. Le choix d'une solution dépendra, d'une part du besoin des

applications, d'autre part du budget disponible. Le tableau suivant est un comparatif des différentes formes de scalabilité.

Tableau 12 : Comparatif des différentes formes de scalabilité

Type de scalabilité	I/O	Nombre de CPU par serveur	Coût par CPU	Améliore la disponibilité	Administration
Verticale	Débit élevé Latence [1]faible	128 ou plus	Elevé	Non	Simple
Horizontale	Débit moyen Latence élevé	1 à 4	Faible	Oui	Complexe
Diagonale	Variable	8 à 24	Moyen	Oui	Intermédiaire

Chacune de ces approches a ses spécificités et elles présentent des avantages et des inconvénients bien différents. La solution scalabilité verticale présente des avantages très intéressants : par exemple l'administration de ce type de solution est simple et l'impact sur la conception et les développements est faible. De plus, la transaction des données se fait à grande vitesse et avec une faible latence car ces échanges se font sur le bus local et il n'y a pas de transaction via le réseau extérieur. En revanche ce type de solution pourrait être très coûteux et la disponibilité n'est pas assurée, comme par exemple les opérations de maintenance nécessitant généralement l'arrêt du système.

Grâce à la redondance des serveurs, la solution scalabilité horizontale s'accompagne d'une augmentation très sensible de la disponibilité du système. Elle offre également un meilleur rapport coût/performance vis-à-vis des superordinateurs pour les applications scientifiques. L'objectif de ce type de solution est de rechercher la performance au moyen du parallélisme des applications. En effet, dans notre système, la mise en place des serveurs calculs pour les travaux « PROOF » correspondent exactement à ce type d'architecture. Ce type de solution a bien ses limites ; par exemple elle nécessite beaucoup d'efforts pour développer des applications avec des contraintes particulières ; l'administration du système n'est pas toujours simple et elle alourdit la charge du réseau en raison des échanges de messages intenses.

[1] Latence : le temps écoulé entre le moment où on initie une action et celui où on constate le résultat de cette action. [11]

La solution scalabilité diagonale est une solution hybride. Elle fait l'objet d'un compromis entre les deux solutions fondamentales qui permettent de cumuler les avantages des deux formes de scalabilité tout en limitant les inconvénients.

Dans le chapitre 10, section 10.3 « les serveurs », nous avons expliqué que les serveurs, pour traiter les travaux « PROOF », ne sont pas assez utilisés. En revanche, lorsqu'un utilisateur soumet des travaux, le temps nécessaire pour finaliser ces travaux est très long. Nous pensons que ce type de problème peut survenir au cours de la mise en place d'une solution de scalabilité horizontale, car il génère des échanges de données à travers le réseau local (LAN) et ce type de réseau présente une latence plus élevée et un débit moindre qu'avec un bus d'échange interne. Ce type de problème peut également venir de la mise en place insuffisante des serveurs calculs « PROOF » pour gérer une montée en charge très élevée des travaux « PROOF ». Enfin pour régler ce problème nous devons analyser et adopter une solution adéquate pendant la phase de stratégie et la conception pour satisfaire les besoins des utilisateurs.

11.2 Les différentes formes de clustering

A l'intérieur du cluster, le rôle des différents nœuds peut être différent. Il existe deux formes distinctes de clustering : les clusters actif/passif et les clusters actif/actif.

Les clusters actif/passif sont aussi appelés « standby cluster ». Ils sont construits sur un principe de redondance passive. Tout les services sont exécutés sur le cluster actif et lorsque ce serveur devient indisponible pour une raison quelconque, un autre cluster, appelé passif, prend la main sur ces services. Ce type de solution améliore le niveau de disponibilité car les serveurs sont montés en parallèle. La mise en place et l'administration est simple pour ce type de cluster car il n'y a qu'un seul serveur actif et il n'y a pas d'accès concurrent à gérer. Par contre, la mise en place de ce type de cluster double le coût d'investissement sur le plan du matériel.

Les clusters actif/actif sont construits sur un principe de redondance active. Le travail est transmis aux serveurs actifs. Lorsqu'un serveur devient indisponible pour une raison quelconque, l'autre serveur prend en charge la totalité du travail. C'est grâce à un montage de composants en parallèle que ce type de cluster permet d'améliorer la capacité de montée en charge et de disponibilité de l'application. En revanche, la mise en œuvre de ce type de configuration est plus complexe que les clusters actifs/passifs.

En analysant la mise en place des formes de clustering du groupe, l'ensemble des serveurs calculs sont de type actif/actif. En revanche, pour les serveurs middleware, la plupart de ces serveurs sont seuls, les redondances actives ou passives sont inexistantes. Le tableau suivant est une représentation des serveurs middleware du groupe.

Tableau 13 : Prséntation des serveurs Middleware du groupe

Lieu	Type du serveur	Nombre	Description
Wisconsin et CERN	Base de données production locale (Wisconsin)	1	C'est une base de données qui contient des informations de la production du groupe ainsi que les informations des Datasets.
	Condor master	1	Il permet de gérer des travaux en utilisant le batch système « Condor »
	PROOF master	1	Il permet de gérer des travaux en utilisant le mode parallèle avec « PROOF »
	Xrootd master	1	Il permet d'orchestrer la gestion des fichiers
Wisconsin (Grid)	Dq2 serveur	1	C'est un middleware Grid développé par ATLAS. Il contient deux principaux composants « Bookkeeping système » et « Local site service ». Il permet de gérer le transfert de données en s'appuyant sur la technologie Grid.
	Grid Gatekeeper	1	Il est le responsable de la gestion des requêtes d'allocation.
	GridFTP serveur	4	Il permet de transférer des données de façon performante et sécurisée.
	LFC serveur	1	Il permet de gérer l'espace de noms logiques des fichiers dans le Grid
	SRM serveur	1	Il permet de gérer l'espace de noms physiques des fichiers et l'allocation de l'espace dans un environnement Grid

Le tableau précédant nous montre qu'au Wisconsin le groupe s'équipe de 12 serveurs middleware et de 4 au CERN. Mis à part les serveurs GridFTP qui ont une redondance active, les autres n'ont ni redondance active, ni redondance passive. Ceci constitue un risque important pour la production car lorsqu'un de ces serveurs tombe en panne cela peut entraîner l'arrêt de la production pour l'ensemble du service. Nous devons prendre ce problème en considération pendant la phase de conception.

11.3 Conclusion

Dans ce chapitre nous avons évoqué les différentes formes de clustering et de scalabilité. Nous avons formulé trois formes de scalabilités, toutes trois nous permettant de faire face à la

montée en charge des travaux. En revanche le choix d'une forme de scalabilité doit prendre en compte la contrainte budgétaire, les moyens techniques et la qualité attendue pour les applications.

Il existe deux formes de clustering, les clusters de type actif/passif et les clusters de type actif/actif. La mise en place d'une forme de clustering permet d'améliorer la disponibilité. En revanche, l'absence de la mise en forme de clustering actif/passif ou actif/actif pour les serveurs middleware augmente fortement le risque d'interruption pour l'ensemble de service. Pour cette raison, nous devons prendre en compte ce type de problème dès la phase d'élaboration de la stratégie et de la conception, pour que notre système soit plus solide, plus réactif et capable de supporter une charge de travail en constante évolution.

Chapitre 12 Le flux de données

Dans les chapitres 10 et 11, nous avons étudié la mise en place d'une infrastructure matérielle et le clustering de notre système. Nous avons étudié les stockages, le réseau et les serveurs pour la mise en place d'une infrastructure. Pour le clustering nous avons étudié les différentes formes de clustering et les différents types de scalabilité. Dans ce chapitre, nous allons nous intéresser au flux de données. Cette étude nous permettra de comprendre pour quelle raison nous avons besoin de transférer des données, de distinguer différents types de flux de données, de connaître leurs caractéristiques et les enchaînements effectués par chaque type de flux.

Selon notre estimation, notre système doit supporter une charge de travail d'environ 500 Go par jour. Une telle charge de travail nous demande de bien veiller sur notre système et de l'optimiser pour en assurer la performance, la sécurité, la robustesse et la fiabilité. Une étude de flux de données nous permettra non seulement de mieux le connaître mais également de réduire les efforts à fournir pour élaborer la stratégie et la conception.

Pour caractériser le flux de données, nous pouvons le classer en deux types de flux : le flux de données interne et le flux de données externe. Le flux de données interne correspond aux données transitant à l'intérieur de notre système. Nous pouvons caractériser le flux de données externe par des transactions de données effectuées entre notre système d'information et une autre VO (Virtuelle Organisation).

12.1 Le flux de données interne

Pour le flux de données interne, il existe deux cas de figure. La première correspond aux données provenant de l'extérieur du site et ensuite distribuées sur différents serveurs fichiers par le Xrootd.

Dans le deuxième cas, le flux de données interne peut être généré par des souscriptions des travaux internes. Ces travaux sont généralement des travaux Condor ou de PROOF, et ils sont envoyés par les physiciens du groupe.

Lorsque les serveurs calculs reçoivent les travaux, les données à traiter vont être envoyées à partir des serveurs fichiers pour aller sur les serveurs de calculs correspondants. Les données vont ensuite être traitées par les serveurs calculs et les résultats vont être renvoyés aux serveurs fichier via Xrootd. Ce type de flux, nous pouvons l'illustrer de la façon suivante :

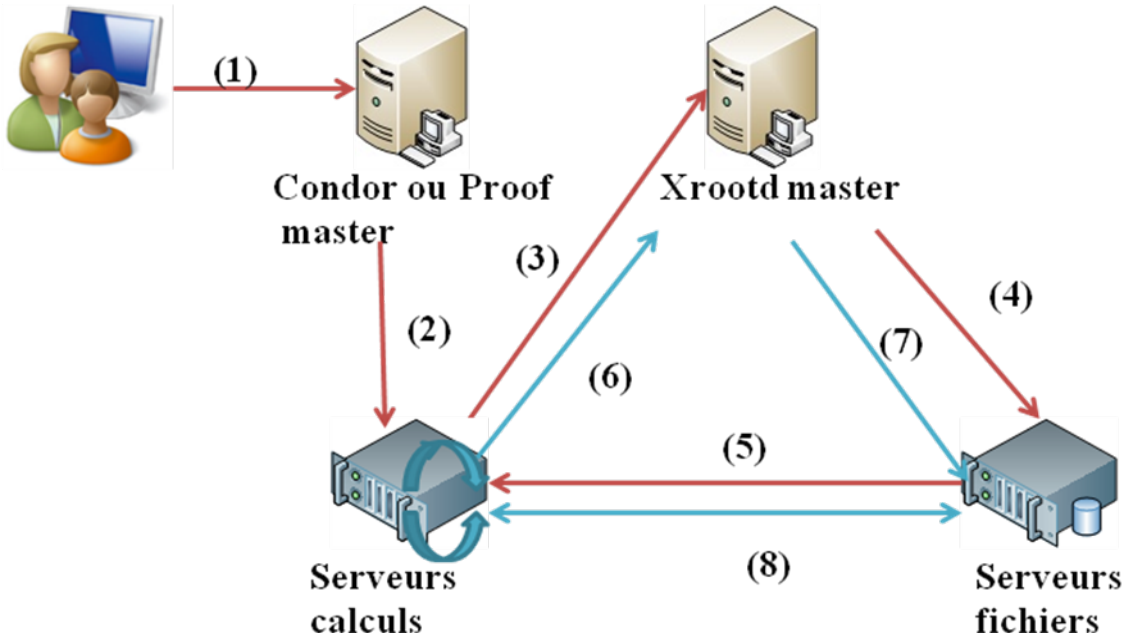

Figure 17 : Illustration du flux de données interne

Dans la figure ci-dessus nous pouvons constater que le flux de données interne peut être découpé en 7 étapes. Les étapes sont les suivantes :

1. Un utilisateur souscrit un travail au système Condor ou PROOF master,
2. Le Condor ou PROOF master va chercher les serveurs disponibles et leur confient la tâche.
3. Les serveurs calculs demandent au Xrootd master où sont les données à traiter,
4. Le Xrootd master envoie la demande pour retrouver les serveurs fichier qui contiennent les données à traiter,
5. Les serveurs fichiers envoient les données correspondantes aux serveurs calculs,
6. Les serveurs calculs obtiennent les résultats de calculs et demandent au serveur Xrootd master les résultats sur lesquels seront envoyés les serveurs fichiers ;
7. Le Xrootd master vérifie la disponibilité des serveurs fichiers et ordonne le transfert des résultats,
8. Les serveurs calculs et les serveurs fichiers établissent des connections et les résultats sont transférés sur les serveurs fichiers.

Voici un exemple concret pour le transfert de données internes. Selon une estimation, après le redémarrage du LHC notre système va accueillir plus de 500Go de données par jour dont une bonne partie pour la transaction interne. Il est donc important d'assurer la qualité de flux de données interne pour pouvoir garantir la performance de notre système.

12.2 Le flux de données externe

Le flux de données externe se distingue par des transferts de données entre les VOs. Un transfert de données peut être prédéfini ou demandé par un utilisateur via le système Panda[1]. Pour qu'un flux de données externe soit fiable et gérable, différents services sont impliqués dans la transaction. Ce sont les services suivants : la planification de transfert de données, le service de la localisation et de transfert, la gestion des files d'attentes, l'agent de transfert.

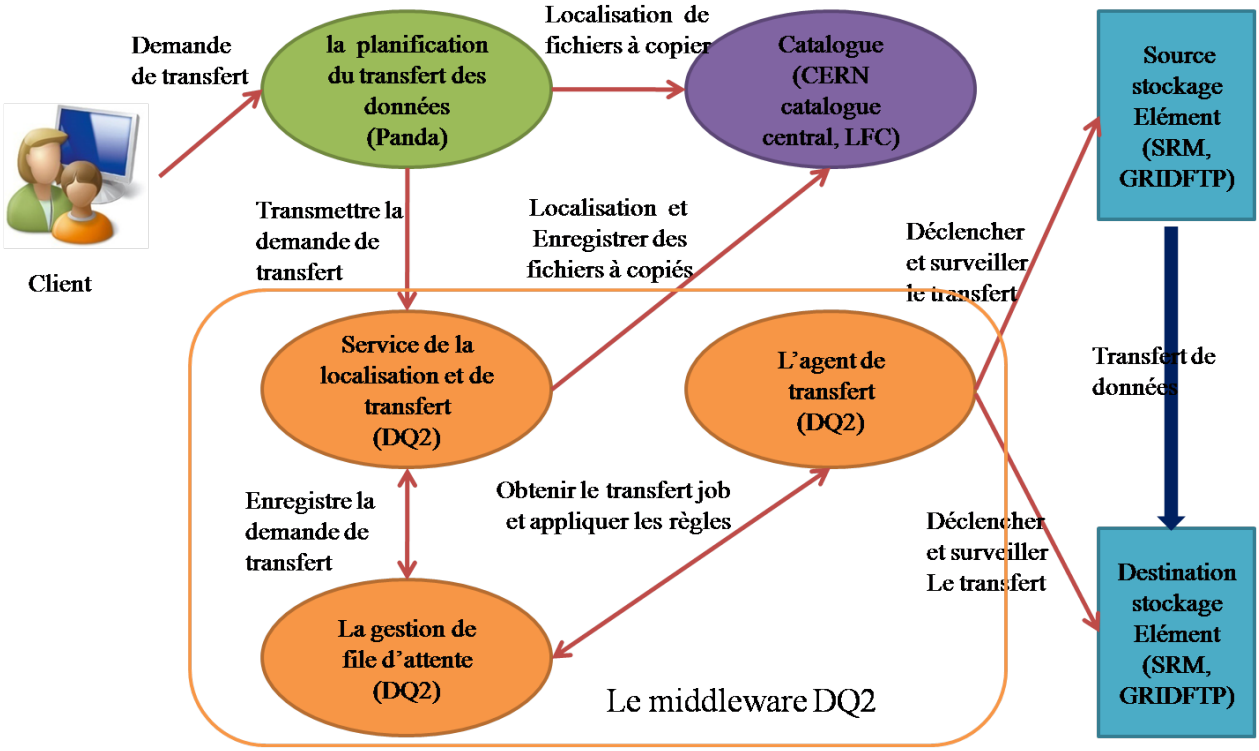

Figure 18 : Présentation du chemin du flux de données externe et ses composants

Dans la figure 18 nous pouvons constater que de nombreuses technologies sont impliquées dans un transfert de données entre différentes VOs. La planification du transfert des données est un service de haut niveau, il gère la politique globale des VOs, il offre une interface qui permet de planifier le transfert de données, il permet également de gérer les tâches de transfert et maintenir la tâche jusqu'à la fin du transfert. Pour soumettre une demande de transfert de données, les utilisateurs peuvent se rendre sur l'interface Web du système Panda pour effectuer une demande. Lorsque le système « Panda » reçoit la demande, il va interroger le « Central Catalogue » pour obtenir les informations nécessaires pour le transfert.

Le « Central Catalogue » est une base de données, et il contient quatre tables. La première table est « Dataset Repository » qui contient tous les noms de Dataset et leurs Unique IDs. La

[1] Panda : il signifie Production And Distributed Analysis system, ce système a été dévéloppé par ATLAS, depuis l'été 2005 pour répondre aux besoins de la gestion des données et de la production distribuée.

deuxième table est « Dataset Content catalog » qui contient des informations concernant les contenus de chaque Dataset. La troisième table est « Data location catalog » qui enregistre l'emplacement de chaque Dataset. La quatrième table est « Dataset Subscription Catalog » qui enregistre l'historique de la souscription de Datasets.

Une fois que l'utilisateur soumet un transfert vers une destination souhaitée, le système Panda va confier la demande avec les informations tels que GUID [1] et LFN [2] au « service de la localisation et de transfert » du middleware DQ2. Ce service est implanté sur les deux sites, le site source et le site destinataire et il est assuré par le middleware DQ2. Quand le site source reçoit la demande de transfert, le DQ2 interroge le LFC en combinant avec GUID et LFN pour obtenir la localisation physique du fichier (SURL [3]). Et quand le site destinataire reçoit la demande de transfert provenant du système Panda, il va également interroger le catalogue LFC pour s'assurer que les fichiers devant être transférés ne sont pas déjà présents sur le site.

Lorsqu'il n'y a pas de problème, le serveur DQ2 de chaque côté du VO va créer une base de données temporaire. Cette base de données sert à manipuler les informations concernant des données à transférer et son statut. Elle appartient au service de la « gestion de file d'attente ». Cette base de données est manipulée directement par l'agent de transfert et le service de la localisation et de transfert.

L'agent de transfert est un service qui permet de réunir différents acteurs du système pour appliquer la politique de chaque VO et la règle du canal de transfert, surveiller l'état de transfert, déclencher et surveiller l'état actuel de transfert de chaque fichier et mettre à jour la base de données temporaire lorsque l'état de transfert est changé.

Le transfert de données entre deux VOs est assuré par le SRM et GridFTP. Le SRM gère l'allocation de l'espace dans un espace de stockage partagé et il permet d'améliorer l'efficacité de transfert. Et le GridFTP occupe le transfert de données physique entre deux VOs. Wisconsin groupe a quatre serveurs GridFTP. La mise en place de quatre serveurs GridFTP permet de réduire la charge de travail de chaque serveur et d'améliorer la performance de transfert à l'intérieur du site. Les données stockées sur les serveurs GridFTP étant temporaires, elles vont être par la suite transférées et sauvegardées définitivement sur les serveurs fichiers. Le transfert

[1] GUID : il signifie Globally Unique Identifier. C'est un identificateur unique, voici un exemple de GUID : guid : 3F2504E0-4F89-11D3-9A0C-0305E82C3301

[2] LFN : il signifie Logical File Name. C'est un lien symbolique qui permet de combiner avec GUID pour retrouver le PFN. Il est sous la forme lfn:/grid/wisconsin/Atlas/test1240.root

[3] SURL : il signifie Site URL, on l'appelle également Physical File Name (PFN). Il est sous la forme srm://atlas07.cs.wisc.edu:8443/srm/v2/server?SFN=/atlas/xrootd/test.root

de données à partir des serveurs GridFTP vers les serveurs fichiers est facilité par le système Xrootd.

Grâce à cette présentation rapide mais relativement complexe, nous pouvons constater que de nombreuses technologies sont impliquées dans ce type de flux et le volume de transfert est considérable. Ce flux nous permet d'accueillir des nouvelles données que nous avons besoin d'analyser. Il est donc très important d'assurer ce flux en état de bon fonctionnement.

12.3 Conclusion

Dans ce chapitre nous avons pris connaissance de deux types de flux de données. Le rôle du flux de données externe est d'assurer le transfert des données via Grid. En revanche le flux de données interne sert aux transactions de données internes de notre système d'information. Le flux de données interne est généralement généré par les utilisateurs internes.

Le redémarrage du LHC entrainera une augmentation de la charge de travail assez conséquente pour notre système, que ce soit pour le calcul scientifique ou pour le réseau. L'étude effectuée dans ce chapitre nous permet non seulement de connaitre les différentes formes de transaction de données effectuées par notre système mais également leurs points forts et leurs points fables. Le constat établi dans ce chapitre nous servira pour proposer des améliorations possibles pendant la phase de modélisation de la stratégie et de conception pour assurer la sécurité, la qualité et la fiabilité du transfert et répondre aux besoins des utilisateurs.

Chapitre 13 La production

Aujourd'hui dans de nombreux projets de différents domaines de recherches scientifiques, le système d'information joue un rôle essentiel pour assurer la qualité de la recherche et son avancement. Bien entendu, le Wisconsin groupe suit logiquement ce chemin. Pour garantir la qualité du système, il est nécessaire de bien gérer les différentes activités de notre système. Dans le chapitre précédent, nous avons étudié le flux de données pour comprendre les étapes parcourues pendant une transmission de données. Dans ce chapitre nous allons nous focaliser sur une autre activité principale de notre système : la production.

Dans les sections suivantes, nous allons présenter et analyser cette activité de notre système, donc la production. Pour cela nous allons aborder les trois types de travaux : ceux provenant du Grid, les travaux Condor et les travaux PROOF. Par la suite nous allons étudier les programmes que nous utilisons pour surveiller et mesurer l'activité du système.

13.1 La production : Grid

Chaque jour notre système reçoit des travaux à traiter, provenant du Grid. Ces travaux sont gérés par le système « Panda ». Par exemple, un chercheur peut souscrire des travaux à notre système, mais dans la plupart des cas, les travaux du Grid sont des tâches prédéfinies à l'avance dans le système « Panda ». Ces travaux sont généralement des transformations des fichiers AOD en fichiers DPD en utilisant des programmes développés par ATLAS. Pour que notre système achève un travail provenant du Grid, il est nécessaire d'enchainer trois étapes :

- La première étape est celle de la réception des données. C'est la VO supérieure qui nous transmet les données correspondantes pour le travail et notre système les enregistre sur les serveurs stockages. Cette démarche est très proche du flux de données externe que nous avons présenté dans le chapitre précédant.
- La deuxième étape est celle du traitement des données. Une fois que la transmission de données est terminée, le système « Panda » va demander à notre système de traiter les données transférées et notre système va exécuter ces ordres.
- La dernière étape est celle de la retransmission des résultats. Lorsque notre système termine le traitement des données, il va sauvegarder une copie des résultats sur notre site.

Voici les trois étapes pour achever les travaux provenant du Grid. Nous constatons que la première étape est un transfert de données et nous avons déjà présenté son fonctionnement dans le chapitre précédent. Lors de la dernière étape, lorsque notre système achève le traitement, il va

sauvegarder les résultats sur nos serveurs fichiers en utilisant Xrootd dont nous avons déjà présenté le principe dans le chapitre 6 qui est consacrée au Xrootd. En revanche dans les paragraphes suivants nous allons comprendre comment le système Grid gère la deuxième étape « le traitement des données ».

Comme dans l'étape de transfert de données, le système « Panda » est impliqué également fortement dans la gestion de traitement de données. En effet le système « Panda » joue un rôle de chef d'orchestre dans la gestion de production de l'expérience ATLAS. La figure suivante est une illustration de traitement des travaux via le système « Panda ».

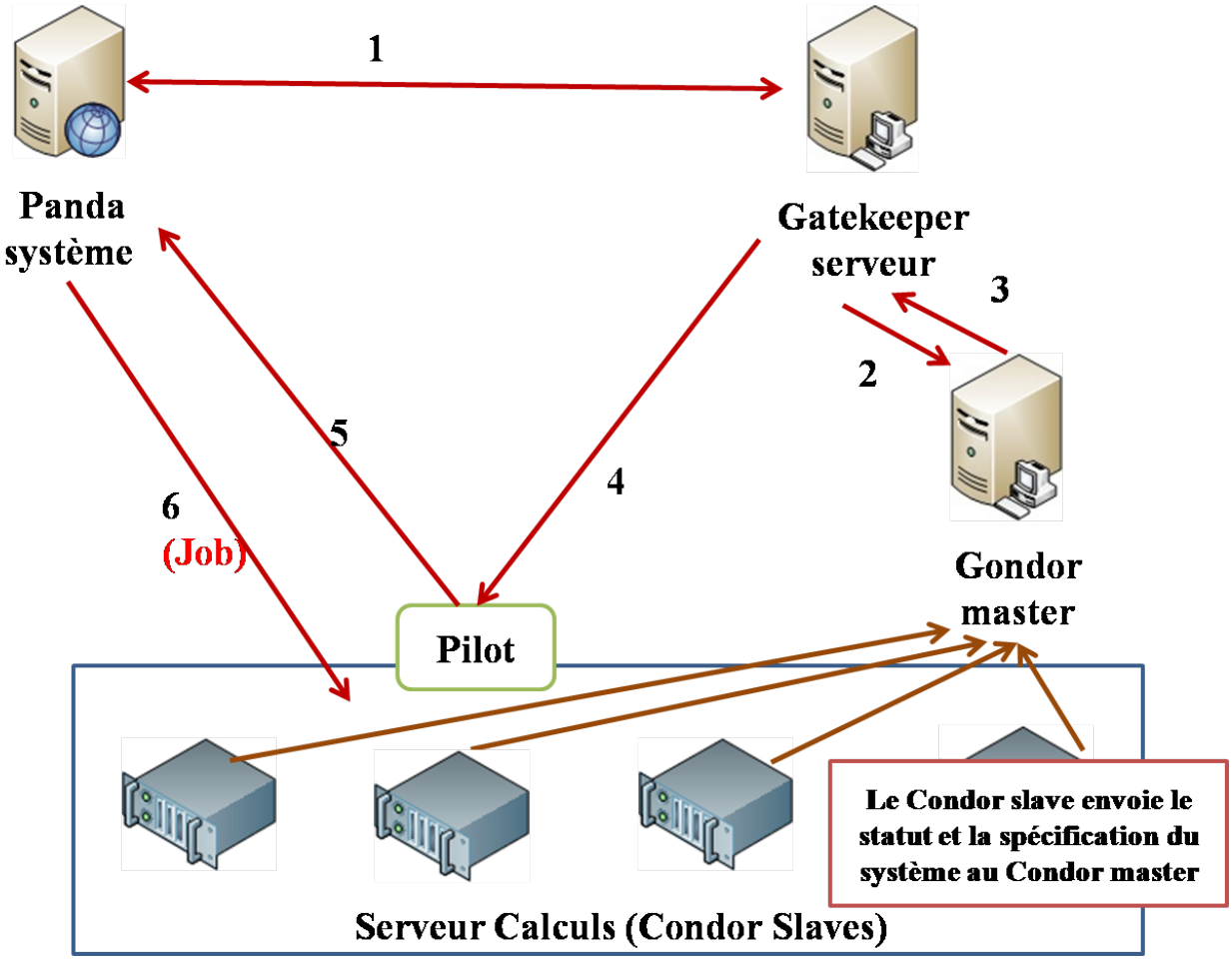

Figure 19 : Travaux Grid - Pilot

Pendant le transfert de données, le système « Panda » commence déjà à organiser le traitement des données. Pour achever les travaux demandés initialement, nous devons enchaîner six étapes pour que notre système les accomplisse. Dans cette phase, différentes technologies sont impliquées dans ce cycle.

Après avoir transféré des données, le système « Panda » reprend des choses en main. Dans un premier temps, il va essayer de se connecter sur le serveur Gatekeeper du Wisconsin en lui fournissant des clés et certificats en adéquation avec la configuration du GSI (voir le chapitre

8). En même temps, le Panda envoie une série de petits programmes qu'on appelle « Pilots[1] » au serveur « Gatekeeper » du Wisconsin via le Condor-G. Ces Pilots servent à réserver des serveurs calculs pour les traitements et envoyer des paramètres des serveurs calculs au système « Panda ». Il faut souligner que dans un premier temps les vrais travaux ne sont pas envoyés à notre système car ils sont stockés temporairement sur le système « Panda ».

Une fois que le serveur Gatekeeper vérifie et accepte la demande de connexion du système Panda, il va accepter de réceptionner les Pilots envoyés par le système « Panda ». Le Gatekeeper va soumettre les descriptions des travaux au Condor master pour essayer de trouver des machines utilisables. Par la suite, dans l'étape 3, le Condor master envoie des informations au serveur Gatekeeper concernant la mise en relation des jobs « Pilots » avec des serveurs utilisables.

Comme nous pouvons constater dans les étapes 4 et 5, après avoir reçu les informations provenant de Condor master, le Gatekeeper serveur confie des tâches (Pilots) aux serveurs calculs. Les Pilots vont être exécutés sur des serveurs calculs utilisables, puis vont inspecter ces serveurs et envoyer leurs paramètres tels que la puissance des CPUs, la capacité et disponibilité des mémoires et la liste des versions des programmes ATALS disponibles sur le site au Panda système. Enfin, le système « Panda » envoie des vrais travaux directement aux serveurs calculs du Wisconsin pour être exécutés.

Le principe de Grid est de partager des ressources afin d'en faire bénéficier à l'ensemble des participants du projet. Pour assurer le bon fonctionnement du partage des ressources, les administrateurs de chaque VO doivent bien veiller à leurs serveurs contrôles car l'arrêt d'un serveur Middleware comme par exemple le serveur Gatekeeper pourrait provoquer un arrêt du service Grid pour notre site. Nous pouvons noter que chaque administrateur du VO peut décider de partager l'ensemble ou partie de sa puissance de calculs au bénéfice du Grid. Le Wisconsin groupe par exemple partage 50 machines du côté Wisconsin pour les travaux Grid. En revanche, l'administrateur du VO n'a pas le droit de manipuler les travaux provenant de Grid, seuls les administrateurs du service Grid ayant ce privilège.

13.2 La production : Condor

Dans le chapitre 5, nous avons pris connaissance du système Condor. Dans ce chapitre, nous avons présenté ses caractéristiques, son fonctionnement et les outils d'administration de ce

[1] Pilot : est un petit programme prévu pour le « Batch système » qui permet de recevoir des travaux dès que des processeurs deviennent disponibles. Il permet également d'envoyer les informations telles que la puissance de CPU, la taille de mémoire disponible et les versions disponibles des programmes ATLAS sur le site.

système. Le système Condor est développé et maintenu par le département « Computer sciences » de l'université Wisconsin et il est basé sur un mécanisme de file d'attente, un schéma de priorité, une politique de régulation des tâches et une classification des ressources.

Le Condor est un système spécialisé dans l'exécution concurrente d'un nombre de tâches très important. Il prend en charge la collection des informations concernant les états de serveurs calculs, la planification de travaux aux serveurs calculs et la distribution de travaux aux serveurs. Dans la section précédente, nous avons expliqué le rôle du système Condor dans un environnement Grid. Le but de cette section est alors d'étudier l'utilisation de ce système dans un environnement local.

Nous savons que le système d'information du groupe Wisconsin est réparti sur deux sites dont un à Wisconsin et l'autre au CERN. Les utilisateurs du groupe peuvent soumettre leurs Condor jobs soit sur le site au CERN soit sur le site à Wisconsin. Ces travaux sont des travaux de transformation des données AOD en données CBNT en utilisant des programmes développés par les physiciens du groupe.

Pour soumettre un Condor job local, les utilisateurs doivent écrire un script. Nous appelons ce type de script le « job description ». Dans un job description, il est possible de définir différents paramètres pour que le système Condor gère des travaux selon les besoins des utilisateurs. La suite est un exemple de la « job description » que les utilisateurs utilisent quotidiennement pour soumettre des travaux au système Condor.

```
#####################################################
# Mon job description Condor
 #####################################################
Univers = vanilla
Notification   = Never
Initialdir =
/atlas/atlasdumpdisk/mc08.105001.pythia_minbias.AOD.etid068111
Executable = /home/hni/dump_test/dumper_14.5.1.6_tag95a.sh
 Arguments =
root://higgs13.cs.wisc.edu:1094//atlas/xrootd/atlasmcdisk/mc08/AO
D/mc08.105001.pythia_minbias.AOD.etid068111/AOD.068111._00078.poo
l.root.1
Output = dump119237.out
Error = dump119237.err
Log = dump119237.log
Priority = 600
 Should_transfer_files = yes
```

```
 WhenToTransferOutput  = ON_EXIT_OR_EVICT
 Transfer_input_files =
/home/hni/dump_test/dumper_14.5.1.6_tag95a_InstallArea.tar.gz
OnExitRemove = TRUE
 Requirements=(HasAFS_Atlas==TRUE && ARCH == "X86_64") &&
(regexp("chtc.wisc.edu",Machine) || (substr(Machine,0,6)=="glow-
c"))
##################################################
```

Nous pouvons constater dans l'exemple ci-dessus qu'une dizaine de paramètres sont définis dans ce script. Dans les paragraphes qui suivent, nous allons interpréter ces paramètres en quelques mots :

- Univers : est un environnement d'exécution Condor. Il existe 7 valeurs possibles dont vanilla, standard, pvm, scheduler, globus, mpi et java. Dans notre environnement, nous utilisons l'Univers vanilla, qui permet d'exécuter un programme sous l'environnement Condor sans l'avoir compilé pour Condor. De plus, il n'y a pas de checkpoint pour cet environnement : La migration d'un programme vers une autre machine n'est donc pas possible,
- Notification : les utilisateurs sont notifiés par mail quand certains évènements se produisent. Il y a 4 valeurs possibles Always, Complete, Error et Never,
- Initialdir : répertoire de travail (par défaut, le répertoire depuis lequel Condor_submit est appelé),
- Executable : le nom de l'exécutable,
- Arguments : liste des arguments passés au programme, on les sépare par des espaces.
- Output : indique le fichier de sortie (stdout),
- Error : le fichier sortie erreur (stderr),
- Log : le fichier de log Condor,
- Priority : la priorité du travail, par défaut est 0. Plus la valeur est élevée plus le niveau de la priorité est haut,
- Should_transfer_files : 3 valeurs possibles Yes, If_needed et No. Il est utilisé pour indiquer si Condor doit transférer des fichiers depuis ou vers la machine d'exécution.
- WhenToTransferOutput : il permet d'indiquer quand les transferts de fichiers doivent être effectués par le Condor. 2 valeurs possibles : ON_EXIT ne permet d'effectuer le transfert qu'à la fin d'exécution du job ; ON_EXIT_OR_EVICT : les fichiers sont transférés si le job est renvoyé de la machine d'exécution (fin du programme ou éviction),

- Transfer_input_files : il permet d'indiquer que le job requiert d'autres fichiers pour fonctionner. La liste de fichiers peut être séparée par des virgules,
- OnExitRemove : 2 valeurs possibles True ou False. Si c'est True, il permet de supprimer l'exécutable et l'argument dans le répertoire de travail,
- Requirements : expression booléenne définissant les spécifications de la machine et ou de l'architecture système sur laquelle doit s'exécuter le travail.

Ces paramètres sont des valeurs d'entrées pour le système Condor. Ils permettent de renseigner le système Condor comment l'exécution doit s'appliquer. Lorsqu'un fichier de soumission est établi, nous pouvons soumettre le job au système Condor à l'aide de la commande Condor_submit.

Lorsqu'un job description est soumis au système Condor, le Condor va chercher à associer le travail à des serveurs utilisables, organiser le transfert de données à partir des serveurs fichiers aux serveurs calculs et démarrer le travail. Pour transférer les fichiers, nous utilisons le protocole Xrootd et le chemin des fichiers est inscrit dans le job description sous le paramètre « Arguments ». Lorsque le job sera terminé, les résultats seront renvoyés et enregistrés sur les serveurs fichiers en utilisant le Xrootd.

Comme pour la production Grid, le traitement des travaux locaux sollicite diverses ressources, en particulier le CPU, la mémoire, le réseau et le disque. Si nous pouvons considérer que notre système pour le moment gère bien la charge due, en revanche, notre système pourrait très vite atteindre ses limites lors du démarrage de LHC. Alors pour que notre système ne soit pas pénalisé par l'augmentation de volume de travail, nous devons réexaminer notre système et proposer des solutions alternatives.

13.3 La production : **PROOF**

Le PROOF est un système qui permet d'effectuer des calculs parallèles sur de multiples processeurs et de multiples machines. Ce type de calculs a pour objectif d'accélérer l'exécution de calculs complexes. Au Wisconsin groupe, nous avons mis en place deux serveurs dédiés à des travaux PROOF sur chaque site. Il existe deux types de méthodes pour soumettre un travail PROOF : la méthode textuelle et la méthode graphique.

La méthode textuelle permet aux utilisateurs de soumettre un travail à l'aide d'une ligne de commande. Cette méthode exige des utilisateurs d'écrire un script en langage C en insérant des options et des arguments à spécifier pour le traitement. La méthode graphique permet aux utilisateurs de soumettre un travail à l'aide d'une interface graphique conviviale et simple. Les

utilisateurs doivent saisir les options et les arguments via l'interface graphique et par la suite ils peuvent soumettre le travail au système PROOF.

Comme nous le savons, le système PROOF est une extension de framework ROOT et il permet d'effectuer des analyses sur un système distribué. Lorsque nous lançons un travail au système PROOF, PROOF va créer une session de travail qui va d'abord à la découverte des ressources disponibles et partagées. Par la suite il va dispatcher le travail aux ressources de façon optimale pour garantir une meilleure performance.

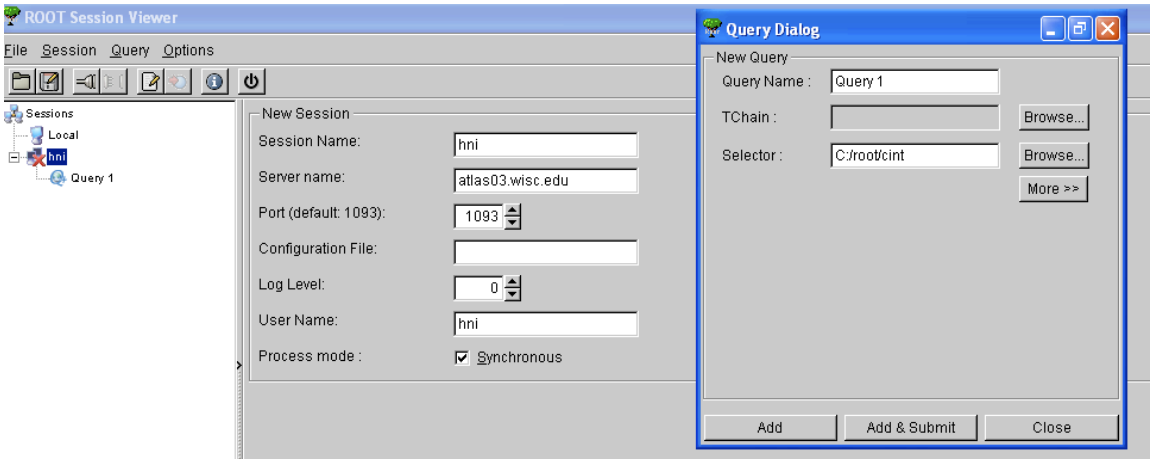

Figure 20 : PROOF interface graphique

La découverte des ressources partagées et l'optimisation de la charge du travail fait partie des deux principales étapes de la gestion des ressources du système PROOF. Les ressources partagées sont prédéfinies dans le fichier configuration du PROOF master. Lorsqu'un utilisateur envoie un job, le système PROOF va tout d'abord vérifier les statuts des ressources partagées et par la suite il va confier le travail aux ressources disponibles et partagées.

L'optimisation de la charge du travail intervient après l'étape de découverte de la ressource partagée. Cette étape est assurée par un module du système PROOF « Packetizer ». Pour accomplir l'optimisation de la charge du travail, le « Packetizer » effectue trois étapes de travail. La première étape est la vérification de la localisation des fichiers. Lorsque nous lançons un job PROOF, nous devons fournir une liste de fichiers à traiter et l'étape de vérification consiste à examiner l'existence de ces fichiers. La vérification est effectuée au niveau du PROOF master en se combinant avec le Xrootd master.

La deuxième étape est la phase de validation. Cette étape consiste à identifier des parties de programme qui ne sont pas interdépendantes, et à cette condition nous pouvons les faire exécuter sur plusieurs processeurs.

La troisième étape est l'étape de création et distribution de paquets. Lorsque les fichiers sont validés, le « Packetizer » va découper des fichiers en paquets et les envoyer sur différents Workers pour être analysés. Cette étape va prendre fin lorsque toutes les données du Dataset sont

transférées aux serveurs calculs (PROOF slave). Lorsque tous les calculs sont terminés les PROOF slave vont envoyer les résultats au PROOF master qui lui va additionner les résultats pour fournir un résultat final.

Dans la phase d'ingénierie des besoins, nous avons beaucoup cité les problèmes avec le système PROOF. Par exemple, la charge de travail des serveurs PROOF est trop restreinte en comparaison avec les serveurs Condor. En revanche lorsqu'on lance un travail PROOF, le temps d'attente est souvent long. En moyenne pour achever un travail PROOF, il faut compter environ 20 minutes. Dans les problèmes que nous avons pu recenser, nous avons aussi mentionné que la gestion des travaux PROOF est inexistante et le temps d'attente pour que les serveurs commencent à traiter des données est long.

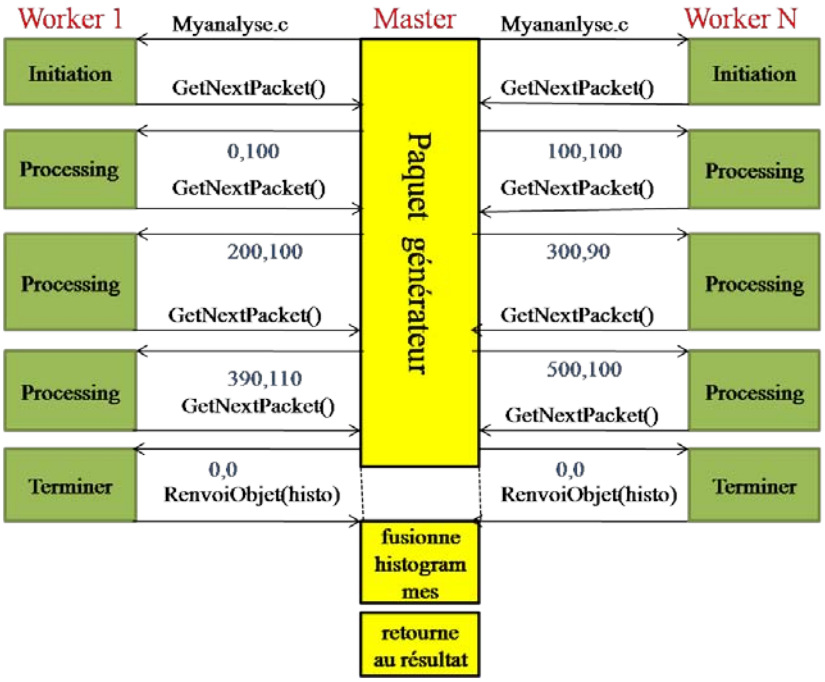

Figure 21 : Le fonctionnement du Packetizer

Le PROOF est un système qui permet d'effectuer un calcul parallèle. Il est capable d'identifier et de partitionner un traitement en tâches élémentaires adaptées et de les faire exécuter simultanément sur plusieurs unités pour réduire le temps d'exécution. Un travail PROOF sollicite beaucoup de ressources. La mise en place insuffisante des ressources dédiées à la production PROOF est principalement le goulot d'étranglement des problèmes de performance. Il est possible d'obtenir un meilleur résultat en allouant plus de ressources au PROOF. En revanche comment réutiliser ces serveurs PROOF pendant qu'ils sont inactifs, est un problème que nous devons résoudre.

13.4 La base de données

Après avoir présenté la production « PROOF », nous allons prendre ici connaissance de la base de données du groupe Wisconsin. Les bases de données étant depuis longtemps au cœur des systèmes d'information, dans notre système d'information (dans lequel existe deux bases de données en production, un au CERN et l'autre au Wisconsin), la base de données est bien un élément indispensable.

La mise à jour de la base de données se fait aider par un script « File_Sync.py » qui se place sur chaque serveur fichier. Ce script est exécuté une fois par jour. Il permet de parcourir l'ensemble des Datasets et de mettre à jour la base de données.

Comme nous le savons, chaque jour, le site Wisconsin reçoit des nouveaux Datasets qui proviennent de Grid. Ces données devraient être transférées sur notre site au CERN. La mise en place de la base de données nous permet d'identifier les nouvelles arrivant sur le site Wisconsin. Et une fois par jour, un programme est exécuté pour transférer ces données à partir du site Wisconsin pour aller sur le site du CERN.

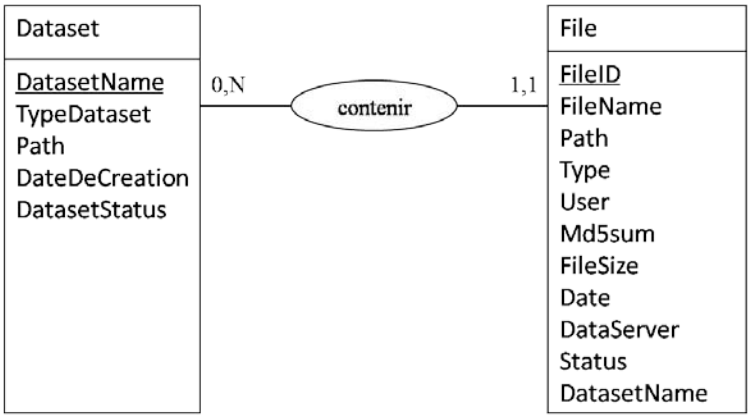

Figure 22 : Modèle des Entités Association de la base de données

Cette mise en place de la base de données est bien utile pour les utilisateurs et le système. En revanche, nous pouvons regretter l'absence d'un mécanisme de mise à jour automatique et la redondance de la base de données alourdit la charge de travail de l'administrateur. Nous pouvons également constater que la mise à jour de la base une fois par jour ne peut certainement pas satisfaire les utilisateurs, d'ailleurs nous avons évoqué ce problème pendant la phase d'ingénierie des besoins. Pour régler ce problème, la solution la plus simple est d'augmenter la fréquence de mise à jour mais cette solution est-elle la meilleure ? Dans la prochaine partie, nous allons tenter de résoudre tous ces problèmes.

13.5 Conclusion

Dans ce chapitre, nous avons commencé par une présentation de la production Grid, et par la suite nous avons évoqué la production Condor et la production PROOF, puis nous avons terminé sur la base de données. Il faut également retenir que, chaque jour, notre système reçoit des centaines de Go de données provenant de Grid, et notre système reçoit également de nombreux travaux locaux émis par les utilisateurs locaux. La charge du travail de notre système est donc très importante. Il faut noter que la production Condor, Grid et PROOF font partie des tâches les plus importantes de notre système. La gestion de la production est une clé importante pour construire un système d'information performant qui permet de répondre aux besoins des utilisateurs.

Conclusion de la troisième partie

Dans cette partie, nous avons commencé par l'ingénierie des besoins et par la suite nous avons présenté le système existant. Ces deux études sont importantes pour l'ensemble de projet. L'ingénierie des besoins nous permet de comprendre ce dont les utilisateurs ont besoin et l'étude du système existant nous permet d'avoir une vue d'ensemble de notre système actuel.

Une mauvaise compréhension des besoins pourraient coûter cher à un projet informatique, d'où l'intérêt d'effectuer ce type d'étude. Cette étude comporte quatre phases dont la découverte, l'analyse, la spécification et la validation de besoins. Ces besoins que nous avons déterminés serviront les objectifs à atteindre pour le projet.

L'étude du système existant s'est déroulée également en quatre étapes. La première étape avait pour but l'étude de l'infrastructure matérielle du système existant. Pour cela nous avons focalisé nos attentions sur le réseau, le stockage et les serveurs. Cette étude nous a permis d'avoir une vision plus claire sur la mise en place de ces éléments dans notre système.

Par la suite, nous avons effectué une étude sur le clustering. Le but de cette étude était de faire un état des lieux sur la mise en place du clustering actuel et en même temps de présenter d'autres solutions possibles pour mettre en place une grappe de serveurs. Un bon design du clustering permet d'améliorer la disponibilité, la capacité de traitement et la performance de notre système.

Dans cette partie nous avons aussi effectué une étude de flux de données. Nous avons présenté leurs caractéristiques et leur fonction pour comprendre les écarts entre différents types de flux de données. Mener une étude sur la production nous permet de prendre connaissance de différents types de travaux effectués sur notre système.

Les études et les analyses que nous avons effectuées dans cette partie sont nécessaires pour l'étape de la modélisation de la stratégie. Comprendre les besoins des utilisateurs, du système existant et de la production nous permet de découvrir les goulots d'étranglement de notre système. Et c'est bien dans la phase de modélisation de la stratégie et de conception que nous devons prendre en compte ces problèmes afin de concevoir un système adapté aux besoins des utilisateurs.

Quatrième partie

Le développement du projet

Dans la partie précédente, nous avons effectué une recherche sur les besoins et sur le système existant. Dans cette partie, nous allons aborder les sujets concernant la réalisation du projet. Nous allons commencer par la modélisation de la stratégie, puis nous aborderons le sujet du découpage du projet et le choix de la méthode de développement. Nous entrerons également dans les détails de la réalisation puis nous en présenterons le bilan.

La modélisation de la stratégie nous permet de définir les objectifs et les actions pour atteindre ces objectifs. Pour ce faire, nous allons nous appuyer sur les études et les analyses que nous avons réalisées auparavant. Dans le même temps, nous devons prendre en compte nos moyens et les risques qui pourraient influencer le résultat final du projet.

L'étape du découpage du projet nous permet de le scinder en plusieurs sous-systèmes et composants afin de mettre en évidence les tâches à réaliser. Le choix de la méthode de développement est basé sur le profil du projet et le modèle adopté nous guidera pour poursuivre notre développement.

L'étape de réalisation consiste à concevoir, développer et intégrer les fonctionnalités. C'est une étape qui nous permet de percevoir le fonctionnement de notre futur système. La réalisation de la conception doit se baser sur les besoins attendus que nous avons déterminés dans la phase d'ingénierie des besoins. Elle doit prendre également en compte la situation actuelle du système existant. Pour le développement et l'intégration de fonctionnalités, nous nous appuyons sur les résultats que nous avons obtenus dans la phase de conception. En revanche, nous devons prendre des précautions car la phase de réalisation et de déploiement ne doit pas nuire à la production actuelle.

Chapitre 14 La stratégie

La stratégie est l'art de diriger et coordonner des actions pour atteindre un objectif. Comme nous l'avons décrit dans le chapitre 4 « La motivation du projet », le but de ce projet est de construire un système extensible, fiable, évolutif, rentable et visible. Alors pour modéliser une stratégie, nous devons tout d'abord adopter une démarche adéquate. Et c'est cette démarche qui nous guidera jusqu'à la détermination de la meilleure stratégie possible.

14.1 La démarche

Pour modéliser la stratégie, nous avons adopté une démarche en cinq étapes. Pour la première étape, nous allons effectuer un état des lieux sur les études que nous avons menées dans la partie précédente. Il nous aidera à revoir les besoins et les problèmes que nous avons découverts au cours de la phase d'ingénierie des besoins et d'analyse du système existant.

Par la suite, nous allons effectuer une analyse sur nos moyens humains, techniques et financiers. Nous devons prendre en compte les résultats de cette analyse pour élaborer notre stratégie. Les ignorer, ce serait prendre le risque de rencontrer des difficultés inattendues pendant la phase de conception et de développement.

Alors pour la troisième étape, nous allons effectuer une autre étude et cette étude consiste à aller repêcher les points qui pourraient influencer la réussite de notre projet. Identifier les risques nous permet de les éviter, afin de créer une condition favorable pour la réalisation du projet.

Au cours de la quatrième étape, nous allons déterminer les objectifs de notre système. Pour cela, nous allons construire un diagramme d'Ishikawa qui nous permettra d'identifier les objectifs de notre SI et les actions qui nous permettront de les atteindre.

Durant la dernière étape, nous allons élaborer un plan de pilotage. Ce plan contiendra des mesures que nous devons suivre pour faciliter la conception et le développement de notre projet. Il va déterminer quelques règles pour en fixer la ligne de conduite et il nous guidera peu à peu vers la phase de suivante.

14.2 Etat des lieux

Dans la partie précédente, nous avons déjà introduit des études sur les besoins et le système existant. Le but de cette section est de faire un court résumé pour présenter les analyses que nous avons effectuées auparavant afin de faciliter la détermination de nos objectifs.

Dans la phase d'analyse, nous avons relevé plusieurs problèmes. Les problèmes concernant la production sont des problèmes liés à la base de données et à la production PROOF.

Pour le problème PROOF, les utilisateurs voudraient réduire le temps d'exécution des travaux PROOF et ce problème est lié à la phase de validation des travaux. Il y a aussi le problème de la gestion des travaux PROOF, car elle est inexistante. En plus de ces deux problèmes, nous avons découvert que dans notre système, la répartition de la charge de travail pose également problème. En effet, la charge du travail des serveurs calculs dédiés à la production PROOF est beaucoup moins élevée que celle des serveurs Condor.

Il y a aussi le problème de la base de données. Son but est de faciliter la gestion de la manipulation des données (Datasets). La mise à jour de celle-ci ne semble que pas satisfaire les attentes des utilisateurs.

Parmi les problèmes que nous avons décrits dans les deux paragraphes précédant, nous avons également détecté le problème potentiel de la disponibilité de service. En effet, la mise en place des serveurs contrôles ne prévoit pas des serveurs de secours, alors si un des serveurs contrôle tombe en panne, c'est peut être l'ensemble du service qui peut être inutilisable.

Dans la partie précédente, nous avons également évoqué le problème du stockage : actuellement les serveurs ont atteint 80% de leur capacité de stockage. Pour résoudre ces problèmes, nous devons effectuer de multiples analyses pour définir une stratégie bien adaptée à nos besoins.

14.3 Les moyens

Une étude sur nos moyens consiste à aller rechercher et analyser la situation actuelle du groupe. Pour cela, nous allons fixer notre attention sur nos moyens techniques, financiers et humains. Ce sont des moyens élémentaires pour la réalisation d'un projet. Bien les connaitre nous permettra de gérer convenablement nos ressources et de créer de bonnes conditions pour élaborer notre stratégie.

Comme nous le savons, par rapport aux autres Tier 3, notre groupe dispose de ressources plus convenables que d'autres. La taille de notre système d'information est comparable à celle de Tier 2. Partager nos ressources sur le Grid, utiliser le système Condor pour traiter des travaux spécifiques et employer le système PROOF pour traiter un travail afin d'obtenir un résultat nous demande de construire un système solide, performant et évolutif. Mettre en place un système d'information de haut niveau demande beaucoup d'efforts humains, un budget pour investir dans les matériels et une bonne connaissance technique.

Le système d'information du groupe utilise de nombreuses technologies qui ne sont pas connues du grand public, comme par exemple le système Condor, le Grid, le système PROOF et le Xrootd. Ces technologies sont développées et maintenues par différentes équipes. En tant qu'utilisateur de ces technologies, il sera difficile de mettre correctement en place un système

sans avoir une parfaite connaissance de ces technologies. Le tableau suivant nous permet de comprendre le niveau de maîtrise de ces technologies.

Tableau 14 : Le degré de la maîtrise des technologies

Technologies	Utilisation	Configuration	Administration	Développement	Relation*
Base de données	Très bien maîtrisé	Très bien maîtrisé	Très bien maîtrisé	Oui	Développé par nous
Condor	Très bien maîtrisé	Très bien maîtrisé	Très bien maîtrisé	Non	Très bonne
Grid	Très bien maîtrisé	Très bien maîtrisé	Pas de tâche administration	Non	Bonne
PROOF	Très bien maîtrisé	Très bien maîtrisé	Pas de tâche administration	Non	Très bonne
Xrootd	Très bien maîtrisé	Très bien maîtrisé	Très bien maîtrisé	Non	Bonne

*relation avec l'équipe de développement

Dans le tableau ci-dessus, nous pouvons constater que nous avons une très bonne maîtrise de technique pour mettre en place notre base de données ce qui n'est pas le cas pour les autres technologies. Par exemple, pour le système Condor, nous en maîtrisons très bien l'utilisation, la configuration et l'administration, De plus nous avons un très bon contact avec l'équipe de développement. En revanche, ce système n'est pas développé par notre groupe et si nous voulons développer des nouveaux composants ou des fonctionnalités, nous dépenserons probablement beaucoup plus d'effort que pour un système développé et maintenu par nous. Dans ce tableau, nous constatons qu'il existe un autre cas figure. Par exemple, pour un système PROOF, il est nécessaire de développer un composant pour que nous puissions pré-valider les données à traiter. En revanche nous ne disposons pas de ressources et la technologie nécessaire pour les développer bien que le groupe dispose de ressources assez importantes.

Le groupe dispose d'un budget à hauteur de 400,000$ par an pour les dépenses dans le domaine informatique. Dans ces 400,000$, 100,000$ sont dédiés au personnel et les 300,000$ restants sont réservés aux achats de matériels.

Le budget « Personnel » représente donc un quart de notre budget informatique, les ¾ restant est principalement dédié aux achats des matériels. Dans la phase d'analyse des besoins, nous avons expliqué que les utilisateurs demandent d'augmenter la bande de passante. Dans la phase d'étude du système existant, nous avons vu que le volume de fichiers stockés sur nos serveurs ont atteint environ 80 pour cent de notre capacité de stockage. Si un jour, nos serveurs

fichiers sont pleins, cette situation pourrait provoquer un arrêt total de notre activité. Bien investir sur les matériels est essentiel pour notre système. Pour nous donner une idée plus précise, nous avons effectué une demande de devis auprès de nos fournisseurs. Le tableau suivant en est le résultat.

Tableau 15 : Le prix des matériels

Article	Prix unitaire	Unité	Total
Switch 48 ports 10Gbit/s	11500$	3	34500$
Câble Réseau 10Gbit/s	240$	234	56160$
Carte réseau 10Gbit/s	1030$	231	237930$
Total			328590$
Serveur 8 Core + 24 disques 750Go	7000$	24	168000$
Total			168000$

Dans le ci-dessus tableau, nous pouvons constater que si nous voulons généraliser le réseau 10 Gbit/s, nous devons débourser plus de 300,000$ et ceci dépasse totalement notre budget. Si nous mettons 12 serveurs fichiers supplémentaires pour chaque site, cela nous permettra de réduire le taux d'occupation des serveurs fichiers jusqu'à environ 50 pour cent. De plus il nous reste encore une part du budget pour améliorer une partie du réseau de notre parc.

14.4 Les risques

Un risque est un danger éventuel plus ou moins prévisible qui peut affecter l'issue du projet. Les risques peuvent avoir des conséquences sur le déroulement d'une phase du projet mais aussi des impacts financiers et sur la durée de celui-ci. Pour éviter ou réduire un risque, nous devons tout d'abord l'identifier, et par la suite, évaluer les impacts possibles afin de définir des actions pour éviter ou réduire un risque inacceptable.

Dans le chapitre 9, nous avons déjà évoqué les différents risques qui pourraient entraîner un échec de notre projet informatique, comme par exemple le manque de compréhension des besoins des utilisateurs. Il existe bien d'autres points qui pourraient représenter des facteurs de risques pour notre projet, comme par exemple la taille du projet et sa difficulté technique.

Il est évident qu'un grand projet est souvent plus difficile à manœuvrer qu'un petit projet, car il réclame plus de ressources et son domaine de couverture est souvent assez large. Alors pour mieux maîtriser un projet, il est recommandé de découper ce projet en plusieurs modules. Et chaque module peut être considéré comme un petit projet, ceci permettant de mieux maîtriser la réalisation du projet.

La difficulté technique correspond au niveau de complexité des technologies utilisées dans ce projet. Le manque de compétences techniques nécessaires pourrait pénaliser la production. Il y a aussi la stabilité de l'équipe de projet. Une équipe instable pose le problème du transfert de connaissances. Et ce problème peut influencer sur le délai du projet et avoir des conséquences sur la cohérence de la conception.

Identifier des risques nous permet de prendre des mesures pour les éviter et faire les modifications nécessaires afin de créer une condition favorable au projet. L'implication des utilisateurs est un point également crucial. En effet, plus les utilisateurs sont impliqués dans un projet plus la probabilité de réussir augmente, alors qu'un projet sans le soutien de la hiérarchie, ni la définition claire des besoins est un projet sans avenir. Il existe bien autres points importants pour la réalisation d'un projet comme par exemple un plan de développement correct et des attentes réalistes. Ces points essentiels pour notre projet doivent être surveillés régulièrement pendant toute la durée du projet. Le tableau suivant est une analyse effectuée pour comprendre le profil de notre projet. Elle est basée sur les points que nous avons décrits dans les paragraphes précédents et que nous avons classés sur une échelle de cinq.

Tableau 16 : Profil du projet

Les critères	**Situation**				
	1	2	3	4	5
Taille du projet		*			
Compréhension des besoins des utilisateurs				*	
Difficulté technique			*		
Implication des utilisateurs					*
Soutien de la hiérarchie				*	
Stabilité de l'équipe de projet				*	

Dans ce tableau, nous pouvons constater que la taille du projet est plutôt moyenne voire petite. Nous avons une équipe de projet stable et nous bénéficions également du soutien de la hiérarchie. La compréhension des besoins des utilisateurs est bonne et les utilisateurs sont bien impliqués. Et la difficulté technique du projet est également moyenne.

14.5 Les objectifs du système d'information

Le but d'élaborer une stratégie est de diriger et coordonner des actions pour atteindre un objectif. Pour notre projet, l'objectif général est clair, nous avons donc besoins d'améliorer la qualité de notre SI. La phase d'ingénierie des besoins nous permet de comprendre quels sont les

besoins et les problèmes correspondants. Nous allons utiliser un diagramme d'Ishikawa pour mettre en évidence des solutions possibles afin d'atteindre notre objectif.

Dans le diagramme d'Ishikawa (figure 23), nous avons défini l'objectif général pour notre système qui est d'améliorer la qualité et la performance de notre système. Pour atteindre ce but, nous nous basons sur six sous-objectifs et ces six sous-objectifs sont les fruits de notre recherche pendant la phase d'ingénierie des besoins et d'analyse du système existant. Pour chaque sous-objectif, nous avons suggéré quelques actions à mettre en œuvre pour notre projet.

Dans la phase d'ingénierie des besoins, nous avons beaucoup cité le problème concernant le système PROOF que avons besoin d'optimiser. Pour accomplir cette mission, nous avons déterminé deux actions. La première consiste à mettre en place un outil pour gérer les travaux PROOF et la deuxième est de pre-valider les Datasets afin de réduire le temps d'exécution.

L'objectif est d'optimiser la transaction de données est d'améliorer la performance. Pour cela nous proposons deux actions : réduire la transaction réseau interne et mettre en place des nouveaux équipements réseau.

Dans notre parc informatique, certains serveurs sont submergés de travail. En revanche, certains serveurs passent beaucoup de temps en veille, comme par exemple pour les serveurs PROOF. Nous devons alors mieux exploiter nos serveurs. Pour cela nous devons revoir notre architecture du système et essayer de réutiliser les CPU mal exploités.

En ce concerne la capacité de stockage, nos serveurs fichiers ont déjà atteint environ quatre-vingt pour cent. Avec plus de 3/4 d'espace disque déjà occupés et nos serveurs fichiers continuant à recevoir des données provenant du Grid, l'augmentation de la capacité du stockage nous semble obligatoire. Notre objectif est donc de baisser le taux d'occupation des disques à environ cinquante pour cent.

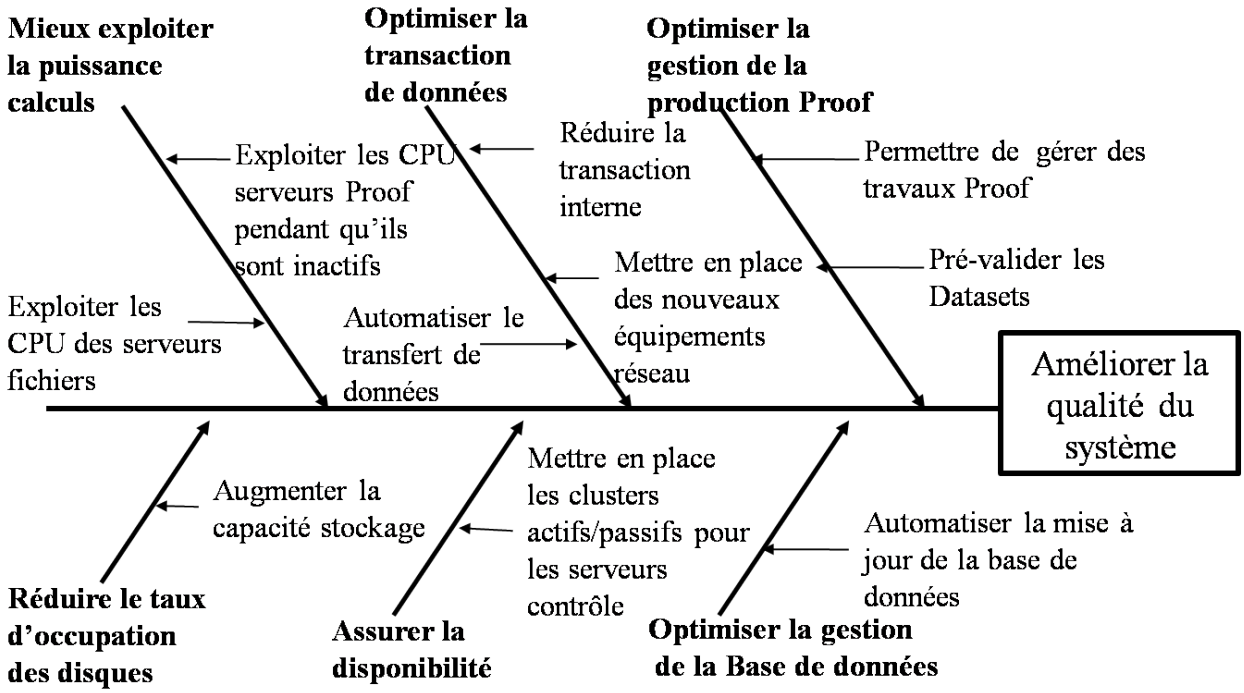

Figure 23 : Objectifs du système d'information

La disponibilité, a un impact direct sur l'activité de notre groupe. Pour atteindre un haut niveau de disponibilité, le projet pourrait être très coûteux. Pour avoir une solution solide et rapide, nous allons mettre en place les clusters actifs/passifs pour les serveurs contrôle afin d'améliorer la disponibilité de notre système.

Enfin, la base de données permet aux utilisateurs de retrouver rapidement des informations nécessaires et de connaître le statut de transfert des données journalières. Il est important d'avoir une base de données lisible et facile à gérer. Pour cela nous allons mener une action qui consiste à automatiser la mise à jour de la base de données.

14.6 Le plan de pilotage

Dans la section précédente, nous avons déterminé les objectifs que nous devons atteindre à la fin de notre projet et nous avons également déterminé les actions pour mener ces objectifs à termes. Dans cette section, nous allons élaborer un plan de pilotage qui fixera la ligne de conduite et les règles à respecter.

Notre projet sera construit sur un système existant. Pour cette raison, dans la phase de conception et de développement nous devons prendre en compte que le système existant est en production. Effectuer des modifications sur le système existant peut entraîner l'arrêt total de notre activité. Pour cela, nous préconisons de réaliser un prototype et nous allons d'abord tester les nouvelles fonctionnalités sur ce prototype avant de les appliquer dans notre système.

L’étude sur nos moyens nous montre que nous manquons d’expérience et de compétence pour mener à termes nos objectifs. Pour cela nous devons collaborer étroitement avec les différentes équipes de développement en utilisant leurs compétences pour nous aider à atteindre les objectifs.

Dans la phase de conception et de développement, nous devons découper le projet en plusieurs modules et chaque module correspond à une fonctionnalité précise. Découper notre projet en plusieurs modules nous permet de mieux maîtriser le risque et le délai de la réalisation. Pour vérifier les résultats de chaque module, nous allons effectuer une série de tests. Ces tests nous permettent d’obtenir de précieuses informations concernant le couplage des composants et le fonctionnement du système. Il permet de découvrir des problèmes cachés afin de les corriger. Ce sont des pratiques bien utiles pour la réalisation du projet et ces pratiques font partie des étapes importantes du cycle de vie du développement.

14.7 Conclusion

Nous avons commencé ce chapitre par une présentation de la démarche que nous allons poursuivre pour élaborer notre stratégie. Par la suite, nous avons consacré une section pour faire un état des lieux sur les problèmes que nous avons détectés dans la phase d’ingénierie des besoins et d’analyse du système existant. Une étude sur nos moyens nous permet de mettre en évidence nos forces et nos faiblesses en termes de moyens techniques, humains et financiers. Une bonne compréhension de nos moyens nous permet d’élaborer une stratégie bien adpatée.

L’analyse des risques est une analyse plus approfondie que nous utilisons pour obtenir le profil de notre projet. Bien connaitre le profil du projet facilitera la détermination de la stratégie. Nous avons alors effectué deux étapes pour achever la détermination de la stratégie. Pour la première étape, nous avons déterminé les objectifs de notre système et les actions qui permettent d’atteindre ces objectifs. Puis, nous avons établi un plan de pilotage qui nous fournit quelques conseils et règles à appliquer dans la phase de conception et de déploiement.

Chapitre 15 Le découpage du projet et la méthode adoptée

Dans ce chapitre nous allons traiter deux sujets : le découpage du projet et les modèles de cycle de vie. Ils interviennent après la détermination de la stratégie. Elaborer la stratégie nous permet de mettre en évidence les objectifs pour notre système futur, alors que l'étape du découpage du projet et l'adoption de la méthode agile nous permettent de définir des sous-ensembles et des démarches adaptées pour préparer à réaliser ce système.

15.1 Le découpage du projet

A travers la phase d'analyse du système existant, de l'ingénierie des besoins et de la détermination de la stratégie nous pouvons faire un état des lieus de notre système, rassembler les besoins des utilisateurs et fixer nos objectifs. L'étape de découpage sert, quant à elle, à décrire les objets à développer dans la phase de réalisation. Pour cela nous devons comprendre puis maîtriser nos objectifs pour mettre en évidence des liens entre les objectifs et les différents composants de notre système.

Le découpage nous permet également de réduire la complexité et la difficulté de notre projet. Le projet est donc plus facile à maîtriser, ce qui permet de mieux gérer nos ressources et de garantir les résultats attendus pour notre futur système.

Il existe différents approches pour découper un projet, comme par exemple le découpage PBS, *Product Breakdown Structure* (structure de décomposition d'un produit). Cette approche correspond au découpage structurel. Elle consiste à découper un projet en plusieurs modules et chaque module assurera une fonction spécifique.

Le découpage WBS, *Work Breakdown structure* (structure de décomposition du travail) correspond à une combinaison des critères de découpage temporel et structurel. Il définit les différents composants de travail nécessaires pour parvenir au résultat tel qu'il est décrit dans le PBS.

Enfin il existe un autre type de découpage, le découpage OBS, *Organisation Breakdown structure* (structure de décomposition de l'organisation). Ce découpage s'appuie sur des éléments de découpage WBS et il intègre l'attribution des ressources aux différents travaux.

Pour bien découper notre projet, nous divisons le procédé en deux étapes. La première étape consiste à reprendre les objectifs et les actions que nous avons déterminés dans la phase d'analyse de la stratégie et par la suite nous les classons dans les différents sous-systèmes. Le tableau suivant est une représentation de cette étude.

Tableau 17: la décomposition du projet en sous-systèmes

Objectif principal	Sous-objectifs	Action	Sous système associé
Améliorer la qualité de notre système	Optimiser le PROOF système	Permettre de gérer des travaux PROOF	La production
	Optimiser le PROOF système	Pré-valider les Datasets	La production
	Optimiser la transaction de données	Réduire la transaction interne	Architecture du système
	Optimiser la transaction de données	Mettre en place des nouveaux équipements réseau	Infrastructure du système
	Mieux exploiter notre puissance calculs	Mieux exploiter les CPU des serveurs fichiers	Architecture du système
	Mieux exploiter notre puissance calculs	Exploiter les CPU serveurs PROOF pendant qu'ils sont inactifs	Architecture du système
	Réduire le taux d'occupation des disques	Augmenter la capacité stockage	Infrastructure du système
	Assurer la disponibilité	Mettre en place les clusters actifs/passifs pour les serveurs contrôle	Architecture du système
	Optimiser la mise à jour de la base de données	Automatiser la mise à jour de la base de données	Base de données.

A travers le tableau ci-dessus nous pouvons constater que nous avons découpé notre projet en quatre parties donc la production, l'architecture, l'infrastructure et la base de données. Chaque sous-système comporte plusieurs objectifs et pour les mener à terme nous avons préconisé différentes actions comme inscrites dans le tableau. La figure suivante est une illustration du découpage de notre projet.

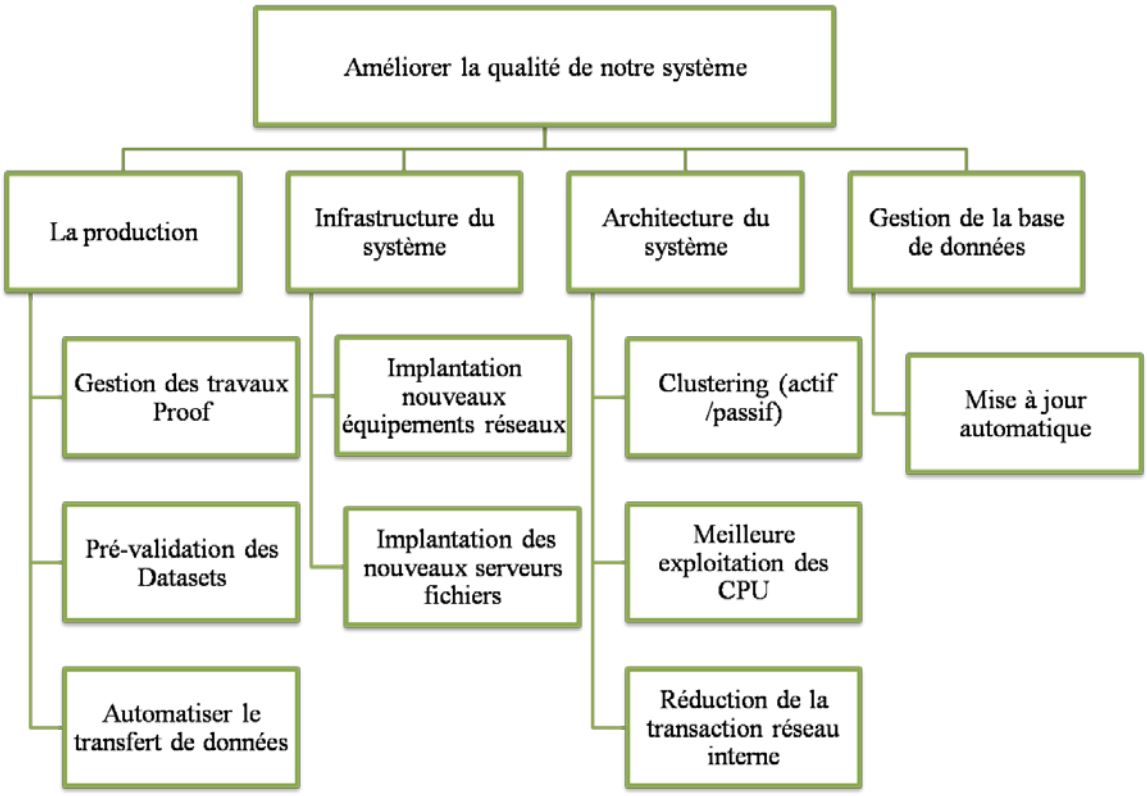

Figure 24: Le découpage du projet

En analysant la figure ci-dessus, nous pouvons constater que nous avons découpé notre projet en quatre sous-systèmes et à l'intérieur de chaque sous-système nous avons déterminé des tâches à accomplir. Il apparaît également que le découpage du projet est basé sur l'approche PBS et l'application de cette approche est fortement influencée par l'adoption du modèle de cycle de vie de notre projet.

15.2 L'adoption de la méthode « agile »

Dans le monde de la gestion de projet, il existe de nombreux modèles de développements. Nous pouvons citer des modèles dits classiques comme par exemple le modèle en V, le modèle de la cascade et le modèle spirale. Ces modèles sont basés sur le découpage temporel qui s'appuie sur les caractéristiques de l'entreprise et du projet.

Il existe aussi un autre type de modèle de cycle de vie qui utilise des méthodes agiles. Les modèles sont construits sur le principe de la méthode agile basée sur le découpage structurel dont chaque composant représente une fonctionnalité spécifique de notre projet.

Selon une étude du group Standish, la réussite d'un projet informatique dépend de divers critères. L'implication des utilisateurs, le soutien de la hiérarchie et la définition claire des besoins sont des points essentiels pour un projet informatique. Alors un plan de développement

correct, des attentes réalistes, le découpage du projet en petite étapes et les compétences dans l'équipe de projet sont également des clés pour la réussite d'un projet.

En regardant les critères de la réussite d'un projet informatique, l'utilisation de ce type de méthode est particulièrement adaptée à la situation actuelle de notre projet. En effet, ce type de modèle exige une collaboration étroite entre utilisateurs et développeurs. La plupart de nos développeurs et utilisateurs travaillent tous au CERN, et ceci facilite la communication entre ces deux parties. De plus, le groupe organise une réunion par semaine avec l'ensemble du personnel ce qui permet de renforcer le lien entre les deux parties.

L'utilisation de ce type de méthode nous permet de bien définir nos besoins, car il est basé sur une collaboration étroite entre les utilisateurs et les développeurs. La méthode agile est basée sur un cycle de vie itérative et incrémentale. Elle n'exige pas de production de dossier d'analyse et de spécification en amont. En revanche, utiliser ce type de méthode nous permet d'éclairer progressivement les différents aspects pour notre futur système afin de mettre en œuvre un plan de développement correct.

Une collaboration étroite avec les utilisateurs permet aux développeurs d'obtenir les feedbacks réguliers des utilisateurs. Ces informations peuvent servir aux développeurs pour modifier leur plan de développement afin de satisfaire les attentes des utilisateurs. Concernant le découpage du projet en petites étapes, cette démarche nous permet alors de concentrer nos efforts sur des points précis et de les amener à terme progressivement.

Dans cette section, nous avons présenté les avantages d'adopter la méthode agile pour la réalisation de notre projet. Comme nous le savons, l'utilisation de la méthode agile ne nous oblige pas à produire des dossiers pour la spécification, en revanche il nous fournit de nombreuses « bonnes pratiques » pour nous aider à réussir dans notre projet.

15.3 Les bonnes pratiques

Dans la méthode agile, il existe plusieurs modèles ; nous pouvons citer par exemple le modèle XP (eXtreme Programming), le RAD (Rapide Application Development) et le DSDM (Dynamic systems development method), chaque modèle a sa spécificité, en revanche ils s'appuient tous sur les valeurs communes de la méthode agile. Dans notre réalisation, nous n'avons pas fixé un modèle particulier pour notre développement mais nous nous sommes plutôt appuyés sur des bonnes pratiques telles que la conception rapide, la mise en place de prototype, etc. Alors, dans les paragraphes qui suivent nous allons présenter les bonnes pratiques que nous utilisons pour le développement du projet.

La conception rapide

La conception rapide est bien une pratique que nous utilisons dans cette phase de réalisation. La méthode que nous utilisons pour la réalisation ne comporte pas d'étape de conception complexe comme utilisée dans la méthode traditionnelle. Lorsque nous allons commencer à réaliser une nouvelle fonctionnalité, nous allons d'abord commencer à déterminer une liste de tâches qui nous permet de construire la fonctionnalité demandée par les utilisateurs. L'étape de conception doit être simple. Pour cela, notre conception doit uniquement se focaliser sur les demandes actuelles sans anticiper sur les besoins de demain.

Après avoir déterminé une fonctionnalité et les tâches associées, nous commençons à élaborer un plan de test. Ce plan est établi même avant le commencement du développement, car développer un plan de test nous permet de vérifier que notre futur système soit conforme aux attentes des utilisateurs.

Dans notre projet, il nous arrive d'utiliser les diagrammes d'UML pour concevoir un composant. Cette pratique est notamment utilisée dans la réalisation du sous-système « Base de données ». Nous utilisons deux diagrammes essentiels pour la conception de la base de données. Un diagramme des Cas d'utilisation nous permet d'identifier les relations entre les acteurs et le système. Elaborer un diagramme de classes nous permet d'identifier les tables, les attributs et les associations entre les tables.

Le développement de prototype

Durant la réalisation du projet, nous avons déployé beaucoup d'efforts pour mettre en place des prototypes. Une dizaine de machines mises à la disposition pour construire un clustering similaire de notre système en production. Les différents composants sont d'abord développés et testés sur le prototype avant d'être intégrés dans notre système actuel.

Il s'avère que le développement de ces prototypes est très utile pour nous car la réalisation d'un prototype présente plusieurs avantages ; par exemple, il permet de mesurer le temps nécessaire pour la réalisation d'une tâche ou d'un composant, ce qui donne une indication sur l'effort nécessaire pour cette réalisation.

La réalisation d'un prototype nous permet également de tester des nouvelles fonctionnalités et de percevoir des avis des utilisateurs. Elle nous permet aussi d'effectuer des essais ou des simulations pour obtenir des résultats concrets sans nuire à la production. A travers ces résultats, nous pouvons effectuer une comparaison avec le système existant ou la version précédente du prototype. Cette démarche nous permet d'optimiser notre réalisation et de corriger des défauts de conception.

Les tests

Comme le développement de prototype, effectuer des tests joue un rôle essentiel pour notre projet. En effet, dans notre projet, la mise en place de tests n'est pas seulement d'intervenir après la finalisation de la réalisation de composants. Les tests sont effectués avant, pendant et après la réalisation de composant.

Avant de passer au développement, nous devons effectuer des tests sur le système existant pour mieux comprendre le fonctionnement du système. Et à travers ces tests, nous pouvons recueillir des résultats concrets et ces résultats peuvent être servis comme des baromètres pour mesurer la qualité de notre réalisation de composants.

Pendant la phase du développement, les tests que nous effectuons sont appelés « tests unitaires ». Ces tests ont pour but d'assurer le bon fonctionnement de la partie « réalisation ». Ces tests nous permettent aussi de vérifier si nos réalisations sont conformes aux scénarii que nous avons définis dans la phase de conception. Si notre réalisation n'est pas conforme au scénario que nous avons défini, nous pouvons ajuster ou modifier notre plan de réalisation.

A la fin de la réalisation d'un composant, nous effectuons également des tests. Ces tests sont réalisés avec les utilisateurs, ils permettent de vérifier que les fonctionnalités prévus sont réalisées et bien conformes aux attentes des utilisateurs.

Comme nous avons pu le voir, les tests sont indispensables dans notre projet. Effectuer des tests avant la réalisation nous permet de mieux connaître notre système existant afin d'anticiper des travaux à réaliser. Intégrer des tests dans la phase de réalisation, nous permet de gagner en réactivité afin de garantir en qualité de réalisation. Et les tests effectués à la fin de réalisation, nous permettent d'assurer la couverture fonctionnelle.

Intégration continue

L'intégration est une activité qui consiste à assembler les composants qui forment un système. La pratique de l'intégration continue sert à répondre aux besoins d'intégration des nouvelles fonctionnalités que nous avons développées. L'adoption de la méthode agile nous conduit à fournir une livraison fréquente des composants. Cette pratique nous permet d'être en permanence en mesure de livrer une version opérationnelle du système.

Appliquer cette pratique dans notre réalisation présente plusieurs avantages. Elle permet de détecter plus rapidement les incidents d'intégration afin de les corriger le plus vite possible. Une livraison fréquente des composants accompagnée d'une intégration continue exige que nous effectuions différents types de tests durant chaque livraison, Par conséquent, plus notre projet avance plus notre système sera testé.

15.4Conclusion

Dans ce chapitre nous avons abordé deux sujets : le découpage du projet et l'adoption du cycle de vie. Le découpage du projet que nous avons effectué est basé sur une décomposition structurelle. Ce découpage nous permet de décomposer notre projet en sous-systèmes puis en composants. Ce type de démarche nous amène à réaliser notre projet en petites étapes. Le projet est donc plus facile à maîtriser et le risque d'échecs diminue.

Dans la deuxième partie de ce chapitre, nous avons présenté l'adoption de la méthode agile et les cycles de vie adoptés pour la réalisation du projet. Le choix de ce type de méthode est basé sur l'environnement du projet et les avantages qu'ils pourraient fournir pour le développement du projet. Par la suite, nous avons présenté les bonnes pratiques que nous allons introduire dans la phase de réalisation. Ces pratiques sont très utiles pour nous et pour la réussite du projet.

Chapitre 16 La réalisation

Dans le chapitre précédent, nous avons effectué un découpage du projet. Ce découpage nous amène à décomposer notre projet en quatre sous-systèmes, qui sont :

- L'architecture du système ;
- La production ;
- L'infrastructure du système ;
- La gestion de la base de données.

Nous avons décomposé chaque sous-système en un ou plusieurs module. Chacun de ces modules correspond à une fonctionnalité ou un élément particulier à réaliser. Dans le chapitre précédent, nous avons aussi évoqué l'adoption de la méthode agile dans notre projet. Cette adoption nous amène à inclure les étapes de conception, de développement et d'intégration dans une même phase. La force de ce type de méthode est la collaboration étroite entre les utilisateurs et les développeurs. Cette pratique nous permet d'être plus réactifs face à des problèmes que nous allons rencontrer.

16.1 Le sous-système « Architecture du système »

Pour réaliser le sous-système « Architecture du système », nous avons focalisé nos efforts sur trois modules essentiels, qui sont :

- Clustering Actif/Passif ;
- Meilleure exploitation des CPU ;
- Réduction de la transaction réseau.

Clustering Actif/Passif

Assurer une haute disponibilité de nos services est une clé essentielle pour maintenir la continuée de la production sur nos sites. Bien que les utilisateurs n'aient pas exprimé ce type de besoin pendant la phase d'ingénierie des besoins, nous l'avons quand même intégré dans notre liste de composants à réaliser.

Dans notre système, il existe une liste très longue de serveurs middleware. Chacun de ces serveurs assure un service bien particulier, comme par exemple le serveur SRM, un composant du Grid, qui permet d'assurer la gestion des ressources stockages. En cas de panne du serveur SRM, notre site ne peut plus recevoir de données et de travaux provenant du Grid.

Il faut noter que le serveur SRM n'est pas le seul pouvant générer ce type de problème, les serveurs tels que le Condor master, PROOF master, Xrootd master, LFC, DQ2 et le Gatekeeper

peuvent tous paralyser notre production, d'où l'importance de chercher une solution pour assurer une haute disponibilité de nos services.

La mise en place d'un clustering Actif/Actif ou Actif/Passif nous permet d'assurer une continuité du service en cas de panne du serveur sur lequel il est situé. La forme de clustering Actif/Actif est déjà utilisée pour la mise en place des serveurs Gridftp. En revanche ce type de clustering ne peut pas être généralisé pour tous les serveurs de contrôles, car certains services comme par exemple, le SRM, le DQ2, le LFC et le Condor-g exigent un seul point d'entrée pour la production. Pour cette raison, nous décidons d'utiliser la forme Actif/Passif pour la mise en place du clustering.

Pour valider le résultat de notre réalisation, nous avons déterminé trois critères à vérifier. Tout d'abord, notre futur système doit être capable de détecter la panne du serveur actif dans un délai très court. Ensuite, lorsque cette panne est détectée, notre système doit pouvoir lancer la procédure de bascule immédiatement. Enfin, la synchronisation des données doit être permanente : lorsque le service bascule vers un autre serveur, les données sont parfaitement à jour et disponibles.

Pour la réalisation, nous devons en tout premier lieu chercher des solutions existantes, plutôt que de développer nos propres solutions. L'idée de cette démarche est de réduire le temps et l'argent consacrés au développement. La solution que nous avons adoptée est d'utiliser deux programmes entièrement libres pour le système Linux.

- L'un s'appelle « Heartbeat », qui signifie le « battement de cœur » : il permet aux serveurs de s'informer mutuellement sur leur fonctionnement. En cas de panne du serveur Actif, le serveur Passif va lancer la procédure de bascule, acquérir l'IP adresse du service et monter le système fichier.

- Pour le deuxième, nous avons intégré le programme DRBD, permettant de synchroniser le contenu de nos dossiers en temps réel et garantissant une copie conforme du serveur Actif. La mise en place de ce programme est bien plus délicate que celui du programme « Heartbeat », car il nécessite la modification du noyau du système d'opération. Pour cette raison, nous avons d'abord essayé d'introduire la solution sur un prototype pour procurer le maximum d'informations concernant l'installation, la configuration et les comportements de ces deux programmes.

La figure 25 est une illustration du fonctionnement de la mise en place de clustering Actif/Passif dans notre système.

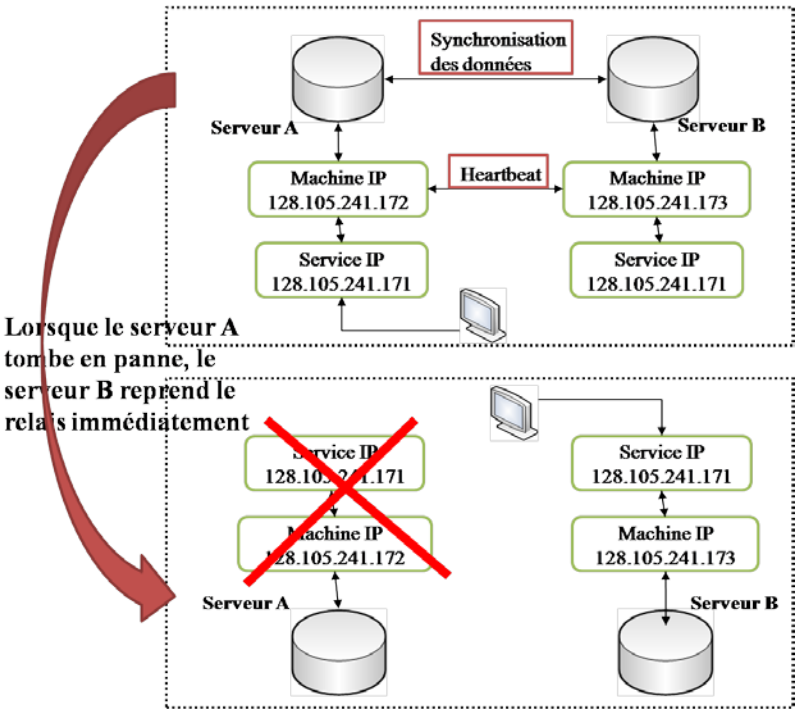

Figure 25 : Illustration du fonctionnement du Clustering Actif/Passif

Meilleure exploitation de nos puissances de calculs

Le Wisconsin groupe a un nombre important de serveurs et chaque serveur a sa fonctionnalité spécifique. Dans la phase d'ingénierie des besoins, la demande d'une meilleure utilisation de nos capacités de calculs se faisait entendre. Dans la phase d'analyse du système existant, nous avons également constaté que les CPU des serveurs fichiers sont très mal exploités. Nous avons aussi évoqué le problème concernant l'utilisation des serveurs calculs dédiés à la production PROOF. Tous ces problèmes sont liés au problème de l'architecture.

Distinguer les serveurs calculs selon leur principale fonction facilite la gestion de l'administration. Cela permet également à chaque type de serveur calculs d'occuper une fonctionnalité particulière, donc il n'y aura pas de conflit entre les différentes productions. En revanche, cette mise en place demande un financement important pour les équipements. De plus, ce type de mise en place peut entrainer un gaspillage de nos ressources si nous n'avons pas assez de travail à fournir.

Pour mieux gérer l'ensemble de nos puissances de calculs, nous avons décidé de revoir l'architecture du système. Pour cela, nous avons d'abord effectué une analyse du système existant. A cet effet, nous avons adopté la solution qui consiste à fusionner des serveurs calculs avec les serveurs fichiers. Cette fusion nous donne la possibilité d'augmenter le nombre de CPU dédié à l'ensemble de la production. Elle nous permet aussi de réutiliser les CPU de serveurs fichiers. En revanche, ce type de solution n'est pas une solution sans contrainte car la gestion de

l'administration sera plus difficile et la gestion de la priorité pour chaque type de production doit être établie.

Pour établir un niveau de priorité pour chaque type de production, nous nous sommes basés sur les caractéristiques de chacune. Par exemple, nous avons classé la production PROOF en haute priorité car ce sont des travaux soumis par les utilisateurs locaux qui veulent obtenir un résultat dans un très court délai. En revanche, pour un travail Condor soumis par un utilisateur du groupe, nous l'avons classé en niveau de priorité médium, ce type de travail étant soumis par les utilisateurs du groupe sans une contrainte du temps très importante. Enfin, nous avons classé tous les travaux provenant du Grid en niveau bas, car ce sont des travaux de long durée soumis par tous les utilisateurs provenant de l'extérieur.

Pour mettre en œuvre cette fusion nous avons commencé par la fusion de tous les serveurs calculs. Cette fusion nous permet de surveiller le comportement des serveurs calculs lorsqu'ils reçoivent des travaux Condor et PROOF simultanément. Nous avons particulièrement veillé à ce que la priorité pour chaque type de production soit respectée.

La deuxième étape consiste à fusionner les serveurs fichiers avec les serveurs calculs. Cette démarche nous permet d'exploiter des CPUs dédiés auparavant uniquement aux serveurs fichiers. La figure 26 est une image de monitoring du taux d'utilisation de CPU pour un serveur calcul. Dans cette figure, nous pouvons constater que nous pouvons exécuter un travail PROOF (en bleu) et Condor (en Jaune) en même temps sur un seul serveur. Et lorsque le système PROOF a besoin de plus de CPU, le système Condor lui donne une partie de CPU car la production PROOF est en haute priorité.

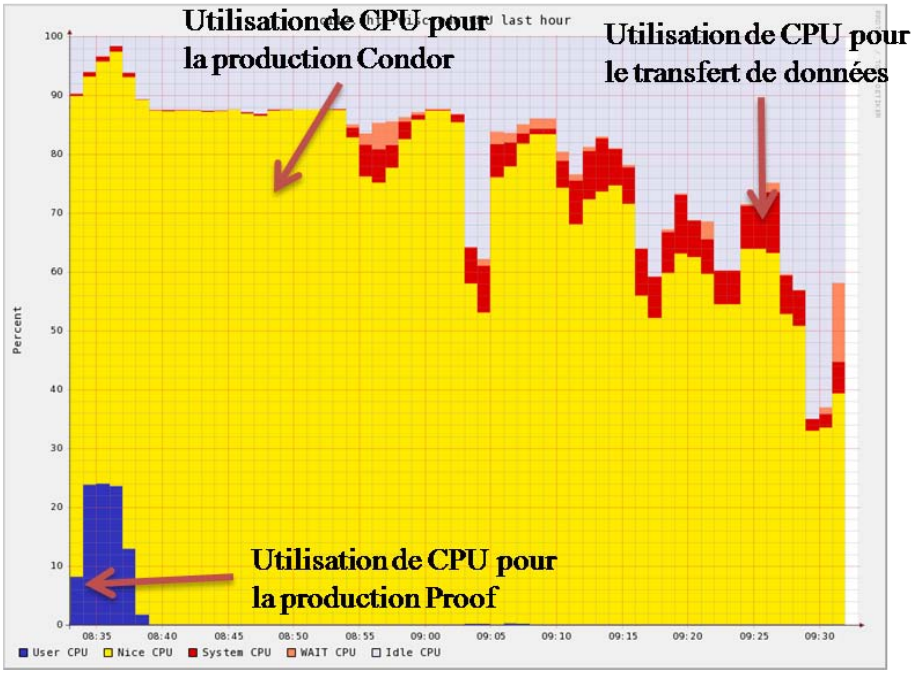

Figure 26 : La co-habitation entre la Production PROOF et Condor

Réduction de la transaction réseau

Le réseau est un élément essentiel pour garantir le bon fonctionnement de notre système. Chaque jour des centaines de Giga de données transitent dans notre système. Ceci est généré par la demande de transfert de données soumis par les utilisateurs du groupe, la production Grid, PROOF et Condor. Comme nous le savons la transaction de données entre les serveurs fichiers et les serveurs calculs sollicite énormément nos ressources. Réduire la transaction réseau nous permet d'assurer un bon fonctionnement de notre système et de limiter les investissements en matière d'équipement.

Les analyses que nous avons effectuées auparavant nous montrent que pour le moment la mise en place du réseau n'est pas mise en cause. En revanche, dans un futur proche, le démarrage du LHC pourrait augmenter fortement la charge réseau au sein de notre système. Pour anticiper cette contrainte, nous avons besoin, d'une part de moderniser nos équipements réseau, et d'autre part de limiter une partie des transactions données à l'intérieur de notre système. Dans cette section, nous allons traiter ce sujet.

Les tests nous montrent que la bande passante de notre réseau est en moyenne de 116Mo/s. Ce résultat est assez proche de la bande passante théorique de 125Mo/s pour un réseau de 1 Gbit/s. En revanche, les tests de débit des disques nous montrent que la mise en place de RAID 5 nous permet d'obtenir une bande passante supérieure à celle du réseau. A partir de ce constat, nous avons commencé à réfléchir à la possibilité d'utiliser cet avantage.

Comme nous le savons, les données sont toutes initialement stockées sur les serveurs fichiers. Lorsque nous avons besoin de traiter un Dataset, ces données du Dataset vont être d'abord transférées sur les serveurs calculs avant d'être exécutées. Ce type de transaction représente plus de 75 % des transactions réseau réalisé chaque jour dans notre système.

Le remaniement d'architecture du système que nous avons effectué auparavant nous offre la possibilité de réduire sensiblement la transaction dans notre système. Permettre aux serveurs fichiers de traiter également des travaux PROOF, Condor et Grid nous ouvre des perspectives de limiter une partie des transactions entre des serveurs fichiers et des serveurs calculs. Pour que cette idée soit réalisable, nous avons commencé par la modification du script de soumission. En effet dans un script nous pouvons spécifier les machines calculs que nous préférons utiliser. Nous pouvons aussi spécifier que certains travaux s'exécutent sur des machines que nous avons sélectionnées. En combinant ces solutions, nous avons réussi à réaliser une diminution des transactions d'environ 10 pour cent, ce qui n'est pas négligeable. La figure 27 illustre le changement que nous avons apporté entre l'ancienne version et la version actuelle de l'architecture.

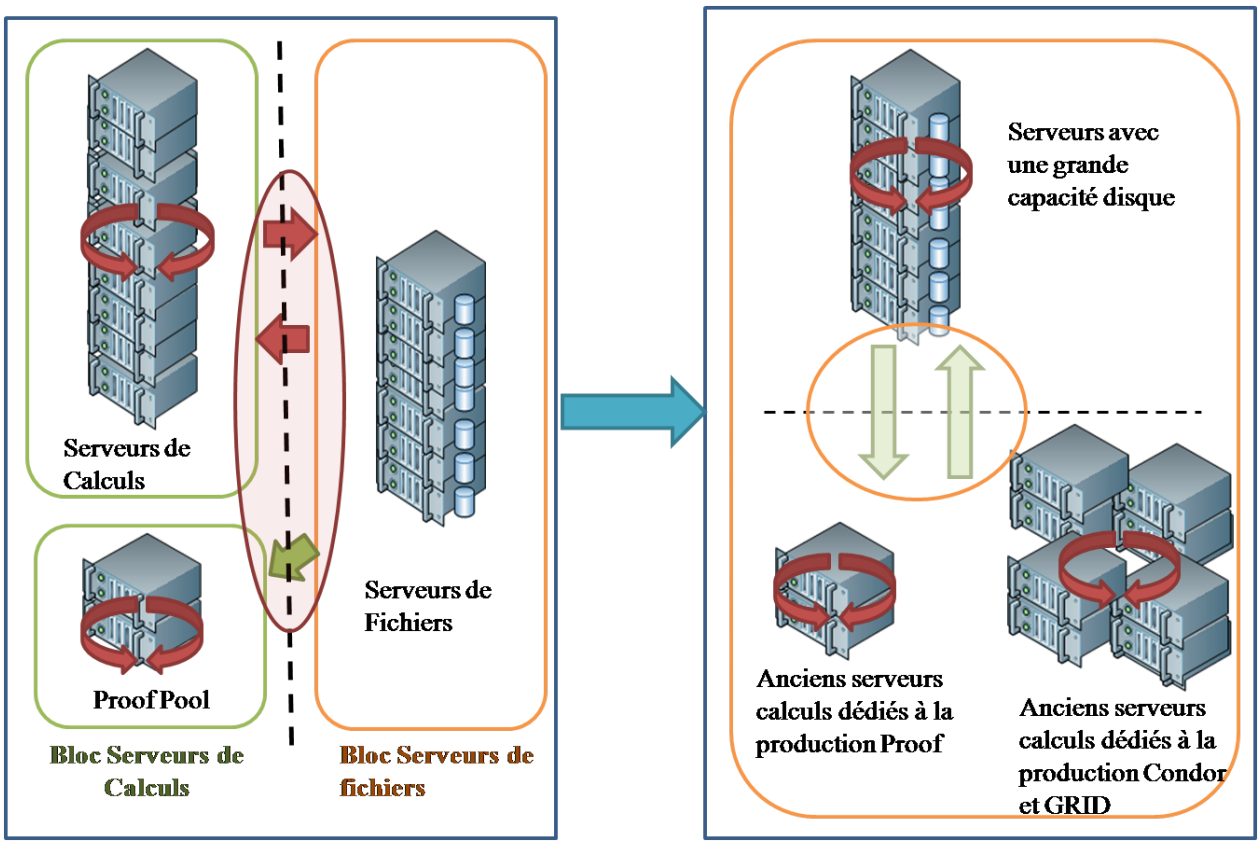

Figure 27 : illustration de la nouvelle architecture et de la réduction de la transaction

16.2 Le sous-système « La production »

La production est une activité principale de notre système. Assurer le bon fonctionnement de la production est le but primordial de ce projet. Pendant la phase d'ingénierie des besoins, nous avons recensé plusieurs demandes de la part de nos utilisateurs pour une amélioration de la production PROOF. Pour cela, nous avons déterminé deux tâches à accomplir qui sont :

- La gestion des travaux PROOF ;
- La pré-validation des Datasets.

La gestion des travaux PROOF

Jusqu'à présent, le système PROOF ne comporte pas d'outil d'administration pour gérer sa production. Le seul moyen pour arrêter un travail lancé par un utilisateur est de quitter lui-même « en force » le programme PROOF. Même un administrateur du système ne dispose pas de moyens d'arrêter et de redémarrer une session PROOF. Cette situation ne peut évidemment pas être acceptée par tout le monde. Pour cette raison nous avons besoin d'un outil qui est capable de gérer la production PROOF.

Pour développer un outil qui permet de gérer la production PROOF, il existe trois types de possibilité. Le premier est de développer un outil de A à Z pour gérer la production PROOF. Le deuxième est d'adopter un outil de gestion des tâches qui permet de travailler ensemble avec le

système PROOF afin d'atteindre notre but. Le troisième est de développer un module ou outil à l'intérieur du système PROOF pour qu'on puisse en gérer la production.

Après longue réflexion, nous avons décidé d'intégrer un gestionnaire de tâches pour pouvoir gérer la production PROOF. Bien entendu, cette décision est basée sur les aspects techniques et financiers. Intégrer un outil de gestion, c'est nous épargner le coût du développement et il est plus facile à gérer que si nous voulons développer nous même un outil ou un module pour le système PROOF.

Pour faciliter le choix d'un gestionnaire, nous avons d'abord consulté les utilisateurs puis nous avons déterminé plusieurs critères pour que le choix d'un gestionnaire soit plus facile. Voici les critères que nous avons déterminés :

L'outil doit permettre de

- lancer une tâche PROOF (une session PROOF) ;
- consulter le statut de la production ;
- gérer le niveau de priorité de chaque tâche ;
- gérer la disponibilité des ressources ;
- suspendre une tâche en cours d'exécution ;
- reprendre une tâche ;
- tuer une tâche en cours d'exécution ;
- arrêter une tâche ;
- redémarrer une session PROOF ;
- protéger les tâches contre les crashs des nœuds.

Hormis ces critères, nous devons aussi prendre en compte que le futur gestionnaire de tâches doit être compatible avec notre système. Il doit être facile à intégrer dans notre système, facile à utiliser et avoir un support technique en cas de problème.

D'après cette liste de conditions, nous avons commencé la recherche d'un gestionnaire. Nous avons pu examiner plusieurs gestionnaires de tâches, comme par exemple le « Job scheduler » et « Maui Cluster Scheduler » et finalement nous sommes revenus sur le gestionnaire de tâches Condor parce que le système Condor est un outil très puissant qui permet de gérer la production. De plus il est déjà utilisé dans notre système avec lequel nous avons déjà plusieurs années d'expérience.

Comme nous l'avons présenté dans le chapitre 13, la production PROOF est gérée par le «Packetizer » du côté PROOF master. Et maintenant, nous voulons confier ce travail au système Condor. Pour cela, notre futur outil doit pouvoir gérer trois fonctionnalités, qui sont les suivantes :

- la soumission : elle permet au Condor master d'envoyer un fichier d'un Dataset à un serveur calcul Condor ;
- l'exécution : elle permet aux serveurs calculs Condor de traiter les fichiers en utilisant le programme ROOT ;
- le réassemblage : il permet de fusionner les résultats de chaque fichier traité et de nous fournir un résultat final.

La réalisation de cet outil est basée sur ces trois fonctionnalités. Les tests nous montrent que le système Condor est bien capable de gérer la production PROOF et il permet de répondre à toutes nos exigences.

La pré-validation des Datasets

Lorsque le Système PROOF reçoit un travail lancé par un utilisateur, le Système PROOF va gérer le travail de la façon suivante :

- La première étape consiste à vérifier et à valider des données à traiter dont leurs tailles, leur intégralités, leur type et leur chemin ;
- La deuxième est de traiter ces données en utilisant la bibliothèque du ROOT ;
- La troisième consiste à fusionner les résultats pour fournir un résultat final.

Dans la phase d'analyse, nous avons découvert que l'étape de vérification et de validation des données prend beaucoup de temps pour être achevée. Alors pour améliorer la performance de la production PROOF, il nous semble que bien gérer l'étape de la validation et de la vérification des données sont des points déterminants. Pré-valider les Datasets nous permet de vérifier et valider les Datasets par avance, lorsqu'un utilisateur lance un PROOF job, l'étape de vérification et de validation seront beaucoup plus rapides car les données sont déjà pré-validées. En revanche pour la réalisation, il reste quelques difficultés.

Le système PROOF est un module du Framework (bibliothèque) ROOT. Le Framework ROOT contient environ 1200 classes et elles sont réparties dans les 19 modules du Framework ROOT. Pour ajouter une nouvelle fonctionnalité dont celle de pré-validation, il faut : une très bonne connaissance et maîtrise de l'environnement du développement de la librairie ROOT, une équipe de plusieurs développeurs d'expérience, du temps et de l'argent.

En regardant nos moyens humains, techniques et financiers, il nous a semblé qu'il serait difficile de tout élaborer par nous même. Pour cela, nous avons pris contact avec l'équipe de développement PROOF pour que nous puissions travailler ensemble pour le développement du composant « La pré-validation des Datasets », sachant que les développeurs de l'équipe PROOF travaillent dans le même bâtiment que nous et que la relation est plutôt bonne entre nos deux équipes.

La collaboration a commencé par la détermination des tâches à accomplir entre nos deux équipes. L'équipe PROOF prend en charge tout ce qui concerne le codage et les testes unitaires. Notre groupe quant à lui se charge de tout ce qui concerne l'ingénierie des besoins et les tests d'intégration.

L'outil que nous avons développé s'appelle PQ2 et il signifie « **P**ROOF **Q**uick **Q**uery ». C'est un outil de gestion qui permet d'accéder aux Datasets pour lister, souscrire, retirer et vérifier des informations. Les informations que nous pouvons obtenir concernent la localisation, la taille et le type des fichiers et des Datasets, etc. L'outil PQ2 contient sept commandes, chaque commande correspondant à une fonctionnalité spécifique. Le tableau ci-dessus est une présentation de l'outil PQ2 et ses commandes.

Tableau 18 : Les commandes d'outil PQ2

Commande	**Utilisation**	**Fonctionnalité**
pq2-ls	pq2-ls	Lister les informations de tous les Dataset
pq2-ls-files	pq2-ls-files *dataset*	Lister les informations d'un Dataset spécifique
pq2-ls-files-server	pq2-ls-files-server *dataset serveur*	Lister les informations des fichiers qui se trouvent dans un même Dataset et même serveur
pq2-info-server	pq2-info-server *serveur*	Lister les informations de tous les fichiers qui se trouvent dans un même serveur
pq2-put	pq2-put *datasetfile*	Enregistrer un Datasets
pq2-verify	pq2-verify *dataset*	Vérifier le contenu d'un Dataset
pq2-rm	pq2-rm *dataset*	Retirer un Dataset de notre système

Ces commandes couvrent tous les besoins nécessaires pour pré-valider les Données. Selon différents besoins, nous pouvons exécuter l'une de ces commandes. Les informations concernant la pré-validation sont stockées sur le PROOF master en mémoire cache. Ces informations peuvent être utilisées lorsqu'une requête est soumise au système PROOF. Les tests de performances nous montrent que les résultats de cette réalisation sont très satisfaisants. Pré-valider un Dataset nous permet de réduire le temps consacré à l'initiation et nous avons pu réduire de moitié le temps pour un travail PROOF, ce qui est loin d'être négligeable. Pour un travail PROOF exécuté sur la nouvelle architecture, nous avons obtenu des scores de vingt cinq secondes en moyenne.

16.4 Le sous-système « Infrastructure du système »

L'infrastructure du système est un élément de base pour construire un système d'information. Dans notre système, il se présente sous la forme d'équipements réseau et de serveurs. Une bonne gestion de nos équipements nous permet de répondre à la demande croissante en matière de la production et d'assurer la stabilité de notre système. Dans cette section, nous allons donc présenter l'implantation des nouveaux équipements réseaux et des serveurs.

Implantation de nouveaux équipements réseaux

La possibilité d'implanter de nouveaux équipements réseaux a été évoquée par les utilisateurs dans la phase d'ingénierie des besoins. Pour assurer le bon fonctionnement de notre système, pour bien gérer une croissance constante des données à traiter, nous devons consolider l'infrastructure de notre système pour mieux en préparer l'avenir. Le réseau fait donc partie d'un domaine prioritaire à gérer. Actuellement, sur nos deux sites de production, tous les serveurs sont équipés une carte réseau de 1 Gbit/s. Et sur le site Wisconsin, nous avons mis en place des Switchs de 10 Gbit/s mais pour le côté CERN les Switchs sont de 1 Gbit/s.

Passer notre réseau en 10 Gbit/s nous permet d'augmenter la bande passante de notre système. Cependant cela nécessiterait un budget très conséquent. En effet, le budget dépasserait notre budget annuel pour l'achat de tous les matériels. Pour cette raison, nous avons proposé de moderniser une partie de notre réseau informatique pour mieux répondre aux besoins actuels tout en respectant notre budget annuel.

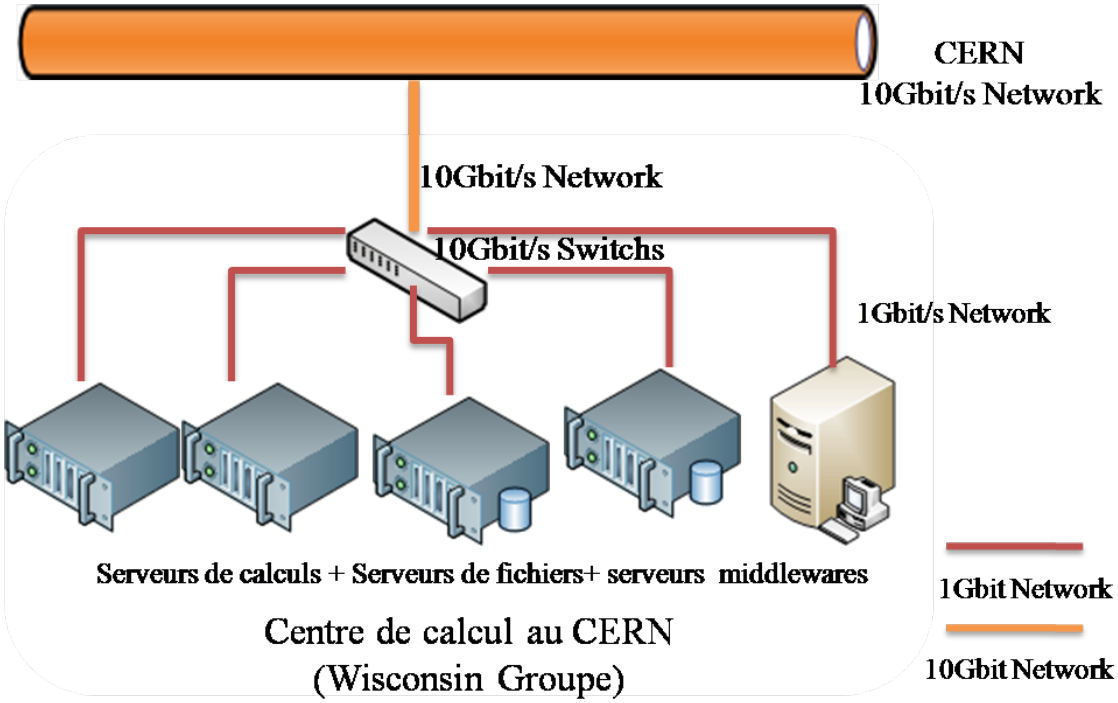

Figure 28 : La mise en place des nouveaux switch pour le site du côté CERN

La figure 28 ci-dessus est une représentation de la modification que nous avons apportée à notre système. Elle permet de généraliser des switchs de 10 Gbit/s pour l'ensemble du système. Malgré le contrainte budgétaire, la mise en place des switchs nous permet de bénéficier de deux avantages :

- Augmentation la capacité de la bande passante à partir de switch vers l'extérieur ;
- Mettre en place une base solide pour la modernisation de notre infrastructure.

Implantation des nouveaux serveurs fichiers

Implanter des nouveaux serveurs fichiers dans notre système nous permet de résoudre le problème du stockage. Actuellement, le volume de fichiers stockés sur nos serveurs a atteint environ 80 pour cent de notre capacité de stockage. Si nos serveurs fichiers étaient pleins, cela pourrait provoquer un arrêt total de notre activité. Cette situation est inacceptable. De plus le redémarrage de LHC nous donnera plus de données à gérer, à traiter et à stocker. Dans la phase d'analyse de la stratégie, nous avons par conséquent préconisé d'implanter des nouveaux serveurs fichiers dans notre système.

Avant d'acheter des nouveaux serveurs, nous avons d'abord effectué une étude pour connaître combien de serveurs devront être achetés pour combler nos besoins. Pour cela, nous avons commencé par une étude d'estimation pour connaître quelle volume de données notre système devrait acquérir chaque année. Nous avons ensuite fixé une limite, cette limite servant à surveiller que les fichiers stockés sur notre système ne dépassent pas 80 pour cent de sa capacité. Cette démarche nous permet de maintenir le bon fonctionnement de notre système.

Finalement, nous avons opté pour un achat de 24 serveurs au total (12 serveurs pour chacun des sites), chaque serveur contenant 8 processeurs de 2.66 GHz, 16 Giga de mémoire et 24 disques de 750 Go. Ce choix a été basé d'après l'estimation que nous avons effectuée. Implanter ces nouveaux serveurs sur chaque site nous permet de baisser le taux d'occupation des serveurs fichiers à environ 50 pourcent.

16.5 Le sous-système « Gestion de la base de données »

Dans notre système, la base de données que nous nous avons mise en place est un système très utile pour aider à la production. Cette mise en place permet de fournir un catalogue fichiers qui aidera les utilisateurs dans leurs recherches à travers la base de données. Cette mise en place nous permet également de classifier les données selon la date d'arrivée sur le site ou leur propriété et ce type de classification nous permet de mieux gérer nos données. Comme nous le savons, notre base de données a besoin d'être améliorée. Dans les paragraphes suivants nous allons présenter la réalisation du sous-système « gestion de la base de données ».

Mise à jour automatique de la base de données

Actuellement nous avons deux bases de données en production, une à l'université Wisconsin et l'autre au CERN. Les deux bases de données contiennent des informations concernant les Datasets et les fichiers, comme leur nom, la date de création, etc. La mise à jour de ces bases de données s'effectue une fois par jour. Comme on le sait, la production « base de données » ne fait pas partie de la production la plus importante de notre système, mais elle permet de fournir des informations très utiles à notre groupe et à nos utilisateurs.

Dans la phase d'ingénierie des besoins et la phase d'analyse de la production, nous avons évoqué plusieurs problèmes. Le premier est que la mise en place d'une base de données sur chaque site alourdit la charge de travail des administrateurs. Le deuxième est celui de la mise à jour : actuellement la mise à jour de la base de données est effectuée une fois par jour. Cette pratique ne convient plus à nos utilisateurs.

Pour ces deux raisons, nous avons décidé de revoir nos bases de données afin de les améliorer pour satisfaire nos utilisateurs. Pour offrir une base de données facile à utiliser, à administrer, pour fournir des informations utiles pour tous, nous avons décidé de commencer par une feuille blanche pour construire une nouvelle base de données permettant de mettre à jour automatiquement avec un rythme plus soutenu, de faciliter la gestion et la maintenance de la base de données et de fournir une interface unique pour faciliter la recherche des informations.

Pour la réalisation de ce composant, nous avons effectué cinq travaux :

- La mise en place d'une nouvelle structure de la base de données ;
- Le développement d'une interface Web qui permet aux utilisateurs d'interroger la nouvelle base de données ;
- Le développement d'un service qui permet de mettre à jour la base de données locale avec les informations concernées ;
- Le développement d'un service qui permet de gérer le transfert de données entre nos deux sites ;
- Le développement d'un service qui nous permet de soumettre des travaux de transformation des données.

La figure 29 est une représentation de la réalisation de la base de données et des programmes associés. A travers cette figure, nous pouvons constater que nous avons déployé une interface Web pour présenter les données enregistrées dans notre base de données locale. Les utilisateurs peuvent consulter et manipuler les données à travers l'interface Web. Nous pouvons également voir que notre nouvelle base de données est alimentée par trois services que nous

avons développés. Ces trois services sont « Transfert_souscription », « Job_soumit » et « DDM_input ».

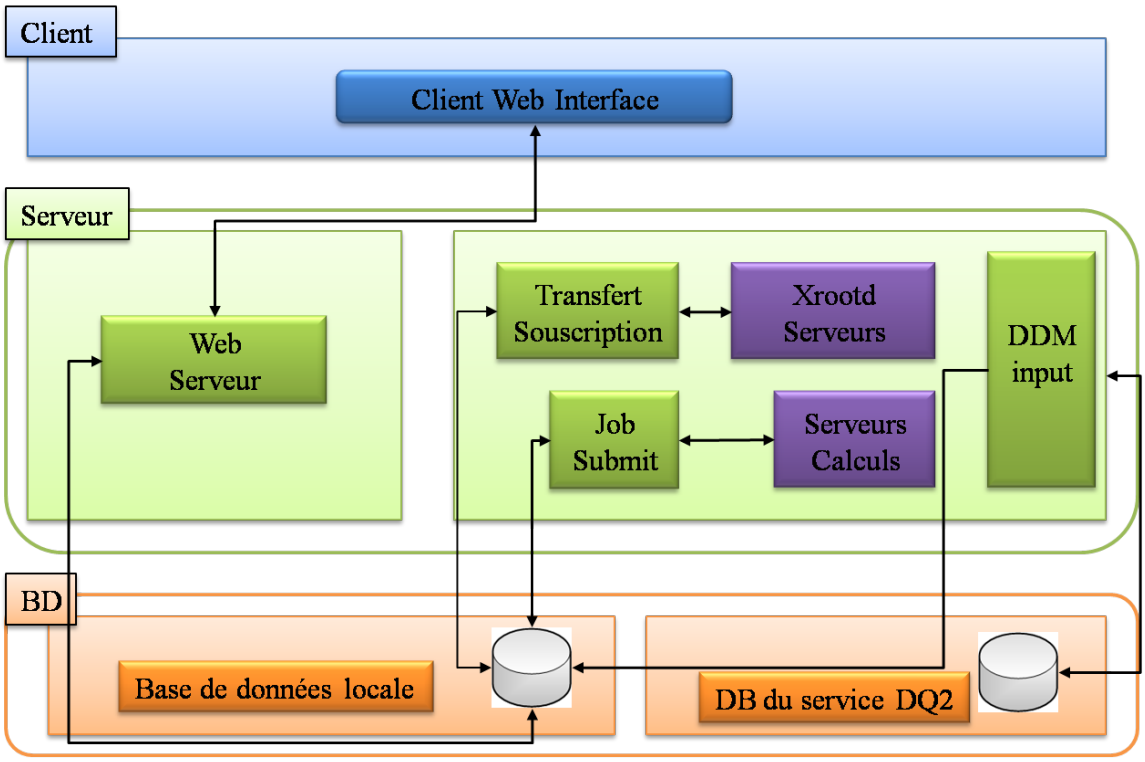

Figure 29 : Diagramme d'architecture de la base de données

Le service « DDM_input » est un programme qui permet d'alimenter notre base de données locale avec des informations concernant des données provenant du Grid. Comme nous le savons, le DQ2 est un middleware Grid qui permet de gérer des données. Le service DQ2 contient une base de données SQL. Cette base de données collecte des informations concernant des données transférées via le Grid. Lorsque des données transférées sont sur notre site, la base de données du service DQ2 va être automatiquement mise à jour. Alors le service « DDM_input » est un programme qui permet de surveiller le mouvement de la base de données du service DQ2 et de mettre à jour notre base de données locale des nouvelles données apparaissant sur notre site.

Le service « Job_submit » est un service qui nous permet de transformer des données AOD à des données DPD ou CBNT. Lorsque des données sont traitées et enregistrées sur nos serveurs fichiers, le service va mettre à jour notre base de données locale avec les informations telles que les noms des fichiers, la version de logiciel utilisée, leurs locations, etc.

Le service « Transfert_souscription » est un service qui permet de souscrire des données à transférer à partir du site Wisconsin au site CERN. Ce service permet également de superviser le transfert de données lorsque des données ont été transférées, il est capable de mettre à jour notre base de données locale avec les informations telles que leurs locations, la date de transfert et leurs chemins, etc.

16.6 Conclusion

Dans ce chapitre nous avons présenté les différentes tâches que nous avons réalisées pour mener bien à notre projet. L'étape de réalisation est basée sur quatre sous-systèmes. La réalisation du sous-système « architecture du système » nous permet d'assurer la continuité de nos productions et d'améliorer la performance de notre système. La réalisation du sous-système « infrastructure du système » nous permet de maintenir le bon fonctionnement de notre système actuel et d'accueillir convenablement les nouvelles données arrivant sur nos sites. La réalisation du sous-système « la production » nous permet de gérer la production PROOF et de pré-valider les Datasets afin d'améliorer la performance de la production PROOF. Et la réalisation du sous-système « gestion de la base de données » permet de consulter les informations de nos données et de les manipuler à travers notre base de données locale et des programmes associés.

Conclusion de la quatrième partie

Dans cette partie, nous avons présenté les différentes actions que nous avons menées pour le développement de notre projet. Nous avons commencé par l'élaboration de la stratégie, et par la suite nous avons découpé le projet en plusieurs sous-systèmes et composants et préconisé une démarche pour la suite de la réalisation. Enfin, nous avons retaillé les différentes étapes de la réalisation que nous avons effectuées pour les composants.

Pour élaborer la stratégie, nous avons d'abord étudié les moyens que nous pouvons réunir pour le développement du projet. Par la suite, nous avons étudié les risques qui pourraient avoir des impacts sur notre projet. L'étude des risques nous permet de bien connaître l'environnement du projet et prendre les précautions nécessaires pour son développement.

Le découpage du projet est basé sur les objectifs que nous avons déterminés dans la phase d'élaboration de la stratégie. Le choix de l'approche PBS nous amène à identifier toutes les fonctions qui composent le livrable à la fin du projet.

L'adoption de la méthode agile pour la réalisation du projet permet aux utilisateurs d'être impliqués dans le développement du projet. Basé sur la conception rapide, le développement de prototype et la validation du composant par des tests, elle offre une très grande réactivité en visant à satisfaire les besoins de nos utilisateurs.

La réalisation du projet est basée sur les quatre sous-systèmes que nous avons dans la phase du découpage du projet. À travers le développement des différents composants, nous avons pu renforcer la fiabilité et la robustesse de notre système. Nous avons également pu améliorer la performance de notre système tout en respectant la contrainte budgétaire.

Le bilan

Le système d'information du groupe Wisconsin est indispensable pour l'activité du groupe. La performance et la fiabilité de notre système représentent un enjeu important, à la fois stratégique, technique, économique et humain.

Le bilan scientifique et technique

Le projet de refondre une nouvelle architecture se déroule dans un contexte particulier. Tout d'abord, le groupe possède un système qui est en cours de production et dans la phase de réalisation, il est indispensable de prendre en compte cette situation. Les technologies déployées dans notre système sont souvent complexes et méconnues, pour les maîtriser nous devons prendre le temps pour bien les connaître. Pour cette raison, nous avons commencé le projet par l'analyse des technologies déployées dans notre système. Ensuite nous avons étudié les besoins des utilisateurs et notre système existant. Après avoir achevé ces deux étapes, nous avons commencé à chercher des solutions possibles pour améliorer notre système.

Au cours de la réalisation du projet, nous avons focalisé nos efforts sur quatre sous-systèmes, qui sont : « L'architecture du système », « La production », « L'infrastructure du système » et « La gestion de la base de données ». Et pour chaque sous-système, nous avons défini un ou plusieurs composants à réaliser.

Pour réaliser le sous système « Architecture du système », nous avons trois composants : « le Clustering actif/passif », « la meilleure exploitation des CPU » et « la réduction de la transaction réseau ». La réalisation du composants « Clustering actif/passif » nous permet d'améliorer la fiabilité et la disponibilité de notre système et la réalisation du composant « meilleure exploitation des CPU » et « la réduction de la transaction réseau » nous permettent de mieux exploiter nos puissances de calculs et de réduire la transaction réseaux interne afin d'améliorer la performance de notre système.

La réalisation du sous-système « La production » fut une collaboration avec l'équipe de développement « PROOF ». Ce sous-système comporte deux composants qui sont « la gestion de travaux PROOF » et « La pré-validation des Datasets ». La réalisation de ces deux composants nous permet non seulement de gérer des travaux PROOF, mais aussi de pré-valider des Datasets afin de réduire le temps d'exécution d'un travail PROOF.

La mise en place du sous-système « Infrastructure du système » est conditionnée par la situation financière du groupe. L'implantation des nouveaux équipements réseau du côté CERN nous permet d'augmenter sa capacité de transaction de données. Et la mise en place des nouveaux

serveurs fichiers sur les deux sites de production nous permet d'accueillir confortablement des données pour les deux prochaines années.

Enfin la réalisation du sous-système « Gestion de la base de données » nous permet de mieux gérer nos données. En associant différents programmes, nous offrons la possibilité d'automatiser une partie de la production et de transférer des données entre les deux sites.

Le bilan personnel

La réalisation de ce projet fut pour moi un grand enrichissement personnel non seulement du point vue technique mais également du point de vue humain. En effet, le succès d'un projet dépend de plusieurs paramètres, comme la compétence technique, la définition claire des besoins, l'implication des utilisateurs, le soutien de la hiérarchie, etc.

Pendant la phase de la réalisation du projet, j'ai pu utiliser différentes approches et méthodes pour accomplir diverses tâches. Ce sont des approches et des méthodes que j'avais apprises au cours des dernières années de ma formation et des recherches personnelles en consultant de nombreux ouvrages. J'ai pu utiliser dans un contexte réel ces différentes approches et méthodes, comme l'ingénierie des besoins, le découpage du projet et l'intégration de la méthode agile.

Tout au long de ce projet, les aspects humains ont joué un rôle très important. Dans la phase d'ingénierie des besoins, une collaboration étroite avec nos utilisateurs m'a permis de définir les besoins fonctionnels et non fonctionenels. Et c'est bien cette collaboration étroite avec nos utilisateurs qui permet de développer différents composants en garantissant la qualité de notre réalisation et la satisfaction de chacun.

Conclusion

Pendant la période de la réalisation du projet, nous nous sommes appuyés sur une collaboration étroite avec nos utilisateurs, que ce soit dans la phase d'ingénierie des besoins que dans la phase de développement. Intégrer la méthode « Agile » dans la phase de développement, nous permet non seulement de travailler avec les utilisateurs, mais également de livrer et intégrer dans notre système des livrables au fur et à mesure.

Notre projet nous permet d'apporter trois avancées majeures pour notre système. Tout d'abord, nous avons renforcé la disponibilité de notre système d'information grâce à la mise en place des serveurs Actif/Passif et la mise en place des nouveaux serveurs fichiers.

Ensuite, nous avons amélioré la performance de notre système. Cette amélioration est due aux optimisations que nous avons apportées à notre système, comme par exemple la réutilisation des CPU des serveurs fichiers pour la production, la mise en place de nouveaux équipements réseaux, la pré-validation des Datasets et la réduction de la transaction réseau interne.

Les améliorations que nous avons apportées à la gestion des données et la gestion de la production PROOF constitue la troisième avancée. Grâce à ces améliorations, consulter des informations des Datasets et organiser les transferts des Datasets entre les deux sites devient possible. Nous pouvons aussi changer la priorité des travaux PROOF.

Le travail que j'ai fourni constitue une avancée sérieuse. Bien que ce projet soit arrivé à son terme, la recherche de la performance et la robustesse ne doivent pas s'arrêter là. Comme nous le savons, dans les années qui viennent, nous aurons de plus en plus de données à traiter et à stocker. Pour préserver la qualité et la performance de notre système, nous devons continuer à migrer notre infrastructure réseau vers 10 Gbit/s et maintenir le taux d'occupation des serveurs fichiers que nous avons préconisé. Nous devons également chercher des solutions pour essayer d'équilibrer la charge de travail sur chaque serveur afin d'améliorer la performance de notre système d'information.

Table des figures

Figure 1 : Les expériences de LHC 12
Figure 2 : Infrastructure du système d'information du groupe (côté CERN) 19
Figure 3 : Infrastructure du système d'information du groupe (côté Université du Wisconsin)..... 20
Figure 4 : Architecture du système d'information du groupe (côté Université du Wisconsin) 22
Figure 5 : Illustration d'organisation du système Condor sur un réseau de clusters 29
Figure 6 : La répartition de charge dynamique Xrootd 34
Figure 7 : Xrootd architecture 35
Figure 8 : Exemple une interface ROOT et son interface graphique 36
Figure 9 : PROOF Architecture..... 37
Figure 10 : PROOF-Lite..... 38
Figure 11 : Architecture Protocolaire de Grid Computing 41
Figure 12 : Le système Condor dans un environnement Grid 44
Figure 13 : Cas d'utilisation sous-système « PROOF »..... 57
Figure 14 : Diagramme de séquence « Gérer des travaux PROOF » 58
Figure 15 : Diagramme de séquence « mettre à jour la base de données » 59
Figure 16 : illustration de l'utilisation de "iperf" 63
Figure 17 : Illustration du flux de données interne..... 77
Figure 18 : Présentation du chemin du flux de données externe et ses composants..... 78
Figure 19 : Travaux Grid - Pilot 82
Figure 20 : PROOF interface graphique..... 87
Figure 21 : Le fonctionnement du Packetizer..... 88
Figure 22 : Modèle des Entités Association de la base de données..... 89
Figure 23 : Objectifs du système d'information 100
Figure 24: Le découpage du projet..... 104
Figure 25 : Illustration du fonctionnement du Clustering Actif/Passif..... 111
Figure 26 : La co-habitation entre la Production PROOF et Condor 112
Figure 27 : illustration de la nouvelle architecture et de la réduction de la transaction 114
Figure 28 : La mise en place des nouveaux switch pour le site du côté CERN 118
Figure 29 : Diagramme d'architecture de la base de données..... 121

Liste des tableaux

Tableau 1: Les caractéristiques des expériences du LHC 13
Tableau 2 : Le coût relatif pour la correction d'un défaut dû à l'ingénierie des besoins 48
Tableau 3 : Les besoins capturés 54
Tableau 4 : L'analyse des besoins 55
Tableau 5 : Spécification des besoins non-fonctionnels 60
Tableau 6 : Test performance du réseau 64
Tableau 7 : Les différents niveaux de RAID 66
Tableau 8 : Test single disque, 8 disques et 24 disques en RAID 5 67
Tableau 9 : Présentation des serveurs du site Wisconsin 68
Tableau 10 : Présentation des serveurs du site du CERN 69
Tableau 11 : Taux d'utilisation des CPU 69
Tableau 12 : Comparatif des différentes formes de scalabilité 72
Tableau 13 : Prséntation des serveurs Middleware du groupe 74
Tableau 14 : Le degré de la maîtrise des technologies 96
Tableau 15 : Le prix des matériels 97
Tableau 16 : Profil du projet 98
Tableau 17: la décomposition du projet en sous-systèmes 103
Tableau 18 : Les commandes d'outil PQ2 117

Bibliographie

[1] Akoka J., Wattiau I., 2008. L'ingénierie des besoins. CNAM., Paris., 106 p

[2] Abbas A., 2004. GRID COMPUTING : A Practical Guide to Technology and Application. CHARLES RIVER MEDIA., Hingham., 407 p.

[3] Berman F., Fox G C., Hey A J G., 2003. Grid Computing Making the Global infrastructure a Reality. WILEY., Chichester., 1012 p.

[4] Boehm B. W. 1981. Software Engineering Economic. Englewood Cliffs, Prentice-Hall., 767p.

[5] Caseau Y., 2007. Performance du système d'information. Dunod., Paris., 253 p.

[6] Contini I., 2002. L'apport de l'ingénierie des besoins à l'ingénierie de l'urbanisme des systèmes d'information (USI). Université de Paris 1 Sorbonne, 8 p.

[7] Foster I., 2002. What is the Grid? A Three Point Checklist

[8] Foster I., Kesselman C., 2004. The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann., San Francisco, 733p.

[9] Freeman R.E., 1984. Strategic Management: A stakeholder approch. Pitman., Boston, 276 p.

[10] Grady Robert B.**,** 1999 "An Economic Release Decision Model: Insights into Software Project Management.", Proceedings of the Applications of Software Measurement Conference, Orange Park, FL: Software Quality Engineering. 227-239 p.

[11] Grojean P., Morel M., Plouin G., 2007. Performance des Architectures IT, DUNOD., Paris, 269 p.

[12] Grosz G., 2002. Ingénierie des besoins : problème et perspectives. Université de Paris 1 Sorbonne, 2 p.

[13] Joliot D., 2003. Performances des SI : vérification, comparaisons, tests et mesures au service du management de l'entreprise, LAVOISIER, Paris, 330 p.

[14] Kulak D., Guiney E., 2000. USE CASES Requirements In Context. ACM Press., New York. 329 p

[15] Leffingwell D., Widrig D., 2003. Managing Software Requirements Second Edition A Use Case Approach., Addison-Wesley. Boston. 502 p.

[16] Li M Z., Baker M., 2005., The Grid core technologies. Wiley. Chichester. 423 p.

[17] Longépé C. 2006. Le Projet d'Urbanisation du S.I. Démarche pratique avec cas concret., DUNOD. Paris. 296 p.

[18] Orfali R., Harkey D., Edwards J., 1999. CLIENT/SERVEUR Guide de survie 3^{e} édition. Vuibert., Paris. 782 p

[19] Rolland C., 2001., Partie 2 Module « Fondements des SI » : Ingénierie des besoins. Université de Paris 1 Sorbonne, 61 p.

[20] Roques P.,2003., Les Cahiers du programmeur UML Modélisation un site e-commerce. EYROLLES., Paris. 150 p.

[21] Sharp H., Finkelstein A., Galal G., 1999. Stakeholder Identification in the Requirements Engineering Process. IEEE CS Press. 387-391 p

[22] Sommerville I., 2007., Software Engineering 8. Addison Wesley., Harlow. 840 p

[23] Tawbi M., 2001., CREWS-L'Ecritoire : Une apporche Guidant l'Ingénierie des Besoins. INFORSID. Martigny Suisse. 123-141 p.

[24] Wiegers K. E. 2006. More About Software Requirements Thorny Issues and Pratical Advice. Microsoft Press., Washington. 201 p.

[25] Yu E., Mylopoulos J., 1998. Why Goal-Oriented Requirements Engineering. Université de Toronto, 6 p.

Webgraphie

ATLAS

http://www.atlas.ch/

http://atlas.web.cern.ch/Atlas/index.html

CERN

http://public.web.cern.ch/

http://fr.wikipedia.org/wiki/Organisation_europ%C3%A9enne_pour_la_recherche_nucl%C3%A9aire

Globus Toolkit

http://www.globus.org/

Grid

http://www.gridcafe.org/

Ingénierie des besoins

http://fr.wikipedia.org/wiki/Partie_prenante

http://en.wikipedia.org/wiki/Requirement_analysis

LHC

http://www.lhc-france.fr/?article6

http://lhc-milestones.web.cern.ch/LHC-Milestones/LHCMilestones-fr.html

RAID

http://fr.wikipedia.org/wiki/RAID_(informatique)

Scalability

http://www.linux-france.org/prj/jargonf/E/extensibiliteac.html

VDT

http://vdt.cs.wisc.edu/index.html

Index

AOD 18
ATLAS 133
Autopilot 83
Boson de Higgs 15
CBNT 18
CERN 133
Condor 133
Condor-G 83
disponibilité 24
DPD 18, 20, 21
ESD 17
Gatekeeper 43, 74, 83
Globus Toolkit 42, 131
GRAM 41
GRID 17, 133
GridFTP 32
GRIS 42
GSI 41
GUID 79
High-Performance Computing 28
High-Throughput Computing 28
LCG 40
LFC 44
LFN 79
MTBF 24
MTTR 24
OBS 102
Panda 78, 83
PBS 102
Proof 133
RAD 105
RAID 19
RAID5 ..... 20, 64
RAW 17
robustesse 24, 133
SRM 44
SUSY 15, 16
tolérance aux pannes 24
UML 49
WBS 102
XP 105
Xrootd 133

Résumé et mot clé

Une nouvelle architecture cluster pour la recherche scientifique : Condor, PROOF et Xrootd

Mémoire d'ingénieur C.N.A.M., Lyon 2010 Hao. Ni

RESUME

Le groupe de l'université du Wisconsin-Madison participe à l'expérience ATLAS au CERN depuis 1993. Il fut le premier groupe américain rejoignant cette expérience. Pour faciliter leurs recherches ce groupe possède une centaine de serveurs qui permettent d'effectuer des calculs scientifiques selon les besoins. Les besoins en fonctionnalité, performance et robustesse nous obligent à revoir l'ensemble de leur système informatique. L'objectif de ce projet est donc de mettre en place une nouvelle architecture pour répondre aux besoins spécifiques du groupe.

Par ailleurs, la mise en place de cette nouvelle architecture présente de nombreuses difficultés : les tailles des fichiers sont souvent très variées (10-1000MB), les charges de travail sont intenses, les technologies employées dans ce projet tel que Xrootd, PROOF, Condor, Grid sont très complexes. De plus, dans le domaine de la mise en place des clusters, il existe peu de méthodologies, de règles formelles pour la phase d'analyse, de conception et de mise en œuvre.

C'est dans ce contexte que je vais mener ma propre démarche à partir de l'analyse du système existant et des besoins des utilisateurs pour élaborer la stratégie du système informatique du groupe, de concevoir et de réaliser une nouvelle architecture afin de répondre aux besoins du groupe.

Mots clés : Condor, Grid, PROOF , Xrootd

SUMMARY

The group of Wisconsin-Madison participates in the ATLAS experiment at CERN since 1993. It was the first American group joining this experiment. To facilitate its works, the group has hundreds of servers to perform scientific researches. The needs for functionalities, performance and robustness ask us to review our computer system to propose a new architecture to the specific needs of the group.

Moreover, the introduction of a new architecture represents many difficulties: the range of file size is usually very large (10-1000 MB), the workload is very hard predicate and the technologies used in our system such as Xrootd, Proof, Condor, Grid are very complex. There is little methodology in the establishment of clusters and also no formal rules for the analysis phase in the design and implementation.

Under this context, I will conduct my own approach based on the analysis of the existing system and engineering requirements to develop a new architecture to meet the group's needs.

Key words : Condor, Grid, PROOF , Xrootd