ORSAY
No. d'ordre:

UNIVERSITE DE PARIS-SUD

CENTRE D'ORSAY

THESE

présentée

pour obtenir

le grade de Docteur en Sciences

par

Denis CALVET

SUJET:

Réseau à Multiplexage Statistique pour les Systèmes de Sélection et de Reconstruction d'Evénements dans les Expériences de Physique des Hautes Energies.

Soutenue le 31 Mars 2000 devant la Commission d'examen

MM.	Maris	Abolins	
	Etienne	Augé	Directeur de thèse
	Pascal	Debu	
	Jean-Pierre	Dufey	Rapporteur
	Alain	Mérigot	Président
	François	Touchard	Rapporteur

REMERCIEMENTS

Je tiens tout d'abord à remercier les physiciens du Service de Physique des Particules (SPP) du DAPNIA[1] (CEA[2] Saclay) impliqués (ou ayant été impliqués) dans le projet de trigger de niveau 2 pour l'expérience ATLAS: Alexis Amadon, Jiri Bystricky, Patrick Le Dû et Dick Hubbard pour leur participation très active qui a apporté de nombreuses idées novatrices à la collaboration ATLAS. Je remercie également Jean Ernwein qui supervise l'ensemble des activités relatives à ATLAS pour le groupe de Saclay.

Je remercie François Darnieaud, chef du Service d'Electronique et d'Informatique (SEI) du DAPNIA, et Michel Mur, chef de groupe, pour la confiance et le soutien qu'ils m'ont accordés pour engager et mener à bien cette thèse. Je remercie mes collaborateurs directs du SEI, notre scientifique du contingent Jean-Marc Conte, Olivier Gachelin, Trandl Hansl-Kozanecka, Michel Huet, Bruno Thooris, et plus particulièrement Irakli Mandjavidze qui a joué un rôle majeur à tous les stades de notre projet.

Mes remerciements vont également à Jean-Pierre Dufey et Mike Letheren du CERN, Genève, qui ont dirigé les premières études sur l'utilisation de la technologie ATM pour les systèmes de sélection et de reconstruction d'événements et ont accueilli le groupe de Saclay dans la collaboration RD-31 en manifestant une grande ouverture d'esprit.

Je remercie Etienne Augé du Laboratoire de l'Accélérateur Linéaire d'Orsay d'avoir été mon directeur de thèse, Jean-Pierre Dufey et François Touchard pour leur patiente relecture de mon manuscrit. Mes remerciements vont également à Maris Abolins (Michigan State University), Pascal Debu (SPP Saclay) et Alain Mérigot (Institut d'Electronique Fondamentale d'Orsay) qui ont accepté de faire partie de mon jury.

Je remercie Masaharu Nomachi du Research Center for Nuclear Physics de l'université de Osaka, Japon, qui a fait mettre à ma disposition le système informatique de cet institut et a obtenu les moyens financiers me permettant de venir exploiter ce système sur le site pendant plusieurs mois.

Je remercie nos collaborateurs de tous les instituts impliqués dans le T/DAQ de l'expérience ATLAS, et plus particulièrement ceux qui ont contribué de manière directe à mon activité du point de vue matériel, scientifique ou humain: personnels du CERN, de l'Argonne National Laboratory et de Michigan State University (USA), du Rutherford Appleton Laboratory (Royaume-Uni), de l'Université de Mannheim (Allemagne), du Centre de Physique des Particules de Marseille et du Nagasaki Institute of Applied Science (Japon).

De manière plus générale, j'exprime ma reconnaissance envers le CEA et l'unité à laquelle j'appartiens pour m'avoir fourni un cadre de recherche de qualité et des moyens adéquats.

1. Département d'Astrophysique, de Physique des particules, de physique Nucléaire et d'Instrumentation Associée
2. Commissariat à l'Energie Atomique

TABLE DES MATIÈRES

Réseaux pour la Reconstruction d'Événements

Protocole de Communication à Haute Performance

Démonstrateurs du Modèle du Technical Proposal

Démonstrateurs du Modèle de Trigger Séquentiel

Projet Pilote du Trigger de Niveau 2 d'ATLAS

LISTE DES ACRONYMES

AAL **ATM Adaptation Layer**
couche d'adaptation pour l'accès à la couche ATM

ASIC **Application Specific Integrated Circuit**
circuit intégré pour application spécifique

ATD(M) **Asynchronous Time Division (Multiplexing)**
acronyme initialement utilisé pour désigner l'ATM

ATLAS **A Toroidal LHC ApparatuS**
expérience en cours de construction au LHC

ATM **Asynchronous Transfer Mode**
technologie de réseau "mode de transfert asynchrone" ou TTA

B-ISDN **Broadband Integrated Services Digital Network**
Réseau Numérique à Intégration de Services (RNIS) large bande

CBR **Constant Bit Rate**
connexion à débit constant (classe de service en ATM)

CCITT **Comité Consultatif International Télégraphique et Téléphonique**
organisme de standardisation dans les télécommunications (devenu ITU)

CERN **Centre Européen pour la Recherche Nucléaire**
centre de recherche situé à proximité de Genève

CMS **Compact Muon Solenoid**
expérience en cours de construction au LHC

CNET **Centre National d'Etude des Télécommunications**
laboratoire de recherche de France Télecom (9 sites en France; 1 en Californie)

CPU **Central Processing Unit**
processeur, unité centrale

CRC **Cyclic Redundancy Check**
code cyclique redondant pour la détection et la correction d'erreurs

CSMA/CD **Carrier Sense Media Access with Collision Detection**
accès multiple avec écoute de porteuse et détection de collisions

DMA **Direct Memory Access**
accès direct en mémoire

DSP **Digital Signal Processor**
processeur destiné au traitement numérique du signal

FDDI **Fibre Distributed Data Interface**
technologie de réseau utilisant des fibres optique (100 Mbit/s)

HIPPI **High Performance Parallel Interface**
technologie de transport de l'information à haut débit (100 Mo/s)

ITU **International Telecommunication Union**
organisme international de normalisation dans les télécommunications

LAN **Local Area Network**
réseau local

LEP **Large Electron Positron collider**
grand collisionneur électron-positon en service au CERN

LHC **Large Hadron Collider**
grand collisionneur de hadrons en construction au CERN

MAC **Medium Access Control**
contrôle d'accès au médium

MAN **Metropolitan Area Network**
réseau métropolitain

PC **Personal Computer**
ordinateur personnel

PCI **Peripheral Component Interconnect**
bus d'interconnexion de périphériques, norme de bus très répandue aujourd'hui

PDH **Plesiochronous Digital Hierarchy**
ancienne technologie pour les réseaux de télécom remplacée par SDH/SONET

PMC **PCI Mezzanine Card**
carte PCI Mezzanine, format de carte électronique utilisant le bus PCI

PVC **Permanent Virtual Connection**
connexion virtuelle permanente

RNIS **Réseau Numérique à Intégration de Services**
réseau unifié pour le transport de la voix, des données et flux multi-média

ROB **Read-Out-Buffer**
carte mémoire tampon utilisée dans l'expérience ATLAS

SAR **Segmentation And Re-assembly (layer, device)**
(élément, couche) de segmentation et ré-assemblage en ATM

SBC **Single Board Computer**
ordinateur sur carte

SCI **Scalable Coherent Interface**
technologie de réseau basé sur des anneaux à haut débit (1 Go/s)

SDH **Synchronous Digital Hierarchy**
équivalent de SONET pour les réseaux de télécommunication en Europe

SONET **Synchronous Optical NETwork**
technologie de transport de l'information (réseaux de télécom Américains)

SSC **Super Superconducting Collider**
projet de collisionneur de particules au Texas (projet abandonné en 1993)

SRAM **Static Random Access Memory**
mémoire statique à accès aléatoire (mémoire statique volatile)

SVC **Switched Virtual Connection**
connexion virtuelle commutée

T/DAQ **Trigger and Data AcQuisition (system)**
(système de) déclenchement et d'acquisition de données

TTA **Transmission Temporelle Asynchrone**
traduction française de l'acronyme ATM

UBR **Unspecified Bit Rate**
connexion à débit non spécifié (classe de service en ATM)

VPI/VCI **Virtual Path Identifier, Virtual Channel Identifier**
identificateur de connexion virtuelle en ATM

WAN **Wide Area Network**
réseau étendu

WWW **World Wide Web**
réseau d'information mondial basé sur Internet

1. Contexte de l'étude

Parmi les nombreux progrès scientifiques du XX^{ème} siècle, notre connaissance des constituants élémentaires de la matière a connu un développement particulièrement spectaculaire. Alors que les Grecs anciens Leucippe et Démocrite imaginaient l'existence d'atomes – matière indivisible – dès le V^{ème} siècle avant notre ère, peu de progrès ont été faits par la suite, jusqu'à la fin du XIX^{ème} siècle avec les travaux de Thomson puis ceux de Rutherford mettant en évidence le proton et prédisant l'existence du neutron qui ne sera découvert par Chadwick qu'en 1932, douze ans après l'hypothèse de Rutherford. Les découvertes de nouvelles particules se sont enchaînées par la suite à un rythme soutenu, parallèlement à l'établissement du "Modèle Standard" de la physique moderne qui permet de décrire toutes les formes de matière connues à l'aide de 6 quarks et de 6 leptons, et d'unifier deux des quatre forces fondamentales connues, la force électromagnétique et la force nucléaire faible [Cot98].

Bien que le Modèle Standard n'ait jamais été contredit par l'expérimentation, certains points ne sont pas satisfaisants, en particulier ceux qui concernent l'explication du mécanisme de brisure de symétrie électrofaible. Elle pourrait être due à des particules inobservées à ce jour: les bosons de Higgs dont l'étude permettrait de comprendre l'origine de la masse inertielle des particules (fermions et médiateurs des interactions).

Une théorie plus générale que le Modèle Standard unifiant à la force électrofaible la force nucléaire forte et, par certaines extensions, intégrant la gravitation est avancée: c'est la "super-symétrie". Une des manifestations observables serait l'existence d'une particule "super-symétrique" associée à chacune des particules élémentaires connues.

La mise en évidence, sous réserve d'existence, du (ou des) boson(s) de Higgs, la découverte d'hypothétiques particules super-symétriques et la recherche de phénomènes inexplicables dans le cadre du Modèle Standard sont parmi les principales tâches expérimentales en physique des particules pour la prochaine décennie [Gun90], [Nat98].

2. Les expériences en physique des hautes énergies

La première particule élémentaire découverte fut l'électron, en 1897, par Joseph John Thomson à l'aide d'un tube à rayons cathodiques et de relativement peu de matériel. La mise en évidence d'autres particules élémentaires a fait appel à du matériel de plus en plus volumineux et complexe. Ces découvertes ont été possible non seulement par les nombreux progrès techniques et industriels de notre société moderne mais aussi grâce à la mise en place d'infrastructures et d'équipes de recherche dotées de moyens suffisants pour réaliser les équipements de production artificielle de particules et les détecteurs associés.

Afin de produire et observer ces particules éphémères, des collisionneurs de particules opérant à des niveaux d'énergie de plus en plus élevés ont été construits. Les bosons W et Z^0,

découverts au CERN en 1983 par les expériences UA1 et UA2, sont aujourd'hui produits en grande quantité dans les expériences du collisionneur LEP: un peu plus de 4 millions de désintégrations de Z^0 ont été enregistrées par chaque expérience à ce jour. Le dispositif expérimental consiste en un anneau de 27 km de circonférence dans lequel circulent en sens contraire un faisceau d'électrons et un faisceau de positons de haute énergie (un peu moins de 100 GeV). En chacun des quatre points de croisement des faisceaux, une expérience est installée. Chacune aura mobilisé une équipe d'un peu plus d'une centaine de physiciens, ingénieurs et techniciens pendant plusieurs années. Les détecteurs installés sont de forme cylindrique (~10 mètres de diamètre et de longueur) et contiennent environ une centaine de milliers de capteurs électroniques.

La recherche du boson de Higgs et de particules super-symétriques nécessite la construction de nouveaux collisionneurs de particules afin d'atteindre des niveaux d'énergies suffisamment élevés pour explorer un large spectre de masse et maximiser le potentiel de découverte des équipements. Deux projets concurrents de collisionneur circulaire ont été proposés à la fin des années 80: le *Super Superconducting Collider* (SSC) de 81 km de circonférence, à construire sur un nouveau site au Texas, et le *Large Hadron Collider* (LHC), au CERN, devant venir en remplacement du collisionneur actuel dans le même tunnel de 27 km de circonférence. L'abandon du projet SSC a été voté par le congrès Américain fin 1993. Seul le LHC sera construit et devrait entrer en service fin 2005 selon le calendrier actuel.

Le LHC est un collisionneur proton-proton permettant d'atteindre une énergie de 7 TeV par faisceau, soit une énergie au centre de masse de 14 TeV. Cela représente un saut en énergie important par rapport au plus puissant collisionneur actuel, le Tévatron du laboratoire Fermi près de Chicago, qui atteint 2 TeV. Quatre expériences sont prévues au LHC, et chacune d'entre elles mobilise des équipes internationales de plus de mille personnes sur une durée qui dépasse une dizaine d'années. L'ensemble des détecteurs de l'expérience ATLAS mesure plus de 40 m de longueur pour un diamètre de 26 m et contient des ensembles très sophistiqués de capteurs totalisant plus d'une centaine de millions de canaux électroniques.

3. Sélection des événements en ligne dans ATLAS

L'extrême rareté des phénomènes que l'on souhaite observer impose d'adopter une fréquence de croisement des faisceaux au coeur des détecteurs très élevée (40 MHz). Chaque croisement est un "événement" au cours duquel se produisent statistiquement ~20 collisions proton-proton. Compte tenu de la fréquence des collisions et du nombre de capteurs sollicités, le flot de données brutes produit par les détecteurs d'ATLAS est supérieur au Téra-octet par seconde. Les contraintes techniques et économiques sur le système d'acquisition, de stockage et de traitement des données imposent d'effectuer une sélection en ligne des événements pour réduire le débit d'information à un niveau jugé acceptable sans compromettre le potentiel de découverte de l'expérience. On estime aujourd'hui que les données relatives à moins d'un événement sur un million seront stockées sur support permanent pour analyse différée. Le tri en ligne, est réalisé par un système de *Trigger/Data AcQuisition* (T/DAQ) ou système de déclenchement en ligne pour l'acquisition des données (souvent noté *trigger* par la suite pour plus de concision). Un premier niveau de sélection basé sur des critères suffisamment simples pour être implantés en électronique câblée à proximité des détecteurs, permet de réduire à la volée le taux d'événements d'un facteur 1000. Par la suite, les triggers de niveaux supérieurs appliquent des critères de plus en plus fins mettant en œuvre des algorithmes de complexité croissante pour réduire le flux de données et extraire les événements contenant les réactions physiques que l'on souhaite analyser finement lors des dépouillements

ultérieurs non contraints par les périodes de fonctionnement de l'expérience.

Compte tenu de la complexité des algorithmes de sélection et de filtrage des événements, le nombre de processeurs nécessaires pour effectuer le traitement en temps réel du flux de données est estimé à 1000, voire davantage. Des solutions utilisant les technologies de l'informatique distribuée sont indispensables. Par exemple, un puissant réseau de communication est nécessaire pour connecter les ~1600 sources de données provenant des détecteurs aux processeurs de traitement. La bande passante nécessaire totale est estimée à plusieurs dizaines de giga-bit par seconde. Les autres expériences au LHC ont des besoins du même ordre de grandeur: de cent à quelques milliers d'éléments à interconnecter avec un débit utile compris entre 10 et 500 Gbit/s. Pour satisfaire les besoins des T/DAQ des expériences antérieures, des systèmes basés sur des bus interconnectés en structure d'arbre étaient en général suffisants. Pour les systèmes de sélection d'événements et d'acquisition de données des expériences modernes, l'utilisation de réseaux performants et optimisés est devenu une nécessité compte tenu du nombre d'éléments à connecter et de la quantité d'information à véhiculer.

4. Les réseaux à haut débit

Jusque dans les années 70, les réseaux à haut débit étaient plutôt destinés aux applications de téléphonie et ce n'est que suite au remplacement dans les entreprises des calculateurs frontaux reliés à des terminaux par une informatique distribuée sur des stations de travail que la notion de réseau informatique local a pris naissance. Parallèlement au développement des réseaux locaux, l'interconnexion de sites distants dans le but d'échanges d'information entre calculateurs s'est mise en place avec la création de l'Internet.

La technologie Ethernet a connu un remarquable succès pour la réalisation de réseaux locaux et la version de base offrant un débit de 10 Mbit/s sur un médium partagé par plusieurs postes de travail a satisfait les besoins de nombreux utilisateurs pendant plusieurs années. Face à la demande croissante des utilisateurs, plusieurs améliorations ont été apportées et d'autres technologies ont été développées. La popularisation des ordinateurs personnels et l'extraordinaire expansion de l'Internet grâce au *World Wide Web* (WWW) depuis le début des années 90 entraînent aujourd'hui une forte croissance dans le développement des réseaux informatiques, locaux et étendus.

Bien que le transport de la voix représente à ce jour 80% du trafic sur les réseaux étendus, le potentiel de croissance du trafic de données (les 20% restants) est bien supérieur, tout du moins dans les pays développés. Cela pousse les opérateurs de télécommunication à diversifier leur offre et entraîne les fournisseurs d'accès à Internet à augmenter leur capacité d'accueil en abonnés, voire à tenter de transporter de la voix sur leur réseau en offrant des tarifs plus avantageux que les opérateurs traditionnels.

Le support scientifique, technique et technologique qu'offrent les universitaires, laboratoires et industriels impliqués dans le domaine des réseaux informatiques représente une somme énorme de connaissances et une source d'approvisionnement en équipements d'une grande diversité. Le dynamisme de ce secteur constitue une aide précieuse pour résoudre les problèmes spécifiques aux communications dans les systèmes de sélection d'événements et d'acquisition de données des futures expériences de physique.

5. But et travail de la thèse

Un des buts principaux de la thèse est de proposer une architecture pour le système de T/DAQ de l'expérience ATLAS. Le travail effectué est une analyse critique de l'architecture proposée initialement puis l'établissement et la justification du modèle défendu.

Dans cette thèse, l'accent est mis sur l'étude des réseaux de communication qui sont des constituants cruciaux du système considéré. Une étape préliminaire permet d'établir les fonctionnalités, les besoins et les spécificités des réseaux d'interconnexion pour les T/DAQ des expériences modernes en physique des hautes énergies et plus particulièrement ceux de l'expérience ATLAS. Le travail consiste à définir des modèles de réseau adaptés à l'application envisagée, à les étudier du point de vue théorique et par simulation, puis à les comparer.

Parmi les technologies de réseau envisagées par la collaboration ATLAS, celle qui est considérée dans cette étude est l'*Asynchronous Transfer Mode* (ATM). Un des buts de la thèse est de justifier ce choix (à posteriori) par comparaison avec les autres technologies candidates et de rechercher si cette technologie est (ou n'est pas) utilisable pour notre application, d'évaluer ses points forts et ses faiblesses. Un aspect important concerne les protocoles de communication: le travail consiste à définir, développer et mettre en œuvre un mécanisme d'échange de messages performant par réseau ATM. Cette tâche comporte le codage et l'optimisation d'un logiciel pilote de carte réseau.

La validation du principe de l'architecture proposée et l'étude de faisabilité de ce système en utilisant la technologie ATM sont des objectifs abordés par l'assemblage de modèles de démonstration à échelle réduite. Le travail consiste à connecter une ou plusieurs dizaines de machines par un réseau ATM, à développer en totalité le logiciel optimisé pour chacun des composants et à effectuer des mesures de performances permettant d'extraire divers paramètres caractéristiques.

L'extrapolation des résultats obtenus sur les bancs de test contribue à la démonstration de faisabilité du système final et à l'évaluation des performances attendues. Une proposition de solution complète pour la partie du T/DAQ d'ATLAS étudiée est présentée. L'extension des concepts développés dans cette thèse à d'autres expériences de physique conclut l'étude.

6. Résumé

Dans ce chapitre, j'ai présenté le contexte de l'étude et j'ai introduit le projet de *Large Hadron Collider* au CERN où sera installé la prochaine génération d'expériences de physique des hautes énergies. J'ai décrit brièvement le système de sélection en ligne d'événements de l'expérience ATLAS et j'ai mentionné l'importance des réseaux à haut débit pour sa réalisation. J'ai exposé le but de la thèse: définition d'une partie de l'architecture du T/DAQ d'ATLAS puis démonstration de faisabilité de ce système au moyen de modèles à échelle réduite basés sur la technologie ATM et extrapolation des résultats obtenus sur ces bancs de test.

CHAPITRE I. POSITION DU PROBLÈME

1.1. Introduction

Dans ce chapitre, je présente le projet de grand collisionneur de hadrons LHC et mentionne les quatre expériences qui y seront installées. Je décris de manière succincte les objectifs de physique principaux des expériences ATLAS et CMS ainsi que la structure des détecteurs d'ATLAS. Je montre le besoin d'effectuer une sélection en ligne des données issues des expériences et j'explique le mécanisme de sélection à niveaux multiples qui sera utilisé dans ATLAS. Je montre le besoin d'utiliser des réseaux informatiques à haut débit pour la réalisation du système correspondant.

1.2. Le *Large Hadron Collider*

Le *Large Hadron Collider* (LHC), grand collisionneur de hadrons, sera la machine la plus puissante au monde pour la production artificielle de particules de haute énergie. Cet instrument scientifique unique permettra d'atteindre une énergie au centre de masse de 14 TeV pour les collisions proton-proton, et d'environ 1300 TeV pour les collisions d'ions lourds de plomb. Cela représente une grande avancée en énergie par rapport à la plus puissante des machines d'aujourd'hui (2 TeV pour le Tévatron du laboratoire Fermi, près de Chicago) et ouvre de nombreuses perspectives dans le domaine de la physique des hautes énergies pour les quinze à vingt prochaines années. Ce projet très ambitieux, à la fois sur le plan technique et humain, sera réalisé au CERN à proximité de Genève en utilisant une partie des infrastructures déjà en place, en particulier le tunnel circulaire de 27 km de circonférence qui abrite le collisionneur électron-positon actuellement en service, le LEP (*Large Electron Positron collider*). Ce dernier sera démantelé aux environs de 2002 pour accueillir le nouveau collisionneur qui devrait entrer en service fin 2005.

Afin de pouvoir atteindre le niveau d'énergie souhaité dans une machine circulaire, les particules accélérées seront des protons qui, à la différence des électrons, ne perdent pas la majorité de leur énergie en rayonnement synchrotron induit par la courbure de leur trajectoire. Compte tenu du diamètre de l'anneau existant, l'intense champ magnétique (induction de 8,2 T) permettant de courber la trajectoire des protons afin de les maintenir sur leur orbite circulaire ne peut être produit qu'à l'aide d'aimants supra-conducteurs. Le LEP est basé sur des aimants classiques et la même source de champ magnétique est utilisée pour contenir les deux faisceaux de particules circulant en sens contraire mettant à profit le fait que celles-ci sont de charges opposées. Au LHC, comme les deux faisceaux circulant en sens contraire sont constitués de particules de même charge électrique, deux tubes distincts et deux champs magnétiques opposés doivent être créés. Environ 1200 dipôles de déflexion et 400 quadrupôles de focalisation sont répartis le long de la circonférence de l'anneau. Tous ces éléments supra-conducteurs sont refroidis à la température de 1,9 K par de l'hélium liquide (superfluide). Chaque faisceau de protons est constitué d'un train de 2835 paquets régulièrement espacés; chaque paquet contient environ 10^{11} particules. Des cavités accélératrices par radio-fréquence sont également placées pour accélérer les particules jusqu'à l'énergie nominale du collisionneur puis compenser les pertes afin de maintenir l'énergie des faisceaux constante.

Les faisceaux se croisent en quatre points le long de la trajectoire circulaire. Chaque point constitue le site où seront placés les détecteurs d'une des quatre expériences prévues. Une vue schématique du site expérimental est présentée Figure 1-1.

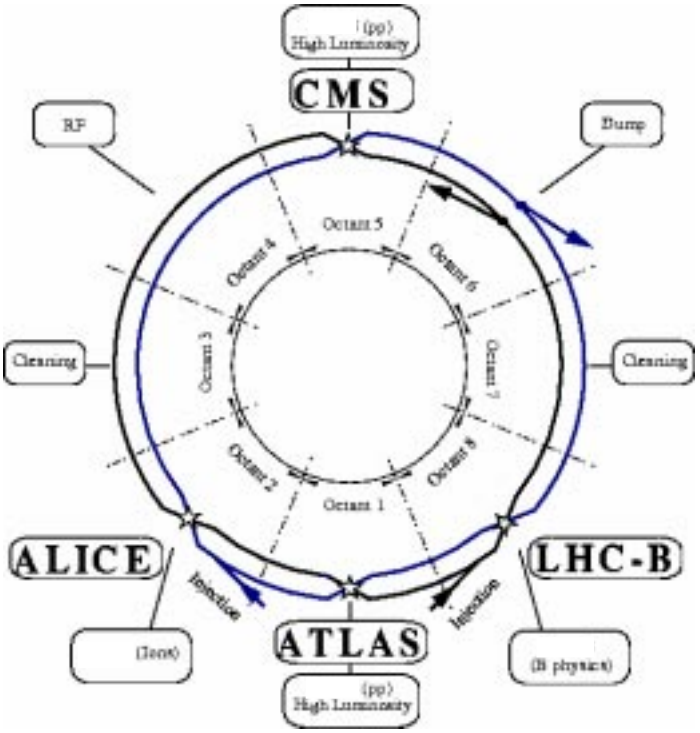


Figure 1-1: Anneau du LHC et emplacement des expériences.

En un point donné, la fréquence de croisement des faisceaux de particules est de 40 MHz (100 kHz au LEP). La luminosité[1] durant une phase initiale de fonctionnement de la machine sera de 10^{33} $cm^{-2}s^{-1}$ et atteindra par la suite la valeur nominale prévue de 10^{34} $cm^{-2}s^{-1}$ (5.10^{31} $cm^{-2}s^{-1}$ au LEP). A chaque croisement des faisceaux, la plupart des 10^{11} protons de chaque paquet se croisent sans interagir et il ne se produit environ que 20 interactions proton-proton. Celles-ci donnent naissance à une multitude d'autres particules et produits de désintégration dont la plupart découlent de réactions physiques connues et non de phénomènes inobservés que l'on cherche à découvrir. La recherche du boson de Higgs relève de deux défis majeurs: la production et la détection de cette particule. La production de particules de Higgs est un phénomène imprévisible qui demeure extrêmement rare malgré les importants moyens techniques mis en œuvre. Sur les 8.10^{8} interactions à chaque seconde, environ une toutes les 100 secondes en moyenne engendrerait un boson de Higgs. La détection de ces bosons est également très difficile. Seuls certains modes de désintégration du boson de Higgs sont susceptibles de signaler clairement cette particule par rapport au bruit de fond dominant. La précision des mesures, l'étalonnage des détecteurs et le suivi de leur fonctionnement sont des aspects cruciaux. La validation de la chaîne de sélection (matériel et logiciel) est fondamentale pour exclure les candidats pouvant résulter des artefacts de procédure. On estime pouvoir détecter sans ambiguïté de l'ordre de 1 boson de Higgs sur 1000 effectivement produits (soit environ une détection de cette particule par jour).

1. Le paramètre appelé luminosité dans un collisionneur est le produit de la fréquence de croisement des paquets de particules par le nombre de particules qu'ils contiennent divisé par un facteur représentatif des caractéristiques du profil horizontal et vertical des faisceaux. La fréquence d'interaction des particules est le produit de la section efficace d'interaction par la luminosité.

Les quatre expériences prévues au LHC sont:

- ALICE (*A Large Ion Collider Experiment at the CERN LHC*) [ALI95] qui exploitera le potentiel du LHC dans le domaine des ions lourds (~900 signataires répartis dans 26 pays);
- ATLAS (*A Toroidal LHC ApparatuS*) [ATL94], expérience disposant de détecteurs à usage général exploitant les collisions proton-proton (~1800 collaborateurs dans le monde entier);
- CMS (*Compact Muon Solenoid*) [CMS94], seconde expérience basée sur des détecteurs à usage général destiné à l'observation des collisions proton-proton (~1700 signatures de physiciens et d'ingénieurs de 130 instituts répartis dans une trentaine de pays);
- LHCb [LHC95], expérience spécialisée dans les études de physique sur le méson B lors des collisions proton-proton (~600 membres de 50 instituts à ce jour);

Une vue des détecteurs de CMS tels qu'ils seront installés dans le hall souterrain de l'expérience est présentée Figure 1-2.

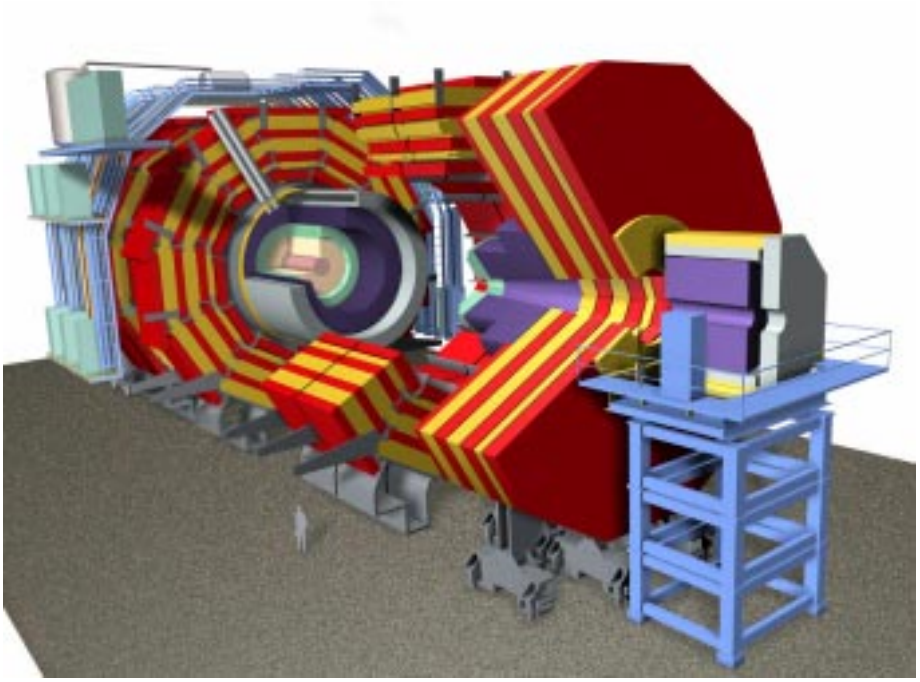


Figure 1-2: Détecteurs de l'expérience CMS.

Les détecteurs, dans leur ensemble, mesurent ~22 m de longueur et ~15 m de diamètre. Leur masse est de 14500 tonnes. Le besoin de pouvoir distinguer des particules de trajectoires séparées d'angles très faibles impose une segmentation des détecteurs en de nombreux canaux. Les limites de miniaturisation des capteurs et l'obtention d'une bonne résolution spatiale pour la mesure précise des trajectoires des particules conduisent à ces dimensions imposantes.

1.3. Objectifs des expériences ATLAS et CMS

Un des objectifs principaux de ces expériences est la recherche du mécanisme à l'origine de la brisure de symétrie spontanée dans le domaine électro-faible du Modèle Standard (*Standard Model* – SM) de la physique des particules. Une des manifestations possibles de ce mécanisme de brisure de symétrie pourrait être l'existence d'un boson de Higgs (H^0, Modèle Standard) ou d'une famille de 5 bosons de Higgs (H^+, H^-, h^0, H^0 et A^0) dans l'extension Supersymétrique Minimale du Modèle Standard (*Minimal Supersymmetric Standard Model* – MSSM).

Une autre voie de la physique est la recherche de signatures de nouvelles particules prédites par la supersymétrie (SUSY). D'autres théories de symétrie, au delà du Modèle Standard, prédisent de nouveaux bosons de jauge W' et Z' qui pourraient être observés au LHC. Les déviations par rapport à la chromo-dynamique quantique du spectre d'énergie des jets peuvent être utilisées pour tester des hypothèses sur le caractère composite des quarks.

Un autre objectif important est l'étude des systèmes contenant des quarks lourds, la fréquence de production du quark *b* (*beauty*) et *t* (*top*) étant relativement élevée (~10^7 paires $t\bar{t}$ par an). Un domaine assez large dans la physique du méson B peut être exploré: mesure précise de la violation CP dans le système B_d^0, détermination des angles du triangle unitaire de Cabibbo-Kobayashi-Maskawa, investigations du système $B\bar{B}$, étude des désintégrations rares du B. Des études détaillées sur le quark *t* sont également envisagées [ATL94].

1.4. Structure des détecteurs d'ATLAS

Les détecteurs des expériences ATLAS et CMS ont une structure relativement proche du point de vue du principe. La réalisation pratique et les technologies utilisées sont quant à elles différentes et résultent des choix d'optimisation faits par les concepteurs. La structure d'une expérience type est en "pelure d'oignon" comme le montre la Figure 1-3. L'axe des détecteurs est constitué par les tubes à ultra-vide véhiculant les paquets de particules du collisionneur.

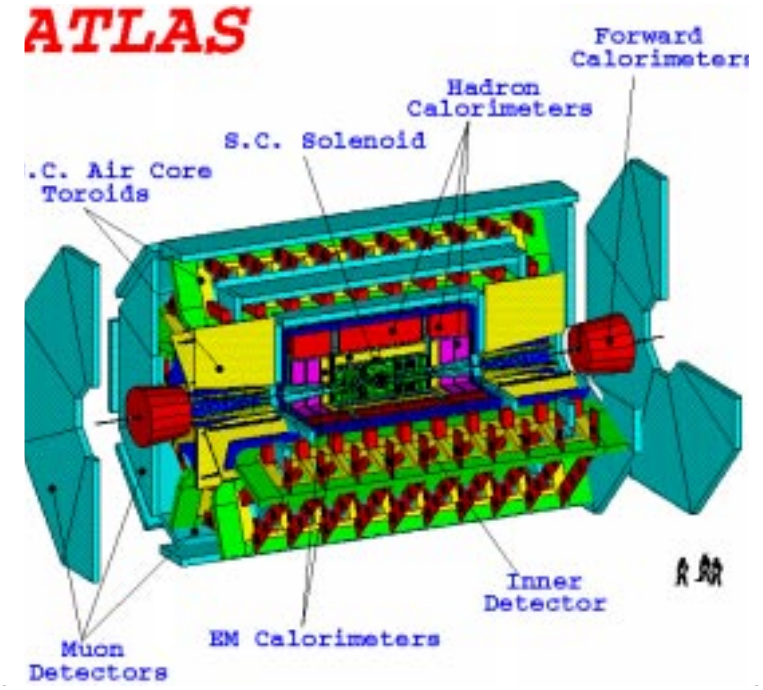


Figure 1-3: Structure des détecteurs d'ATLAS.

Le centre de l'expérience est situé au point de croisement des deux faisceaux de particules. La partie dans l'axe des faisceaux est le "tonneau" (*barrel*); ce dernier est fermé aux extrémités par deux "bouchons" (*end-caps*).

1.4.1. Détecteur interne

La partie centrale de l'expérience située à proximité du point d'interaction est appelée

détecteur interne (*inner detector*). Le sous-détecteur central est un détecteur à traces (*inner tracker*) dont la fonction est de repérer le passage des particules chargées et de permettre la reconstruction précise de leurs trajectoires. Le détecteur interne baigne dans un champ magnétique uniforme (4 T pour CMS et 2 T dans ATLAS) créé par un solénoïde englobant permettant de courber la trajectoire des particules chargées en vue de la détermination de leur impulsion. Le sens de la courbure d'une trajectoire permet d'identifier le signe de la particule, le rayon de courbure est d'autant plus grand que l'impulsion de la particule est importante. L'environnement dans lequel baigne le détecteur interne est particulièrement hostile avec une dose de radiations ionisantes de 300 kGray et un flux de 5.10^{14} neutrons par cm^2 intégré sur les dix ans de fonctionnement prévu. Cela impose le recours à des technologies particulières de composants électroniques durcis aux radiations.

Le détecteur interne d'ATLAS comporte trois sous-ensembles. La partie la plus centrale est formée de capteurs de type "pixels" de granularité très fine (chaque élément sensible mesure 50 μm x 300 μm) afin d'obtenir des mesures de grande précision au voisinage immédiat du point d'interaction. Dans la partie longitudinale du détecteur, les pixels sont arrangés en trois cylindres concentriques de 8, 20 et 26 cm de diamètre. A chacune des deux extrémités du détecteur, les pixels sont groupés en cinq disques placés dans l'axe du détecteur. La surface totale couverte par cet ensemble est de 2,3 m^2 et le nombre total de pixels est d'environ 130 millions. L'information délivrée par chaque pixel est binaire: touché ou non touché par une particule.

Le second sous-ensemble du détecteur central d'ATLAS est formé de 6,2 millions de bandes de silicium (*silicon strips*) de 6,4 cm de longueur par 80 μm de large agencées en 4 cylindres concentriques dans la partie longitudinale et 9 disques parallèles à chaque extrémité. La surface totale de silicium est d'environ 61 m^2.

Le troisième élément du détecteur interne est le *Transition Radiation Tracker* (TRT – détecteur de traces par radiations de transition) composé de "pailles" remplies de xénon comportant en leur centre un fil de collecte des charges générées par l'ionisation locale du gaz au passage d'une particule. La mesure du temps de dérive des charges dans une paille permet de déterminer la position d'un impact par rapport au centre de la paille avec une résolution de ~170 μm. Le diamètre des pailles est de 4 mm et leur longueur maximale est de 1,5 m. Le TRT comporte au total 420.000 voies.

1.4.2. Calorimètres

Autour du détecteur à traces se trouve l'ensemble des calorimètres permettant la mesure précise de l'énergie des particules et l'identification des électrons, des photons et des hadrons. Il est constitué d'un calorimètre électro-magnétique et d'un calorimètre hadronique. Le partie centrale du calorimètre électro-magnétique d'ATLAS est constituée d'absorbeurs en plomb et d'électrodes de Kapton en forme d'accordéon placés dans de l'argon liquide. Le calorimètre électro-magnétique comporte 3 ou 4 couches segmentées de manière différente en fonction de leur position par rapport au point d'interaction des faisceaux de protons. Le nombre total de canaux de ce détecteur est ~190.000. Le calorimètre hadronique est composé de deux blocs distincts. Dans la partie centrale du détecteur, des "tuiles scintillantes" et absorbeurs en inox sont utilisés. A chaque extrémité du détecteur, le calorimètre hadronique est du type argon liquide avec des absorbeurs en cuivre.

En plus des calorimètres *barrel* et *end-caps*, sont placés légèrement à l'extérieur du corps de l'expérience, les calorimètres "vers l'avant" (*forward*).

1.4.3. Spectromètre à muons

Ce dispositif vient en complément du détecteur interne pour la mesure de l'impulsion des

muons de haute énergie. Le principe du spectromètre à muons repose sur la mesure de la déflexion de la trajectoire des muons sous l'influence d'un champ magnétique créé par un toroïde supra-conducteur à air (26 m de longueur et 20 m de diamètre). De telles dimensions sont justifiées par le besoin de précision pour les mesures et par le fait que la déflexion des muons les plus énergétiques est faible, de l'ordre de quelques millimètres sur une trajectoire de plusieurs mètres. La connaissance précise de la position dans l'espace des différents capteurs (à ~100 μm près) est indispensable pour la qualité des mesures. Cela pose de nombreuses difficultés techniques compte tenu des dimensions des différents éléments.

Un premier type de chambre à muons fournissant des informations en un temps relativement bref est destiné au trigger alors que le second type de chambre effectue des mesures de précision. Les chambres pour le trigger comportent deux types de détecteurs: les *Resistive Plate Chambers* (RPCs) et les *Thin Gas Chambers* (TGCs). L'élément de base d'une RPC est formé de deux plans de bakélite résistive séparés par un isolant et rempli d'un mélange gazeux adapté ($C_2H_2F_4$ avec une faible proportion de SF_6). Les TGCs sont constituées de chambres à fils baignant dans un gaz formé d'un mélange de CO_2 et de n-pentane (n-C_5H_{12}) hautement inflammable ce qui entraîne la prise de mesures de précaution adéquates. Les RPCs comportent 355.000 canaux répartis en 596 chambres; les TGCs totalisent 440.000 canaux répartis en 192 chambres. La surface totale couverte par ces deux sous-ensembles est de 4500 m^2. Les chambres à muons destinées aux mesures précises sont également réparties en deux groupes: les *Cathode Strip Chambers* (CSCs, chambre à fils) et les MDTs (*Monitored Drift Tubes*, tubes en aluminium remplis d'un mélange de gaz ininflammable). Les CSCs (MDTs) sont réparties en 32 (1194) chambres pour un total de 67.000 (370.000) canaux. La surface couverte est de 27 m^2 et 5500 m^2 respectivement pour les CSCs et les MDTs.

1.5. Sélection d'événements en ligne dans ATLAS

Le nombre total de capteurs dans les détecteurs d'ATLAS est de l'ordre de 10^8. Toutes les 25 ns, à chaque croisement des faisceaux de protons, un capteur sur mille (valeur très approximative) produit une information significative. En conséquence, le flot total de données brutes produit est supérieur à 1 Téra-octet par seconde. Seule une infime partie des collisions engendrent des réactions qui pourraient correspondre aux phénomènes que l'on souhaite observer (une sur 10 à 100 millions).

Niveau de sélection	Fréquence d'entrée	Flux de données	Facteur de rejet	Délai de décision	Données utilisées	Réalisation
1	40 MHz	~10 Tbit/s	1000	2,5 μs fixe	partielles - précision réduite	logique câblée spécifique
2 ("Sélection")	100 kHz max.	10 -20 Gbit/s	~100	1 - 100 ms variable	régions d'intérêt	matériel du commerce
3 ("Filtre")	1 kHz	10 - 20 Gbit/s	10 - 100	0,1 - 1 s variable	événement complet	matériel du commerce
stockage	10 - 100 Hz	~1 Gbit/s	-	-	événement complet	disques, bandes magnétiques...

Table I-1: Sélection à niveaux multiples dans ATLAS.

L'énorme quantité d'information produite ainsi que le fait que seule une très faible partie des événements comporte un intérêt pour les objectifs de physique imposent d'effectuer une sélection en

ligne des événements ainsi qu'une compression des données pour réduire le débit d'information à un niveau jugé acceptable. En prenant en compte divers aspects techniques et économiques concernant le système de sélection en ligne face au système de stockage et d'analyse des données, on estime que seul un événement sur un million sera stocké sur support permanent en vue de l'analyse ultérieure. Le débit correspondant est de ~1 Gbit/s. La sélection des candidats à retenir se fait en plusieurs étapes qui utilisent des critères de sélection relativement simples au début, puis de plus en plus sophistiqués à mesure que la baisse du flux d'informations permet de consacrer les ressources suffisantes à l'exécution d'algorithmes plus complexes sur les événements restants. Le principe de sélection d'événements dans l'expérience ATLAS est basé sur trois niveaux dont les caractéristiques sont présentées Table I-1.

1.5.1. Trigger de niveau 1

Le premier niveau de sélection doit effectuer le tri des événements à 40 MHz. Compte tenu du faible intervalle de temps entre le croisement des faisceaux (25 ns), ce système applique des critères relativement simples, implantés en électronique câblée (en partie analogique) à proximité des détecteurs. Il permet de réduire à la volée le taux d'événements d'un facteur ~1000 en émettant une décision toutes les 25 ns avec un retard fixe de 2,5 μs par rapport au croisement considéré (système synchrone en pipe-line). Les données utilisées sont celles des calorimètres (en granularité et précision réduite) et du spectromètre à muons (chambres du trigger seulement et non celles des chambres de précision). Ces informations permettent l'identification des candidats électrons, photons, gerbes (de hadrons), muons et la mesure de l'énergie transverse totale et l'énergie manquante qui sont les primitives recherchées pour décider de poursuivre le processus de sélection des événements concernés.

Pour chaque événement accepté par le premier niveau, le résultat fourni comporte le jeu des conditions qui sont satisfaites pour justifier de retenir l'événement ainsi qu'une liste de pointeurs sur les régions des détecteurs dans lesquelles des particules d'énergie suffisante ont été repérées. Chaque pointeur définit à travers les détecteurs une pyramide fictive dont la pointe est dirigée vers le point de collision des faisceaux. Ces portions d'angle solide de détecteurs sont appelées "régions d'intérêt" (*Regions of Interest* – RoIs) et servent à guider la poursuite du processus de sélection. L'analyse porte en premier lieu sur ces régions d'intérêt avant de considérer la totalité des informations fournies par les détecteurs. Ce mode de guidage des triggers de niveau 2 et 3 au moyen de régions d'intérêt identifiées par le trigger de niveau 1 est un concept que l'on retrouve également dans l'expérience HERA-B [Dam97].

1.5.2. Triggers d'ordre supérieur – niveau 2 et niveau 3

Après le premier niveau de sélection, les triggers de niveaux supérieurs appliquent des critères de plus en plus fins mettant en œuvre des algorithmes de complexité croissante pour réduire le flux de données et extraire les événements contenant les réactions physiques que l'on souhaite observer. Dans le trigger de niveau 2 de l'expérience ATLAS, l'analyse commence par les régions d'intérêt qui ont été identifiées par le premier niveau de sélection. Le guidage au moyen de RoIs permet de réduire la quantité d'information à transmettre et à analyser car seules les données correspondant aux régions d'intérêt sont utilisées (~1-10 % des données de l'événement complet).

Le facteur de rejet attendu par le trigger de niveau 2 est de l'ordre de 100. Les études pour optimiser les algorithmes permettant d'atteindre ce facteur de rejet sont en cours et plusieurs méthodes sont étudiées. Celle décrite dans le *Technical Proposal* [ATL94] consiste à analyser l'ensemble des régions d'intérêt pour en extraire un certain nombre de primitives qui une fois

combinées permettent l'identification d'objets physiques (électrons, photons...). Le nombre, les caractéristiques et la position de ces objets forment une "signature" qui est comparée à la liste des signatures des phénomènes que l'on souhaite observer ("menus" du trigger) pour émettre la décision de détruire l'événement ou de le faire suivre au niveau de trigger suivant.

Alors que le trigger de niveau 2 n'utilise qu'une fraction des données de chaque événement, le modèle actuel pour le trigger de niveau 3 (aussi dénommé "filtre d'événements") repose sur l'utilisation d'algorithmes sophistiqués, semblables à ceux utilisés *off-line*, faisant appel à la totalité des données disponibles pour chaque événement. De même, chaque événement à conserver est stocké en rassemblant la totalité des données correspondantes. Une approche alternative consiste à envisager la possibilité de reconstruction et stockage d'événements partiels, selon les cas: données complètes pour les candidats boson de Higgs, fraction des données pour les événements Z^0 de calibration, etc. Cette approche pourrait permettre une économie de bande passante de réseau et de moyens de stockage. Cependant, le gain en capacité de stockage n'est que marginal si la majorité des enregistrements provient des événements que l'on souhaite observer le plus finement possible pour atteindre les objectifs de physique de l'expérience.

1.6. Besoins de réseaux de communication

Les données des événements acceptés par le premier niveau de sélection sont tout d'abord transmises par des liens haut débit (~1 Gbit/s) depuis les cartes de lecture des canaux des détecteurs vers ~1600 mémoires tampons permettant le stockage temporaire des données jusqu'à ce que la décision de conserver ou de détruire chaque événement ait été prise par le (ou les) processeur(s) en charge de se prononcer. Compte tenu de la complexité des algorithmes de sélection et de filtrage des événements, le nombre de processeurs nécessaires pour effectuer le traitement en temps réel du flux de données est estimé à 1000, voire davantage.

Pour le trigger de niveau 2, seules les données correspondant aux régions d'intérêt sont transmises. Pour le trigger de niveau 3, la totalité des données de chaque événement doit être rassemblée dans un des processeurs en charge du processus de filtrage. Les données des événements acceptés doivent être dirigées vers des périphériques de stockage et être accessibles pour les études différées. En plus de l'ensemble des données des événements, les messages nécessaires au dialogue entre les différents éléments, ceux relatifs à la configuration et à la surveillance du bon fonctionnement de l'ensemble du système doivent être transportés. Un puissant réseau de communication est nécessaire pour connecter les sources de données aux processeurs de traitement. La bande passante nécessaire totale est estimée à plusieurs dizaines de giga-bit par seconde. Les autres expériences au LHC ont des besoins du même ordre de grandeur: de cent à quelques milliers d'éléments à interconnecter avec un débit utile compris entre 10 et 500 Gbit/s.

1.7. Résumé

Dans ce chapitre, j'ai présenté le *Large Hadron Collider*. J'ai décrit brièvement les objectifs de physique des expériences ATLAS et CMS ainsi que la structure des détecteurs d'ATLAS. J'ai présenté le principe de sélection à niveaux multiples qui sera mis en œuvre dans cette expérience. J'ai justifié le besoin de réseaux de communication à haut débit pour la réalisation du T/DAQ des expériences modernes par le nombre d'éléments à inter-connecter et la quantité d'information à véhiculer.

CHAPITRE II. ARCHITECTURE INITIALE DES T/DAQ D'ATLAS ET DE CMS

2.1. Introduction

Dans ce chapitre, je présente l'architecture du système de *Trigger/Data AcQuisition* (T/DAQ) initialement proposée pour l'expérience ATLAS et les hypothèses qui ont conduit à cette proposition. Je présente également l'architecture du T/DAQ de l'expérience CMS et la justification qui en est faite.

2.2. T/DAQ de l'expérience ATLAS

Le système de tri en ligne des événements dans ATLAS comporte les trois niveaux décrits dans le chapitre précédent. Le premier niveau de sélection réalisé en électronique câblée à proximité des détecteurs ne rentre pas dans le cadre de cette étude et n'est pas décrit dans ce document. La structure des triggers d'ordre supérieur – de niveau 2 et 3 – est détaillée ci-après. Les choix définitifs concernant l'architecture du système n'étant à ce jour pas encore arrêtés, il existe plusieurs versions possibles et différentes options. Par rapport au schéma de base présenté dans le *Technical Proposal* d'ATLAS [ATL94], diverses suggestions et modifications ont été apportées. Au fur et à mesure des études, une meilleure compréhension du problème a permis de faire évoluer l'architecture, d'écarter certaines options, de préciser les choix technologiques, etc. Cette période d'étude se poursuivra jusqu'à la phase de construction du système qui devrait démarrer en 2001, selon le calendrier actuel. Afin de justifier la version de l'architecture qui sera établie et défendue dans cette thèse, il est utile de décrire le modèle du *Technical Proposal* qui constituait la référence dans ce domaine au moment où la thèse a débuté.

2.2.1. Démarche de conception

Le modèle du *Technical Proposal* d'ATLAS repose sur un certain nombre d'hypothèses relatives à la physique, à la technologie et sur des considérations d'ingénierie des systèmes.

2.2.1.1. Hypothèses relatives aux algorithmes de physique

Le facteur de rejet visé pour le trigger de niveau 2 est de ~100. Ce système doit opérer jusqu'à une fréquence d'événements venant du trigger de niveau 1 de 100 kHz[1] (réduite à 75 kHz dans la phase initiale d'exploitation de l'expérience).

Pour la physique relative aux particules de haute impulsion (physique des "hauts p_T"), il est considéré qu'un algorithme effectuant le traitement de la totalité des régions d'intérêt identifiées par

1. Les événements étant de nature aléatoire, il s'agit d'une fréquence moyenne. Des contraintes techniques sur les mémoires tampon et les circuits de numérisation des détecteurs imposent un temps mort de 4 croisements de faisceaux après chaque événement accepté par le trigger de niveau 1 (i.e. 125 ns au minimum entre événements; fréquence d'événements de 8 MHz en crête), ainsi qu'une limitation à 8 événements acceptés dans une fenêtre de temps glissante de 80 μs (durée maximale d'une rafale d'événements à 100 kHz).

le trigger de niveau 1 permet d'obtenir le taux de rejet désiré. Le traitement de chaque RoI est effectué au niveau de chaque détecteur et permet d'extraire des caractéristiques nommées "*features*", par exemple la valeur d'un dépôt d'énergie. Pour une RoI donnée, la combinaison des *features* en provenance de chaque détecteur permet d'identifier (ou de rejeter) le candidat particule (électron, hadron...). La combinaison de l'ensemble des particules identifiées dans les RoIs donne une vision globale de l'événement. Elle constitue une signature qui est comparée à la liste des phénomènes que l'on souhaite observer et permet d'élaborer la décision d'accepter ou de rejeter l'événement. Seules les données des RoIs sont analysées par l'algorithme de niveau 2. Elles sont identifiables dès que la liste des RoIs de cet événement est disponible.

Pour la physique correspondant aux particules de faible énergie transverse (physique du méson B envisagée lors du fonctionnement du LHC à basse luminosité), le guidage par RoI n'est pas possible et le traitement de l'événement doit être fait de manière séquentielle. Les données à utiliser ne peuvent pas être définies à priori comme dans le cas du guidage par RoI, mais sont déterminées au cours de l'exécution de l'algorithme de sélection, en fonction de résultats intermédiaires. Il est considéré que le taux d'événement devant suivre ce traitement séquentiel sans guidage par RoI est faible (~4% du total après la première étape de réduction opérée par le trigger de niveau 2) et ne devrait, de ce fait, pas présenter de difficulté majeure.

Le trigger de niveau 3 vise un facteur de rejet de ~10. Il doit opérer jusqu'à une fréquence d'événements de 1 kHz venant du trigger de niveau 2. Pour tous les types de physique considérés (physique du méson B ou des hauts p_T), il est considéré que l'algorithme à utiliser à ce niveau doit être relativement proche de celui employé pour l'analyse différée. Le potentiel de rejet d'algorithmes basés sur des données partielles étant largement exploité par le trigger de niveau 2, la reconstruction complète de l'événement est envisagée pour le trigger de niveau 3. De manière systématique, l'ensemble des données disponibles pour chaque événement ayant passé le trigger de niveau 2 est collecté et analysé par un programme sophistiqué.

2.2.1.2. Hypothèses sur la technologie

Il est considéré que la capacité des réseaux de communication reliant les cartes frontales des détecteurs aux processeurs de traitement est un élément à évaluer avec beaucoup d'attention. Malgré l'évolution prévue des matériels de réseau, il n'est pas envisagé que des réseaux offrant plusieurs centaines de giga-bit par seconde de bande passante seront disponibles voire possibles économiquement pour l'expérience ATLAS. Afin de réduire les besoins en bande passante pour le trigger de niveau 2, seules les données correspondant aux RoIs sont transmises; soit environ 5-10% du volume de données total. Comme seule une fraction des données est transférée des cartes frontales vers les processeurs en charge du trigger de niveau 2, il est nécessaire de conserver, dans des mémoires tampon situées en amont du réseau, le reste des données jusqu'à ce que la décision de conserver ou de rejeter l'événement soit prise. Le stockage temporaire de ces données est considéré comme critique, en particulier si ces mémoires sont placées à proximité des détecteurs (contraintes d'encombrement, de dissipation thermique et de tenue des composants aux radiations). Afin de diminuer la taille de ces mémoires tampon et de réduire le temps mort induit en cas de débordement, l'accent doit être porté sur la réduction du temps de décision du trigger de niveau 2 ("temps de latence"). Ce temps de latence est variable, vaut ~1 ms en moyenne et ~10 ms au maximum. La puissance de calcul nécessaire pour le trigger de niveau 2 ne semble pas une limitation majeure dans l'hypothèse de doublement de la puissance des processeurs tous les 18 mois (loi de Moore).

Pour le trigger de niveau 3, transporter l'ensemble des données est considéré comme possible et il n'est pas envisagé de faire un effort particulier afin de réduire les besoins en bande passante de

réseau. Dès qu'un événement a été transféré vers le trigger de niveau 3, les données correspondantes sont détruites dans les mémoires tampon liées aux cartes frontales. De ce fait, le temps de latence du trigger de niveau 3 est sans influence sur la capacité de ces mémoires tampon. La puissance de calcul nécessaire pour le trigger de niveau 3 est considérée comme problématique, mais serait disponible le moment venu grâce à l'évolution des processeurs et/ou l'utilisation de machines multi-processeurs.

2.2.1.3. Aspects système

Compte tenu des contraintes différentes pour les triggers de niveau 2 et 3 et afin de permettre une partition du système pour effectuer des développements, installations et tests de manière indépendante, il est envisagé de séparer physiquement ces deux sous systèmes.

2.2.2. Réalisation technique

Le schéma du système de sélection d'événements en ligne et d'acquisition de données décrit dans le *Technical Proposal* d'ATLAS est reproduit Figure 2-1. Les détecteurs ainsi que les cartes d'électronique frontale permettant la lecture, mise en forme et numérisation des signaux en provenance des détecteurs ne sont pas représentés sur la figure.

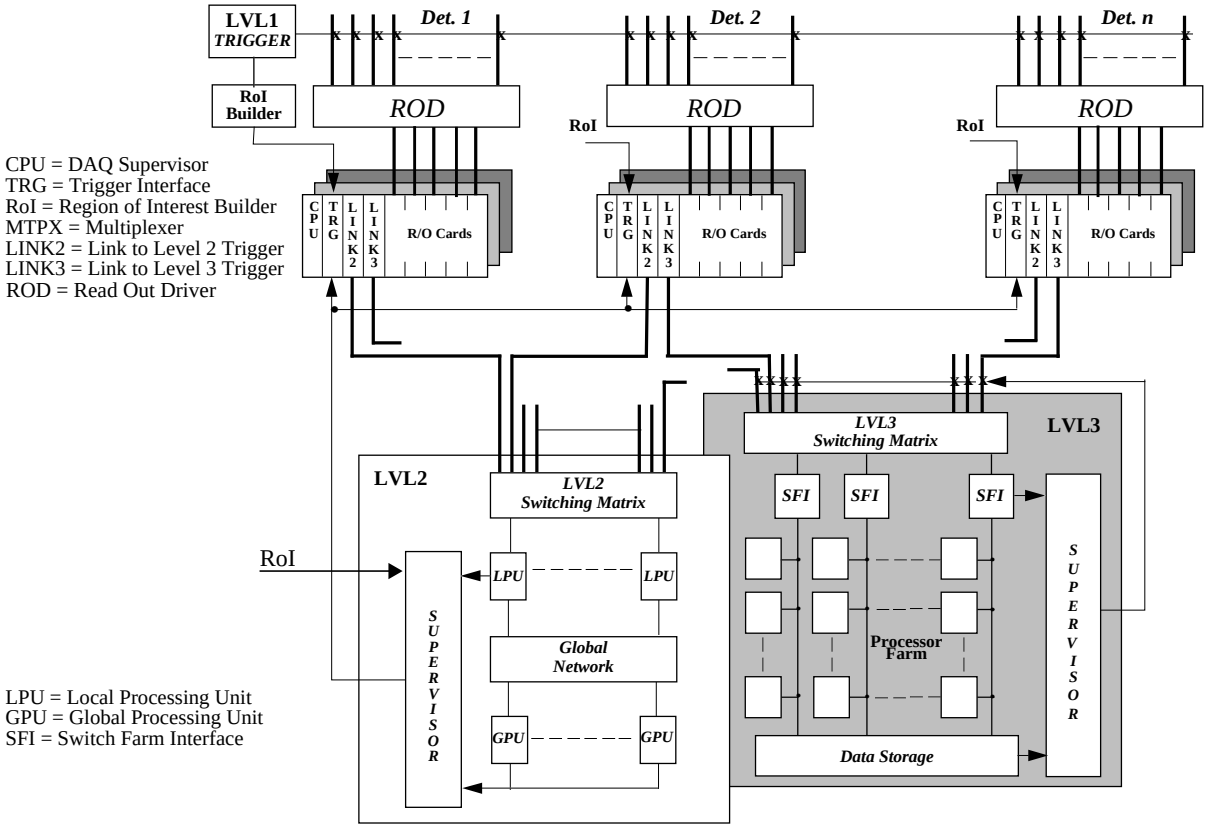


Figure 2-1: Architecture du T/DAQ d'ATLAS d'après le *Technical Proposal.*

Le *RoI Builder* ("élaborateur de régions d'intérêt") identifie, à l'aide des informations fournies par le trigger de niveau 1, les régions géographiques des détecteurs contenant des candidats particules. Ces régions d'intérêt guident le trigger de niveau 2. L'information est transmise par une liaison point-à-multi-points du *RoI Builder* vers un élément appelé "superviseur" et vers l'ensemble des châssis contenant les mémoires tampon liées aux détecteurs.

Pour tous les événements acceptés par le trigger de niveau 1, la totalité des données est transmise par chaque *Read-Out Driver* (ROD), au moyen d'une liaison point-à-point de bande passante adaptée (~1 Gbit/s) vers un élément servant de mémoire tampon appelé *Read-Out Buffer* (ROB, noté R/O *Card* sur ce schéma) dont le rôle est de conserver les données tant que cela est nécessaire. Les données peuvent être effacées des ROBs lorsque la décision du trigger de niveau 2 est de rejeter l'événement. Pour les événements acceptés par le trigger de niveau 2, les données des ROBs sont transmises au système en charge de la reconstruction complète de l'événement pour le trigger de niveau 3. Le nombre total de RODs est estimé à 1600; le nombre de ROBs est identique suivant un modèle simple de correspondance directe entre ces deux éléments.

Un certain nombre de ROBs (~16) sont regroupés pour former un des ~100 "châssis d'acquisition de données" ("*DAQ crate*" selon la terminologie employée). Les autres éléments placés dans chacun de ces châssis sont:

- un module relié au réseau de contrôle global du système et contenant un processeur en charge de la configuration, surveillance locale et espionnage du fonctionnement du *DAQ crate* (module CPU sur la figure),
- un module recevant du *RoI Builder* par un lien spécifique la liste des RoIs pour chaque événement accepté par le trigger de niveau 1, et recevant par un autre lien les décisions du trigger de niveau 2 (module TRG),
- un (ou plusieurs) lien(s) vers le réseau véhiculant les données qui seront utilisées pour les traitements du trigger de niveau 2 (noté LINK 2),
- un (ou plusieurs) lien(s) vers le réseau véhiculant les données qui seront utilisées par les processeurs du trigger de niveau 3 (noté LINK 3).

Suivant ce modèle, les triggers de niveau 2 et de niveau 3 sont physiquement séparés depuis les *DAQ crate* qui possèdent deux voies de sortie distinctes. Chaque trigger peut fonctionner de manière indépendante ce qui permet un développement et une mise au point facilités.

Le trigger de niveau 2 est également composé d'éléments modulaires indépendants. Chaque groupe de *DAQ crates* des différents détecteurs est relié par un réseau particulier à un groupe de processeurs nommés processeurs "locaux" (LPU). Chaque processeur local a un accès potentiel à toutes les données du détecteur auquel il est lié (et seulement à celles de ce détecteur). Un des buts de cette segmentation par détecteur vise à utiliser plusieurs commutateurs de capacité et nombre de ports modestes plutôt qu'un seul élément de forte capacité (celle solution est cependant envisageable). Chaque groupe de processeurs locaux est par ailleurs relié au moyen d'un autre réseau à un groupe de processeurs (processeurs "globaux", GPU) en charge de combiner, pour chaque événement, les résultats venant des processeurs locaux et de notifier au superviseur la décision de conserver ou de rejeter l'événement. Les notifications émises par les processeurs globaux sont transférées par un autre réseau vers le superviseur en charge de prendre la décision finale pour le trigger de niveau 2, de distribuer ces décisions aux ROBs et au trigger de niveau 3 (événements acceptés) en utilisant un réseau de diffusion. Le trigger de niveau 3 comporte un réseau unique reliant l'ensemble des *DAQ crates* à des fermes de processeurs de traitement. Ces derniers sont également connectés à des unités de stockage. Le nombre de processeurs nécessaire est estimé à près de mille unités pour l'ensemble des triggers de niveau 2 et 3 (puissance de calcul de $\sim 2.10^6$ MIPS).

2.2.3. Principe de fonctionnement

Lorsqu'un événement a été accepté par le trigger de niveau 1, *le RoI Builder* transmet la liste des RoIs au superviseur et à l'ensemble des *DAQ crates*. Le superviseur alloue un processeur local par RoI et par détecteur ainsi qu'un processeur global pour l'événement. Tandis qu'un contrôleur

dans chaque *DAQ crate* recherche dans les ROBs concernés les données des RoIs pour chaque événement, le superviseur transmet à tous les *DAQ crates* l'identificateur des processeurs locaux et du processeur global à utiliser. Chaque *DAQ crate* retourne au processeur local alloué, le bloc de données correspondant à la RoI et à l'événement considéré. Le processeur local collecte l'ensemble des fragments de la RoI dont on l'a chargé, puis effectue le traitement destiné à extraire les caractéristiques de cette région d'intérêt. Dans le but de minimaliser le temps de latence du trigger de niveau 2, le traitement des RoIs se fait avec un parallélisme à double niveau: d'une part au niveau de la multiplicité des RoIs et d'autre part au niveau de chaque détecteur pour une même RoI. En considérant le traitement de ~5 RoIs à travers 5 détecteurs (calorimètre, muon, TRT, silicium et pixels), le processus de sélection de chaque événement est distribué sur ~25 processeurs locaux et un processeur global. Lorsque le traitement d'un processeur local est terminé, le résultat est envoyé au processeur global qui combine les résultats de tous les processeurs locaux, suivant les détecteurs pour chaque RoI dans un premier temps, puis au niveau de l'ensemble des RoIs pour l'événement complet. Une décision est soumise au superviseur qui reste maître pour émettre la décision finale du trigger de niveau 2. Il diffuse alors à l'ensemble des *DAQ crates* l'ordre d'effacement des événements rejetés, et envoie au *Data Flow Manager* du trigger de niveau 3 (DFM) l'identifiant des événements acceptés. Le DFM alloue une destination pour chaque événement accepté et diffuse l'identifiant de cette destination à l'ensemble des *DAQ crates*. Lorsqu'un *DAQ crate* est informé d'une décision positive du trigger de niveau 2, la transmission de toutes les données relatives à cet événement vers la destination spécifiée a lieu. En fin de transmission, les données sont alors effacées de la mémoire des ROBs. Concernant le trigger de niveau 3, le recours à un parallélisme explicite de traitement n'est pas envisagé. Chaque machine effectue la totalité de l'algorithme considéré pour les événements qui lui ont été confiés. Dans le cas où les machines utilisées sont du type multi-processeurs, le fractionnement et la distribution des tâches ne sont pas imposés par un choix d'architecture comme pour le trigger de niveau 2. Le parallélisme exploitable à ce niveau est implicite et s'effectue de manière transparente.

L'objectif visant à réduire la taille de l'équipement réseau est satisfait par:

- une séparation physique en plusieurs sous-réseaux indépendants de taille modeste, pour chaque détecteur et pour les triggers de niveau 2 et de niveau 3,
- le transfert des données limité aux régions d'intérêt pour le trigger de niveau 2.

Pour réduire le temps de latence du trigger de niveau 2 on fait appel:

- à l'utilisation de réseaux à faible temps de transit,
- à la parallélisation explicite de l'algorithme du trigger de niveau 2,
- au déclenchement du transfert des données des ROBs vers les processeurs par un chemin dédié allant du superviseur vers les *DAQ crates* sans passer par les processeurs locaux ou globaux.

2.3. T/DAQ de l'expérience CMS

Le système de sélection en ligne des événements dans CMS comporte également trois niveaux [CMS94]. Comme pour ATLAS, le premier niveau de sélection est réalisé en logique câblée synchrone implantée à proximité des détecteurs. La présentation de ce système dépasse le cadre de cette étude. Seuls les triggers d'ordre supérieur sont décrits ci-après.

2.3.1. Démarche de conception

Le modèle proposé pour les triggers d'ordre élevé de l'expérience CMS est basé sur des

hypothèses relatives à la physique, à la technologie et à d'autres considérations dont certaines sont proches de celles d'ATLAS alors que d'autres s'en écartent. Au final, l'architecture proposée pour le T/DAQ de CMS est fort différente de celle décrite dans le *Technical Proposal* d'ATLAS.

2.3.1.1. Hypothèses relatives aux algorithmes de physique

Le taux de rejet visé pour le trigger niveau 2 est d'au moins 10. Il doit opérer jusqu'à une fréquence d'événements de 100 kHz venant du trigger de niveau 1. Contrairement à ATLAS dont le trigger de niveau 2 est guidé par les régions d'intérêt identifiées par le trigger de niveau 1, CMS n'a pas choisi ce mécanisme de guidage. Dans une première phase, seules les données des calorimètres et du spectromètre à muons sont analysées. Le traitement de la totalité de ces données permettrait de réduire le taux d'événements d'au moins un facteur 10. Après cette étape, le trigger de niveau 3 utilise l'ensemble des données disponibles pour effectuer la reconstruction complète des événements. Un facteur de rejet global de ~1000 est attendu pour l'ensemble des triggers de niveau 2 et de niveau 3. Chaque événement représente ~1 Mo de données; la production journalière de l'expérience est de l'ordre de 1 To (en format compressé).

2.3.1.2. Hypothèses sur la technologie

Les concepteurs du T/DAQ de CMS proposent d'exploiter au maximum les composants commerciaux, ordinateurs courants et matériel réseau de forte capacité. Les hypothèses sont les suivantes. Pour un prix de revient constant, la puissance de calcul des processeurs augmente d'un facteur 10 tous les 5 ans. De même, la capacité des circuits mémoire augmente d'un facteur 4 tous les 2 ans, toujours à coût constant. Les développements dans le domaine des télécommunications se poursuivront au même rythme qu'aujourd'hui pendant au moins les dix prochaines années.

Suivant ces hypothèses, il ne semble pas utile de chercher à minimaliser les quantités de données à transférer. Le concept visant à ne véhiculer que les données des régions d'intérêt qui est fondamental pour ATLAS n'apparaît pas comme une nécessité pour les concepteurs du modèle de CMS. Si la bande passante du réseau, disponible de manière économique, autorise le transfert systématique de la totalité des données des détecteurs utilisés pour le trigger de niveau 2, il est inutile de compliquer le système pour n'identifier et ne transporter que la fraction contenant des informations significatives.

La recherche de réduction du temps de latence du trigger de niveau 2 était fondamentale dans l'élaboration de l'architecture d'ATLAS mais semble inutile suivant l'hypothèse de la disponibilité de mémoires tampon de capacité suffisante à coût acceptable. Pour le trigger de niveau 3, le choix est similaire à celui d'ATLAS: la reconstruction complète des événements est envisagée.

2.3.1.3. Aspects système

Contrairement au concept proposé pour ATLAS qui vise à une séparation physique des triggers de niveau 2 et de niveau 3, le modèle défendu pour CMS repose sur l'approche inverse: le système est uniforme et la distinction entre les deux niveaux de trigger n'est que conceptuelle (le trigger de niveau 2 est qualifié de "virtuel"). Le but recherché est de réduire la disparité des éléments afin de faciliter le développement et la maintenance. Un effort particulier a été fait pour avoir une architecture simple, flexible et facile à mettre à niveau. Un autre aspect consiste à prévoir d'utiliser les ressources de calcul des triggers de niveau 2 et de niveau 3 pour l'analyse différée des événements lors des périodes d'arrêt de l'expérience.

2.3.2. Réalisation technique

Le schéma du système de sélection d'événements et d'acquisition de données décrit dans le *Technical Proposal* de CMS est reproduit Figure 2-2. Pour tous les événements acceptés par le trigger de niveau 1, les données des cartes frontales du détecteur sont transmises à ~1000 unités de mémorisation temporaire nommées *Dual Port Memories* (DPMs) ayant une fonctionnalité proche des ROBs, suivant la terminologie employée dans la collaboration ATLAS. Ces mémoires tampon sont reliées par un réseau unique à un ensemble de ~1000 unités de traitement de données (machines mono- ou multi-processeurs). La bande passante utile du réseau est de 500-1000 Gbit/s. Un système de supervision réalise l'interface avec le trigger de niveau 1 et se charge de la distribution des tâches en commandant l'envoi des données depuis les DPMs vers les unités de traitement de données en utilisant un réseau dédié. L'architecture est simple et ne comporte que peu d'éléments différents. Les DPMs comportent une mémoire de taille suffisante (au moins 128 Mo) de sorte que pour un flux d'entrée de 100 Mo/s, un historique de plus d'une seconde est conservé. Cela réduit de beaucoup les contraintes sur le temps de réponse du trigger de niveau 2.

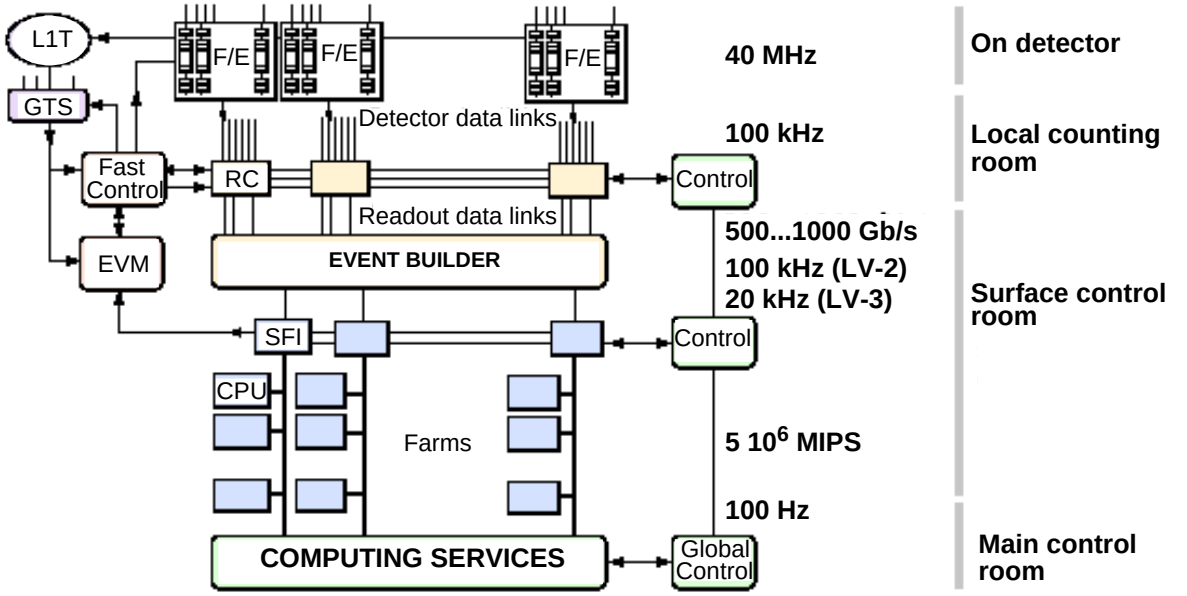


Figure 2-2: Structure du T/DAQ de CMS.

2.3.3. Principe de fonctionnement

Pour chaque événement accepté par le trigger de niveau 1, l'*Event Manager* (EVM) décide d'allouer cet événement à une des unités de traitement (un CPU attaché à un *Switch to Farm Interface* – SFI). Le transfert systématique vers les SFIs de la totalité des données (~1 Mo) à la fréquence de sortie du trigger de niveau 1 (~100 kHz) correspondrait à un débit d'information de ~100 Go/s. Compte tenu des difficultés techniques et économiques d'une telle approche, une solution basée sur un transfert de données en deux étapes successives est envisagé. Il s'agit du trigger de niveau 2 "virtuel" qui opère comme suit. Dans un premier temps, l'*Event Manager* ne diffuse l'identificateur de l'unité de traitement allouée qu'à l'ensemble des DPMs correspondant aux calorimètres et au spectromètre à muons. La totalité des données de ces deux détecteurs est transmise à l'unité de traitement considérée. Lorsque celle-ci dispose des informations nécessaires, l'algorithme de sélection (trigger de niveau 2 virtuel) est exécuté et la décision de poursuivre l'analyse ou de rejeter l'événement est communiquée à l'*Event Manager*. Cette décision est diffusée

à l'ensemble des DPMs. Si l'événement est à rejeter, il est effacé de la mémoire des DPMs; dans le cas contraire, les données des détecteurs qui n'ont pas encore été transmises sont envoyées. Lorsque l'unité de traitement a reçu ce complément d'information, elle dispose de la totalité des données pour effectuer la reconstruction complète de l'événement. Lorsque cette opération est terminée, l'événement est soit effacé soit sauvegardé. L'unité de traitement signale alors sa disponibilité à l'*Event Manager*.

Cette approche est, dans le principe, plus simple que celle d'ATLAS et a pour résultat une architecture très homogène. Le concept de traitement de régions d'intérêt n'étant pas envisagé pour CMS, les besoins en bande passante de réseau sont plus importants que pour ATLAS. Les deux expériences ont approximativement le même nombre de canaux et ont un trigger de niveau 1 pouvant fournir des événements jusqu'à 100 kHz. La bande passante totale utile pour le transfert des données dans ATLAS est estimée à quelques dizaines de giga-bit par seconde, alors que pour CMS, un réseau offrant au minimum 500 Gbit/s de bande passante est requis.

Le temps de latence du trigger de niveau 2 n'est pas considéré comme critique dans le cas de CMS et les concepts de parallélisation et de partage explicite des algorithmes correspondants, prônés par les concepteurs du T/DAQ d'ATLAS, sont ici totalement absents. Le même processeur est en charge de la totalité du processus de sélection des événements qui lui ont été assignés.

2.4. Résumé

Dans ce chapitre, j'ai décrit l'architecture des systèmes de sélection et de reconstruction d'événements des expériences ATLAS et CMS telle que chacune a été introduite et justifiée dans le rapport de proposition technique des collaborations. Ces deux architectures sont fort différentes: d'une part pour ATLAS un système physiquement segmenté effectuant le traitement de chaque événement en parallèle en répartissant la tâche sur plus d'une vingtaine de processeurs, et d'autre part pour CMS, un système homogène sans répartition du traitement d'un événement sur plusieurs unités de calcul. Les évolutions proposées et les modifications effectivement apportées à l'architecture du T/DAQ d'ATLAS seront présentées ultérieurement dans ce document, après une analyse critique des modèles de référence précédemment décrits.

CHAPITRE III. COMMUNICATIONS DANS LE T/DAQ D'ATLAS

3.1. Introduction

Dans ce chapitre, je décris les fonctionnalités du réseau de communication du système de sélection et de reconstruction d'événements d'ATLAS. Je présente diverses méthodes de contrôle du flot de données pour la reconstruction partielle ou totale d'événements. Je discute les avantages et les inconvénients de ces différents protocoles. Je définis une série de critères de qualité et de points critiques afin d'évaluer les performances des différentes configurations de réseau qui seront étudiées et de comparer les différentes solutions techniques proposées.

3.2. Fonctionnalités du réseau de communication

Le schéma abstrait sous forme d'un diagramme de classes [Mul97] du Trigger/DAQ d'ATLAS est présenté Figure 3-1.

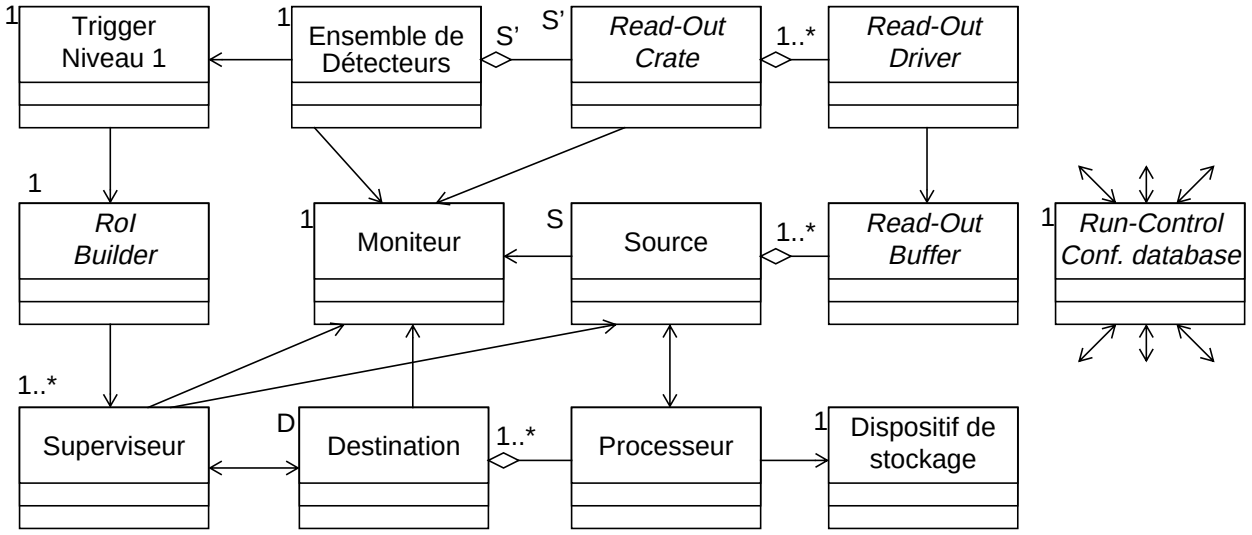


Figure 3-1: Diagramme de classes du système de Trigger/DAQ d'ATLAS.

Les principales fonctionnalités du point de vue de la communication dans un système de sélection et de reconstruction d'événements sont décrites ci-après. Cette liste ne préjuge pas des implémentations possibles en particulier concernant la possibilité de partition en plusieurs sous-réseaux physiquement distincts.

3.2.1. Fournir une interface avec le niveau de sélection précédent

Dans un système de sélection d'événements à niveaux multiples, l'entrée d'un niveau donné est le lot des événements acceptés par le niveau précédent. Dans ATLAS, l'interface entre le trigger de niveau 1 et celui de niveau 2 est le *RoI Builder*. Le système chargé de distribuer aux processeurs

de traitement les informations relatives aux événements acceptés par le trigger de niveau 1 est le(s) superviseur(s). Cet élément est connecté d'une part au *RoI Builder* (communication point-à-point) et d'autre part à tous les éléments en charge de la sélection des événements (éléments nommés "destinations" sur le diagramme). La distribution de messages se fait d'un point unique vers plusieurs points. Les messages sont de taille relativement faible: l'information à transmettre est le numéro d'identification de l'événement ainsi que la liste des RoIs (bloc de 50-200 octets). La fréquence d'envoi des messages est celle du trigger de niveau 1 (100 kHz) en faisant l'hypothèse qu'un seul message d'allocation est envoyé pour chaque événement. Le nombre de superviseurs est 1-10, celui des destinations est 100-1000 suivant les hypothèses actuelles.

La connexion requise entre le *RoI Builder* et les superviseurs est du type point-à-point, uni-directionnelle allant du *RoI Builder* vers chaque superviseur (avec éventuellement un chemin de retour pour le contrôle de flux). Pour la fonction de distribution des événements, la connexion requise entre chaque superviseur et les processeurs de traitement est du type point-à-point, uni-directionnelle depuis les superviseurs vers les processeurs destinations.

3.2.2. Transporter les données des détecteurs vers les processeurs

Il s'agit d'une des fonctions principales du réseau de communication. Les données de chaque événement à traiter doivent être regroupées dans le (ou les) processeur(s) affectés. Le transfert des données brutes des détecteurs se fait via les RODs, qui transmettent aux ROBs des données numérisées, calibrées et encapsulées dans des paquets structurés. Les liaisons sont du type point-à-point uni-directionnelles (~1600 paires ROD/ROB à connecter). Les ROBs sont groupés en sous-ensembles nommés "sources" (~200-1000 sources au total). Les *DAQ Crates* sont l'implémentation de ces éléments fonctionnels. Le nombre de sources participant à un événement donné varie selon les traitements effectués: il ne s'agit que d'une fraction de la totalité des sources dans le cas de l'analyse des RoIs, de l'ensemble des sources d'un détecteur complet (sources du calorimètre pour le calcul de l'énergie manquante) ou de la totalité des sources pour la reconstruction complète de l'événement. La taille des blocs de données en provenance de chaque source est variable de quelques octets à quelques kilo-octets. Le flux de données est estimé à plusieurs dizaines de gigabit par seconde et constitue la majorité du trafic à travers le réseau.

Les connexions entre les éléments sources et destinations sont du type point-à-point, uni-directionnelles allant des sources vers les destinations processeurs de traitement. Si l'envoi des données doit être fiable et que le réseau ne garantisse pas l'intégrité de la communication de bout-en-bout, un chemin de retour pour véhiculer les accusés de réception, ordre de retransmission après erreur et contrôle de flux doit être ajouté.

3.2.3. Transporter les messages du protocole d'échange de données

Dans le cas où seule une fraction des sources est concernée par le traitement d'un événement, ces sources doivent être informées de leur participation. Que l'ensemble ou une fraction des sources soit concernée, ces dernières doivent dans tous les cas connaître le destinataire des fragments d'événements à envoyer. Diverses méthodes seront exposées dans ce chapitre pour préciser la méthode de distribution des informations nécessaires. Selon les modèles, les messages sont émis par les processeurs destination ou le(s) superviseur(s) en direction des sources.

Les connexions requises sont du type point-à-point entre chaque processeur destination (ou superviseur) et chaque source, et point-à-multi-points pour les groupes de sources particuliers (groupe de toutes les sources de chaque détecteur et de l'expérience complète).

3.2.4. Véhiculer les résultats

3.2.4.1. Décisions de trigger

Les sources doivent être informées de la décision de conserver ou de rejeter chaque événement afin de libérer les ressources allouées au stockage temporaire des données au moment opportun: dès que la décision de rejet est connue pour les événement rejetés ou bien lorsque les données ont été déplacées à l'extérieur des sources pour les événements à conserver. L'information à transmettre provient du (des) superviseur(s) ou des processeurs de traitement et est destinée à l'ensemble des sources. Des connexions point-à-multi-points, uni-directionnelles sont requises.

3.2.4.2. Résultats des traitements

A l'issue du trigger de niveau 2, l'opération envisagée dans le *Technical Proposal* est de transférer les résultats des processeurs ayant effectué la sélection des événement vers un *Read-Out-Buffer* d'un type particulier faisant l'interface avec le trigger de niveau 3. Les processeurs du trigger de niveau 3 collectent la totalité des données de chaque événement en incluant celles de ce ROB particulier. Cette solution permet de conserver des sous-systèmes et réseaux indépendants pour le trigger de niveau 2 et celui de niveau 3 mais crée un élément critique supplémentaire (un défaut de cet élément unique bloque l'ensemble du système). Elle requiert des connexions du type point-à-point entre les processeurs du trigger de niveau 2 et le ROB d'interface, et d'autres connexions du même type entre ce ROB et les processeurs du trigger de niveau 3. Une autre solution est de conserver les résultats dans chaque processeur du trigger de niveau 2 et de les transférer directement au processeur du trigger de niveau 3 concerné sans passer par un élément intermédiaire. Cette option requiert des connexions du type point-à-point entre les processeurs effectuant la sélection des événements et ceux effectuant le processus de filtrage (trigger de niveau 3). Une dernière possibilité est de faire suivre l'algorithme de sélection par le processus de filtrage dans la même destination en collectant les données permettant de compléter l'événement. Cette solution ne requiert pas de communication par réseau pour le transfert des résultats du trigger de niveau 2 vers le filtre d'événements. Dans tous les cas, les données à conserver pour les analyses de physique différées doivent être véhiculées jusqu'aux périphériques de stockage.

3.2.5. Permettre la surveillance du fonctionnement du système

Un aspect important pour un système de T/DAQ est de pouvoir vérifier son bon fonctionnement en cours d'opération, identifier les éléments défectueux et transmettre aux opérateurs humains ces informations significatives. Cette surveillance continue se fait à plusieurs niveaux dans le système: vérifications du bon fonctionnement du réseau, de la chaîne de transport des messages, des processeurs, sources de données, des résultats des algorithmes de sélection, des processus de physique identifiés... Au niveau de la vérification du matériel et logiciel du réseau et de chaque élément connecté, des messages de test doivent être périodiquement émis. Un mécanisme de collecte des statistiques permettant d'apprécier le fonctionnement de chaque élément est nécessaire. Les connexions requises sont du type point-à-point entre tous les sous-ensembles du système et le point central servant d'interface avec l'utilisateur ("moniteur"). Le débit d'information pour ces opérations est bien moindre que celui du flot de données principal.

3.2.6. Permettre la configuration du système

Cette activité comprend deux volets: la configuration du système au moment du démarrage et la mise à jour régulière des paramètres qui évoluent au cours de l'expérience (constantes de

calibration par exemple). L'ensemble des paramètres est stocké par une base de données à laquelle doivent avoir accès de manière directe ou indirecte tous les éléments à configurer ainsi que ceux qui font appel à des informations extérieures en cours de fonctionnement. Pour cette tâche également, l'objectif est davantage de fournir un chemin de transfert d'information plutôt que d'optimiser les performances du point de vue du débit.

La Table III-1 présente en résumé les besoins de communication entre les différents éléments du système. PP désigne une connexion du type point-à-point, PMP désigne une connexion du type point-à-multi-points; un indice subjectif suit pour mentionner si la liaison à réaliser comporte un impératif de performance (bande-passante, débit de messages) ou bien s'il s'agit simplement d'un chemin non critique à cet égard.

Depuis... \ vers:	Superviseur	Source	Destination	Moniteur	Périphériques de stockage	Base de données de configuration
Superviseur	PP[a]	non requise	PP performante	PP lente	PP lente	PP lente
Source	PP PMP performante	non requise	PP PMP performante	PP PMP lente	PP selon modèle[b]	PP lente
Destination	PP performante	PP performante	PP[c] performante	PP PMP lente	optionnelle[d]	PP lente
Moniteur	PP lente	PP lente	PP lente	PP[e]	optionnelle	PP lente
Périphériques de stockage	PP lente	optionnelle	PP performante	PP lente	non significative	PP lente
Base de données de configuration	PP lente	PP lente	PP lente	optionnelle	PP lente	non significative

Table III-1: Communications entre les éléments du modèle de T/DAQ.

a. si le système comporte plusieurs superviseurs.
b. dans le cas ou les périphériques de stockage sont attachés au même réseau que les sources, celles-ci peuvent transmettre les données des événements à conserver sans transiter en totalité par les processeurs de traitement en ligne.
c. dans le cas où la sélection et le filtrage d'un événement donné sont effectués par deux processeurs physiquement distincts.
d. pour transmettre les accusés de réception par exemple si cela est nécessaire.
e. si le système comporte plusieurs moniteurs.

Les connexions entre les RODs et ROBs ainsi que celles entre le trigger de niveau 1 et le *RoI Builder* ne sont pas reportées dans la table; il s'agit de liaisons point-à-point ne constituant pas véritablement un réseau. Les liaisons entre le *RoI Builder* et le(s) superviseur(s) sont également du type point-à-point et constituent un petit réseau particulier.

3.3. Mécanismes de contrôle du flot de données

Après avoir présenté les fonctionnalités requises du point de vue de la communication entre les différents éléments du système, je présente différents mécanismes de contrôle du flot de données en provenance des sources liées aux détecteurs vers les processeurs de traitement.

3.3.1. Modèle “données poussées” – *push dataflow*

Dans ce mode de fonctionnement, les données sont envoyées par les sources dès que possible vers un processeur de traitement. Les sources impliquées dans l’événement considéré ainsi que le choix des données à envoyer est effectué à priori. L’ensemble des sources est concerné et la totalité des données est envoyée dans le cas de la reconstruction complète d’événements. Une autre possibilité est de transférer de manière systématique l’ensemble des données d’un ou plusieurs détecteurs (par exemple celles des calorimètres et du spectromètre à muons pour le trigger de niveau 2 virtuel de l’expérience CMS). Avec le guidage par RoI utilisé dans ATLAS, les sous-ensembles de sources concernées sont déterminés à chaque événement.

Quelle que soit la méthode de sélection des sources, il s’agit d’une architecture “flot de données”: la totalité des données dont pourra avoir besoin l’unité de traitement lui est envoyée de manière systématique sans intervention de sa part. L’allocation de l’unité de traitement peut être implicite: chaque unité de traitement peut être affectée par construction, de manière statique, à une zone géographique d’un détecteur. Une autre possibilité est d’allouer les unités de traitement de manière récurrente en se basant par exemple sur les numéros d’événements. L’avantage d’une affection statique est (en plus d’un contrôle trivial) de pouvoir réduire la taille du réseau liant les unités de traitement aux sources dans la mesure où chacune d’entre elles n’a besoin de recevoir des informations que de la fraction des sources correspondant à la zone géographique du détecteur qui lui est affectée. Cette solution se prête relativement bien à une réalisation en logique câblée compte tenu de son mode de contrôle peu complexe et des simplifications possibles dans la structure du réseau liant les sources de données aux unités de traitement. Un des inconvénients de l’allocation géographique statique de ressources est la répartition figée de ces dernières. Certaines zones d’un détecteur peuvent être le siège de plus d’activité que d’autres, et l’équilibrage dynamique de la charge des unités de traitement est problématique. Une affectation cyclique des unités de traitement permet de tirer un meilleur parti des ressources disponibles mais supprime l’avantage potentiel de localité des communications entre sources et processeurs par régions géographiques de taille limitée. Dans ce mode de fonctionnement, toute unité de traitement doit pouvoir recevoir des informations de n’importe quelle source de données. Bien que les moyens matériels à mettre en œuvre soient plus complexes que dans le cas de l’allocation statique, cette solution offre davantage de flexibilité, notamment pour ajuster le nombre d’unités de traitement, et présente une meilleure tolérance aux pannes dans la mesure où l’arrêt d’une unité de traitement ne perturbe que temporairement le fonctionnement du système et ne crée pas de zone aveugle.

Dans le mode de fonctionnement “données poussées”, il n’y a pas d’action (ou de rétro-action) des unités de traitement sur les sources de données et un réseau uni-directionnel est suffisant. Compte tenu du fait que les sources de données émettent vers les unités de traitement dès que cela est possible, le délai de transmission des informations est minimisé. Ce mode permet de contribuer à l’optimisation du temps de latence des décisions du système de sélection d’événements.

Cette méthode de contrôle du flux de données est adoptée par différentes expériences, par exemple NA48 [Anv97] et D0 Run II [Bla97a]. Elle constitue la proposition initiale pour les expériences ATLAS et CMS.

3.3.2. Modèle "données tirées" – *pull dataflow*

Dans ce mode de fonctionnement, les données des sources sont transmises à la demande des unités de traitement. Dans un premier temps, une unité de traitement reçoit, par un élément central de distribution (i.e. superviseur), une indication d'affectation à un événement donné. Cette unité de traitement identifie les sources dont elle souhaite obtenir les données et leur transmet sa requête. Les sources concernées répondent à cette requête en retournant les données demandées. Contrairement au modèle "données poussées", ce mode de contrôle offre une grande flexibilité dans la détermination des sources concernées et des données à collecter. Le choix est laissé en totalité à l'unité de traitement qui peut décider à la volée de collecter les donnés nécessaires au moment opportun. Il est inutile de transférer, suivant un critère fixé à priori, l'ensemble des données susceptibles d'être utilisées par l'algorithme de traitement. Il est possible d'effectuer le traitement des événements par étapes successives. A chaque étape, seules les données nécessaires sont collectées par l'unité de traitement qui peut décider après analyse de cette fraction des données de rejeter l'événement ou de poursuivre la sélection en demandant un complément d'information à d'autres sources. Les résultats intermédiaires de l'algorithme de traitement peuvent être utilisés dans la détermination des sources qui participeront à l'étape suivante. Cette opération n'est pas possible avec le mode de fonctionnement par "données poussées" car le choix des sources est effectué à priori sans intervention des unités de traitement. De même, la sélection d'un événement par une séquence de transferts de données suivis de phases de traitement n'est pas possible à réaliser avec le modèle "données poussées" sans introduire un mécanisme de rétro-action des unités de traitement sur les sources de données (ce qui rendrait l'architecture nettement plus complexe).

Les objectifs principaux du modèle "données tirées" sont de minimaliser les transferts de données entre les sources et les unités de traitement, d'autoriser une grande flexibilité dans le choix des sources de données pour chaque événement, et de faciliter le traitement séquentiel des événements. Cela permet de réduire les besoins en bande passante pour le réseau de communication liant les sources aux processeurs de traitement et permet également de minimaliser la puissance de calcul nécessaire par l'utilisation d'algorithmes adaptés évitant les traitements systématiques pouvant se révéler inutiles dans certains cas. Ces optimisations se font au détriment du délai de transmission des événements vers les processeurs: les sources doivent attendre d'être sollicitées avant d'émettre leurs données. La capacité de mémorisation requise au niveau des sources est plus importante que dans le cas du modèle "données poussées". Pour un algorithme de traitement ne comportant qu'une seule phase de transfert puis de traitement de données, le temps de décision du système de sélection est également plus important avec le modèle "données tirées" qu'avec l'autre modèle. Le réseau reliant les sources de données aux unités de traitement doit être bi-directionnel et doit permettre la communication entre toutes les sources et tous les processeurs. Le contrôle du système est également plus complexe avec le modèle "données tirées" et ce dernier se prête plus difficilement à des réalisations en logique câblée.

Le modèle "données tirées" est mis en œuvre dans les expériences HERA-B [Dam97] et EUROBALL [Bar96], mais la possibilité de transfert des données d'un événement en de multiples étapes n'est pas véritablement exploitée.

3.3.3. Comparaison des modèles "données tirées" et "données poussées"

Les caractéristiques principales de ces deux modes de fonctionnement sont résumées dans la Table III-2. Dans le modèle "données poussées", les données sont transmises par les sources "à l'aveugle" vers les unités de traitement. La concentration simultanée des données de nombreuses sources vers l'unité de traitement concernée peut créer des points de congestion à l'intérieur du

réseau de communication et également dans l'unité de traitement elle-même qui n'est pas nécessairement toujours prête à absorber ce flot de données. A l'inverse, dans le modèle "données tirées" une unité de traitement peut solliciter les données des sources en séquence et en fonction de ses capacités de traitement (régulation et contrôle de flux par le consommateur). L'utilisation d'un réseau bi-directionnel permet de faciliter le recouvrement en cas d'erreur de transmission par la mise en place d'un protocole d'accusés de réception et de ré-émission des données perdues. Ce type de transfert sécurisé n'est cependant pas toujours nécessaire: on peut tolérer la perte de quelques événements à cause d'erreurs de transmission (si cela ne biaise pas les mesures).

paramètre	"données poussées"	"données tirées"
objectif principal	minimisation du temps de latence des décisions du trigger	réduction de la bande passante du réseau pour les transferts de données
contrôle	simple	complexe
réseau	uni-directionnel, possibilité de connectivité réduite	bi-directionnel, connectivité complète sources/destinations
choix des sources	décidé à priori	choix libre
transferts de données séquentiels	non	possible
flexibilité	moindre	supérieure
recouvrement d'erreurs	difficile	possible
réalisation	logique câblée ou processeurs et réseaux du commerce	processeurs et réseau du commerce

Table III-2: Caractéristiques des modèles "données poussées" et données tirées".

Le mode de fonctionnement "données poussées" est bien adapté aux applications pour lesquelles le temps de décision du trigger est critique et l'algorithme de sélection fait appel toujours au même type de données. Avec le modèle "données tirées" davantage de flexibilité est permise pour l'algorithme de sélection.

3.4. Critères d'évaluation

Après avoir défini de manière qualitative les besoins en terme de communication dans un système de T/DAQ, je propose un certain nombre de critères d'évaluation ainsi qu'une liste de paramètres quantifiables pour permettre la comparaison de différentes architectures.

3.4.1. Critères généraux

3.4.1.1. Complexité de la solution

Cet aspect est apprécié selon le nombre d'éléments, de réseaux différents pour réaliser les fonctionnalités nécessaires. Compte tenu de la diversité des types de trafic à véhiculer, une des solutions proposées dans ATLAS consiste à utiliser différents réseaux conçus et optimisés pour chaque type de communication: réseau pour véhiculer les informations de contrôle (requêtes aux sources, décisions de trigger...), second réseau pour transporter les données des sources vers les processeurs en charge de la sélection des événements, troisième réseau pour transmettre les données

en vue de la reconstruction des événements, etc. Une réalisation basée sur plusieurs réseaux distincts, éventuellement de technologies différentes, présente à la fois des avantages et des inconvénients. Un des avantages est de permettre de découper plus facilement un système complexe en plusieurs systèmes relativement découplés. L'inconvénient principal est de multiplier le nombre d'éléments (cartes réseaux, commutateurs) ce qui rend le système moins homogène et plus difficile à maintenir, configurer, fiabiliser... Une solution basée sur un réseau unique supportant la totalité des fonctionnalités est en un sens une solution optimale (en quantité de matériel par exemple).

3.4.1.2. Flexibilité de la solution et possibilité d'évolution

Le système est flexible s'il est possible de le modifier sans difficultés: affecter un élément prévu à l'origine pour une certaine tâche à une autre, modifier les algorithmes de traitement de données, augmenter ou réduire les ressources affectées à une partie du système pour les transférer vers une autre, etc... La possibilité d'évolution du système est un aspect particulièrement important pour les expériences au LHC. Le collisionneur fonctionnera initialement avec une luminosité de 10^{33} $cm^{-2}s^{-1}$ avant d'atteindre par la suite la luminosité nominale de 10^{34} $cm^{-2}s^{-1}$. Il est souhaitable que cette montée en puissance (multiplication du taux de collisions d'un facteur 10) n'entraîne pas de modification majeure des systèmes de sélection et d'acquisition de données des expériences. L'ajout de nouveaux éléments (processeurs plus rapides, unités de stockage...) doit être possible dans un système de T/DAQ pour exploiter au mieux l'évolution du potentiel de l'expérience et bénéficier des progrès technologiques constants. Pour l'expérience ATLAS, le système de sélection d'événements devra, dans sa version initiale, pouvoir fonctionner à 75 kHz et sera amélioré pour atteindre 100 kHz (limite fixée pour l'électronique frontale des détecteurs et le trigger de niveau 1). Les réseaux de communications et le nombre d'unités de traitement devront être adaptés en conséquence. Pour satisfaire ce critère de flexibilité et faciliter l'évolution du système, l'architecture doit être relativement homogène; il est préférable de limiter la diversité des éléments et d'utiliser en priorité des éléments à usage général courants.

3.4.1.3. Effort de développement

Ce type de critère ne s'appuie pas sur des considérations scientifiques ou techniques mais est lié aux ressources humaines qui peuvent être affectées à l'étude et la réalisation d'une solution donnée. L'effort est minimal lorsque les matériels et logiciels existants sont adaptés, ou requièrent peu de modifications. En ce qui concerne les réseaux, l'utilisation d'équipements disponibles dans le commerce est très vivement recommandée dans de nombreuses collaborations, et en particulier dans ATLAS. La motivation centrale des expériences est la physique, et le recours à des développements spécifiques ne peut se justifier que lorsque l'existant n'offre pas une solution satisfaisante ou que les adaptations nécessaires en diminuent considérablement l'intérêt.

3.4.1.4. Support industriel et maintenance à long terme

Cet aspect est également très important pour les expériences au LHC qui auront une durée de vie supérieure à une dizaine d'années. Il est souhaitable de disposer de plusieurs sources d'approvisionnement pour tous les éléments du système afin d'en assurer le remplacement en cas de défaut. Les technologies utilisées doivent être des standards industriels répandus et avoir une durée de vie compatible avec celle de l'expérience. Compte tenu de la mobilité des personnels, il est fort probable que les personnes qui auront participé à la conception et à la construction du système ne seront pas celles qui en assureront l'opération et la maintenance à long terme. Il est donc nécessaire de bénéficier d'un important support industriel. Une solution optimale en ce sens est une solution basée sur du matériel commercial courant. Les standards moins répandus et les développements

spécifiques sont jugés moins favorables.

3.4.1.5. Aspect économique

Cet aspect est à l'évidence déterminant dans la réalisation d'un système opérationnel réel. Quels que soient les mérites théoriques d'une solution ou les aspects pratiques d'une technologie donnée, la réalisation doit satisfaire aux contraintes budgétaires de l'expérience. Ces contraintes vont en général dans le sens d'une limitation des développements spécifiques et d'une utilisation des standards et matériels les plus répandus. Les aspects économiques ne sont donc, le plus souvent, pas contradictoires avec les autres critères précédemment évoqués.

3.4.2. Critères quantifiables

3.4.2.1. Ressources de traitement de données

Une optimisation possible du système est de viser à réduire le nombre de processeurs nécessaires au traitement des événements. Le moyen principal pour réduire ce besoin de ressources est l'utilisation d'algorithmes de traitement ayant un bon rapport simplicité/efficacité. Ce type d'optimisation est souvent une des préoccupations principales des concepteurs d'un T/DAQ. Dans le cas d'ATLAS, cette optimisation est primordiale compte tenu des besoins en ressources de calcul.

3.4.2.2. Besoins au niveau du réseau de communication

Le système peut être optimisé de manière à diminuer le nombre d'éléments à connecter (par des regroupements d'éléments, réduction du nombre de processeurs de traitement...). Une autre optimisation vise à réduire la bande passante nécessaire pour le réseau de communication: transfert de la fraction des données réellement nécessaire, compression des données... Ces deux optimisations sont importantes dans ATLAS compte tenu du nombre d'éléments à connecter et de la bande passante globale.

3.4.2.3. Temps de collecte des données

Le système est optimal suivant ce critère lorsque l'intervalle de temps pour rassembler les données d'un événement est minimal. Il peut être intéressant de minimaliser le temps de collecte des données lorsque le délai pour émettre les décisions de trigger est une contrainte du système (due par exemple à la capacité de mémorisation limitée des sources de données). Cette contrainte est très forte pour le trigger de niveau 1 des expériences au LHC. Pendant l'exécution des algorithmes du trigger de niveau 1, les données sont en général conservées dans des circuits électroniques installés sur les détecteurs. La capacité de mémorisation est relativement faible (typiquement une centaine d'événements) ce qui impose des délais de décision pour le trigger de niveau 1 très courts (quelques micro-secondes) afin de limiter les temps morts. Pour les niveaux de trigger suivants, le temps de latence des décisions du trigger est un paramètre dont l'importance est très différente d'une expérience à l'autre.

3.4.2.4. Capacité de mémorisation

Dans un système de T/DAQ, des mémoires tampon sont en général présentes dans trois emplacements distincts.

3.4.2.4.1. Au niveau du réseau

Dans les matériels de réseau à commutation de paquets, des mémoires tampon sont souvent placées dans les commutateurs. Suivant le profil du trafic à acheminer et la charge sur les différents

liens, des engorgements peuvent se produire en certains points du réseau. Cette surcharge peut entraîner des pertes de messages à cause des débordements dans les mémoires tampon des éléments de commutation du réseau. On peut optimiser le système de manière à réduire ce risque (par un contrôle de flux, profilage du trafic...). Dans la plupart des matériels du commerce, la capacité de mémorisation interne des commutateurs n'est pas modifiable. Elle varie de quelques centaines de kilo-octets à plusieurs méga-octets par port selon les modèles (placée parfois en entrée, le plus souvent en sortie, partagée entre tous les ports ou non...).

3.4.2.4.2. Au niveau des sources

Au niveau des sources, les données doivent être conservées tant que cela est nécessaire (i.e. jusqu'à ce que la décision de conserver ou rejeter l'événement ait été prise). Le système peut être optimisé de sorte que la taille des mémoires tampon dans chaque source soit minimale. Lorsque les éléments de mémorisation des sources sont situés sur les détecteurs, les nombreuses contraintes (encombrement, intégration dans un ASIC comportant de nombreuses autres fonctions, dissipation thermique, tenue aux radiations,...) rendent ce critère primordial (cas du trigger de niveau 1 de nombreuses expériences dont ATLAS). Lorsque les éléments de mémorisation se situent dans un environnement moins hostile, ce critère peut se révéler moins important. Suivant la technologie actuelle, chaque source réalisée sur une carte électronique de format courant peut disposer de plusieurs dizaines de méga-octets de mémoire sans que cela ne pose de problème particulier. Il est important de vérifier que cette capacité mémoire est suffisante pour l'application envisagée et d'effectuer une optimisation relative à ce paramètre le cas échéant.

3.4.2.4.3. Au niveau des destinations

Chaque destination doit être capable de recevoir au moins les données d'un événement. Dans le cas où de multiples événements sont en cours d'assemblage dans une destination donnée, le besoin de mémoire est augmenté d'autant. Une optimisation possible peut être de viser à réduire la taille des mémoires tampon au niveau des processeurs destinations. L'utilisation de machines du commerce (PC ou ordinateurs sur cartes) permet à ce jour de disposer d'un minimum de 64 Mo et davantage si nécessaire, ce qui ne devrait pas constituer de limitation sur le nombre d'événements en cours d'assemblage dans notre cas.

3.5. Résumé

Dans ce chapitre, j'ai présenté les fonctionnalités requises du point de vue de la communication entre les différents éléments du système de sélection et d'acquisition de données de l'expérience ATLAS. J'ai décrit deux mécanismes de contrôle des flux de données dans le système étudié. J'ai donné certains des paramètres pouvant être optimisés et des critères permettant de comparer les implémentations du point de vue de la réalisation technique et des performances.

CHAPITRE IV. ETABLISSEMENT DE L'ARCHITECTURE PROPOSÉE

4.1. Introduction

Ce chapitre est consacré à l'établissement du modèle du système de sélection et de reconstruction d'événements proposé pour l'expérience ATLAS. J'effectue une analyse critique des modèles initiaux des T/DAQ des expériences ATLAS et CMS et discute la validité de ces architectures dans le contexte technologique actuel et celui du futur proche. Je décris les conséquences des études relatives aux algorithmes de physique sur la stratégie globale de sélection des événements dans l'expérience ATLAS. Je présente et justifie les modifications que nous avons proposées pour le T/DAQ de l'expérience ATLAS. J'indique celles qui ont effectivement été prises en compte par la collaboration et celles qui demeurent des options à l'étude.

4.2. Analyse critique des T/DAQ d'ATLAS et de CMS

4.2.1. Contrôle du flot de données

Les modèles initiaux des systèmes de sélection et de reconstruction d'événements pour ATLAS et CMS sont décrits dans les rapports de proposition technique de ces deux expériences. Dans les deux cas, le mode de contrôle du flot de données allant des sources vers les processeurs de traitement est le modèle "donnée poussées" couramment employé par les expériences antérieures. Les raisons de ce choix sont différentes pour ATLAS et CMS. Pour les concepteurs du trigger d'ATLAS, la réduction du temps de latence du trigger de niveau 2 entre dans la justification du modèle "données poussées" alors que pour CMS, c'est l'aspect de simplicité du contrôle qui prime.

Dans le cas d'ATLAS, le choix des sources de données concernées par chaque événement est effectué dynamiquement en se basant sur la position des RoIs identifiées. Cela nécessite de diffuser l'information correspondante à l'ensemble des sources à la fréquence du trigger de niveau 1 (100 kHz). Le choix décrit dans le *Technical Proposal* d'ATLAS est un réseau dédié reliant le *RoI Builder* à tous les *DAQ Crates* (~100 unités). En considérant une taille de message à diffuser pour chaque événement de ~50 octets, la bande passante utile globale requise pour ce réseau est de 500 Mo/s. La mise en place de ce réseau complique l'architecture, n'est pas triviale du point de vue technique et a un certain coût.

Chaque source reçoit l'information relative aux RoIs par un chemin dédié venant du *RoI Builder* et celle concernant les processeurs affectés aux traitements correspondants par un autre chemin dédié venant du superviseur. Les sources doivent effectuer la mise en correspondance de ces deux informations en se basant sur le numéro d'identité de l'événement. Cette approche alourdit inutilement la tâche des sources en doublant le nombre de messages à traiter et en ajoutant une étape de mise en correspondance de messages relatifs aux mêmes événements mais provenant de deux origines différentes. L'alternative est de transmettre à chacune des sources concernées (et seulement à celles-ci) l'information relative à chaque RoI et l'identité du processeur affecté à son traitement dans un même message.

Une complication supplémentaire dans ATLAS résulte du choix de traitement parallèle des RoIs (multiples RoIs par événement et traitement de chaque RoI par détecteur) et de la séparation explicite de l'algorithme de sélection en tâches "locales" suivi d'un traitement global. Plusieurs unités de traitement doivent être allouées pour chaque événement ce qui rend relativement complexe la tâche d'identification des destinations par les sources. Chaque source doit en effet savoir quel processeur local est affecté à une région d'intérêt donnée. Comme le nombre de fragments à collecter en provenance des sources est variable et que les fragments arrivent sans ordre précis dans les processeurs locaux, chaque source doit ajouter au fragment de données qu'elle envoie le nombre total de fragments que comporte la RoI considérée. Un processeur local doit également être informé de l'identité du processeur global affecté pour l'événement considéré. De même, le processeur global doit connaître le nombre de processeurs locaux dont il attend des résultats (il s'agit du produit du nombre de RoIs par le nombre de détecteurs). L'allocation géographique des processeurs conduirait à une mauvaise utilisation des ressources (à cause de la répartition non uniforme des RoIs dans les détecteurs) tandis que l'allocation dynamique n'est pas une tâche triviale (allouer ~26 processeurs par événement à 100 kHz).

Je pense que la complexité du matériel et du logiciel ajoutée au système de contrôle pour l'identification des sources et destinations diminue de beaucoup les avantages potentiels du modèle "données poussées" et rend cette approche difficile à réaliser techniquement.

Dans le modèle de CMS, la situation est très différente car:

- le choix des sources de données est effectué de manière statique: il s'agit de l'ensemble des sources des calorimètres et du spectromètre à muons, ou de la totalité des sources;
- une seule unité de traitement est affectée par événement.

Cette approche permet effectivement de réduire sensiblement la complexité du système de contrôle du T/DAQ mais impose un transfert de données plus important que dans le cas d'ATLAS. Cela a pour conséquence d'augmenter considérablement les besoins en bande passante du réseau reliant les sources de données aux processeurs de traitement (de quelques dizaines de gigabit par seconde à plus de 500 Gbit/s). Compte tenu des technologies disponibles et du coût actuel des équipements de commutation, je pense que cette approche est difficilement réalisable aujourd'hui. L'objectif de minimiser les besoins en bande passante de réseau me semble fondamental et je pense que le choix du mécanisme de RoIs dans ATLAS est pleinement justifié.

4.2.2. Structure des réseaux de communication

Les choix sont également très différents dans ATLAS et CMS: multiples réseaux d'une part et réseau homogène pour les triggers de niveau 2 et de niveau 3 d'autre part. Dans les deux cas, le réseau pour le transfert des données des sources vers les processeurs de traitement peut être uni-directionnel. Le flux d'information dans un système de sélection et de reconstruction d'événements est principalement uni-directionnel, allant des sources de données vers les processeurs de traitement. L'utilisation d'un réseau bi-directionnel symétrique en bande passante n'est donc pas optimal car cela conduit à devoir utiliser un réseau de capacité double de ce qui est nécessaire. Cependant, compte tenu du fait que la plupart des technologies de réseau offrent des liaisons bi-directionnelles (à de rares exceptions près, telle la technologie HIPPI), je pense qu'en pratique, il est difficile d'éviter cette perte de ressources. De plus, le trafic en provenance des processeurs de traitement n'est pas complètement nul: transmission d'accusés de réception des données transmises par les sources, information au superviseur de la disponibilité des processeurs, transport de messages de surveillance... Au lieu de rajouter des réseaux de contrôle dédiés en plus du réseau de transport principal de données, je pense qu'il est préférable de tenter d'exploiter au mieux les possibilités de

ce réseau principal.

Un des choix proposés pour ATLAS est d'utiliser des réseaux différents pour le contrôle, le transfert des données (réseau séparé pour les triggers de niveau 2 et de niveau 3), et éventuellement segmenté à l'intérieur du trigger de niveau 2: partage par détecteur, réseau connectant les processeurs locaux et globaux, autre réseau spécialisé pour la distribution des informations relatives aux RoIs... Cette séparation entraîne l'utilisation de nombreux éléments disparates et accroît le nombre de composants nécessaires. Cela rend le système inutilement complexe en multipliant les chemins de transfert dédiés entre les différents éléments. Je pense que seule une distinction logique entre les différents types de communication doit être faite, et que chaque élément du système doit être équipé d'une liaison bi-directionnelle à un réseau commun d'usage général.

Pour CMS, le réseau principal est unique pour les triggers de niveau 2 et 3 et permet la communication de n'importe quelle source de données avec n'importe quel processeur de traitement. L'existence d'un chemin de retour par ce même réseau depuis les processeurs vers les sources n'apparaît pas sur les diagrammes de principe, mais risque à mon sens d'être imposé par la technologie de réseau qui sera adoptée. Cela est particulièrement critique pour CMS car afin de bénéficier d'une bande passante utile de 500 Gbit/s entre les sources et les unités de traitement, le réseau bi-directionnel symétrique à utiliser doit offrir 1 Tbit/s de bande passante. Cette perte de ressource me paraît inacceptable. Je pense que les concepteurs de CMS devront adapter leur modèle pour tenir compte de ce problème. Une possibilité de réalisation technique d'un réseau uni-directionnel à partir d'un commutateur équipé de ports bi-directionnels est d'utiliser séparément la partie d'émission et de réception de chaque port: la partie de réception est connectée à une source de données et la partie d'émission à un processeur de traitement. Au niveau de la carte réseau des sources (destinations), seule la partie d'émission (de réception) est utilisée. Ce type de fonctionnement d'un commutateur est non standard et peut, dans certains cas, ne pas être possible.

Je pense que l'approche basée sur un réseau homogène défendue pour CMS est plus simple et plus flexible que celle proposée pour ATLAS.

4.2.3. Objectif de réduction du temps de latence du trigger de niveau 2

Une des justifications du modèle "données poussées" et du recours au traitement parallèle des événements dans ATLAS est l'objectif de réduction du temps de latence des décisions du trigger de niveau 2. Comme les données des événements sont conservées dans les mémoires tampon des sources durant le processus de sélection, la capacité de mémorisation de ces dernières est proportionnelle au temps de latence des décisions du trigger de niveau 2. Ce paramètre est critique dans certains cas, par exemple lorsque ces mémoires tampon sont placées à proximité des détecteurs où les contraintes de place, de consommation électrique et de tenue aux radiations sont fortes.

Diverses expériences, par exemple NA48 [Anv97] et D0 [Bla97a], ont été construites avec une capacité de mémorisation, au niveau des sources, correspondant aux données produites pendant ~100 μs de fonctionnement de l'expérience. Afin de réduire le temps mort du système d'acquisition de données, la décision de l'algorithme de sélection doit être émise en moins de 100 μs. Cela impose une limite à la complexité des algorithmes de traitement, nécessite la mise en place de réseaux à faible temps de transit, fait appel à de la logique câblée dans certains cas, impose l'utilisation de systèmes d'exploitation temps réel optimisés, etc. Le besoin de déterminisme et de faible temps de latence est une contrainte forte sur le système.

A l'origine, pour ATLAS, le temps de latence envisagé pour les décisions du trigger de niveau 2 était de 1 à 10 ms en moyenne. Bien que moins sévère que la centaine de micro-secondes

imposée dans d'autres expériences, je pense que cette contrainte demeure trop forte et n'est pas justifiée. Je montre cette affirmation par le raisonnement suivant. Après avoir envisagé diverses options pour le stockage temporaire des données pendant l'exécution des traitements du trigger de niveau 2, le modèle adopté est basé sur des *Read-Out-Buffers* situés en partie dans un local en surface et pour le reste dans une salle à proximité des détecteurs (sans contrainte sévère de radiations ni d'encombrement). Pour tous les événements acceptés par le trigger de niveau 1, l'ensemble des données disponibles est mis en forme par les *Read-Out Drivers* puis est transmis aux *Read-Out Buffers* (incorporant la mémoire tampon nécessaire). Suivant l'hypothèse de fragments d'événements de ~1,6 ko, la bande passante de chaque liaison ROD-ROB est de ~160 Mo/s pour un fonctionnement à la fréquence maximale du trigger de niveau 1 de 100 kHz. En faisant l'hypothèse que chaque ROB dispose d'une mémoire de 16 Mo pour le stockage temporaire des événements (facile avec les circuits mémoire actuels), 100 ms de prise de données peuvent être stockées: je pense qu'il est possible de lever la contrainte sur le temps de latence des décisions du trigger de niveau 2 (valeur moyenne et caractère déterministe)[1]. L'objectif de réduction de ce paramètre ne doit pas être une priorité guidant les choix d'architecture pour le trigger de niveau 2.

Il est fondamental que le système puisse accepter le flux d'événements nominal de sortie du trigger de niveau 1, mais le temps de décision lui-même importe moins. Le temps moyen d'exécution de l'algorithme de sélection d'événement dans un processeur permet de fixer le nombre de processeurs que le système doit comporter pour pouvoir absorber le flux d'entrée. Le temps de latence du trigger est la somme du temps d'exécution dans l'unité de traitement et du temps passé dans différents éléments (temps d'attente dans les sources, transit à travers le réseau, etc). Il paraît difficile d'envisager des algorithmes utilisant en moyenne plus de 10 ms de temps de calcul par événement dans la mesure où ce choix conduirait à un système comportant plus de 1000 processeurs ce qui pose des problèmes pratiques et économiques. Une capacité de mémorisation de 100 ms au niveau des ROBs me semble donner une marge de sécurité suffisante car je pense que la limite sur le temps de latence moyen du trigger est davantage liée aux ressources de traitement des événements qu'à la capacité des mémoires tampon des ROBs.

Dans l'expérience CMS, le temps de latence du trigger de niveau 2 n'est pas considéré comme critique et la mémoire tampon de chaque DPM a une capacité prévue de l'ordre de 256 Mo. A mon sens, cela est surdimensionné, mais cette configuration constituera peut-être effectivement la taille de mémoire courante des ordinateurs sur carte de la prochaine décennie.

4.2.4. Flexibilité et caractère évolutif du système matériel

L'architecture du trigger de niveau 2 décrite dans le *Technical Proposal* de l'expérience ATLAS est conçue et optimisée pour le traitement des régions d'intérêt identifiées par le trigger de niveau 1. Les traitements se font de manière systématique sur l'ensemble des RoIs et dans chaque détecteur. Le trigger de niveau 3 est physiquement séparé et utilise un réseau et des processeurs distincts de ceux du trigger de niveau 2. Un des inconvénients de cette approche est que les ressources (réseau et processeurs) sont liées à l'un des deux sous-systèmes de manière figée. Les besoins en bande passante réseau et puissance de calcul pour le trigger de niveau 3 sont directement proportionnels au produit de la fréquence d'événements en sortie du trigger de niveau 1 par le facteur de rejet du trigger de niveau 2. Le but est d'obtenir un facteur de rejet de 100 pour le trigger de niveau 2, mais le facteur qui sera effectivement obtenu est très dépendant des caractéristiques des détecteurs, de l'efficacité et de la complexité des algorithmes utilisés, etc. Si le taux de rejet effectif

1. voir l'étude de modélisation et la simulation présentée dans l'Annexe 1 de cette thèse.

n'est pas suffisant, il est toujours possible d'ajuster les seuils de déclenchement de sorte que le facteur de rejet désiré soit obtenu, mais cette méthode triviale peut dégrader le potentiel de physique de l'expérience. Dans le cas où le facteur de rejet obtenu est inférieur à 100, le trigger de niveau 3 devra être en mesure d'accepter cette surcharge; à l'inverse, si le trigger de niveau 2 est plus efficace que prévu, le trigger de niveau 3 pourra se révéler surdimensionné. Je pense que l'utilisation de ressources distinctes pour les triggers de niveau 2 et de niveau 3 conduit à surdimensionner ces deux systèmes dans la mesure où les ressources doivent être fixées en prenant des marges de sécurité suffisamment larges sans qu'il soit possible de transférer des ressources d'un système à l'autre. Je pense qu'il est souhaitable d'utiliser le même type de ressources partagées pour les deux systèmes de sorte que l'ajout de processeurs et/ou de bande passante de réseau puissent être bénéfiques globalement à l'ensemble du système. De plus, je pense qu'il est souhaitable que la répartition des ressources en bande passante de réseau et puissance de calcul entre les triggers de niveau 2 et de niveau 3 puisse être modulée facilement. Ce type d'approche est voisin de la stratégie adoptée par CMS: le même réseau est utilisé pour le transfert des données du trigger de niveau 2 et de niveau 3. De même, les processeurs utilisés sont tous du même type et enchaînent sans distinction les algorithmes du trigger de niveau 2 puis ceux du niveau 3. Le partage des ressources disponibles s'effectue dynamiquement et de manière automatique.

Dans le trigger de niveau 2 d'ATLAS, le problème d'allocation de ressources est également présent. Considérons le modèle où les processeurs de traitement locaux sont sur des réseaux distincts par détecteur et les processeurs de traitement global forment eux aussi un groupe à part. Le nombre de processeurs de chaque groupe et les ressources en bande passante de chaque sous-réseau sont très dépendants du type d'algorithme exécuté. Par exemple, les algorithmes utilisés aujourd'hui pour le traitement des données du calorimètre prennent environ 10 fois moins de temps que ceux du traitement du détecteur de traces. Avec un modèle de partage explicite des réseaux et groupes de processeurs, chaque groupe doit être dimensionné à la construction en tenant compte de cette disparité. En cas d'erreur d'estimation ou d'évolution dans la durée des différents algorithmes, les excédents de ressources affectées à un détecteur ne peuvent pas être transférés pour compenser un manque dans un autre détecteur. Une amélioration consiste à utiliser un réseau unique et à fusionner les différents groupes de processeurs. Je pense que cette approche permet une grande flexibilité dans l'allocation des ressources, en garantit l'utilisation efficace indépendamment des algorithmes exécutés tout en permettant une évolution homogène du système.

4.2.5. Flexibilité sur les algorithmes de sélection

L'architecture du trigger de niveau 2 du *Technical Proposal* d'ATLAS est calquée sur la stratégie de traitement des régions d'intérêt et est mal adaptée à l'implémentation de toute autre approche du processus de sélection. Dans le modèle "données poussées" le choix des données à utiliser est effectué à priori. Il est aujourd'hui difficile d'affirmer que seul le traitement des régions d'intérêt sera suffisant pour obtenir le facteur de rejet désiré. Il est envisageable d'effectuer des traitements complémentaires si les ressources disponibles le permettent. La frontière entre les algorithmes relevant du trigger de niveau 2 et ceux du niveau 3 peut fluctuer. Je pense que l'architecture du système ne doit pas imposer une frontière stricte dans le partage des algorithmes. Le transfert de tâches initialement prévues pour le trigger de niveau 3 vers le trigger de niveau 2 doit pouvoir se faire sans difficultés (ou bien le transfert inverse).

Dans le cas du trigger destiné à la physique du méson B, la méthode de sélection est différente du cas précédent car les régions d'intérêt ne sont pas identifiables par le trigger de niveau 1. La méthode proposée à ce jour est séquentielle avec une étape de traitement initiale consistant en la

recherche de traces de particules dans l'ensemble du *Transition Radiation Tracker*. Cette étape permet d'identifier des candidats particules et produit une liste de régions d'intérêt dont chacune peut être analysée en collectant les données correspondantes. La différence principale avec le mode de fonctionnement guidé par les RoIs identifiées par le trigger de niveau 1 est que le choix des sources de données qui seront utilisées par l'algorithme de sélection n'est pas connu au moment où le traitement de l'événement commence. Le modèle "données poussées" n'est pas compatible avec ce mode de fonctionnement. Une solution où le superviseur est capable d'initier de multiples transferts de données sur ordre des processeurs de traitement a été esquissée dans [ATL98], mais reste à valider compte tenu de sa complexité.

Je pense que l'architecture du système de sélection d'événements de l'expérience ATLAS doit être identique pour les différents types de physique étudiés et ne doit pas restreindre le choix des algorithmes à une stratégie de sélection particulière. Le modèle proposé pour CMS est de ce point de vue beaucoup plus souple: tous les processeurs peuvent avoir accès à l'ensemble des données et peuvent exécuter tout type d'algorithme sans changement de l'architecture du système.

4.3. Sélection séquentielle des événements dans ATLAS

Une méthode de sélection séquentielle des événements dans ATLAS a été proposée dans [Bys97]. Les principes et bénéfices sont rappelés ici. Les simulations de physique actuelles montrent que le nombre moyen de RoIs par événement est de l'ordre de 5. Deux types de RoIs peuvent être distingués. Les "RoIs primaires" sont celles qui ont causé la décision positive du trigger de niveau 1 (1,2 par événement en moyenne). Le trigger de niveau 1 identifie également des "RoIs secondaires" (3,8 par événement en moyenne) qui à elles seules n'auraient pas entraîné la décision positive du trigger, mais contiennent cependant des signes d'activité particulière. La distribution du nombre de RoIs primaires et du nombre total de RoIs pour des événements du type *di-jets* (~70% des événements acceptés par le trigger de niveau 1) est reproduite Figure 4-1 (d'après [Bys97]).

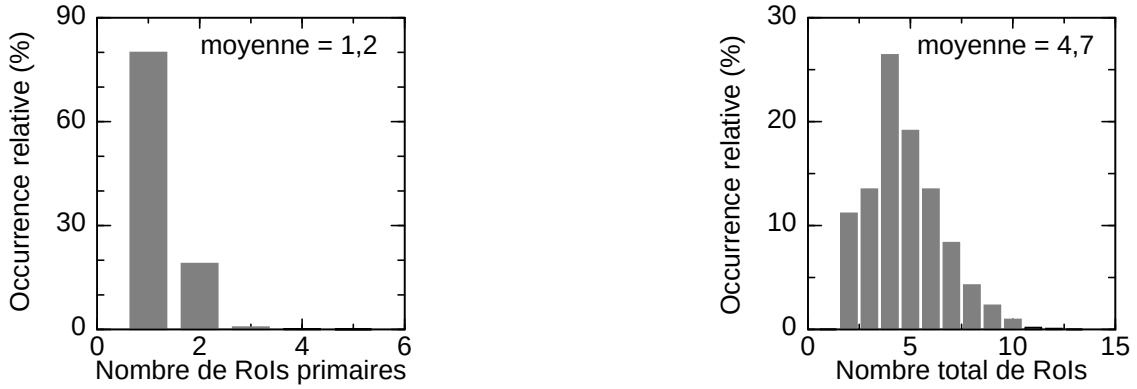


Figure 4-1: Distribution du nombre de RoIs par événement.

Au lieu de traiter de manière systématique l'ensemble des RoIs, on peut ne s'intéresser dans un premier temps qu'aux RoIs primaires pour affiner les résultats du trigger de niveau 1. Compte tenu du fait que le trigger de niveau 2 a davantage d'information que le trigger de niveau 1 (meilleure précision et segmentation des capteurs plus fine), cette étape préliminaire pourrait permettre de réduire le taux d'événements d'un facteur estimé entre 5 et 10. Pour les événements restants, les RoIs secondaires sont analysées et les primitives identifiées sont comparées aux signatures des phénomènes que l'on souhaite observer plus finement. L'avantage en terme de réduction des besoins en ressources de calcul et de bande passante de réseau par rapport à une analyse systématique de

l'ensemble des RoIs est fonction du facteur de rejet que l'on peut obtenir à l'issue de la première étape. L'évaluation de ce paramètre relève d'études de physique dépassant le cadre de ce travail. Dans le cas le plus défavorable (aucun rejet à l'issue de la première étape), les ressources nécessaires sont identiques pour le traitement séquentiel des RoIs ou leur analyse systématique.

Une autre possibilité d'économie de ressources consiste à effectuer le traitement d'une RoI en séquence, détecteur par détecteur. Il serait par exemple possible de réduire d'un facteur ~5 le taux de RoIs correspondant à des candidats électrons en analysant seulement les données du calorimètre avant de procéder à l'analyse dans les autres détecteurs.

D'autres algorithmes plus complexes, ne faisant pas nécessairement appel aux RoIs, sont également envisagés pour compléter le processus de sélection avant le trigger de niveau 3. D'une manière générale, les études de physique récentes conduisent à des séquences d'algorithmes plus complexes que le seul traitement systématique de l'ensemble des RoIs. Un traitement séquentiel avec possibilité de rejet de l'événement à chaque étape est aujourd'hui la stratégie de sélection la plus efficace admise par la collaboration ATLAS pour le trigger de niveau 2.

4.4. Architecture proposée pour le T/DAQ d'ATLAS

4.4.1. Démarche de conception

Par rapport au modèle du *Technical Proposal* d'ATLAS un certain nombre de modifications ont été proposées par moi-même et les membres de notre groupe en se basant sur les hypothèses et principes de conception suivants:

- La réduction du temps de latence des décisions du trigger de niveau 2 n'est pas une priorité. Le modèle "données poussées" et le traitement en parallèle des régions d'intérêt est abandonné.
- L'objectif prioritaire est de réduire les transferts de données et la puissance de calcul nécessaire. Les transferts de données "à l'aveugle" et l'exécution d'algorithmes de manière systématique sur toutes les RoIs à travers tous les détecteurs sont supprimés. Les transferts de données sont effectués si cela est nécessaire et sur ordre du processeur de traitement. L'algorithme de sélection est composé d'étapes conçues de manière à pouvoir rejeter les événements dès que possible.
- Le système est adapté à un mode de traitement séquentiel et autorise une grande souplesse dans le choix des algorithmes. Le mode de contrôle "données tirées" est adopté.
- L'architecture est identique pour le fonctionnement à haute et à basse luminosité, et est indépendante des algorithmes envisagés. Le partage explicite de l'algorithme du trigger de niveau 2 entre des unités de traitement fonctionnant en parallèle est abandonné. La totalité de l'algorithme de sélection est confiée à un processeur unique effectuant les traitements en série.
- Les ressources peuvent être réparties facilement entre le trigger de niveau 2 et celui de niveau 3. La séparation physique entre ces deux systèmes est supprimée. Le même réseau est utilisé pour le transport des données dans les deux cas. Le même type de processeur est utilisable pour le trigger de niveau 2 et celui de niveau 3. Deux options sont considérées après une décision positive du trigger de niveau 2: soit le traitement de l'événement est poursuivi dans le même processeur, soit il est confié à un processeur différent. Dans le premier cas, les processeurs ne sont pas différenciés. Dans le second cas, deux groupes sont distingués: les membres du premier groupe sont affectés au trigger de niveau 2 et ceux du second au trigger de niveau 3. L'appartenance d'un processeur à l'un ou à l'autre groupe n'est qu'une distinction logique, les transferts de ressources d'un groupe vers l'autre pouvant se faire de manière simple.
- La reconstruction complète des événements pour le trigger de niveau 3 n'est plus systématique.

Selon les types d'événements, il est possible de n'effectuer qu'une reconstruction partielle de ces événements, éventuellement en plusieurs étapes. Le traitement séquentiel des événements avec plusieurs phases de transferts de données est possible pour le trigger de niveau 3 de la même manière que pour le trigger de niveau 2. Cette possibilité vise à réduire la quantité de données à véhiculer à travers l'*event builder* pour chaque événement ce qui permet soit une économie de ressources réseau, soit d'augmenter le débit du système en nombre d'événements par seconde à capacité de réseau constante[1].

- Le nombre de réseaux est réduit et les chemins dédiés sont supprimés. Les sous-réseaux indépendants et les réseaux spécialisés pour la diffusion des RoIs et des décisions de trigger sont rassemblés en un réseau bi-directionnel unique qui relie tous les éléments du système (sources de données, processeurs, superviseurs). Ce réseau transporte à la fois les données, les messages de protocole, de surveillance du fonctionnement du système, etc. Un réseau supplémentaire pour la configuration du système peut néanmoins être mis en place.

4.4.2. Schéma de principe

Le schéma de principe de l'architecture proposée est présenté Figure 4-2.

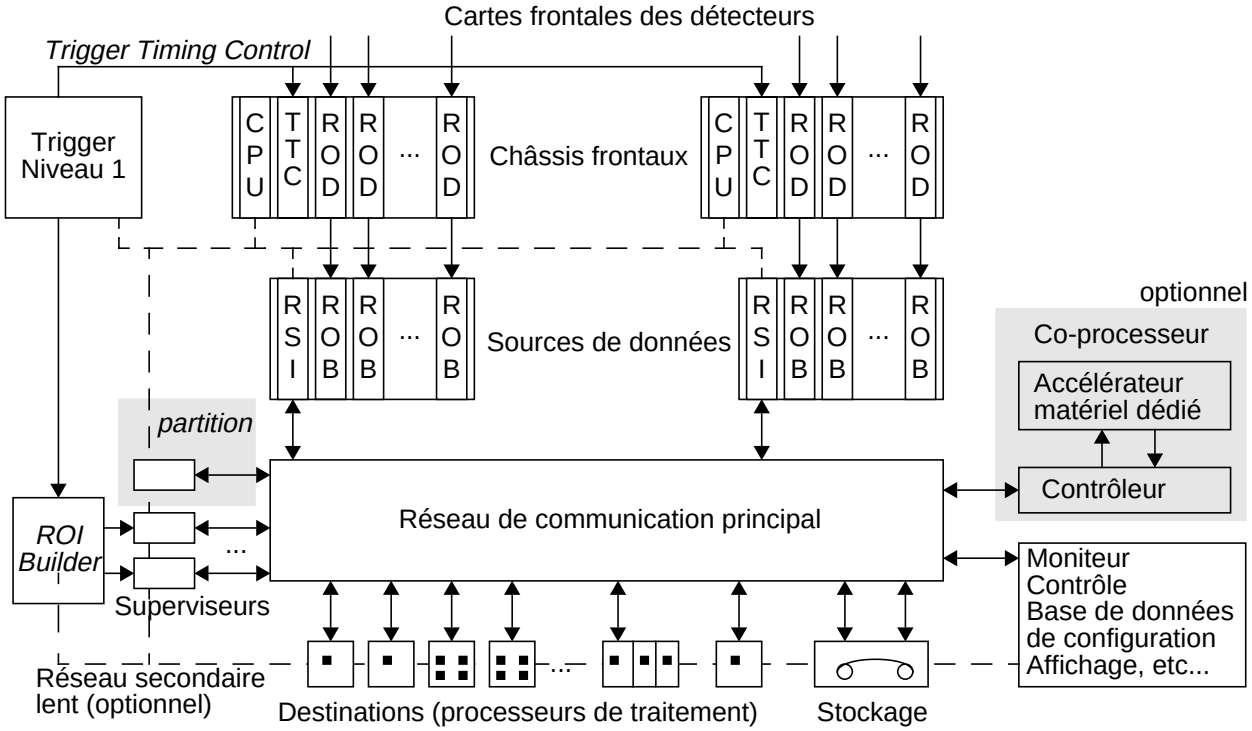


Figure 4-2: Schéma de principe de l'architecture proposée.

Le système s'articule autour d'un réseau de communication principal reliant la plupart des

1. Ce concept pourrait s'appliquer au TRT *Full Scan*. La fréquence de ce traitement est ~10 kHz et le volume de données du TRT est 100-200 kB. Lorsque cet algorithme est exécuté par le trigger de niveau 2, 1-2 GB/s de bande passante réseau est utilisé. Si ce traitement est déplacé au trigger de niveau 3 (flexibilité envisagée par les physiciens), effectuer la reconstruction complète des événements à 10 kHz décuple les besoins au niveau de l'*event builder* (de ~1-2 GB/s à 10-20 GB/s) ce qui pose de sérieux problèmes techniques et économiques. A l'inverse, transférer dans un premier temps uniquement les données du TRT ne fait que doubler les besoins au niveau de l'*event builder* (2-4 GB/s). Les ressources en bande passante et moyens de traitement sont transférés du trigger de niveau 2 vers le niveau 3.

éléments. Un réseau additionnel lent est cependant prévu pour la configuration du système au démarrage, le dialogue avec les bases de données, la mise à jour de paramètres, le dépannage, etc. Le choix d'utilisation d'un ou de deux réseaux distincts est à la fois technique et pratique. Aujourd'hui, la plupart des ordinateurs du commerce (PC et ordinateurs sur cartes) sont équipés d'une interface Ethernet intégrée. Si le réseau principal est de type Ethernet on peut envisager d'utiliser l'interface intégrée de chaque machine sans mettre en place un réseau secondaire. A l'inverse, si une carte réseau spécifique doit être placée dans chaque machine, il est alors possible d'utiliser l'interface intégrée pour constituer un réseau secondaire sans engendrer de surcoût important ni une grande complexité additionnelle.

Chaque carte de lecture des canaux des détecteurs (ROD) est reliée à un *Read-Out Buffer* par une liaison point-à-point. Les ROBs sont groupés en nombre variable dans des châssis du même style que les *Read-Out Crates* du modèle du *Technical Proposal*. Tous les ROBs d'un châssis sont reliés par un bus de fond de panier à un contrôleur commun nommé RSI (*ROB to Switch Interface*). Ce contrôleur est relié au réseau de communication principal par un lien bi-directionnel. Le nombre de ROBs par châssis est ajusté en fonction des possibilités du bus de fond de panier, de la bande passante du lien au réseau et de la sollicitation particulière des ROBs considérés (groupe de 1 à 16 par exemple). Un groupe de ROBs et le contrôleur associé forment un ensemble appelé "source". Les différences avec les *Read-Out Crates* du modèle du *Technical Proposal* sont:

- la suppression du module TRG et des deux réseaux associés (pour la diffusion des RoIs et des décisions du trigger de niveau 2),
- le regroupement des interfaces *LVL2 link* et *LVL3 link* en une interface commune (RSI),
- la suppression du module TTC (*Trigger Timing and Control*),
- la suppression du module CPU en charge de l'acquisition des données locales pour la surveillance du système (*monitoring* du *Local DAQ*).

La manière d'assurer les fonctions qui étaient prises en charge par les éléments que je propose de supprimer est décrite dans la section suivante.

Le(s) superviseur(s) comporte(nt) une liaison d'interface avec le *RoI Builder* et une liaison bi-directionnelle avec le réseau principal. Afin d'obtenir la performance désirée, il est probable que plusieurs superviseurs seront requis. Le système comporte au moins un superviseur de plus que nécessaire pour assurer le bon fonctionnement du système en cas de défaillance d'un superviseur. Un ou plusieurs superviseurs peuvent être assignés à des sources de trigger différentes du *RoI builder*, par exemple lors du test/calibrage d'une partie du système.

Les processeurs de traitement sont appelés "destinations" en raison du rôle complémentaire qu'ils jouent par rapport aux éléments sources. Chaque destination peut être une machine mono-processeur, ou une machine multi-processeurs voire un petit groupe de processeurs reliés par un bus commun (ou un réseau local). Les destinations sont soit séparées en deux groupes distincts, soit non différenciées. Pour le premier modèle ("fermes séparées"), on distingue les destinations affectées au trigger de niveau 2 et celles allouées au trigger de niveau 3. Pour le second modèle ("ferme commune"), tous les processeurs ont le même rôle.

Compte tenu de l'évolution rapide des matériels et des améliorations qui seront apportées au système au fil de son exploitation, différents types de destinations (processeurs, version de système d'exploitation, etc) devront coexister.

Le "moniteur" est un élément en charge de la surveillance du bon fonctionnement du système (mesure de performance, statistiques) et de l'interface utilisateur pour la présentation de ces informations et la prise en compte des paramètres fournis par les opérateurs humains. Pour des raisons de robustesse, il est également possible de dupliquer cet élément.

Le dispositif de stockage des événements à conserver est également connecté au réseau principal. Cette disposition peut être modifiée s'il apparaît préférable de connecter les périphériques de stockage aux processeurs utilisés pour le trigger de niveau 3 par un réseau séparé. L'étude de cette partie de l'architecture dépasse le cadre de cette thèse.

4.4.3. Fonctionnement

Lorsqu'un événement a été accepté par le trigger de niveau 1, le *RoI Builder* produit la liste des régions d'intérêt identifiées et la transmet à un superviseur (sélection cyclique par exemple ou mode de distribution plus élaboré). Chaque superviseur dispose d'une liste initiale de destinations disponibles et affecte l'événement à une destination en lui envoyant un message d'allocation comportant le numéro d'identification de l'événement ainsi que la liste des RoIs identifiées. A la réception de ce message, la destination confie l'événement à un processeur (implicite dans le cas d'une destination mono-processeur) qui identifie le type de traitement à effectuer et la liste des sources contenant les données requises pour effectuer ce traitement. Le processeur envoie une requête aux sources concernées. Le contrôleur de chaque source identifie localement les ROBs impliqués et leur adresse une requête. Les ROBs identifient l'emplacement des données à retourner et informent le contrôleur de leur disponibilité. Les données des différents ROBs sont rassemblées, éventuellement traitées localement par le contrôleur, puis envoyées à la destination qui avait émis la demande. La destination rassemble les fragments d'événements des différentes sources et informe le processeur concerné de la disponibilité des données. Le processeur traite ces données et décide de poursuivre ou non le processus de sélection de cet événement en fonction des résultats obtenus. En cas de poursuite du processus, le processeur identifie le traitement à effectuer et les sources contenant les données nécessaires, envoie des requêtes, etc.

Compte tenu de l'intervalle de temps important entre l'émission des requêtes et la disponibilité des données demandées dans un processeur (en comparaison au temps d'accès à des données situées en mémoire locale), il n'est pas souhaitable qu'un processeur demeure inoccupé pendant cette période. La solution proposée consiste à avoir plusieurs événements en cours de traitement dans chaque processeur. Pendant les phases d'attente des données d'un événement particulier, un processeur est en mesure de traiter les données d'un événement précédent devenues disponibles ou bien d'envoyer les requêtes pour l'événement suivant. Chaque processeur possède une queue de tâches à effectuer et alterne les opérations entre différents événements. Ce mécanisme est facilement réalisable avec les machines utilisant un système d'exploitation multi-tâches ou *multi-threads*, c'est-à-dire avec la majorité des systèmes d'exploitation modernes. Le cycle de requêtes, collecte des données puis traitement se poursuit jusqu'à ce que la décision de conserver ou de rejeter l'événement puisse être prise. Deux modèles sont possibles:

- modèle "ferme commune":
  L'algorithme du trigger de niveau 2 et de niveau 3 est exécuté dans la même destination (éventuellement dans un processeur différent si la destination est une machine multi-processeurs). A la fin du processus, la destination dispose de toutes les données à conserver et de tous les résultats de calcul intermédiaire. La destination envoie alors un message au superviseur qui à son tour informe toutes les sources puis tous les ROBs. Les ROBs libèrent la place mémoire correspondante. La destination stocke localement les données des événements acceptés ou bien les transmet au système de stockage.
- modèle "fermes séparées":
  A la fin de l'algorithme du trigger de niveau 2, la suite des opérations est poursuivie dans une destination distincte. La destination ayant achevé le traitement du trigger de niveau 2 envoie sa

décision au superviseur. Si l'événement est rejeté, le superviseur en informe les sources; les ROBs libèrent la place mémoire utilisée. Si l'événement est accepté, le superviseur alloue une destination du groupe réservé au trigger de niveau 3. Lorsque la destination du trigger de niveau 3 reçoit le message d'allocation du superviseur, elle affecte un processeur, identifie les sources concernées (toutes les sources suivant le modèle actuel, mais un choix plus restreint est également possible), envoie les requêtes... Le mode de fonctionnement est identique à celui du trigger de niveau 2 bien que le traitement envisagé à ce jour par la collaboration ne comporte qu'une seule phase de collecte de données puis de traitement. Lorsque les sources ont envoyé les données au processeur correspondant, l'événement est effacé de la mémoire des ROBs. Le processeur du trigger de niveau 3 effectue les traitements nécessaires puis conserve ou rejette l'événement. De manière semblable au modèle "ferme commune", les données sont stockées localement ou transmises à un dispositif extérieur.

Les deux modèles présentent à la fois des avantages et des inconvénients. Dans le concept, le modèle "ferme commune" est le plus simple. Un problème peut venir de difficultés d'intégration de logiciel entre les algorithmes utilisés pour le trigger de niveau 2 et le code utilisé pour le trigger de niveau 3. Un des avantages du modèle "fermes séparées" est de partager le système en deux blocs relativement indépendants ce qui facilite l'organisation et la répartition du travail entre les différents groupes impliqués. Du point de vue technique et économique, il paraît prématuré de se prononcer en faveur de l'une ou l'autre des deux options.

Dans le modèle proposé, un objectif est de réduire à un minimum la tâche de surveillance locale des *ROBs* afin qu'elle puisse être effectuée par le RSI. Cependant, un processeur séparé (LDAQ) pourrait se réveler nécessaire si cette charge de travail demeure trop lourde pour être prise en charge par le RSI.

Le système peut être partagé en plusieurs sous-systèmes fonctionnant de manière indépendante et disposant chacun de sa propre source de déclenchement. Ce mode d'opération est utile dans les phases d'installation, pour la mise au point et le calibrage des différents détecteurs. Alors qu'une partie du système fonctionne en utilisant les déclenchements fournis par le trigger de niveau 1, un ou plusieurs détecteurs peuvent fonctionner en mode test ou calibrage avec pour chaque partition une source de déclenchement propre. Au niveau des RODs, la diffusion des déclenchements est assurée par le système de *Trigger Timing and Control* (TTC) [ATL98b] que l'émetteur soit le trigger de niveau 1 ou une autre source: le TTC peut être partagé en ~35 segments pouvant chacun véhiculer les ordres de déclenchements fournis par le trigger de niveau 1, par un générateur aléatoire interne ou par une des trois entrées de déclenchement extérieur. Dans le modèle du *Technical Proposal*, la distribution de l'information au niveau des *DAQ Crates* est effectuée par le module TRG pour les déclenchements émis par le trigger de niveau 1 et par le module TTC pour le fonctionnement en mode segmenté. Dans l'architecture proposée, l'information provient dans les deux cas des superviseurs via le réseau de communication principal. Pour le fonctionnement du système en mode segmenté, au moins un superviseur est affecté à une partition. Soit ce superviseur est la source de déclenchement pour la partition considérée en injectant des ordres dans le segment de TTC ainsi que vers les RSIs concernés, soit il puise les ordres de déclenchement dans le segment de TTC considéré pour les diffuser aux RSIs correspondants. L'intérêt de l'approche proposée est de supprimer nombre des interfaces des *DAQ Crates*: les interfaces au TTC, au *RoI Builder* et au superviseur sont rassemblées en une seule interface également utilisée pour les transferts de données. Le flux de données va dans la direction opposée à celle des informations de contrôle ce qui limite les interférences entre ces différents flux et permet d'exploiter au mieux les liaisons bi-directionnelles des RSIs (utilisées dans un seul sens dans le modèle du *Technical Proposal*).

4.5. Modèle et options considérés par la collaboration

Dans la collaboration ATLAS, le modèle désormais accepté pour le trigger de niveau 2 est très proche de celui précédemment décrit: algorithme et traitement séquentiel, modèle "données tirées", réseau unique pour transporter les données et les messages de protocole. Concernant la mise en commun des réseaux et processeurs pour le trigger de niveau 2 et de niveau 3, la décision n'a pas été prise. Le modèle du *Technical Proposal* n'est de ce fait pas remis en cause. Le modèle proposé dans cette thèse constitue toujours une option différente du schéma de base.

D'autres options sont envisagées pour les triggers de haut niveau dans ATLAS:

- le développement d'un trigger de niveau intermédiaire ("1,5") réalisé en logique câblée re-programmable et devant permettre de réduire d'un facteur 5-10 le taux d'événements à l'entrée du trigger de niveau 2 [Boc97]. L'ajout de ce bloc ne modifie pas l'architecture des triggers de niveau 2 et de niveau 3 mais réduit les besoins en bande passante de réseau et puissance de calcul pour ces parties du système. Ce concept demeure à étudier et à valider.
- l'ajout de matériel spécifique destiné à l'accélération de certains algorithmes. Une possibilité est d'ajouter une carte spécifique à chaque destination (sans modification de l'architecture du système). Une autre possibilité est d'utiliser une destination d'un type particulier constituée par une machine dédiée jouant le rôle de "co-processeur" pour les processeurs destinations (Figure 4-2). La tâche envisagée pour le co-processeur est la recherche de traces dans l'ensemble du TRT (*Full TRT scan*) qui requiert une puissance de calcul importante. Lorsqu'un processeur souhaite effectuer le *Full TRT Scan*, il envoie une requête au co-processeur. Celui-ci envoie à son tour une requête à l'ensemble des sources correspondant au TRT. Lorsque les données ont été collectées (en utilisant le réseau de communication principal ou bien un réseau annexe), le programme de recherche de traces est exécuté sur la machine dédiée et les résultats sont retournés au processeur qui en avait fait la demande.

4.6. Résumé

Dans ce chapitre j'ai analysé de manière critique les modèles initiaux des systèmes de sélection d'événements et d'acquisition de données des expériences ATLAS et CMS. J'ai décrit et justifié le modèle de T/DAQ pour ATLAS proposé dans cette thèse. J'ai présenté le fonctionnement de ce modèle et j'ai mentionné les différentes options d'architecture envisagées.

CHAPITRE V. RÉSEAUX À HAUT DÉBIT

5.1. Introduction

Dans ce chapitre, je présente un bref historique sur les réseaux informatiques. Je montre l'utilisation de certaines technologies pour la réalisation de systèmes d'acquisition de données d'expériences de physique. Je décris les technologies qui sont étudiées pour la réalisation du T/DAQ de l'expérience ATLAS et évalue de manière qualitative leur potentiel en vue de cette application afin de faire ressortir celles qui paraissent les plus prometteuses. Je présente en particulier la technologie *Asynchronous Transfer Mode* (ATM) qui a été utilisée pour cette étude.

5.2. Historique des réseaux

Le premier câble de communication transatlantique ayant fonctionné fut posé en 1866 par la Telegraph Construction and Maintenance Company [Dum97]. Un message du Président des Etats-Unis à la Reine Victoria fut transmis à la vitesse de 37 lettres par minute, soit, exprimé en unités actuelles, un débit équivalent à environ 4 bit par seconde. Dix ans plus tard, Alexandre Graham Bell inventait le téléphone permettant l'échange de messages de vive voix. Le premier commutateur téléphonique opérationnel fut installé en Suède en 1916. Parallèlement à la mise au point des premiers ordinateurs électro-mécaniques et électroniques peu après la seconde guerre mondiale, Shannon publiait en 1948 sa théorie mathématique de l'information [Sha48], et la naissance du transistor était annoncée cette même année par les Bell Telephone Laboratories. Les premiers mini-ordinateurs apparurent dans les années 60, et commença alors à se poser le problème d'échange d'informations entre ordinateurs géographiquement éloignés. Alors que les systèmes de téléphonie reposaient sur des réseaux effectuant la commutation de circuits électriques, un nouveau mode de communication "par paquets" (*packet switching*) fut proposé par Baran en 1964 [Bar64]. Une connexion expérimentale entre des ordinateurs de 4 sites – l'université de l'Utah, le Stanford Research Institute, l'université de Californie à Santa Barbara et Los Angeles – fut mise en place en 1969 dans un projet financé par l'Advanced Research Projects Agency (ARPA) du Department of Defense Américain. Ce réseau évoluera pour donner naissance à l'Internet puis au *World Wide Web* (WWW) qui connaît un engouement planétaire depuis quelques années. La très rapide croissance du nombre de sites et d'utilisateurs entraîne la mise en place d'infrastructures de communication à haut débit. Suite à la généralisation de l'utilisation de l'informatique et de la micro-informatique à tous les niveaux de notre société est apparu le besoin de communication entre ordinateurs situés sur un même site (réseaux locaux, *Local Area Networks* – LANs), dans la même zone géographique (réseaux métropolitains, *Metropolitan Area Networks* – MANs) et ceux répartis à l'échelle des pays et du monde entier (réseaux étendus, *Wide Area Networks* – WANs). Les réseaux locaux ont connu un développement rapide à partir des années 70, et plus rapide encore depuis quelques années avec l'accroissement du parc informatique dans les entreprises, la mise en place d'Intranet, de serveurs WWW, etc.

Pour la réalisation d'un réseau de communication, deux fonctions principales peuvent être

distinguées: le transport et l'aiguillage de l'information. Dans le domaine des réseaux étendus, le transport de l'information peut se faire au moyen de lignes téléphoniques classiques, de câbles en cuivre ou de fibres optiques, de faisceaux hertziens (satellites)... Les techniques modernes de transmission de l'information (ADSL – *Asymetric Digital Suscriber Line*) permettent de transporter un flux de plusieurs méga-bit par seconde sur une ligne téléphonique classique. Les développements des composants optiques, des lasers et de l'électronique rapide autorisent aujourd'hui un débit de ~320 Gbit/s sur une seule fibre optique, par la technique du multiplexage dense de longueur d'onde (*Dense Wavelengh Division Multiplexing* – DWDM). Dans le domaine des réseaux locaux, le support physique pour le transport de l'information peut être un câble coaxial, des câbles en paires torsadées blindés ou non blindés (*Unshielded Twisted Pairs* – UTP), ou des fibres optiques. Les débits sont compris entre 1 Mbit/s et 1 Gbit/s.

Les équipements d'aiguillage de l'information dans les réseaux étendus (commutateurs ou routeurs) permettent de diriger le trafic dans la direction appropriée. Les plus gros commutateurs permettent d'aiguiller un flux total de quelques centaines de giga-bit par seconde et comportent jusqu'à plusieurs milliers de liens (les routeurs, dispositifs comportant davantage "d'intelligence" implantée en logiciel pour effectuer des décisions de routage plus complexes, ont des performances plus modestes en terme de débit). Dans les réseaux locaux, des concentrateurs sont en général utilisés pour regrouper un petit nombre de machines (8 à 16) sur un lien commun relié à un commutateur central. Les commutateurs centraux les plus puissants disponibles à ce jour permettent de connecter un peu plus de 200 liens et offrent un débit global de quelques dizaines de giga-bit par seconde.

5.3. Technologies utilisées dans les expériences de physique

Pour les systèmes d'acquisition de données d'expériences de physique, la problématique relève de la conception d'un réseau local: interconnexion de machines sur un même site, faibles distances, nombreux éléments à connecter, réseau privé sans impératif de continuité du service, pas de facturation des communications, etc. Lorsqu'il ne s'agit pas de développements entièrement spécifiques, les réseaux utilisés sont basés sur des technologies de type LANs (standards plus ou moins répandus, solutions propriétaires et combinaisons diverses).

Le "*Data Merger*" de l'expérience NA48 rapporté dans [Boy94] effectue le ré-assemblage de fragments d'événements en provenance de ~15 sources à l'aide de matériel dédié concentré dans un seul châssis d'électronique. Les cartes portant les liens d'entrée en fibre optique fonctionnent à 266 Mbit/s et utilisent le codage 8/10 bit de la technologie Fibre Channel. Le groupement des fragments d'événements utilise un bus de fond de panier spécifique basé sur des composants utilisés pour FutureBus+. Un réseau additionnel de type Token Ring relie l'ensemble des cartes du châssis. Le canal de sortie est adapté au standard HIPPI (*High Performance Parallel Interface*) et offre un lien à 100 Mo/s. Ce lien de sortie est relié à un commutateur HIPPI à 4 entrées et 4 sorties permettant la distribution des événements à 4 stations de travail effectuant le stockage temporaire des données. Les 4 stations de travail sont reliées: 1) au commutateur HIPPI par une carte développée à cet usage, 2) à un réseau Ethernet utilisé pour la configuration et la surveillance du système, 3) à un concentrateur FDDI (*Fiber Distributed Data Interface* [Sha93]). Les données sont acheminées par ce concentrateur FDDI via un lien HIPPI série puis un second concentrateur FDDI jusqu'à la base centrale de traitement et de stockage des données du CERN située à 2 km du site expérimental. Les liens FDDI du second concentrateur sont connectés à une machine parallèle Meiko CS-2 comportant 128 processeurs effectuant le traitement des données puis le stockage sur bande magnétique (périphériques de stockage connectés par un autre groupe de liaisons FDDI). Bien que relativement

complexe et hétérogène, ce système est opérationnel et satisfaisant [Gia97].

Les triggers de niveau 2 et 3 de l'expérience HERA-B font appel à des processeurs de traitement de signal (*Digital Signal Processors* – DSPs) disposant de liens de communication [Dam97]. Environ 140 cartes comportant 6 processeurs SHARC (Analog Devices) sont prévues pour construire le réseau de communication reliant les sources de données des détecteurs aux ~150 PCs effectuant les traitements. Chaque DSP dispose de 6 liens pouvant fonctionner jusqu'à 40 Mo/s (format et protocole propriétaire). Le débit utile global à atteindre pour ce réseau est de ~200 Mo/s. Les processeurs de traitement sont reliés à un quatrième niveau de trigger par des liaisons de type Fast Ethernet. Le besoin en bande passante est plus modeste à ce niveau (~5 Mo/s). Ce système est en cours de construction et de mise au point. L'ensemble est assez homogène et possède l'originalité d'utiliser un réseau de communication réalisé à l'aide de DSPs.

Le réseau du trigger de niveau 2 de l'expérience L3 est basé sur des commutateurs de type C104 [SGS94] (voir section 5.4.2) connectés à ~20 Transputers[1] T9000 [Blai97]. Ce système est opérationnel et satisfaisant, mais la technologie des Transputers est aujourd'hui dépassée.

D'autres expériences utilisent pour leur système de sélection et d'acquisition de données des technologies plus répandues telle Fibre Channel dans Euroball [Bar96], Fast Ethernet dans Babar [Pav98], et ATM dans CDF Run 2 [Fro98].

5.4. Technologies de réseau envisagées pour ATLAS

Malgré le très large éventail de technologies disponibles sur le marché et compte tenu des contraintes et de la spécificité de l'application envisagée, la démonstration de faisabilité des réseaux pour les systèmes d'acquisition de données des expériences au futur LHC n'est pas une tâche facile. Afin de mieux sélectionner les technologies susceptibles de répondre aux besoins de ces expériences, le *Detector Research and Development Committee* (DRDC au CERN) a approuvé deux projets de recherche consacrés à l'étude de technologies de réseau en vue de leur application aux expériences du LHC. Les projets RD-24 "*Application of the Scalable Coherent Interface to Data Acquisition at LHC*" [Bog96] et RD-31 "*NEBULAS: a high performance data-driven event building architecture based on asynchronous self-routing packet switching network*" [Cos95] ont été mis en place en 1992 et se sont achevés environ trois ans plus tard. En plus de ces projets du DRDC, d'autres études ont été faites dans des cadres de recherche différents. Après la fin des projets RD-24 et RD-31, les travaux ont été poursuivis au niveau des collaborations de chaque expérience du LHC et non plus par des projets de recherche communs.

Les technologies de réseau envisagées pour ATLAS sont présentées dans les sections suivantes.

5.4.1. La technologie ATM

5.4.1.1. Historique

La technique d'*Asynchronous Time Division Multiplexing* (ATDM – multiplexage temporel asynchrone) tire ses origines des travaux des chercheurs de Bell Labs dans les années 60. Trop difficile à réaliser avec la technologie de l'époque, la technique ne sera réellement développée qu'au

1. processeur de conception originale dont le nom est formé par l'assemblage de TRANSistor et comPUTER commercialisé par la société INMOS dans les années 80.

début des années 80 par le Centre National d'Etude des Télécommunications (CNET) de Lannion afin d'offrir un moyen de transport pour la télévision par câble ainsi qu'une méthode unifiée pour le transport de la voix et d'autres types de trafic. Initialement désignée par l'acronyme ATD (*Asynchronous Time Division*), celui-ci deviendra ATM (*Asynchronous Transfer Mode* – mode de transfert asynchrone). Cette technologie a été adoptée en 1986 par le Comité Consultatif International Télégraphique et Téléphonique (CCITT) comme le mode de transport à utiliser pour le *Broadband Integrated Services Digital Network* (B-ISDN, Réseau Numérique à Intégration de Services – RNIS large bande). De nombreux travaux du point de vue théorique, liés à la normalisation de l'ATM et au développement de produits ont eu lieu par la suite. Parmi les dizaines d'ouvrages consacrés à l'ATM, on pourra se référer à [Cut93] et [McD94].

5.4.1.2. Principe

ATM est une technologie de commutation par paquets qui repose sur le multiplexage statistique temporel asynchrone de blocs d'information de taille fixe appelés "cellules" sur un support physique commun. Avec les techniques de transmission synchrone (ou plésiochrone) utilisées en téléphonie, un intervalle de temps fixe et récurrent est alloué de bout-en-bout à travers le réseau pour la communication entre deux abonnés. L'avantage pour l'utilisateur est de disposer d'un canal de bande passante fixe et garantie (64 kbit/s pour une conversation téléphonique classique). Cette approche est mal adaptée à la transmission d'information en provenance de sources ayant des débits très variables dans la mesure où les ressources affectées à une communication le sont pour toute la durée de celle-ci, que des informations soient transmises ou non. A l'inverse, le multiplexage statistique de différents types de trafic sur un même support permet d'utiliser au mieux les ressources disponibles: la bande passante peut être fournie à la demande. Les sources à débit constant peuvent obtenir une bande passante fixe; les sources à débit variable peuvent bénéficier temporairement d'un surplus de bande passante tout en n'utilisant pas de ressources pendant les périodes d'inactivité. Les sources les moins exigeantes peuvent utiliser la bande passante disponible, selon la capacité et la charge instantanée du réseau.

ATM est une technologie "orientée connexion" (*connection oriented*): l'échange d'information entre deux points du réseau ne peut se faire qu'après une procédure préalable d'établissement de la connexion. Au cours de cette procédure sont négociées les caractéristiques de la communication (bande passante et qualité de service), et un "chemin virtuel" est établi de bout-en-bout à travers le réseau. La transmission d'information peut alors avoir lieu et dure jusqu'à ce que la procédure de fermeture de la connexion soit invoquée. Le chemin virtuel est alors détruit et les ressources correspondantes sont libérées.

Les cellules ATM sont les unités d'information élémentaires véhiculées par le réseau. Elles comportent deux parties: un en-tête destiné à recevoir les informations nécessaires au routage suivi de la charge utile de la cellule comportant les données à transmettre. Afin de faciliter l'implantation matérielle des composants ATM dans des circuits intégrés et réduire divers problèmes (délai, gigue, écho), le choix s'est porté sur des cellules courtes de taille fixe: 5 octets pour l'en-tête afin de pouvoir distinguer au moins 1 million de connections différentes et stocker d'autres informations, 48 octets pour la charge utile suite à un compromis entre la valeur proposée par les Européens (32 octets) et celle défendue par les Américains (64 octets). Le format de la cellule ATM est présenté Figure 5-1. Les champs sont les suivants:

- GFC[1]: *Generic Flow Control* – contrôle de flux générique.
- VPI / VCI: *Virtual Path Identifier, Virtual Channel Identifier* – ce champ permet d'identifier de manière unique en un point du réseau la connexion virtuelle à laquelle appartient la cellule. Cette

information permet le routage de la cellule dans les commutateurs situés aux nœuds du réseau. Le label VPI / VCI est une information valide localement: pour une connexion virtuelle donnée, les cellules qui entrent dans un commutateur avec un label X peuvent être renommées pour ressortir avec un label Y différent ou non de celui d'origine.

- PT: *Payload Type* – type de la charge utile de la cellule. Il peut s'agir de données utilisateur, d'information de signalisation, de maintenance...
- CLP: *Cell Loss Priority* – si ce drapeau est activé, les éléments du réseau sont autorisés à détruire la cellule en cas de besoin, lorsque les ressources disponibles ne permettent pas son acheminement normal.
- HEC: *Header Error Control* – code de protection contre les erreurs. Ce champ donne la possibilité de corriger 1 bit erroné dans l'en-tête de la cellule et de détecter les erreurs sur plusieurs bits.

Les 48 octets, constituant la charge utile de la cellule, suivent l'en-tête.

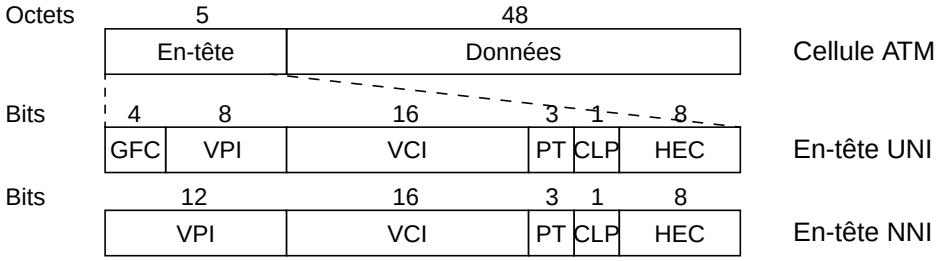


Figure 5-1: Format de la cellule ATM.

Afin de transmettre des messages dont la taille dépasse celle d'une cellule, une couche d'adaptation est nécessaire entre l'application et la couche ATM, il s'agit de l'*ATM Adaptation Layer* (AAL). Plusieurs AAL sont définis:

AAL 1: transport isochrone, débit constant (voix, vidéo non compressée...),

AAL 2: transport isochrone, débit variable (audio ou vidéo compressée),

AAL 3/4: pas de relation de synchronisation entre émetteur et récepteur, débit variable,

AAL 5: version simplifiée du précédent.

La couche AAL 5 est la plus simple et la plus utilisée, en particulier pour le transport de données informatiques. Elle permet le transport de messages d'une taille maximale de 64 ko. Un message à transmettre est segmenté en fragments de 48 octets véhiculés par des cellules ATM. La taille du message est ajustée au multiple de 48 octets supérieur et la dernière cellule est marquée par une valeur spécifique dans le champ PTI pour indiquer la fin du message. La dernière cellule, contient un champ indiquant la taille effective du message et un mot de CRC 32 bit (*Cyclic Redundancy Check*) destiné à la détection/correction des erreurs sur les données de l'ensemble du paquet. Les cellules sont transmises au rythme fixé par les caractéristiques de la connexion virtuelle. A la réception, le contenu des cellules d'une même connexion virtuelle est ré-assemblé pour restituer le message d'origine. En fin de message, le CRC calculé localement est comparé à celui calculé par l'émetteur afin de détecter d'éventuelles erreurs de transmission ou pertes de cellules. La couche ATM ne garantit pas le transport des cellules de bout-en-bout; si des cellules ont été détruites, les actions nécessaires (demande de re-transmission du paquet ou perte de l'information) doivent être

1. Ce champ est spécifié seulement au niveau de l'interface utilisateur au réseau (*User Network Interface* – UNI) et est utilisé pour porter à 12 bits le champ VPI dans le cas d'une interface entre réseaux (*Network to Network Interface* – NNI).

prises en charge par les couches supérieures. La fonction de segmentation et ré-assemblage (*Segmentation And Re-assembly* – SAR) est le plus souvent réalisée à l'aide de circuits spécifiques disponibles chez de nombreux vendeurs de semiconducteurs: NicStar™ de Integrated Device Technology [IDT97], SARA de TransSwitch [Tra92], Lasar de PMC-Sierra [PMC97]...

Les cellules ATM ne sont pas transportées directement sur le support physique, mais sont le plus souvent encapsulées dans des trames SDH/SONET (*Synchronous Digital Hierarchy* – hiérarchie numérique synchrone, *Synchronous Optical NETwork* – réseau optique synchrone). SONET est la technologie de transport de l'information utilisée dans les réseaux de télécommunication aux Etats-Unis; SDH est la version utilisée en Europe. Sous la pression de l'Américain Bellcore, inventeur de SONET dans les années 80, et compte tenu des difficultés pour basculer en une seule étape l'ensemble des réseaux publics depuis l'ancien système PDH (*Plesiochronous Digital Hierarchy* – hiérarchie numérique plésiochrone) vers l'ATM, le CCITT a adopté l'ATM comme solution de commutation et SDH/SONET comme technologie de transport de l'information. De nombreuses valeurs de débit sont normalisées pour le transport de cellules ATM: 1,544 Mbit/s; 2,048 Mbit/s; 25 Mbit/s (encodage 4/5 bit); 34,368 Mbit/s, 51,84 Mbit/s; 100 Mbit/s (basé sur FDDI); 155,52 Mbit/s; 622,08 Mbit/s, etc... La valeur la plus courante pour les matériels utilisés dans les réseaux locaux est 155,52 Mbit/s. Elle est identifiée par l'acronyme OC3-C (*Optical Carrier 3 Concatenated*) suivant la terminologie utilisée pour SONET et STM-1 (*Synchronous Transport Module 1*) suivant celle utilisée pour SDH.

5.4.1.3. Intérêt pour l'application envisagée

Destinée à l'origine aux télécommunications, la technologie ATM fut pressentie au début des années 90 comme une solution également intéressante pour les LANs, en remplacement de FDDI. La collaboration RD-31 a suivi cette tendance, et l'étude de l'ATM en vue des systèmes d'acquisition de données des expériences LHC a débuté en 1992. L'apparition de Fast Ethernet puis Gigabit Ethernet a aujourd'hui beaucoup réduit les perspectives de mise en place généralisée de l'ATM au niveau des postes de travail et dans les LANs, mais cette technologie conserve de nombreux atouts pour l'application envisagée:

- notion de qualité de service, connexions point-à-multi-points, bande passante à la demande,
- large gamme de produits et de commutateurs de différentes capacités,
- support industriel important et utilisation dans le domaine des télécommunications garantissant la pérennité de la technologie sur le long terme,...

L'utilisation de la technologie ATM en vue du T/DAQ de l'expérience ATLAS est rapportée dans cette thèse. Ses avantages et ses inconvénients pour cette application, les performances et les facteurs de limitation identifiés seront présentés en détail dans les chapitres suivants.

5.4.2. La technologie DS-Link

5.4.2.1. Historique

La technologie DS-Link (*Data Strobe Link*) a été développée à l'origine par la société INMOS (puis par SGS-Thomson après le rachat de cette société) pour interconnecter des Transputers. Le nom DS-Link vient de l'encodage sur deux liens nommés D (*Data*) et S (*Strobe*) utilisé pour la transmission des signaux. Ce codage astucieux permet de reconstruire le signal d'horloge par une simple combinaison logique des signaux D et S. La synchronisation au niveau du récepteur est donc particulièrement simple à réaliser et économique (pas de boucle à verrouillage de phase, large gamme de fréquence de fonctionnement, ajustement automatique du récepteur à la

fréquence de l'émetteur). En raison de ses qualités, cet encodage a été repris pour la technologie "FireWire" (norme IEEE-1394) destinée au marché de l'électronique grand public (liaison entre PCs et périphériques vidéo, caméscopes, etc).

Englobée par la norme IEEE-1355 [IEE95], la technologie DS-Link permet de réaliser des interconnexions à très bas prix entre micro-processeurs, avec un débit de base de 100 Mbit/s. D'autres versions fonctionnant à 200 Mbit/s et 400 Mbit/s sont disponibles ou prévues, ainsi que la version HS-Link fonctionnant à 1 Gbit/s.

5.4.2.2. Principe

DS-link est une technologie de transmission par paquets. Chaque paquet comporte un en-tête (de taille non imposée) destiné au routage, une charge utile de taille fixe ou variable suivie d'une indication de fin de paquet. La norme IEEE-1355 laisse beaucoup de liberté concernant la taille et le format des paquets: les composants sont à usage général, et peuvent véhiculer des cellules ATM, des paquets Fibre Channel, des trames Ethernet, Token Ring, ou des paquets d'un format propriétaire. Un des éléments importants de la technologie DS-Link est le composant ST C-104 [SGS94]. Ce circuit est un commutateur 32 ports bi-directionnels 100 Mbit/s d'une grande flexibilité: réseau de type *crossbar* non bloquant, algorithme de routage minimalisant le temps de transit (*wormhole routing*), contrôle de flux par matériel, possibilité de supprimer 0, 1 ou 2 octets d'en-tête à la traversée des paquets pour le routage dans un réseau multi-étage, routage en deux phases (aiguillage aléatoire puis acheminement effectif) afin d'éviter les éventuels points chauds,...

5.4.2.3. Intérêt pour l'application envisagée

Beaucoup d'études sur la technologie DS-Link ont eu lieu dans le cadre de projets financés par la communauté européenne (ESPRIT[1] *Open Microprocessor Initiative*), par exemple les projets ARCHES et MACRAME[2] ayant pour but d'acquérir la maîtrise de cette nouvelle technologie et de démontrer par une réalisation pratique l'ensemble de ses qualités. Un commutateur à 1024 ports construit à base de ST C-104 a été construit [Haa97]. L'utilisation de ce commutateur comme système de démonstration pour le trigger de niveau 2 d'ATLAS a été rapportée dans [Dem98]. Un des problèmes qui demeure à résoudre concerne la réalisation de connexions du type point-à-multi-points qui ne sont pas prévues dans le circuit ST C-104. L'implémentation du trigger de niveau 2 avec la technologie DS-Link devrait probablement faire appel à un réseau supplémentaire pour les diffusions de messages. Ce réseau additionnel demeure à définir et à valider.

La démonstration de faisabilité d'un commutateur de grande capacité basé sur la technologie DS-Link a été un succès, mais l'arrêt de la fabrication des Transputers et l'obsolescence du circuit ST C-104 semble dès aujourd'hui compromettre l'utilisation de cette technologie pour ATLAS. La version HS-Link pourrait connaître davantage de succès, mais cela reste incertain. Je pense que cette technologie est mal adaptée à ATLAS dans la mesure où peu de produits sont disponibles sur le marché et qu'il s'agit davantage de composants électroniques plutôt que de blocs complets (commutateurs, cartes réseau et logiciel associé). Malgré un coût attractif, en apparence, et des performances convenables, une solution basée sur les DS/HS-Link me semble nécessiter un effort trop important de développement de matériel et de logiciel et comporter des risques sur le long terme concernant la pérennité de cette technologie.

1. European Strategic Program for Research and development in Information Technology
2. Multi-processor Architectures: Connectivity, Routers And Modelling Environment

5.4.3. La technologie Ethernet

5.4.3.1. Historique

Inventée en 1973 par R. Metcalfe et D. Boggs des laboratoires Xerox à Palo Alto, puis présentée au public en 1976, la technologie Ethernet a été développée par Xerox, Intel et Digital et rendue compatible avec la norme relative aux réseaux locaux établie en 1980 par le groupe de travail 802 du comité IEEE. La norme IEEE 802.3 spécifie la méthode d'accès au médium (MAC – *Media Access Control*) par CSMA/CD (*Carrier Sense Media Access with Collision Detection* – accès multiple avec écoute de porteuse et détection de collisions) utilisée dans la version initiale d'Ethernet. Des sous-groupes du projet 802 ont spécifié d'autres méthodes d'accès au médium: 802.4 pour le Token Bus (bus à jeton), 802.5 pour le Token Ring (anneau à jeton) et FDDI, 802.6 pour le DQDB MAN (*Distributed Queue Dual Bus Metropolitan Area Network*)...

La version de base d'Ethernet offre un débit de 10 Mbit/s sur un médium partagé entre tous les utilisateurs connectés sur le même brin. Grâce au succès de cette technologie, à la demande croissante des utilisateurs et au dynamisme des industriels impliqués, de nombreuses améliorations ont été apportées au fil des ans: Ethernet commuté, liaisons *full-duplex*, passage à 100 Mbit/s (Fast Ethernet) en préservant la compatibilité avec l'ancienne génération. La version finale de la norme Fast Ethernet (100 Mbit/s) a été ratifiée par l'IEEE en Juin 1995 (spécifications 802.3u); le mode *full-duplex* et le contrôle des flux sont décrits par les spécifications 802.3x.

Par des prix très attractifs et en raison de la compatibilité totale avec les infrastructures en place, la seconde génération d'Ethernet connaît un vif succès. Cela confirme la position dominante de cette technologie dans le domaine des réseaux locaux au détriment des technologies concurrentes, Token Ring (peu attractive aujourd'hui), FDDI (trop chère) et ATM (plus complexe). De nombreux constructeurs d'ordinateurs intègrent aujourd'hui une interface Ethernet sur la carte mère de leurs machines pour un surcoût dérisoire. Néanmoins, pour la réalisation d'épines dorsales de réseaux d'entreprises, de campus ou dans les réseaux métropolitains, Ethernet est relativement moins présent (en général il s'agit d'infrastructures FDDI et plus récemment d'ATM). Cette situation évolue avec l'essor de la troisième génération d'Ethernet, Gigabit Ethernet normalisée par l'IEEE en Juillet 1998 (norme 802.3z).

Un ouvrage de référence sur Ethernet et ses évolutions est [Kad98].

5.4.3.2. Principe

Ethernet est une technologie de commutation par paquets de taille variable appelés "trames". Le format d'une trame Ethernet est présenté Figure 5-2.

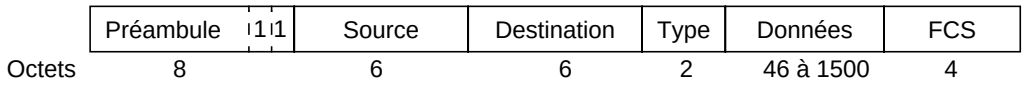


Figure 5-2: Format d'une trame Ethernet.

La trame comporte:

- un préambule composé d'une alternance de "1" et "0" destiné à la synchronisation,
- l'adresse de destination de la trame,
- l'adresse de la source,
- la taille de la trame,
- le *Frame Check Sequence* – séquence de vérification de trame – destiné à la détection/correction d'erreurs.

Avec la méthode CSMA/CD pour l'accès au médium partagé, l'émission d'une trame se fait comme suit: l'émetteur "écoute" la porteuse sur le médium; lorsque une période d'inactivité nommée *Inter-Packet Gap* (gap entre paquets fixé à 9,6 μs) s'est écoulée, la transmission peut débuter avec l'émission du préambule de synchronisation puis du reste de la trame. Si plusieurs stations étaient en attente, il est possible qu'elles commencent à émettre approximativement en même temps: il se produit alors une collision. Les émetteurs doivent stopper l'émission, transmettre une séquence "*jam*" pour marquer la collision, observer un temps d'arrêt d'une durée pseudo-aléatoire (fixée suivant l'algorithme "*truncated binary exponential backoff*" décrit dans la norme IEEE 802.3) avant de tenter à nouveau la transmission de la trame ayant subi la collision. Une station peut tenter jusqu'à 16 fois de transmettre une trame; en cas d'échec, la trame est perdue et l'erreur est signalée au niveau supérieur.

La seconde génération d'Ethernet (Fast Ethernet) conserve la couche d'accès au médium par la méthode CSMA/CD, mais dans le cas du fonctionnement *full-duplex*, cette fonction est désactivée. Pour l'Ethernet commuté, un mécanisme de contrôle de flux a également été prévu: les trames *PAUSE* commandent à une station d'arrêter son émission pendant le temps indiqué (valeur codée sur 16 bits avec pour unité de base le temps de transmission de 512 bits, soit une durée de pause variable de 5,12 μs à 335 ms). La norme IEEE 802.x laisse une certaine liberté concernant la génération des trames PAUSE; le support de cette fonctionnalité dans les équipements est optionnel.

Pour la troisième génération d'Ethernet (Gigabit Ethernet), l'utilisation du même mécanisme d'accès au médium que pour les générations antérieures poserait des problèmes pour détecter les collisions compte tenu du temps de propagation des signaux et de la durée d'émission des trames les plus courtes. La solution retenue par le groupe de travail IEEE 803.z a été de porter à 512 octets la taille minimale d'une trame (ajout d'octets de bourrage). Cette solution est pénalisante pour la transmission de trames courtes; un mécanisme d'envoi de trames "en rafale" (*frame bursting*) permet d'améliorer la situation. La version de Gigabit Ethernet fonctionnant sur un médium partagé sera probablement peu utilisée, et c'est plutôt la version *full-duplex* qui devrait être mise en place par les utilisateurs. Dans ce mode de fonctionnement, il s'agit uniquement de liaisons point-à-point (topologie en étoile utilisant des commutateurs). La couche MAC par CSMA/CD est désactivée. Il ne se produit plus de collisions car le médium n'est pas partagé entre plusieurs stations; les éventuels conflits (présence simultanée dans un commutateur de plusieurs trames destinées à un même port de sortie) sont gérés au niveau des commutateurs à l'aide de mémoires tampon, et éventuellement par la génération de trames *PAUSE*.

5.4.3.3. Intérêt pour l'application envisagée

La technologie Ethernet est la plus répandue dans les réseaux locaux. Les équipements disponibles dans le commerce sont très nombreux et les prix particulièrement attractifs. Compte tenu de la base installée et des évolutions récentes permettant de multiplier par 100 le débit par rapport à la version de base, l'avenir d'Ethernet est assuré. Il paraît donc naturel de considérer cette technologie comme étant l'une des plus prometteuses pour la réalisation des réseaux de communication des T/DAQ des futures expériences au LHC. Je partage cette opinion bien qu'un certain nombre de points soient à étudier pour démontrer la faisabilité de ces systèmes avec Ethernet: existence d'équipements de commutation de capacité suffisante, performances réelles de ces éléments, efficacité des mécanismes de gestion des flux, comportement en cas de surcharge, disponibilité d'une qualité de service adaptée à notre application, mise en place des communications point-à-multi-points, utilisation avec ou sans protocole réseau standard (TCP/IP)... Tout comme lors des investigations sur l'ATM, l'ensemble de ces points doit être abordé. Un groupe de travail à l'intérieur de la collaboration ATLAS mène l'étude relative à Fast et Gigabit Ethernet depuis 1997

[Dob98]. Les progrès et les résultats sont très attendus.

5.4.4. La technologie Fibre Channel

5.4.4.1. Historique

Faisant suite à la technologie HIPPI, le développement de Fibre Channel a débuté en 1988 et la norme ANSI correspondante a été finalisée en 1994 [X3T93]. Cette technologie est principalement destinée à la connexion entre ordinateurs et périphériques de stockage même si des applications dans d'autres domaines sont envisageables (épine dorsale d'un réseau local par exemple). Un ouvrage de référence sur la technologie Fibre Channel est [Fib94].

5.4.4.2. Principe

Un réseau Fibre Channel est, soit un réseau en étoile composé d'éléments reliés à un commutateur central, soit une boucle (*Fibre Channel Arbitrated Loop*) comportant plusieurs unités en série. Les deux topologies peuvent être combinées en reliant à un port de commutateur une boucle comportant plusieurs machines ou périphériques.

Le modèle en couches adopté pour les spécifications de Fibre Channel comporte 4 niveaux:

- la couche FC-0 définit la couche physique (fibre optique, câble électrique, connecteurs...) et les valeurs de débit des liens (133 Mbit/s, 266 Mbit/s, 530 Mbit/s et 1 Gbit/s).
- la couche FC-1 définit le protocole d'encodage et de décodage des caractères et des mots de contrôle. Chaque caractère de 8 bits est encodé en un mot de 10 bits. Ce type d'encodage 8/10 bit est également utilisé pour Gigabit Ethernet.
- la couche FC-2 définit la structure des trames échangées, le mécanisme de signalisation et le protocole de transport des trames. La taille des trames est variable, de 36 à 2148 octets (2112 octets de charge utile). Plusieurs classes de transport sont définies. La classe 1 recouvre les communications en mode connecté (équivalentes à des connexions physiques une fois établies). Un accusé de réception est retourné pour chaque trame reçue; la préservation de l'ordre des trames entre un émetteur et son récepteur est garantie. La classe 2 définit l'échange de datagrammes (communications en mode non connecté). Comme pour la classe 1, un accusé de réception est retourné pour chaque trame reçue. L'ordre des trames à la réception peut être différent de celui de l'émission. La classe 3 est identique à la classe 2, mais ne comporte pas d'accusés de réception (mode de transfert non connecté et non sécurisé). La classe 4 définit un mode de fonctionnement basé sur des circuits virtuels avec possibilité d'allocation de bande passante. La classe 6 (la classe 5 n'est pas clairement définie) traite des transferts uni-directionnels et point-à-multi-points, par exemple en vue de la diffusion de séquences vidéo.
- la couche FC-3 définit des fonctionnalités avancées telles les communications multi-points.
- la couche FC-4 définit l'interface avec les applications, par exemple l'encapsulation dans des trames Fibre Channel de paquets HIPPI, de datagrammes IP, de paquets ATM AAL 5...

5.4.4.3. Intérêt pour l'application envisagée

La technologie Fibre Channel a paru séduisante pendant quelques années: large gamme de produits disponibles, débit de 1 Gbit/s pour un coût modéré, nombreuses applications possibles dans le domaine du multi-média (serveurs vidéo)... Aujourd'hui l'enthousiasme est un peu tombé, en partie avec l'arrivée sur le marché de Gigabit Ethernet qui offre des services à peu près équivalents en plus de la compatibilité avec les réseaux de la génération précédente. La technologie Fibre Channel connaît un relatif succès pour les liaisons entre ordinateurs et périphériques de stockage

(marché visé à l'origine). Dans la collaboration ATLAS, les études sur cette technologie continuent principalement en vue de la réalisation du système de reconstruction d'événements (dans le modèle d'architecture comportant des réseaux de communication séparés pour le trigger et le DAQ).

Pour ma part, je pense que la technologie Fibre Channel est mal adaptée au T/DAQ de l'expérience ATLAS. Les commutateurs disponibles sur le marché disposent le plus souvent d'un faible nombre de ports (de l'ordre de 32 au maximum); le fonctionnement en cascade de commutateurs reste à valider. Les produits sont très différents d'un constructeur à l'autre, certains n'intègrent qu'une partie des classes de service normalisées. La technologie Fibre Channel est optimisée pour les transferts de larges blocs de données; pour les messages courts, la taille relativement importante des en-têtes et les temps de commutation dégradent les performances de manière inacceptable [Let97]. Du point de vue commercial, Fibre Channel demeurera probablement dans la niche du marché des périphériques de stockage et ne me semble pas en mesure de concurrencer Gigabit Ethernet et ATM dans leurs domaines respectifs. De ce fait le dynamisme industriel et commercial relatif à la technologie Fibre Channel ne me semble pas suffisant pour faire évoluer les produits de manière satisfaisante pour notre application.

5.4.5. La technologie SCI

5.4.5.1. Historique

Les études sur le *Scalable Coherent Interface* (SCI) ont débuté en 1988 lors du projet de normalisation du bus Futurebus (norme IEEE P896) quand certains membres de ce groupe de travail ont remis en cause la notion même de bus pour l'interconnexion de processeurs en vue de la réalisation des machines parallèles à hautes performances du futur. Le but du SCI est de fournir un moyen d'interconnexion entre processeurs, mémoires et périphériques offrant les avantages d'un bus (très haut débit, faible temps d'accès) sans ses inconvénients (limitation du nombre d'éléments connectés, dimensions mécaniques) [Gus92]. Après de nombreuses contributions académiques et venant des industriels, la norme ANSI/IEEE-1596 relative au SCI a été finalisée en 1992 [IEE92]. La technologie SCI est aujourd'hui au coeur de certains modèles de machines multi-processeurs chez différents constructeurs (par exemple Data General, Sequent, Hewlett Packard...). L'utilisation de cette technologie est également envisagée pour les systèmes d'acquisition de données [Gus90].

5.4.5.2. Principe

Le SCI offre des fonctionnalités comparable à celle d'un bus d'interconnexion avec l'avantage de pouvoir croître en taille pour réaliser des systèmes à grande échelle (jusqu'à 65520 nœuds connectés) sans saturation des performances. La topologie de base est un anneau où chaque nœud est relié à son voisin de droite et de gauche (liaison uni-directionnelle). Des topologies plus complexes peuvent être formées en connectant plusieurs anneaux au moyen de commutateurs. Au niveau de la couche physique des liens, il existe plusieurs versions offrant des débits de 200 Mo/s, 500 Mo/s et 1 Go/s sur câble parallèle, et 1 Gbit/s sur fibre optique série. Le but du SCI étant d'offrir une bande passante comparable à celle des bus classiques, ce choix de débits élevés est justifié (cependant la bande passante est partagée entre tous les nœuds connectés à un même anneau).

Les entités échangées par les différents nœuds dans un réseau SCI sont des paquets de taille variable comportant un en-tête de taille fixe (16 octets) utilisé pour l'acheminement du message et une charge utile (commande ou données) de 0, 16, 64 ou 256 octets. L'interface SCI de chaque nœud connecté à un anneau fait suivre les paquets qui ne lui sont pas destinés, retire et traite ceux qui la concerne, et insère dans l'anneau les paquets qu'elle émet.

Une des caractéristiques fondamentales et particulièrement attrayante du SCI est d'offrir un mécanisme de mémoire partagée semblable à celui réalisable avec un bus classique. A chaque nœud SCI est affectée une adresse sur 16 bits (unique dans un même réseau) permettant de l'identifier. Par l'intermédiaire de son interface SCI, chaque nœud rend visible pour les autres nœuds une plage mémoire de taille variable (adresses codées sur 48 bits). L'échange d'information entre nœuds se fait ou moyens d'actions élémentaires appelées "transactions". Les transactions permettent de lire, écrire ou déplacer des blocs de 16, 64 ou 256 octets situés dans les zones de mémoire partagée de chacun des nœuds. L'initiateur d'une transaction émet une requête comportant sa propre adresse, celle du nœud cible, l'adresse concernée dans la zone de mémoire partagée du nœud cible, le code de l'opération à effectuer et un éventuel bloc de données (dans le cas d'une demande d'écriture). Le nœud cible retourne dans un premier temps un accusé de réception de la requête, puis un paquet contenant la réponse (dans le cas d'une requête de lecture de données). L'initiateur retourne un accusé de réception de la réponse venant du nœud cible. Ces opérations sont effectuées par matériel de manière extrêmement rapide de sorte que le temps d'accès à des données situées dans la mémoire d'un nœud distant est en général de l'ordre de quelques micro-secondes.

Une deuxième caractéristique importante du SCI est d'offrir la possibilité d'effectuer des transactions en conservant la cohérence des anté-mémoires (*caches*) des différents nœuds (la deuxième lettre de l'acronyme SCI signifie "*Coherent*"). La cohérence de mémoires dans un système multi-processeurs à mémoire partagée est un aspect fondamental pour le bon fonctionnement de ce type de machine. Une bonne introduction à ce problème et une revue de différents mécanismes de gestion de cohérence de mémoire cache sont présentées dans [Arc86]. Le mécanisme de cohérence de mémoire cache est optionnel en SCI. Un troisième mode de fonctionnement possible pour le SCI, est l'utilisation en tant que simple moyen de transport de messages (système à échange de messages et non plus modèle à mémoire partagée; les deux modèles pouvant en principe co-exister).

5.4.5.3. Intérêt pour l'application envisagée

La technologie SCI offre un concept particulièrement séduisant alliant la performance et la simplicité d'un bus d'interconnexion à la flexibilité et l'extensibilité des réseaux. Le modèle de mémoires partagées réparties avec possibilité de gestion automatique de cohérence de cache et le modèle simplifié à échange de message permettent une grande flexibilité dans la conception des systèmes utilisant le SCI. Proposant une évolution logique pour les systèmes d'acquisition de données composés jusqu'alors principalement de bus interconnectés en structure d'arbre (par exemple le Fastbus [Fas83]), le SCI a suscité un très vif intérêt dans la communauté de physique des hautes énergies et particulièrement en vue de la réalisation des systèmes d'acquisition de données des expériences au LHC [Div92]. Un important effort dans l'étude du SCI a été accompli par la collaboration RD-24 et de nombreux résultats ont été obtenus [Bog96], [Ska97]. Pour l'expérience ATLAS, les investigations sur le SCI se poursuivent dans une partie du groupe en charge du trigger de niveau 2. Ces études et les résultats obtenus sont disponibles dans [Bog97] et [Hug98].

Malgré les caractéristiques séduisante de la technologie SCI, je pense que ses chances de succès dans le domaine de l'acquisition de données, et en particulier pour l'expérience ATLAS, sont relativement faibles. Le besoin de cohérence de cache ne me semble pas clairement établi dans le cas de l'application envisagée: l'acquisition de données est la lecture permanente de données fraîches, sans opération d'écriture ni de modification de ces données dans leur emplacement d'origine.

Le modèle de trigger basé sur un paradigme de mémoires partagées réparties est certes très séduisant du point de vue conceptuel: chaque processeur effectue ses traitements sur des événements comme si les données correspondantes étaient dans sa propre mémoire; en cas d'absence des

données (*cache miss*), l'interface SCI du processeur considéré effectue le chargement automatique de la ligne de cache manquante en lisant les données correspondantes dans le nœud distant qui les contient. Pour les processeurs, les algorithmes de traitement des événements s'exécutent de manière totalement transparente et indépendante de la localisation des données. De plus, seules les données réellement utilisées sont transmises à travers le réseau, ce qui permet de ne pas gaspiller de la bande passante. En pratique, ce modèle de système à mémoires partagées réparties me semble assez difficile à réaliser. Par exemple le mécanisme de translation automatique d'adresses afin que les accès des processeurs de traitement puissent se faire sur la base de numéros d'événements dans les mémoires contenant les données en provenance des détecteurs me semble délicat à réaliser.

Un autre problème est le temps d'accès aux données distantes. Pendant le temps de chargement des lignes de cache non présentes dans un processeur, celui-ci risque de manquer rapidement d'instructions exécutables: cela force un processeur à générer des cycles d'attente qui réduisent l'efficacité de ses unités de traitement. Les processeurs modernes (500 MHz soit 2 ns par cycle d'instruction) sont capables de fonctionner sans perte de performance lorsque les temps d'accès à des données situées en dehors de leur mémoire cache sont de l'ordre de quelques centaines de nano-secondes (anticipation de la lecture des lignes de cache, exécution d'instructions dans le désordre puis remise en séquence...). Dans un système de taille modeste, les temps d'accès à des données distantes à travers un anneau SCI peuvent être de l'ordre de la micro-seconde, ce qui semble acceptable, mais dans un système à l'échelle du T/DAQ d'ATLAS comportant plusieurs centaines de nœuds, préserver des temps d'accès aussi faibles me semble difficile. Une solution possible est de modifier l'application (éventuellement de manière automatique à la compilation) pour prendre en compte ces temps d'accès plus long que la normale et commander le pré-chargement des données suffisamment tôt. Cette voie pourrait être à explorer et à valider mais me semble demander un effort important pour des résultats incertains. Un système basé sur des échanges de messages est plus classique et plus simple à réaliser. Il peut l'être avec des technologies de réseau classiques ce qui laisse une marge de manœuvre pour la réalisation technique. A l'inverse, la gestion automatique de mémoires partagées (avec éventuellement cohérence des caches) est une caractéristique relativement unique de la technologie SCI. Le fonctionnement du SCI en mode passage de messages est possible, mais n'apporte pas d'avantage majeur par rapport à d'autres technologies.

Une autre difficulté est la réalisation d'un réseau de grande dimension. La topologie en anneau est limitée en taille: saturation de l'anneau (voir chapitre suivant), problème de robustesse. Un réseau comportant près d'un millier de nœuds doit faire appel à des commutateurs SCI permettant l'interconnexion d'anneaux SCI. Peu de commutateurs sont disponibles sur le marché, et ces derniers comportent un faible nombre de ports (4 à 16 au maximum) ce qui impose de les assembler pour disposer d'un nombre de ports plus élevé [Wu94]. Cette approche demeure à valider du point de vue de la faisabilité technique et des performances. Un autre problème concerne les communications du type point-à-multi-points. La norme du SCI prévoit la possibilité de faire ce type de communications, mais les produits disponibles à ce jour n'implémentent pas cette fonctionnalité.

D'une manière plus générale, je pense que la technologie SCI ne sera utilisée qu'enfouie dans certaines machines multi-processeurs haut de gamme et ne devrait pas être largement diffusée en dehors de ce marché très restreint. Pour l'interconnexion de machines en grappes (*cluster computing*), les produits disponibles sont peu nombreux. A cause des difficultés de principe, des problèmes techniques qui demeurent à résoudre et des maigres perspectives commerciales de la technologie SCI, je pense que cette voie n'est plus à considérer pour le Trigger/DAQ de l'expérience ATLAS.

5.4.6. Autres technologies

En dehors des technologies précédemment mentionnées, fort peu d'autres options sont envisagées par la collaboration ATLAS. Les résultats d'études de veille technologique font par exemple ressortir les technologies Myrinet [Bod95] et RACEway [Rob97].

Myrinet est une technologie destinée à interconnecter des stations de travail ou PCs de manière économique et performante. Les produits disponibles comprennent des cartes réseau (1,28 Gbit/s *full-duplex*) et des commutateurs de type *crossbar* (16 ports au maximum) pouvant être cascadés. Les performances semblent très bonnes (simplicité des commutateurs, logiciel pilote des cartes réseau optimisé, etc) et les prix sont très attractifs. La couche physique et la couche liaison de données de Myrinet sont normalisées depuis 1998 (norme ANSI/VITA 26-1998). Un système comportant ~100 stations de travail interconnectées par un réseau Myrinet a été rapporté [Cul97]. Une étude approfondie sur Myrinet est en cours dans la collaboration CMS; des résultats sont disponibles dans [Mej00].

RACEway est une technologie développée à l'origine par la société Mercury. Elle a été normalisée en 1994 (ANSI/VITA norme 5-1994). Des interfaces réseau, des commutateurs de type *crossbar* et d'autres modules sont disponibles dans le commerce. Aucune étude avancée portant sur cette technologie n'est en cours dans la collaboration ATLAS.

Les approches basées sur une réalisation spécifique complète des réseaux de communication pour un système de Trigger/DAQ sont aujourd'hui abandonnées [Bar90], [Sas93]. Ce type de solution peut paraître attractive par son apparente simplicité et ses performances, mais demande un effort très important de développement puis de maintenance. Je pense néanmoins que cette approche aurait pu être poursuivie et validée afin d'offrir une alternative en cas d'échec (relativement improbable) des solutions basées sur des technologies de réseau standards.

5.5. Résumé

Après avoir présenté un bref historique sur les réseaux informatiques et mentionné certaines des technologies utilisées dans les expériences de physique, j'ai décrit les solutions qui sont envisagées pour la réalisation du système de T/DAQ de l'expérience ATLAS. Les deux seules technologies qui me semblent être des candidats réalistes aujourd'hui sont ATM et Fast/Gigabit Ethernet. L'étude de faisabilité du système en utilisant l'ATM est l'objet de cette thèse; celle relative à l'Ethernet est prise en charge par d'autres membres au sein de la collaboration ATLAS.

CHAPITRE VI. RÉSEAUX POUR LA RECONSTRUCTION D'ÉVÉNEMENTS

6.1. Introduction

Dans ce chapitre, je définis et étudie un certain nombre de réseaux de communication adaptés au problème de reconstruction d'événements. Je montre comment le trafic particulier à cette application peut-être véhiculé par ces réseaux en précisant les contraintes à respecter pour le bon fonctionnement du système. Je décris une méthode de profilage du trafic pour la réalisation de systèmes de reconstruction d'événements basés sur la technologie ATM. Je montre comment réaliser des réseaux de grande capacité par assemblage de matrices de commutation de manière optimale.

6.2. Paramètres caractéristiques

Avant de présenter les réseaux de communications étudiés, un certain nombre de paramètres caractéristiques et les notations utilisées sont définis.

• Nombre de sources

Notation: S ou N

Signification: Nombre de sources de données des détecteurs liées au réseau. Une source peut correspondre à une ou à un regroupement de plusieurs cartes de lecture frontale des détecteurs (ROBs dans ATLAS). Lorsque le nombre de sources est égal au nombre de destinations, celui-ci est noté N.

Contrainte: Le nombre maximal de sources est fixé par les caractéristiques du système de lecture des détecteurs. Le facteur de regroupement des cartes de lecture est ajustable dans une certaine mesure. Si ce regroupement est fait par un bus, la limitation est fixée par la bande passante du bus et par le nombre d'emplacements disponibles dans le châssis d'électronique utilisé (au maximum 10 à 30 selon les cas).

• Nombre de destinations

Notation: D ou N

Signification: Nombre de destinations liées au réseau comportant un ou plusieurs processeurs de traitement des données en provenance des sources.

Contrainte: Le nombre de destinations est un paramètre libre. Il doit correspondre aux besoins matériels pour effectuer le traitement des données des détecteurs. La puissance de calcul et le nombre de processeurs par destination est également ajustable (de 1 à 16 processeurs par exemple pour les machines les plus courantes).

• Fragment d'événement

Notation: E_i, (valeur moyenne E)

Signification: Taille en octets du fragment d'événement de la source S_i pour un événement

donné. L'ensemble des fragments des S sources est l'événement complet de taille E_{tot}.

Contrainte: Les caractéristiques dépendent de l'électronique des détecteurs, certains fournissent des blocs de taille fixe alors que d'autres sont variables d'un événement à l'autre.

• Nombre de fragments à rassembler

Notation: k

Signification: Pour un événement donné, les fragments en provenance de k sources doivent être rassemblés dans une destination particulière.

Contrainte: Ce paramètre dépend de l'organisation des sources et des besoins des algorithmes de sélection des événements. Il peut varier de quelques unités à la totalité des S sources dans le cas de la reconstruction complète de l'événement.

• Fréquence (période) d'arrivée des événements

Notation: F_a, (T_a)

Signification: Fréquence (période) d'arrivée des événements après le trigger de niveau 1.

Contrainte: La valeur maximale de F_a est fixée par le cahier des charges du système. La capacité du système à pouvoir traiter le flux d'événements maximal avec un temps mort spécifié (typiquement de l'ordre de 1%) est un objectif fondamental. L'intervalle de temps entre événements peut être constant ou suivre une distribution exponentielle.

• Bande passante des liens des sources

Notation: Notation: B_S, (B_D)

Signification: Bande passante du lien au réseau de chaque source (destination).

Contrainte: Paramètre libre dont l'ordre de grandeur est donné par le débit de la source correspondante et la valeur exacte est fixée par les valeurs normalisées de la technologie, du produit ou du standard utilisé. La valeur maximale est fixée par les limites technologiques du moment et/ou par des contraintes économiques (2 à 10 Gbit/s est l'hypothèse adoptée).

• Temps de transit du réseau à vide

Notation: T_{N0}

Signification: Temps de transit d'un message à travers le réseau à vide. Ce temps est supposé constant quelle que soit la taille du message pour un réseau de type *cut-through routing* (routage raccourci), et composé d'une partie constante et d'une partie proportionnelle à la taille du message pour un réseau de type *store and forward* (stockage puis ré-émission).

6.3. Paramètres d'observation

• Charge sur un élément

Notation: α

Signification: Rapport entre l'utilisation courante d'une ressource et sa capacité d'utilisation maximale. Dans le cas d'un élément simple, par exemple un lien au réseau, il s'agit du rapport de la bande passante utilisée sur la bande passante théorique. Dans un système complexe, α ne peut, en général, pas s'exprimer simplement: la limitation vient de l'élément le plus sollicité ou bien d'une combinaison des diverses limitations de chacun des éléments.

Contrainte: Ce facteur ne peut pas être supérieur à l'unité, et doit, dans certains cas, être inférieur à ~0,75 afin d'assurer un temps de réponse acceptable de la ressource pour la stabilité et le bon fonctionnement du système. Afin d'utiliser au mieux les ressources disponibles, ce facteur doit être aussi proche de l'unité que possible.

• Temps de ré-assemblage

Notation: t_R

Signification: intervalle de temps pour collecter les k fragments d'événement des sources concernées dans la destination en charge du traitement de cet événement (incluant le temps de transfert à travers le réseau).

• Capacité mémoire des sources (destinations) (éléments du réseau)

Notation: M_S, (M_D)(M_R)

Signification: quantité de mémoire nécessaire au niveau de chaque source (destination) (élément du réseau) pour conserver les fragments d'événements en provenance des détecteurs (messages) pendant le temps nécessaire.

• Utilisation des liens des sources (des destinations) (du réseau)

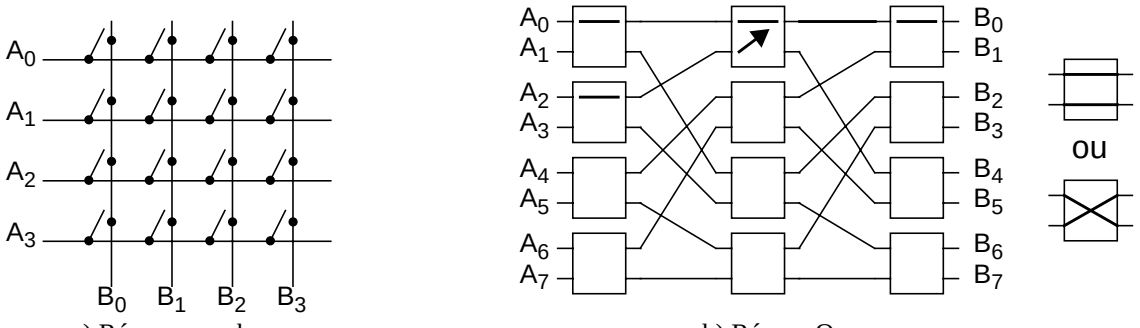
Notation: U_S, (U_D), (U_R)

Signification: fraction de la bande passante du lien utilisée par rapport à sa capacité totale. Plus ce facteur se rapproche de l'unité, mieux la ressource disponible est utilisée.

6.4. Réseau à permutation synchronisé et sans mémoire

6.4.1. Présentation

Depuis le début de l'ère téléphonique, la construction de réseaux permettant de relier des circuits d'entrée et de sortie par paires a donné lieu à de nombreux travaux. Un des soucis principaux des premiers concepteurs était de construire un réseau modulaire, avec le minimum d'éléments internes, pouvant assurer la connexion simultanée de n'importe quelles paires de circuits d'entrée et de sortie. Un exemple simple de réseau est la matrice *crossbar* présentée Figure 6-1a.



a) Réseau crossbar b) Réseau Omega

Figure 6-1: Exemples de réseau.

En fermant l'interrupteur correspondant, ce réseau permet de connecter n'importe quelle entrée A_i à n'importe quelle sortie B_j à condition que celle-ci ne soit pas déjà connectée à une autre entrée. Le réseau est qualifié de "non bloquant". Le nombre de points de connexion varie comme le

carré du nombre d'entrées-sorties N, ce qui constituait une limite à la taille des premiers commutateurs de téléphonie. Aujourd'hui, compte tenu des progrès de la micro-électronique, cette limite à été considérablement augmentée. Un *crossbar* à 128 entrées a par exemple été rapporté dans [Sim95] (limitation en taille due au nombre de broches disponibles sur le boîtier du circuit intégré).

Afin de réduire le nombre d'éléments de commutation par rapport au *crossbar*, de nombreuses structures de réseau ont été proposées. Elles sont regroupées dans la littérature sous le terme *Multi-stage Interconnection Networks (MINs)*, réseaux d'interconnexion à plusieurs étages [Ahm94]. Un exemple de ce type de réseau est présenté Figure 6-1b. L'élément de base de ce réseau est un *crossbar* qui permet de connecter chacune des deux entrées à la sortie en regard ou bien de croiser la connexion. Le nombre d'éléments pour réaliser ce type de réseau varie comme *N.logN*. Il a été démontré que ce type de réseau permet d'effectuer de nombreuses permutations des entrées vers les sorties: décalage circulaire,... Le problème principal est qu'il n'est pas toujours possible d'établir une connexion entre un couple d'entrée-sortie donné même lorsque la sortie considérée n'est pas utilisée par ailleurs. Supposons que le chemin entre A_0 et B_0 soit établi. Il paraît clair sur la Figure 6-1b qu'il n'est pas possible de connecter A_2 à B_1, car la liaison interne qu'il faudrait utiliser est déjà en service. En raison de cette limitation, le réseau est qualifié de "bloquant". Diverses solutions ont été proposées pour obtenir un réseau capable d'effectuer toutes les permutations possibles des entrées vers les sorties (i.e. non bloquant): ajout d'un réseau de tri en amont d'un MIN pour le réseau Banyan-Batcher [Bat68], réseau de Clos [Clo53]. Alors que pour de nombreuses applications, l'aspect non-bloquant du réseau est un point fondamental, dans le cas d'un système de reconstruction d'événements, cet aspect est moins critique. Ce point est explicité ci-dessous.

Pour la reconstruction d'un événement, chaque source doit transmettre un fragment d'événement vers une destination donnée. A priori, il s'agit du cas le plus défavorable pour le réseau car une destination ne peut être connectée qu'à une seule source à la fois. En réalité, des événements sont en cours de reconstruction dans les autres destinations, et chaque source peut émettre le fragment correspondant à un événement antérieur à la destination à laquelle elle peut être connectée sans engendrer de conflit d'utilisation des ressources internes du réseau. Pour chaque destination, il est suffisant d'être en liaison tour à tour avec chacune des sources, sans qu'il soit nécessaire de suivre un ordre précis. De manière symétrique, chaque source doit être en liaison périodique avec chacune des destinations (sans ordre précis). Un réseau capable d'effectuer au moins un type de permutation de manière non bloquante est suffisant. Un exemple est le "décaleur à barillet" (*Barrel Shifter*) qui permet d'effectuer les permutations du type "décalage circulaire". Toutes les familles de permutations établissant toutes les paires possibles d'entrée-sortie au cours de N re-configurations du réseau sont adéquates pour l'application de reconstruction d'événements. La totalité des MINs bloquants courants (et aussi les non bloquants) répondent à cette contrainte.

Un modèle de réseau adapté à la reconstruction d'événements est présenté Figure 6-2. Le nombre de sources S et de destinations D est identique. Chaque source ne peut émettre vers une destination donnée que lorsque la configuration du réseau central la met en contact avec cette destination. Un mécanisme de synchronisation entre les sources et le système de changement de configuration du réseau est nécessaire (d'où le qualificatif "synchronisé" pour ce réseau). A intervalles réguliers, les connexions établies par le réseau sont changées pour mettre en relation d'autres couples de source-destination. Au bout de N changements de la configuration du réseau, toutes les sources ont été connectées tour à tour avec chacune des destinations. A chaque source ne correspond qu'une seule destination à tout instant. Les couples de source/destination sont tels qu'il ne se produit aucun blocage dans le réseau. Si des files d'attente se forment, elles sont situées dans les sources. En une période, chaque source envoie un fragment d'événement vers une destination; au bout de N périodes, N événements initialement répartis dans les N sources ont été rassemblés dans

les N destinations (1 événement rassemblé dans chacune des N destinations; les fragments sont de taille identique).

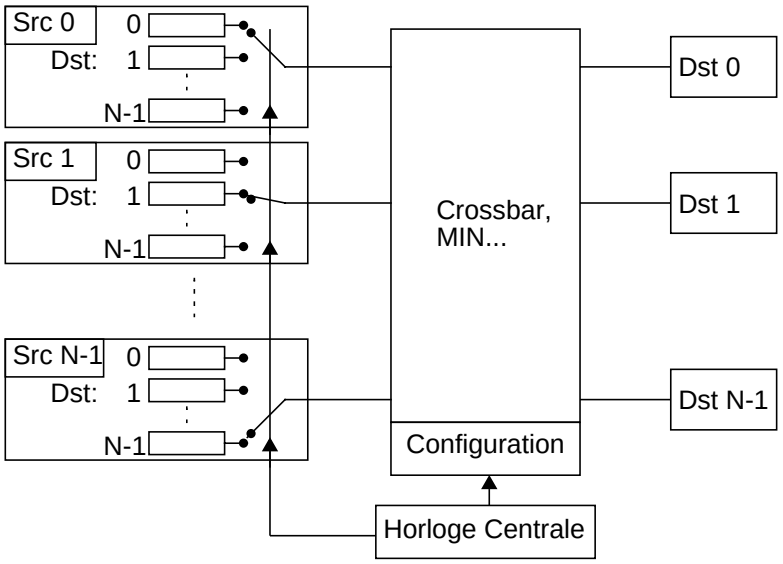


Figure 6-2: Réseau synchronisé et sans mémoire.

La topologie et la structure du réseau sont relativement libres: il peut s'agir d'un MIN bloquant (ou non bloquant) voire de tout autre système capable d'effectuer de manière cyclique au moins un type de permutation d'interconnexion de tous les couples source-destination possibles. La réalisation technique du réseau est également libre: il peut s'agir de commutation de circuits électriques ou d'un réseau à commutation de paquets adapté en conséquence. Dans le cas de commutation de paquets, il n'est pas nécessaire d'avoir des mémoires tampon dans le réseau (ce dernier opère sans blocage interne), d'où le qualificatif de réseau "sans mémoire".

6.4.2. Etude de performance

Le temps de transit à travers le réseau, T_{N0}, est supposé constant, indépendant de la taille des messages et de la charge du réseau. Soit C la période de changement de la configuration des connexions du réseau. On suppose que le temps de re-configuration du réseau est négligeable par rapport aux temps de transfert de données. Les sources et le réseau sont (par exemple) programmés de sorte qu'à l'instant nC, chaque source S_i (pour i dans [0;N-1]) est connectée à la destination $D_{(i+n)Mod\ N}$ (permutation circulaire cyclique des couples source/destination). Lors d'un changement de configuration du réseau, on suppose que chaque source arrête son émission en un temps nul, et est en mesure d'émettre vers son nouveau partenaire instantanément. Dans le cas où ces hypothèses ne sont pas vérifiées, on peut considérer que le temps mort dû à la synchronisation des divers éléments est équivalent à une baisse de la bande passante utile de chaque lien d'une valeur égale à la quantité d'information que pourrait transmettre le lien dans l'intervalle de temps perdu.

6.4.2.1. Fréquence maximale de ré-assemblage d'événements, F_{aMax}

Pour la reconstruction complète d'événements, en considérant tous les liens identiques et des fragments d'événement de mêmes caractéristiques statistiques, on a:

$$F_{aMax} = \frac{B_{SD}}{E} = \frac{B_{SD} \cdot N}{E_{tot}} \qquad \textbf{(6-1)}$$

La fréquence maximale de fonctionnement est l'inverse du temps de transfert d'un fragment de taille moyenne. Comme chaque événement comporte N fragments, le débit total du système de reconstruction d'événements est proportionnel à N.

6.4.2.2. Temps de ré-assemblage

Le temps de ré-assemblage comporte trois composantes: le temps d'attente d'un fragment dans une source, le temps d'envoi de ce fragment et le temps de transfert à travers le réseau. Comme il n'est pas possible que plusieurs fragments entrent en concurrence pour occuper le lien de sortie de chaque destination, il ne peut pas se former de file d'attente à la sortie du réseau (si la bande passante de chaque lien de sortie est suffisante pour absorber le trafic de la source la plus rapide).

6.4.2.2.1. Fonctionnement à faible charge

On considère que l'intervalle de temps inter-événement T_a est arbitrairement grand, de sorte qu'un seul fragment d'événement au plus est présent dans chaque source. Une source devant émettre vers une destination donnée dispose d'un intervalle de C unités de temps toutes les N.C périodes. Le temps nécessaire pour envoyer un fragment de taille E_i d'une source S_i vers une destination est:

$$t_S = t_C + \frac{E_i}{B_{SD}} + (N-1)\cdot C \cdot Int\left(\frac{E_i}{B_{SD}\cdot C}\right) \qquad \textbf{(6-2)}$$

Le premier terme de cette expression, t_C, est l'intervalle de temps entre l'arrivée du fragment d'événement et l'instant à partir duquel la configuration du réseau est telle que son émission peut débuter. En considérant que l'arrivée des événements est décorélée de la synchronisation du réseau, t_C suit une distribution uniforme dans l'intervalle [0;(N-1)C]. Le second membre est le temps de transfert d'un message de taille donnée à travers un lien de bande passante fixe. Le troisième terme est le temps correspondant aux créneaux pendant lesquels la source a dû stopper son émission vers la destination concernée suite aux changements périodiques de la configuration du réseau.

Les deux cas suivants peuvent être considérés:

- cas a: $N.C << Sup_i(E_i)/B_{SD}$

Ce cas correspond à un changement de la configuration du réseau fréquent par rapport au temps d'envoi du plus grand des fragments d'événement et un temps de début d'émission du fragment peu différent du temps d'arrivée de ce fragment. Le temps de ré-assemblage est donné par le temps d'émission du plus grand fragment d'événement. La bande passante moyenne équivalente du lien entre une source et une destination est B_S/N car une source ne peut émettre vers une destination particulière que durant un intervalle de temps C tous les N.C périodes. On a donc:

$$t_R = T_{N0} + Sup\left(\frac{E_a}{\frac{B_{Sa}}{N}}, \frac{E_b}{\frac{B_{Sb}}{N}}, \ldots, \frac{E_k}{\frac{B_{Sk}}{N}}\right) \qquad a, b, \ldots, k; distincts \in [0, S-1] \qquad \textbf{(6-3)}$$

Le premier terme, T_{N0}, est supposé constant. Le second terme est noté t_{RS}. En faisant l'hypothèse que tous les liens des sources ont une bande passante identique et que la taille de tous les fragments d'événement suit la même loi statistique, évaluer les caractéristiques de t_{RS} se ramène à étudier des séries de k tirages aléatoires issus de cette loi. La fonction de répartition de t_{RS} est:

$$F_{t_{RS}}(x) = F_E(x)^k \qquad \textbf{(6-4)}$$

La densité de probabilité s'obtient par dérivation:

$$f_{t_{RS}}(x) = k \times F_E(x)^{k-1} \times f_E(x) \quad \textbf{(6-5)}$$

Lorsqu'une solution analytique ne peut pas être trouvée, le calcul numérique ou la simulation permet de déterminer ces distributions. Un exemple d'application est présenté dans [Cal97b][1].

- cas b: $N.C >> Sup_i(E_i)/B_{SD}$

Ce cas correspond à un changement de la configuration du réseau lent par rapport au temps d'envoi du plus grand des fragments d'événement. Le temps de ré-assemblage est le temps mis pour collecter le fragment correspondant dans le cycle de permutation des connexions à la dernière source mise en relation avec la destination concernée. Ceci est valable quel que soit le nombre de fragments à collecter, le dernier fragment pouvant provenir de n'importe quelle source. On a:

$$t_{RS} = t_C + \frac{E_{last}}{B_{SD}} \quad \textbf{(6-6)}$$

Le second terme de cette expression est le temps de transfert du dernier fragment d'événement. Ce terme suit la même loi de probabilité que E_i.

La premier terme est la valeur dominante. Ce temps t_C est constant et vaut (N-1)C dans le cas particulier où N fragments de taille non nulle sont collectés: il faut toujours un et un seul cycle complet de changements de configuration du réseau pour reconstruire l'événement. Dans le cas de la collecte de k fragments (k<=N), au plus un cycle est nécessaire car les (N-k) dernières sources ne participent pas à l'événement à reconstruire. Le temps t_C est réparti uniformément dans l'intervalle [(k-1)C, (N-1)C]. Dans le cas où le nombre de fragments à collecter k n'est pas constant, la distribution de t_C ne suit plus une distribution uniforme, mais une fonction faisant intervenir la probabilité d'avoir les (N-k) dernières sources non concernées par l'événement à reconstruire. Cette probabilité est $f_k(0)^{(N-k)}$. Un exemple d'illustration est donné dans [Cal97b][1].

6.4.2.2.2. Fonctionnement en charge

L'étude du fonctionnement en charge est complexe. Elle est effectuée par simulation. Des résultats sont présentés dans [Cal97b][1]. Parmi les résultats principaux, on notera que le système fonctionne de manière satisfaisante tant que la charge α sur les liens reste inférieure à ~75%.

6.4.2.3. Communications

- communication point-à-point

Pour les messages courts, le délai de transmission est dominé par une composante aléatoire, uniformément répartie dans l'intervalle [0, (N-1)C]. La bande passante de la liaison équivalente pour chaque couple source/destination est:

$$B_{SD}' = \frac{B_{SD}}{N} \quad \textbf{(6-7)}$$

Par rapport à l'utilisation d'une liaison point-à-point de bande passante B_{SD}, le temps de transmission de messages de taille importante est multiplié par un facteur N. Le réseau à permutation synchronisé est mal adapté aux communications point-à-point.

1. Cette publication est incluse dans l'Annexe 2 de cette thèse.

- communication point-à-multi-points
  Ce type de communication ne peut pas être réalisé de manière simple.

6.4.2.4. Capacité de mémorisation

Le réseau à permutation synchronisé sans mémoire est, à l'évidence, optimal en ce qui concerne la capacité de mémorisation requise au niveau du réseau. Il ne se forme jamais de files d'attente ni à l'intérieur du réseau. Au niveau des sources, un certain nombre de résultats concernant l'occupation des files d'attente sont présentés dans [Nom93] (expression analytique de la valeur moyenne de la taille des files d'attente, distribution en fonction de la charge).

6.4.3. Réalisation pratique

Ce modèle de réseau synchronisé a fait l'objet de divers prototypes tel le *Scalable Parallel Open Architecture* (SPOA) *Data Acquisition System* [And91]. La partie centrale est un décaleur à barillet réalisé à partir de circuits intégrés *crossbar* du commerce. Elle comporte 32 voies d'entrée et de sortie. Les interfaces d'entrée effectuent la liaison avec chaque source de données pour former des paquets de taille fixe sur la base de chaque événement. Ces paquets sont envoyés suivant un ordre bien précis pour être transmis sans blocage au travers du commutateur central vers les interfaces de sortie. Chaque interface de sortie reçoit, suivant une séquence fixe, un paquet en provenance de chaque interface d'entrée. Au terme d'un cycle, l'ensemble des paquets constituant un événement a été rassemblé dans une interface de sortie, et peut être transmis à un processeur de traitement.

Du point de vue des fonctionnalités et des performances, cette réalisation est satisfaisante. Le défaut majeur est la spécificité du matériel développé. Le projet a nécessité d'importants efforts de conception et de réalisation. La plupart des composants utilisés sont aujourd'hui obsolètes, l'évolution et la maintenance du système pose problème.

Un autre exemple de système de reconstruction d'événements basé sur un réseau synchrone est celui de l'expérience BELLE [Nag97]. Ce système utilise des éléments de type *crossbar* 2x2 pour réaliser un décaleur à barillet à 8 entrées et sorties (commutation de circuits électriques). Chaque voie est équipée d'une liaison série à 1 Gbit/s. La synchronisation des sources, destinations et du réseau est effectuée par une horloge centrale distribuée à tous ces éléments par un réseau séparé, à plus faible débit (100 Mbit/s). Ce système, en fonctionnement dans une expérience, démontre la faisabilité de cette approche, mais plusieurs critiques peuvent être formulées. La solution est relativement complexe puisqu'elle met en œuvre trois réseaux différents: le réseau principal pour le transport des données, un réseau annexe pour la synchronisation des éléments et un réseau local classique pour la configuration et la surveillance de l'ensemble. Comme pour le projet SPOA, un effort important a été nécessaire pour concevoir et réaliser le coeur de commutation, les cartes d'interfaces d'entrée et de sortie, le réseau de synchronisation et l'ensemble du logiciel de contrôle.

6.4.4. Discussion

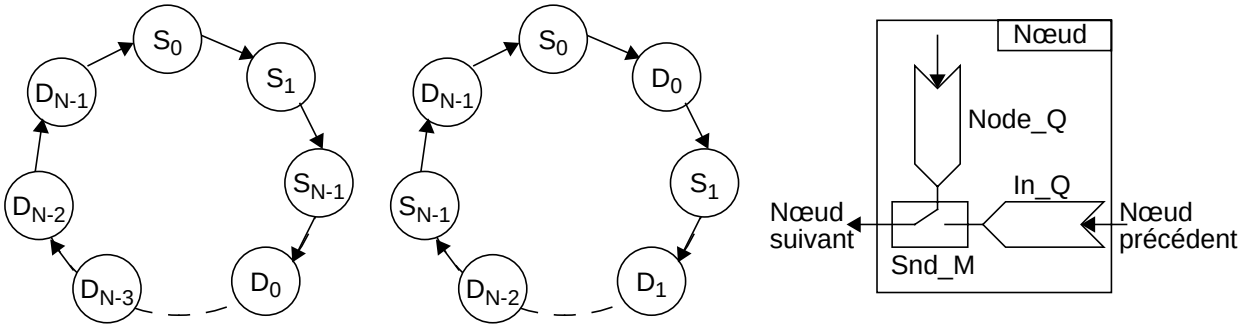
Le réseau à permutation synchronisé est d'une structure très simple pour la partie constituant les chemins de données. Elle peut être réalisée à l'aide de composants électroniques courants (commutateurs de type *crossbar*) interconnectés suivant une topologie de réseau multi-étage classique (Banyan, Omega, etc). Les éléments centraux effectuent des commutations de circuits électriques; cela laisse toute liberté pour adapter les interfaces d'extrémité à un standard, composant ou une technologie particulière. Le mécanisme permettant la synchronisation des sources et du réseau ajoute à la complexité de la solution et requiert la mise en place d'un réseau secondaire. La mise en œuvre de cette solution demande un important effort de développement ou d'adaptation de

matériel et de logiciel car ce type de dispositif n'est pas disponible dans le commerce. Malgré la simplicité apparente du réseau à permutation synchronisé, cette solution est en réalité très imparfaite.

6.5. Anneau à commutation de paquets

6.5.1. Présentation

Ce type de réseau est présenté Figure 6-3a,b. Le système comporte N nœuds du type Source (S) figurant les sources de données à transmettre, et N nœuds du type Destination (D) devant collecter ces données. Deux arrangements sont considérés: nœuds S placés les uns à la suite des autres suivis par les nœuds D ou bien alternance de nœuds S et D. Chaque élément est connecté à ses deux voisins par un lien uni-directionnel (anneau "à sens unique"). Toutes les liaisons sont identiques, de bande passante B_{SD}.



a) Anneau nœuds groupés b) Anneau nœuds alternés c) Modèle d'un nœud

Figure 6-3: Anneau à commutation de paquets.

Le modèle d'un nœud est présenté Figure 6-3c. Il comporte deux files d'attente: une est destinée à recevoir les paquets transmis par le nœud voisin (In_Q), la seconde sert a conserver les paquets devant être émis par le nœud (Node_Q). Chaque paquet contient un en-tête permettant d'identifier le nœud destination du message. Lorsqu'un message est présent dans la file In_Q, la destination du message est analysée: si le destinataire est le nœud considéré, le paquet est retiré (message reçu), dans le cas contraire, le paquet est placé dans l'automate de transmission (Snd_M) qui le transmet au nœud suivant. Le temps de transmission est proportionnel à la taille du message (en octets). Pour transmettre un message, un nœud place le paquet correspondant dans la file Node_Q servie par l'automate Snd_M. Si un paquet est présent dans la file In_Q et qu'un autre paquet est présent simultanément dans Node_Q pour un nœud donné, l'automate Snd_M sert en priorité la file In_Q (discipline limitant l'engorgement de l'anneau semblable à celle utilisée aux rond-points pour la circulation automobile).

La transmission des paquets est du type *store and forward*: un nœud commence à faire suivre un paquet au nœud suivant seulement après avoir reçu en totalité ce paquet. Toutes les files d'attente sont supposées de capacité infinie.

6.5.2. Etude de performance

6.5.2.1. Fréquence maximale de ré-assemblage d'événements

Considérons la reconstruction complète d'événements. Pour chaque événement, un fragment est ajouté dans la file Node_Q de chaque source. Les fragments d'un événement donné doivent être

rassemblés dans une des destinations. La destination affectée à la collecte de chaque événement est déterminée de manière aléatoire. Le mode de contrôle du flux de données est le modèle "données poussées" (pas de trafic allant des nœuds destination vers les sources).

- Arrangement "sources et destinations groupées"

Le trafic sur le lien de sortie de chaque source est la somme du trafic de toutes les sources précédentes et de son propre trafic. Le lien le plus chargé est celui entre la dernière source et la première destination. Le trafic sur le lien de sortie de chaque destination est la somme du trafic destiné aux destinations suivantes. Il décroît linéairement et devient nul pour la dernière destination.

La fréquence maximale de fonctionnement pour la reconstruction complète d'événements est déterminée par la bande passante des liens et la taille des événements:

$$F_{aMax} = \frac{B_{SD}}{E_{tot}} \qquad \textbf{(6-8)}$$

Le débit total du système de reconstruction d'événements est au maximum égal au débit d'un lien de l'anneau. La répartition de la charge sur les liens de l'anneau est inhomogène et le système comporte un goulet d'étranglement évident au niveau du dernier nœud source.

- Arrangement "sources et destinations alternées"

La limitation provient à nouveau des liens des sources. On détermine leur sollicitation lors de la reconstruction de N événements (1 par destination). Chaque lien de sortie d'une source véhicule N fragments provenant d'elle-même, N-1 fragments de la source précédente, etc.

La participation moyenne, notée P_S, d'un lien d'une source par événement est:

$$P_S = \frac{1}{N} \cdot (N + (N-1)\ ) = \frac{N+1}{2} \qquad \textbf{(6-9)}$$

Chaque événement comportant N fragments, on déduit:

$$F_{aMax} = \frac{B_{SD}}{E_{tot}} \cdot \frac{N}{P_S} = \frac{B_{SD}}{E_{tot}} \cdot \frac{2}{1+\frac{1}{N}} \qquad \textbf{(6-10)}$$

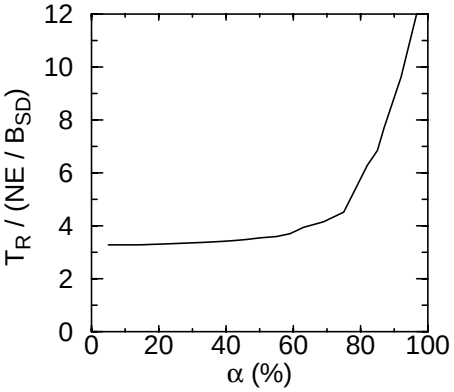
Le débit maximum total du système de reconstruction d'événements tend vers le double du débit d'un lien de l'anneau lorsque N augmente. L'augmentation de la taille du système n'entraîne ni une dégradation ni un accroissement du débit global. A taille de fragment d'événement donnée, F_{aMax} est inversement proportionnelle à la taille du système (saturation de performance).

6.5.2.2. Temps de ré-assemblage

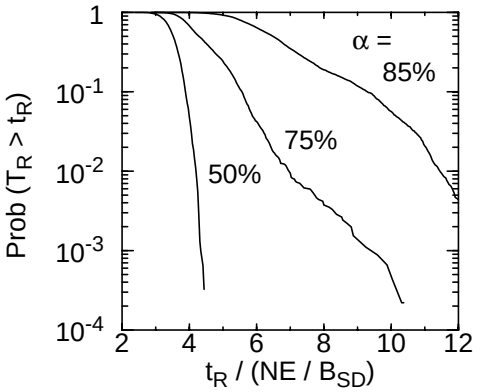
Cette étude est effectuée par simulation. La valeur moyenne du temps de ré-assemblage (T_R) pour un système comportant 32 sources et 32 destinations est présentée Figure 6-4a (configuration sources et destinations alternées, distribution uniforme pour la taille des fragments, intervalle de temps inter-événements constant). Lorsque la charge approche 100%, le système devient instable. La distribution de T_R est présentée Figure 6-4b. Par une autre série de simulations, on observe qu'à charge donnée la valeur moyenne de T_R augmente linéairement avec N.

Le temps de ré-assemblage est également dépendant de la discipline d'allocation des événements aux destinations. Certains cas sont défavorables, tels une allocation affectant l'événement suivant à la destination voisine de la destination courante, alors que d'autres disciplines

d'allocation périodique permettent de mieux répartir le trafic dans l'anneau.



a) T_R en fonction de la charge



b) Distribution de T_R

Figure 6-4: Temps de ré-assemblage.

6.5.2.3. Communications

- communication point-à-point

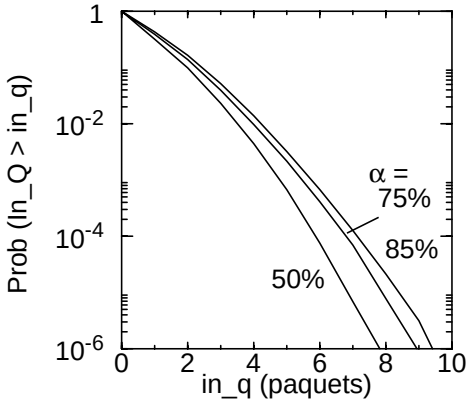
Avec le modèle *store and forward*, le délai de communication entre deux points augmente proportionnellement à la taille du message et à l'éloignement relatif des deux nœuds considérés (entre 1 et N). La bande passante de la liaison équivalente pour chaque couple source/destination est B_{SD}: la communication entre deux points peut utiliser la pleine bande passante de l'anneau.

- communication point-à-multi-points

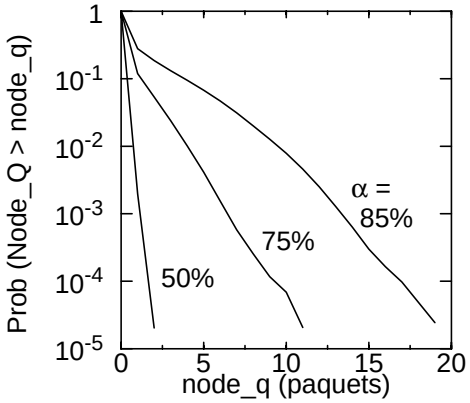
Ce type de communication peut être réalisé simplement en faisant subir aux messages considérés un tour complet sur l'anneau (diffusion à l'ensemble des nœuds). Optimiser la diffusion à un ensemble restreint de nœuds est plus complexe: chaque nœud doit savoir si au moins un nœud parmi tous les suivants dans l'anneau (jusqu'au nœud émetteur) est concerné par le message considéré pour décider de le faire suivre.

6.5.2.4. Capacité de mémorisation

Comme la file In_Q transporte les messages de tous les nœuds précédents dans l'anneau alors que la file Node_Q contient seulement les messages émis par le nœud considéré, In_Q est prioritaire.



a) Distribution de la taille de la file In_Q



b) Distribution de la taille de la file Node_Q

Figure 6-5: File d'attente In_Q et Node_Q.

La distribution de la taille de la file In_Q (Node_Q) est présentée Figure 6-5a (b) pour le

même jeu de paramètres que précédemment. La valeur moyenne de la taille de la file d'attente In_Q (Node_Q) s'établit à 1,4; 1,6; 1,7 (1; 1,2; 1.9) paquets pour une charge respective de 50, 75 et 85%.

Par une autre série de simulations, on observe que la taille de la file d'attente In_Q est peu dépendante de la taille de l'anneau. La capacité de mémorisation requise au niveau des éléments du réseau est modeste (10-20 paquets) ce qui ne doit pas poser de difficultés de réalisation majeure.

6.5.3. Réalisation pratique

Les technologies utilisables sont FDDI, SCI ou Fibre Channel (*arbitrated loop*). Pour ATLAS, l'*event builder* doit avoir un débit total de ~1 Go/s. La bande passante théorique minimale des liens à utiliser est de 500 Mo/s d'après l'interprétation de la formule (6-9) (en fonctionnement à 100% de charge). Cela correspond à du matériel disponible en SCI, mais laisse une marge de sécurité assez faible et peu de possibilités d'évolution: augmenter le débit global impose de changer les interfaces de tous les nœuds. Concernant les besoins de CMS, obtenir un débit pour l'*event builder* de ~60 Go/s est techniquement (et économiquement) irréaliste avec une structure en anneau compte tenu de la difficulté de réalisation de liens fonctionnant à ~30 Go/s.

Un autre problème avec une topologie en anneau est la tolérance aux pannes. Le mauvais fonctionnement d'un élément de l'anneau perturbe l'ensemble du système. En cas d'interruption de l'anneau, le système complet est bloqué. L'insertion d'un nouvel élément dans l'anneau est également source de perturbations. Pour des systèmes de grande taille comportant plusieurs centaines d'éléments à connecter, la structure en anneau est mal adaptée.

6.5.4. Discussion

La structure en anneau peut convenir lorsque deux conditions sont remplies: le débit global requis est compatible avec le débit d'un lien de l'anneau, et le nombre d'éléments à connecter est relativement faible (moins de ~50). Les arrangements mixtes comportant des anneaux connectés par des commutateurs ne sont pas étudiés dans cette thèse. On pourra consulter [Wu94] à ce sujet.

6.6. Réseau à multiplexage statistique

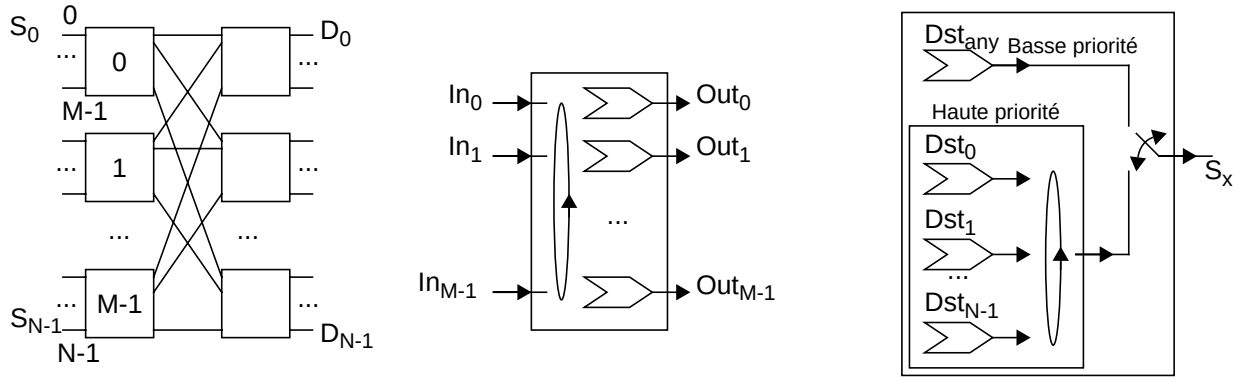
6.6.1. Présentation

Avec le réseau à permutation synchronisé, une part importante de la complexité de réalisation vient du mécanisme de synchronisation des sources. En supprimant cette synchronisation, des conflits internes de ressources dans le réseau peuvent survenir: on introduit des mémoires tampon dans les éléments de commutation.

La structure étudiée est présentée Figure 6-6a. Elle comporte deux étages de M matrices de commutation à M ports d'entrée et de sortie pour former un réseau à $N=M^2$ ports. Chaque commutateur (Figure 6-6b) est supposé non bloquant et dispose de mémoires tampon en sortie de capacité infinie. Les éléments véhiculés par le réseau sont de taille fixe (cellules). Le cycle de fonctionnement du commutateur est le temps de transmission d'une cellule. Pour chacun de ces cycles, le commutateur effectue les opérations suivantes: service des M liens d'entrée, prise en charge d'une cellule pour chaque entrée active, ajout de chaque cellule dans la file d'attente de sortie correspondant à sa destination, service des M files d'attente de sortie (*output queuing*) et émission sur le lien de sortie de la cellule de tête de chaque file d'attente non vide.

Le modèle d'une source est présenté Figure 6-6c. Il comporte deux catégories de files

d'attente: une file de basse priorité, Dst_{any}, pour laquelle les paquets sont servis en mode "premier arrivé, premier servi", et une série de N files (une par destination), Dst_x, de priorité plus haute, servies de manière séquentielle. A chaque cycle de fonctionnement d'une source (correspondant au temps de transfert d'une cellule), la file d'attente Dst_i (haute priorité) courante est servie: si elle comporte un message à transmettre, une cellule est émise; sinon la file d'attente basse priorité, Dst_{any}, est servie. Au cycle suivant, la file d'attente $Dst_{(i+1) Mod N}$ est servie, si elle est vide, la file Dst_{any}, est à nouveau servie. Les sources effectuent le cycle de service des files d'attente Dst_i dans le même ordre, mais à chaque instant les sources sont en des points différents dans ce cycle: chaque source possède son propre compteur de référence dont la valeur initiale est aléatoire, non corrélée avec celle des autres sources.



a) Réseau à deux étages b) Modèle d'un commutateur c) Modèle d'une source

Figure 6-6: Réseau à multiplexage statistique.

Ce modèle est celui d'un circuit de segmentation et ré-assemblage ATM (*Segmentation And Re-Assembly* – SAR). Les files d'attente de priorité plus élevée correspondent à des connexions virtuelles du type CBR de bande passante B_{SD}/N (rafale maximale d'envoi: 1 cellule ATM), la file d'attente de priorité plus basse représente des connexions du type UBR pouvant utiliser la pleine bande passante du lien.

6.6.2. Etude de performance

6.6.2.1. Fréquence maximale de ré-assemblage d'événements

Pour la reconstruction complète d'événements, F_{aMax} est identique à celle du réseau à permutation synchronisé (formule 6-1).

6.6.2.2. Temps de ré-assemblage

6.6.2.2.1. Fonctionnement à faible charge

Dans le cas où toutes les sources utilisent la file d'attente de basse priorité, le temps de ré-assemblage est le temps de transfert de l'événement:

$$T_R = 2.T_{N0} + \frac{E_{tot}}{B_{SD}} \qquad \textbf{(6-11)}$$

Dans le cas d'utilisation des files d'attente de haute priorité, T_R est déterminé par le temps d'envoi du plus grand fragment d'événement (formule 6-3). Ce temps est toujours supérieur à celui correspondant au premier cas (sauf lorsque les fragments d'événement sont de taille constante).

6.6.2.2.2. Fonctionnement en charge

Des résultats de simulation sont présentés dans [Cos95] et [Cal97b][1].

6.6.2.3. Communications

- communication point-à-point

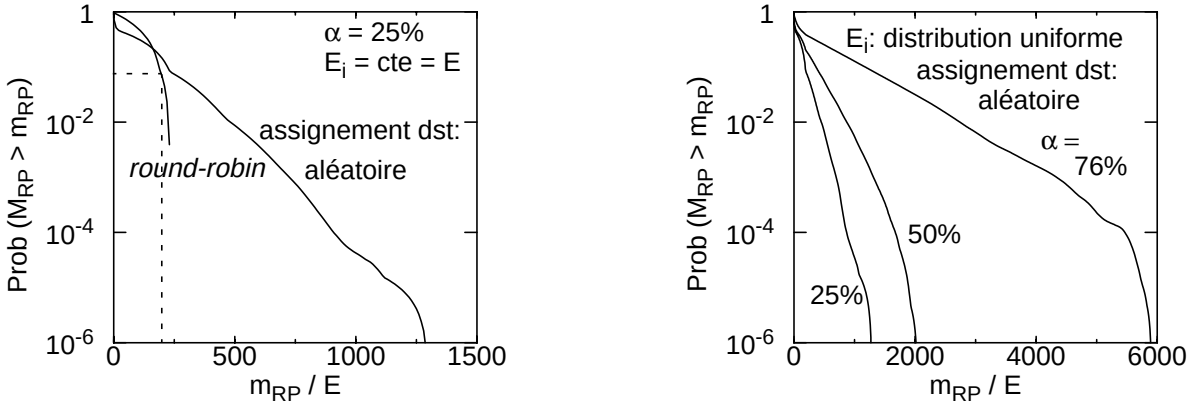
Le délai de communication entre deux points augmente proportionnellement avec la taille du message et varie suivant le nombre d'étages de commutateurs traversés. Dans le cas d'éléments en technologie ATM, le temps de traversée d'un commutateur est indépendant de la taille du message (paquet AAL 5). La bande passante de la liaison équivalente pour chaque couple source/destination est, selon la file d'attente choisie, B_{SD} ou B_{SD}/N, ce qui permet une certaine flexibilité.

- communication point-à-multi-points

Dans le cas de la technologie ATM, ce type de communication est réalisé par du matériel spécifique intégré aux commutateurs.

6.6.2.4. Capacité de mémorisation

Seule la capacité de mémorisation au niveau des commutateurs est considérée dans cette étude. Dans un premier cas, on effectue la reconstruction complète d'événements en utilisant seulement la file d'attente de basse priorité dans les sources. Avec une allocation des destinations cyclique (type *round-robin)*, le nombre maximum de fragments d'événements par port de sortie du commutateur est égal à N-1 pour un réseau à 1 seul étage: la quantité de mémoire maximale par port (M_{RPMax}) vaut $(N-1)E_{Max}$. Dans un réseau à deux étages M_{RPMax} est non bornée (sauf dans le cas de fragments d'événements de taille constante). Lorsque l'allocation des destinations est aléatoire, M_{RPMax} est non bornée dans tous les cas. Le Figure 6-7 donne la distribution de M_{RP} pour un réseau à 2 étages comportant 256 paires source/destination (réseau de 32 commutateurs à 32 ports). L'intervalle de temps inter-événements est constant.



a) M_{RP} vs. discipline d'allocation des destinations

b) M_{RP} vs. α

Figure 6-7: Distribution de M_{RP}.

Compte tenu de la capacité mémoire limitée des commutateurs, des pertes de messages peuvent survenir. Considérons des fragments d'événement de taille moyenne 4 ko (événement complet de 256 x 4 = 1 Mo correspondant à ATLAS). Les commutateurs ATM courants disposent en général d'un maximum de ~16384 cellules par port, soit 768 ko. Il est peu probable que davantage de mémoire soit intégrée car les applications visées par l'ATM requièrent en général des délais de

1. Cette publication est incluse dans l'Annexe 2 de cette thèse.

transmission faibles (voix, multi-média interactif) ce qui conduit à vouloir éviter les longues files d'attente. D'après le graphe 6-7a, on peut prédire un débordement de capacité dans chaque commutateur pour 10% des fragments transmis en fonctionnement à 25% de charge. Cette situation n'est pas acceptable, en particulier si le protocole d'échange de messages n'assure pas la re-transmission des messages perdus (ce qui est le cas avec le protocole de communication optimisé qui sera proposé dans cette thèse). En supposant qu'un protocole de re-transmission est utilisé, les pertes étant dues à la congestion des éléments du réseau (et non à des erreurs sporadiques), re-transmettre des messages ne fait qu'accroître la congestion, et non la résorber.

Diverses techniques de profilage du trafic ont été proposées pour réduire les risques de débordement des mémoires tampon de commutateurs [Man93], [Man94], [Cos95], [Cal97b]. La méthode adoptée dans cette étude est l'utilisation de connexions virtuelles à débit constant. Dans ce cas, la reconstruction d'événements est effectuée en utilisant les files d'attente "haute priorité" (voir Figure 6-6b). Le service des files d'attente dans chaque source suit la discipline *round-robin* (déterminée par programmation du circuit de SAR de la carte réseau de chaque source) quelle que soit la discipline d'allocation des événements aux destinations. De plus, chaque source ne peut pas émettre plusieurs cellules à la suite vers une destination donnée (limitation à 1 cellule ATM également fixée par programmation du circuit SAR). Dans le cas le plus défavorable où tous les circuits SAR servent la même file d'attente dans le cycle, on a:

$$M_{RPMax} = \frac{N}{M} \cdot (M-1) \qquad (cellules) \qquad \textbf{(6-12)}$$

La capacité mémoire au niveau des commutateurs est considérablement réduite et ne dépend plus ni de la taille des fragments d'événements transportés, ni de la discipline d'allocation des événements aux destinations. On peut garantir que si chaque port de sortie d'un commutateur a une capacité d'au moins N cellules ATM, il ne se produit jamais de débordement des files d'attente dans le commutateur quelle que soit la taille des fragments d'événement et la charge du réseau. Avec les hypothèses précédentes (16384 cellules par port), cette condition est largement satisfaite pour les valeurs de N envisagées (au maximum N=~2000 si aucun groupement de ROBs n'est effectué).

6.6.3. Réalisation pratique

Le modèle des sources est directement réalisable en ATM. Avec d'autres technologies, l'implantation d'un mécanisme de qualité de service et de priorité de trafic peut s'envisager par logiciel, bien que la même finesse dans la granularité de l'allocation de la bande passante et dans la gestion des priorités semblent difficile à obtenir. Il est important de vérifier que la capacité de mémorisation dans les commutateurs est suffisante pour assurer le bon fonctionnement du système.

6.6.4. Discussion

Le réseau à commutation de paquets permettant l'allocation de bande passante et la gestion des priorités possède de bonnes caractéristiques. Le temps de ré-assemblage est minimum lorsque les sources utilisent la file d'attente basse priorité, il est identique à celui du réseau à permutation synchronisé si les sources utilisent les files à débit réduit. Dans le premier cas, cette minimisation du temps de ré-assemblage se fait au détriment de la capacité de mémorisation requise au niveau des commutateurs. Dans le second cas, la taille mémoire requise demeure modeste (particulièrement avec ATM) sans qu'il soit nécessaire d'effectuer une synchronisation particulière des différents éléments connectés au réseau. La performance globale du système augmente linéairement avec la taille du système (pas de saturation comme dans la topologie en anneau). Dans le cas de la

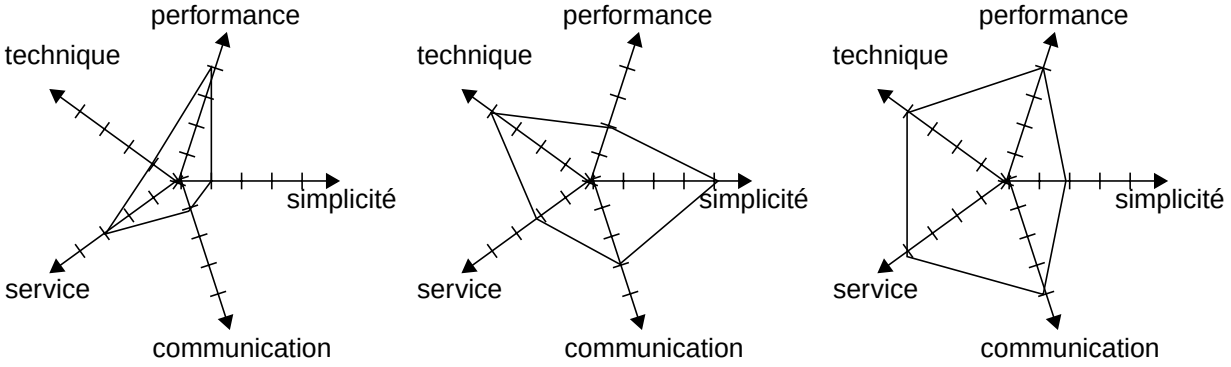
technologie ATM, les connexions point-à-point et point-à-multi-points sont réalisables en standard.

6.7. Comparaison des différents réseaux

Pour cette comparaison, 5 critères sont retenus et une note de 0 à 4 est affectée à chaque type de réseau. Les critères sont les suivants:

- performance
  Ce critère est relatif à la fréquence de ré-assemblage maximale (F_{arMax}) et à son évolution lors de l'agrandissement du système. L'anneau est le moins bien noté.
- simplicité (de la structure)
  L'anneau est le mieux noté car cette structure n'utilise pas de commutateur central.
- communication
  Ce critère est relatif aux communications point-à-point et point-à-multi-points (débit, délai,...).
- service
  Ce critère est relatif à la qualité du service fournie: allocation de la bande passante, risque de perte d'information en cas de débordement de mémoires tampon...
- technique
  Ce critère est représentatif de la qualité technique de la solution: effort de développement, utilisation d'une technologie courante,...

Les diagrammes comparatifs des différents réseaux sont présentés Figure 6-8.



a) Réseau à permutation synchronisé b) Anneau c) Réseau à multiplexage statistique

Figure 6-8: Diagrammes comparatifs des différents réseaux.

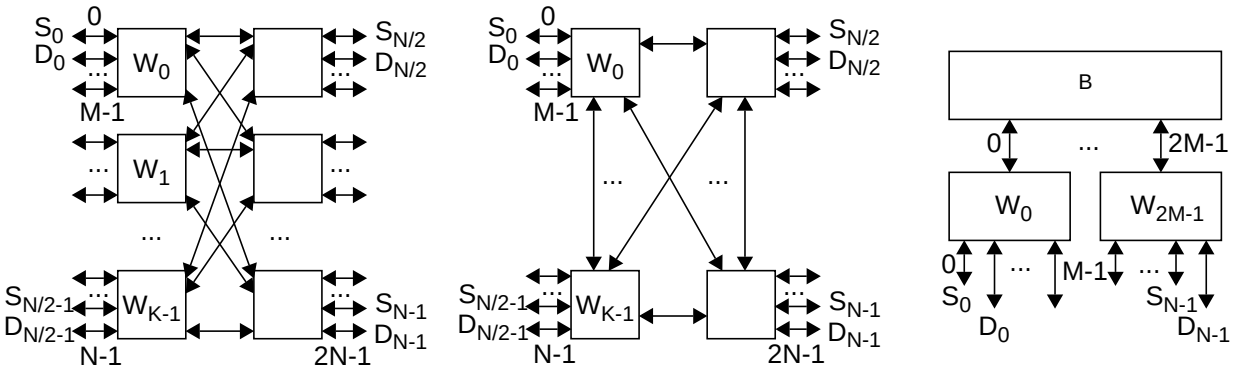
La topologie en anneau possède l'avantage de la simplicité mais présente des performances limitées. Le réseau à multiplexage statistique est plus complexe mais présente globalement les meilleures caractéristiques.

6.8. Assemblage de matrices de commutations

Un des problèmes pour la réalisation de systèmes de reconstruction d'événements de grande dimension, est la disponibilité de commutateurs comportant un nombre de ports suffisants. Compte tenu de la technologie actuelle, des commutateurs ATM disposant d'un maximum de ~200 ports 155 Mbit/s sont des éléments relativement courants et d'un coût abordable (~1300 US$ par port). On peut donc réaliser avec un seul commutateur de ce type un système ayant une capacité (théorique) de 15 Gbit/s comportant 100 paires source/destination. Des commutateurs disposant de quelques

centaines de ports sont disponibles pour le marché des télécommunications[1] (160 Gbit/s maximum est la configuration la plus courante), mais risquent d'être d'un coût prohibitif (prix de départ de 7.000 à 10.000 US$ par port 155 Mbit/s selon les informations actuelles). Une telle différence de prix est justifiée par la différence de qualité des produits, le marché visé, etc. L'apparition de commutateurs destinés au marché des réseaux d'entreprises offrant ~160 Gbit/s de bande passante et pouvant être équipés de plusieurs centaines de ports 155 Mbit/s est attendue mais n'est à ce jour pas encore effective. L'assemblage de commutateurs de taille modeste est une méthode qui pourrait se révéler plus économique que l'emploi d'un commutateur unique de forte capacité. De plus, le partage du système en plusieurs unités indépendantes (éventuellement mises au point dans des lieux différents) est plus facile à réaliser.

Les arrangements de commutateurs étudiés sont présentés Figure 6-9. Dans ces schémas, tous les nœuds sont du type source ou destination. En pratique, quelques-uns seraient du type superviseur ou moniteur, mais ce nombre est suffisamment faible pour ne pas être pris en compte dans cette étude. On cherche à évaluer le mérite relatif de chaque arrangement par la quantité totale de ressources pour la réalisation d'un *event builder* comportant N sources et N destinations: nombre de commutateurs, type et nombre de ports (P). On calcule le coefficient de foisonnement F_P par le rapport entre P et le minimum théorique (P_{min}=2N ports).



a) MIN à deux étages b) Réseau en étoile c) Dorsale + commutateurs d'accès

Figure 6-9: Assemblage de commutateurs.

6.8.1. MIN à deux étages

Le réseau Figure 6-9a est un MIN à deux étages semblable à celui de la Figure 6-6a, mais comportant des liens bi-directionnels. Les nœuds source et destination sont alternés sur les 2N ports du réseau formé par l'assemblage de 2K commutateurs à 2M ports (M.K=N). Suivant les cas, un seul port relie chaque matrice du premier étage au second (lorsque K=M), ou M/K ports sont groupés en parallèle pour effectuer cette liaison. On a:

$$F_P = \frac{2K \cdot 2M}{2N} = \frac{4N}{2N} = 2 \qquad \textbf{(6-13)}$$

Ce résultat est évident en observant que la moitié des ports de chaque commutateur est utilisée pour les liaisons entre commutateurs. On remarque également que pour un MIN à n étages, F_P vaut n ce qui justifie que seul le cas n=2 soit considéré dans cette étude.

1. Les fournisseurs sont par exemple: Alcatel, Ericsson, Fujitsu, Lucent, NEC, Siemens,...

6.8.2. Réseau en étoile

Le réseau Figure 6-9b comporte K commutateurs identiques connectés en étoile. Chaque commutateur est relié aux K-1 autres par des liaisons véhiculant le trafic de M/2 sources et M/2 destinations. Comme le trafic émis par les nœuds destination est négligeable par rapport à celui des sources, que tous les liens utilisés sont bi-directionnels et que sur chaque commutateur la moitié seulement des nœuds connectés sont des sources, on peut réaliser une économie sur le nombre de ports reliant les commutateurs. Il est suffisant de disposer d'une liaison de bande passante équivalente à M/2K ports entre tous les commutateurs pris deux à deux. On a:

$$F_P = \frac{K \cdot (M + (K-1) \cdot M/2K)}{MK} = \frac{3}{2} - \frac{1}{2K} \qquad \textbf{(6-14)}$$

Le coefficient de foisonnement F_P est minimal pour K=2 (il vaut alors 1,25) et tend vers 1,5 lorsque K augmente. Le réseau en étoile permet une économie sur le nombre de ports par rapport au MIN à 2 étages. La réduction du nombre de connexions entre commutateurs n'introduit pas de limitation dans le cas de la re-construction complète d'événements, mais peut constituer un goulet d'étranglement pour un autre type de trafic (à évaluer selon les cas).

6.8.3. Réseau à épine dorsale et commutateurs d'accès

Cette structure est présentée Figure 6-9c. Elle comporte un commutateur central (épine dorsale du réseau) relié à K commutateurs dont M ports sont utilisés pour M/2 sources et M/2 destinations. La liaison entre un commutateur d'accès et le commutateur central utilise (K-1)M/2K ports. Le commutateur central comporte (K-1)M/2 ports. On a:

$$F_P = \frac{(K-1) \cdot M/2 + K \cdot (M + (K-1) \cdot M/2K)}{MK} = 2 - \frac{1}{K} \qquad \textbf{(6-15)}$$

Du point de vue du nombre de ports, cet arrangement est intermédiaire entre le MIN à deux étages et le réseau en étoile. Cependant, il requiert un commutateur central (de forte capacité) en plus des K commutateurs d'accès.

6.8.4. Comparaison

Du point de vue de la performance, F_{aMax}, les trois arrangements sont strictement identiques. Le réseau en étoile est optimal au sens du nombre de ports de commutateurs. La structure avec un commutateur central et commutateurs d'accès nécessite davantage de ressources mais demeure attractive (connexion directe de certaines sources ou destinations au commutateur central et ajout de commutateurs d'accès pour accroître le nombre total de ports sans augmentation de la bande passante globale de l'*event builder*).

6.9. Résumé

Dans ce chapitre j'ai décrit différents réseaux adaptés à la re-construction d'événements. J'ai étudié leurs performances et les ai comparées. J'ai exposé une méthode de profilage du trafic permettant d'éviter la congestion des éléments du réseau et d'éliminer les pertes de messages potentiellement induites. J'ai montré comment connecter des matrices de commutation pour réaliser un réseau de plus grande dimension en minimisant le nombre d'inter-connexions entre matrices.

CHAPITRE VII. PROTOCOLE DE COMMUNICATION À HAUTE PERFORMANCE

7.1. Introduction

Dans ce chapitre, je justifie la nécessité de disposer d'un mécanisme d'échange de messages efficace pour réaliser un système de T/DAQ performant. Je présente différentes techniques permettant d'augmenter les performances, du point de vue de la communication, de machines connectées en réseau. Je décris l'implémentation d'un mécanisme de communication efficace basé sur ces techniques en utilisant du matériel en technologie ATM. Je présente les mesures de performance relatives à ce protocole de messagerie. Je donne les limites de cette approche d'optimisation et évoque les problèmes non résolus à ce jour.

7.2. Nécessité de communications efficaces

Comme cela a été exposé dans la description de l'architecture proposée pour le T/DAQ d'ATLAS au paragraphe 4.4, le système étudié fonctionne par des échanges permanents de messages entre les différents nœuds interconnectés par un réseau central. Les superviseurs distribuent aux processeurs des messages d'allocation d'événements à traiter; les processeurs envoient des requêtes de données aux sources, ces dernières retournent les données demandées, etc.

Une des caractéristiques importantes de l'application envisagée est le niveau de performance requis du point de vue de la communication entre les différents éléments du système. Le bloc des superviseurs doit être en mesure de distribuer les événements aux processeurs, collecter les décisions de trigger et les redistribuer aux sources à la fréquence de 100 kHz. Cela correspond à l'émission ou à la réception d'un message toutes les 5 µs en moyenne. Malgré la vitesse des processeurs modernes, cela demeure un objectif difficile à atteindre si l'on utilise des protocoles réseau classiques. La mise en parallèle de multiples superviseurs rend moins critique le besoin de performance pour chaque superviseur mais pose d'autres problèmes techniques.

Au niveau des sources de données groupant plusieurs ROBs, la contrainte est également relativement sévère. Les sources doivent répondre aux sollicitations des processeurs et traiter les messages contenant les décisions de trigger en provenance des superviseurs. Par le mécanisme de régions d'intérêt, seule une faible fraction des sources est sollicitée pour chaque événement. En faisant diverses hypothèses (détecteur considéré, groupement des ROBs...), la fréquence de requêtes à servir pour les sources les plus sollicitées est estimée à 10-20 kHz. Le temps de service moyen pour recevoir et analyser le message contenant une requête, obtenir les données des différents ROBs et retourner le message ainsi formé doit être au plus de 50 à 100 µs (un peu moins en pratique pour éviter la saturation du serveur). Le nombre de sources est déterminé par les caractéristiques du système de lecture des canaux des détecteurs. Le seul paramètre d'action au niveau des sources est le nombre de ROBs regroupés: plus le nombre de ROBs connectés à une source est élevé, plus la fréquence de sollicitation de cette source augmente. A la limite, il est possible de n'avoir qu'un seul ROB par source mais cela a diverses conséquences néfastes (accroissement du nombre de sources, donc de la taille du réseau, fragmentation plus importante des données).

Au niveau des processeurs de traitement, l'aspect de performance est lui aussi particulièrement important. Si la limitation du système, en terme de capacité de traitement d'événements, provient des processeurs (ce qui doit être le cas pour exploiter au mieux les données de l'expérience du point de vue de la physique), le temps moyen de traitement par événement détermine directement le nombre de processeurs requis pour absorber le flux total. Ce nombre influe également sur la taille du réseau et le coût global du système. En considérant qu'un nouvel événement à traiter arrive toutes les 10 μs en moyenne (trigger de niveau 1 fonctionnant à 100 kHz), chaque augmentation du temps de traitement moyen d'un événement de 10 μs entraîne l'ajout d'un processeur de traitement dans le système. Si l'on tient compte du fait que le taux d'occupation moyen des processeurs doit se situer autour de 60% afin de conserver des temps de réponse acceptables et préserver la stabilité du système, ce n'est l'ajout que de ~6 μs sur le temps de traitement moyen par événement qui entraîne l'augmentation d'une unité du nombre de processeurs. En plus des phases d'exécution des programmes relatifs à la physique, qui sont la tâche véritablement "utile" des processeurs, ces derniers effectuent un nombre important de communications pour dialoguer avec les superviseurs et les sources (plusieurs sources sont concernées à chaque étape de l'algorithme de sélection). Afin de réduire le nombre de processeurs et la taille du réseau, ou bien d'accroître le potentiel de traitement d'événements du système en disposant d'une quantité de ressources donnée, il est nécessaire de pouvoir disposer d'un protocole d'échange de messages performant.

Après avoir donné un ordre de grandeur du niveau de performance visé pour les communications entre les éléments du T/DAQ d'ATLAS, il faut comparer ces chiffres à ceux que l'on peut obtenir en utilisant les protocoles de communication standards. Le modèle de référence d'architecture de réseau est le modèle OSI en 7 couches [Puj95], mais en réalité, le modèle TCP/IP [Com91] s'est imposé au fil des ans comme standard de fait. De nombreuses études et mesures de performance sur TCP/IP [Cla89], [Cam97] font ressortir un certain nombre de points:

- la performance pour l'envoi et la réception de messages de faible taille (<1 ko) est le plus souvent limitée par le sur-débit (*overhead*) du protocole et non par la bande passante du lien. Par exemple, des mesures effectuées sur un PC moderne (400 MHz WindowsNT 4.0), montrent que le temps nécessaire pour envoyer des datagrammes de ~100 octets par la couche UDP/IP et Ethernet est de ~50 μs, la réception demandant ~100 μs.
- la charge sur le CPU de la machine hôte est élevée lors de l'envoi et de la réception de messages par TCP/IP à cause des multiples copies de données, du calcul de *checksum* par programme,...

D'après [Cam97] une station de travail Digital Alpha (175 MHz) est effectivement capable de saturer un lien ATM 155 Mbit/s lors d'un test d'émission continue de messages de 64 ko, mais cette simple tâche monopolise plus de 80% du temps du processeur.

L'utilisation de TCP/IP ou UDP/IP comme protocole réseau pour le T/DAQ d'ATLAS me paraît donc difficile aujourd'hui compte tenu des objectifs de performance. Malgré les progrès constants concernant la vitesse de traitement des processeurs et les améliorations possibles dans l'implémentation du logiciel des couches TCP/IP, je pense qu'un protocole de messagerie plus efficace est (et demeurera) nécessaire pour obtenir le niveau de performance souhaité. Les objectifs principaux de performance sont de permettre l'émission et la réception de messages de faible taille à fréquence élevée (>20 kHz), l'obtention d'un débit égal à la bande passante théorique du lien lors du transfert de messages de taille moyenne (~1 ko), et la minimisation de l'intervention du CPU local pour toutes les opérations de communication afin de conserver le maximum de capacité pour effectuer la partie de traitement utile des données des événements pour la physique.

7.3. Techniques d’optimisation des communications

7.3.1. Suppression des couches TCP-UDP/IP

La première étape dans l’optimisation d’un mécanisme de communication est la suppression des couches TCP-UDP/IP. Les communications se font en soumettant les messages à transmettre et en lisant les messages reçus au niveau de la couche liaison de données (couche 2 du modèle OSI) par manipulation directe de trames Ethernet ou de paquets AAL 5 en ATM. Cette approche entraîne la perte des nombreuses fonctionnalités de TCP/IP (établissement et terminaison des connexions, routage, contrôle de flux, transport de données fiable...) mais permet de gagner en performance.

La Figure 7-1[1] présente une comparaison de performance d’un simple programme d’échange de messages entre deux machines connectées par une liaison ATM en utilisant différents protocoles de communication. La machine client envoie des requêtes (1 cellule ATM) à la machine serveur qui retourne un bloc de données dont la taille est le paramètre que l’on fait varier. Le client envoie les requêtes en cascade afin de saturer le serveur pour s’affranchir de l’influence du temps d’aller-retour des messages à travers le lien ATM. Les machines client et serveur sont des stations de travail SUN Sparc 20 (50 MHz, Solaris I) équipées d’interfaces ATM 155 Mbit/s de la société Interphase.

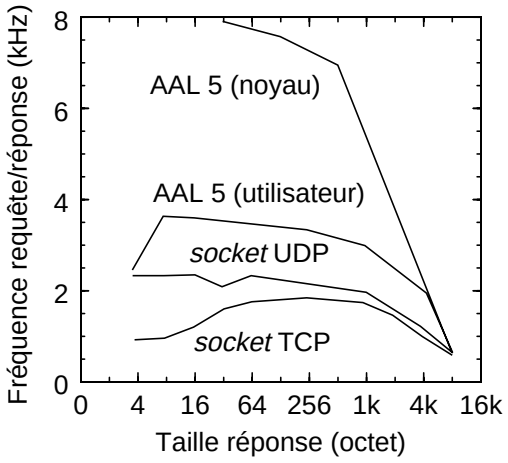


Figure 7-1: Performance comparée TCP/IP, UDP/IP et ATM AAL 5.

En utilisant TCP/IP au dessus d’ATM, le serveur est capable de retourner des messages de 128 octets au client à une fréquence maximale de 1,8 kHz, soit une bande passante effective au niveau du lien de ~2 Mbit/s. Avec UDP/IP, le gain est d’environ 10%. En effectuant les transferts directement au niveau de la couche AAL 5, la fréquence de service des requêtes est de 3,5 kHz. Par une optimisation supplémentaire consistant à faire fonctionner l’application serveur dans le noyau du système d’exploitation, il est possible d’atteindre ~7 kHz (soit un débit effectif de ~7 Mbit/s pour un lien 155 Mbit/s). On notera la très mauvaise performance de ces matériels au regard des besoins pour le T/DAQ d’ATLAS (serveurs de données devant fonctionner à 10-20 kHz). Cela justifie la démarche d’optimisation des communications qui avait été initiée à la suite de cette étude.

7.3.2. Réduction du nombre d’opérations de copie de données

Dans la plupart des implémentations des couches TCP/IP et des pilotes de cartes réseau, plusieurs recopies des messages sont effectuées par le processeur de la machine hôte. Du point de

1. présentation faite au meeting RD-31 les 18-19 Janvier 1996; disponible au CERN ou auprès de l’auteur.

vue de la performance, ces opérations sont pénalisantes car elles créent une charge de travail supplémentaire pour le processeur, la mémoire et les bus de la machine pour effectuer une tâche qui n'est pas "productive". Les processeurs modernes ont une fréquence de fonctionnement interne plusieurs fois supérieure à celle de leurs bus externes et les opérations de recopie de données faisant appel à la mémoire externe sont le plus souvent limitées par la capacité d'entrée/sortie du processeur. Un ordre de grandeur de performance pour la recopie de données de la mémoire externe vers le cache du processeur sur un PC fonctionnant à 400 MHz (carte mère 100 MHz) est ~50 Mo/s. Cela correspond à une utilisation du processeur de ~20 µs pour une copie d'un bloc de données de 1 ko. En considérant que pour l'application envisagée, la quantité de données en provenance des sources pour l'algorithme de sélection d'un événement est de ~10 ko et que l'on s'affranchit de 2 opérations de copie par optimisation du protocole de communication, le gain sur la durée de traitement de chaque événement est de ~400 µs soit une économie de 400/6=~66 processeurs au niveau du système. Ces chiffres approximatifs permettent de donner un ordre de grandeur du gain potentiel.

Supprimer le protocole TCP/IP standard permet de s'affranchir d'une ou deux opérations de recopie des messages (selon les implémentations), mais en général plusieurs copies subsistent. Dans le cas d'envoi d'un message, les opérations sont les suivantes:

- une première copie est effectuée au niveau de l'application pour formater le message à transmettre (couche "présentation" effectuant, entre autre, la conversion suivant l'ordre d'octets de la machine et l'ordre externe);
- une seconde copie est effectuée au niveau du pilote de la carte réseau pour conserver le message dans la mémoire du noyau du système d'exploitation;
- une troisième copie est faite depuis la mémoire du processeur hôte vers la carte réseau (en général par DMA sans l'intervention du CPU local);
- selon les modèles de carte réseau, le message est soit stocké temporairement dans la mémoire de la carte réseau avant envoi, soit il est envoyé depuis la mémoire de la machine hôte.

Dans le cas de la réception d'un message, les opérations sont:

- réception d'un message par la carte réseau et transfert par DMA vers la mémoire de la machine hôte (espace mémoire du noyau du système d'exploitation),
- génération d'une interruption par la carte réseau pour signaler l'arrivée du message,
- analyse du message et aiguillage vers l'application,
- recopie du message depuis l'espace mémoire du noyau vers une mémoire tampon accessible pour l'utilisateur,
- recopie/formatage du message pour être exploité par l'application.

L'opération de formatage des messages pour être exploitable par l'application est une opération indispensable. La seule opération de recopie qui peut sembler superflue est lors du passage entre l'espace utilisateur et l'espace du noyau du système d'exploitation. L'optimisation possible à ce niveau consiste à utiliser la même zone de mémoire visible à la fois par l'utilisateur, le pilote de la carte réseau et le matériel de carte réseau. L'envoi d'un message se fait directement par transfert de la zone mémoire fournie par l'application vers la carte réseau. La réception d'un message se fait dans une zone de la mémoire de la machine hôte directement exploitable par l'application.

Cette approche permet d'effectuer des transferts de messages en "zéro-copie", minimisant le temps de transfert et l'intervention du CPU des machines hôtes [Wel97]. Le désavantage de cette méthode non standard est de devoir développer des pilotes de carte réseau spécifiques et d'adapter l'application pour que la coopération entre l'application, le pilote de la carte réseau et le matériel soit possible. Au niveau de la carte réseau, le choix devra se porter en priorité sur les cartes équipées d'un DMA maître et utilisant directement la mémoire de la machine hôte pour le stockage de messages

sans utilisation d'une mémoire tampon intermédiaire située sur la carte réseau.

7.3.3. Contrôle direct de la carte réseau par l'application

Afin de découpler les programmes d'application d'un matériel périphérique, la méthode classique consiste à utiliser un pilote de périphérique (*device driver*). Pour des raisons de sécurité de fonctionnement, les opérations effectuées par le pilote sur le matériel ne sont pas accessible par l'utilisateur de manière directe. Le pilote est un morceau de code sûr fonctionnant au niveau du noyau du système d'exploitation, alors que les applications des utilisateurs sont sujettes à erreurs et ne doivent pas pouvoir perturber les autres tâches de la machine. Le passage de l'application au pilote nécessite un changement de contexte et différents appels au système d'exploitation qui prennent du temps (de l'ordre de ~10 µs suivant les machines et les systèmes d'exploitation).

Une optimisation possible afin de s'affranchir du temps passé lors de l'appel au pilote d'une carte réseau est de rendre visibles dans l'espace mémoire de l'application les ressources matérielles permettant le contrôle de la carte. L'utilisation de la carte réseau se fait alors par des appels à des fonctions d'une bibliothèque liée à l'application et non plus par l'intermédiaire d'un pilote. L'avantage en performance se fait au détriment de la sécurité de fonctionnement dans la mesure où l'application peut engendrer des erreurs pouvant entraîner le dysfonctionnement du matériel périphérique. De plus, le développement conjoint de l'application et du logiciel de contrôle de la carte réseau devient nécessaire pour assurer l'inter-opérabilité.

7.3.4. Gestion dynamique des interruptions de la carte réseau

La réception de messages à travers le réseau est une opération asynchrone par rapport aux tâches effectuées par le processeur hôte. Le mécanisme le plus couramment employé pour signaler l'arrivée d'un nouveau message est l'interruption. Une tâche sur le processeur hôte est consacrée à la lecture des messages venant du réseau. Lorsqu'il n'y a pas d'activité sur le réseau, cette tâche est en attente sur un élément de synchronisation (par exemple un sémaphore) et ne consomme pas de ressources CPU. Lors de l'arrivée d'un message, une interruption est générée par la carte réseau. La routine d'interruption signale la tâche en attente qui se charge du traitement du message reçu. Selon les modèles de carte réseau, il est possible de générer une interruption pour chaque paquet reçu, éventuellement après un certain laps de temps, ou lorsque le nombre de paquets reçus dépasse un certain seuil. Afin de minimiser le délai de prise en charge des messages, il est préférable d'informer le processeur de la machine hôte le plus tôt possible. Le traitement des interruptions et la signalisation d'une tâche en attente font appel à des primitives du système d'exploitation qui consomment des ressources CPU. Sur un PC 400 MHz fonctionnant sous WindowsNT, le temps CPU associé à la gestion de chaque interruption est de l'ordre de 40 µs.

L'optimisation possible à ce niveau consiste à n'autoriser la carte réseau à générer des interruptions que lorsque cela est nécessaire. Une première possibilité est d'interdire complètement ces interruptions. C'est l'application qui interroge périodiquement la carte réseau (*polling*) pour s'informer de l'arrivée de nouveaux messages. Ce mode de fonctionnement est adapté aux applications mono-tâche pour lesquelles le temps CPU non utilisé par l'application peut être consacré au *polling* sans perte de performance. Dans le cas d'une application multi-tâches (ou *multi-threads*), le recours aux interruptions demeure préférable au *polling* pour des raisons d'efficacité. Considérons la tâche dédiée à la réception des messages. Initialement, les interruptions de la carte réseau sont inhibées. La tâche de réception des messages commence par interroger la carte réseau. Tant que des messages sont disponibles, ils sont pris en compte et traités. Lorsque plus aucun message n'est disponible, la carte réseau est autorisée à générer des interruptions, et la tâche de

réception des messages se bloque alors en attente de signalisation. Dès qu'une interruption survient, les interruptions de la carte réseau sont inhibées. Les messages reçus sont traités jusqu'à épuisement et le cycle reprend. Au lieu de générer de manière systématique une interruption par message, cette méthode permet d'en réduire le nombre selon les fluctuations de l'intervalle de temps entre l'arrivée des messages et les traitements effectués par le processeur de la machine hôte. Dans le cas de tests effectués sur un démonstrateur qui sera décrit ultérieurement, le rapport entre le nombre de paquets reçus et le nombre d'interruptions générées est de l'ordre de 10, ce qui se traduit par un gain moyen sur le temps d'occupation du processeur de ~30 µs pour la réception de chaque message. Pour une application fonctionnant en mode *polling* sans interruptions, le gain en temps d'occupation du CPU est égal au temps de traitement de l'interruption soit ~40 µs par message.

7.4. Etude des communications en ATM

7.4.1. Choix d'une carte réseau ATM

Parmi les cartes ATM disponibles sur le marché, notre choix s'est porté sur celles qui utilisent le circuit de segmentation et ré-assemblage NicStar [IDT97] de la société IDT. Ce choix est justifié par:

- les caractéristiques du circuit NicStar
  L'interface PCI intégrée simplifie la carte réseau en réduisant le nombre de composants nécessaires. La possibilité de gérer plusieurs milliers de connexions virtuelles simultanément est nécessaire pour l'application envisagée car toutes les connexions qui seront utilisées doivent être établies par avance et rester actives pendant toute la durée de fonctionnement du système. Selon la quantité de mémoire incorporée sur la carte réseau, NicStar peut gérer jusqu'à 4096 (16384) connexions simultanées avec 128 ko (512 ko) de SRAM externe. Les cartes disponibles dans le commerce peuvent le plus souvent gérer 4096 connexions simultanées ce qui, en première approche, est jugé suffisant pour notre application. Afin de réduire les délais de communication, il est préférable que la segmentation et le ré-assemblage des paquets ATM utilisent directement la mémoire de la machine hôte et non de la mémoire embarquée sur la carte réseau. Ce mode d'opération est celui de NicStar qui dispose de canaux de DMA performants assurant les transferts de données pour l'émission et la réception de cellules ATM avec un minimum d'intervention de la part du processeur de la machine hôte.
- l'existence d'une documentation complète et publique
  La documentation relative au circuit NicStar, le manuel de programmation du circuit, l'information détaillant la conception des cartes d'évaluation intégrant NicStar sont la base nécessaire pour tout développement de matériel et de logiciel pilote. Selon les vendeurs de circuits de SAR, ces informations ne sont pas toujours disponibles pour l'ensemble du public. Leur mise à disposition se fait parfois par l'achat d'un système complet d'évaluation qui nécessite un investissement que nous avions jugé excessif lors de notre choix.
- l'utilisation de NicStar sur des cartes ATM de plusieurs vendeurs
  A ce jour, le circuit NicStar est utilisé sur certains modèles de cartes réseau de IDT, FORE Systems, 3Com, CES, Soliton, Prosum... Cela permet l'utilisation de notre logiciel pilote avec ces différents produits (avec cependant de très légères variations). Ce large choix de vendeurs et l'existence de plusieurs sources d'approvisionnement sont des considérations importantes pour assurer le suivi du produit pendant plusieurs années. Nous avons également développé une carte réseau basée sur NicStar au format *PCI Mezzanine Card* (PMC) [Cal97].

7.4.2. Connexions virtuelles

ATM est une technologie de réseau basée sur des communications en mode connecté faisant appel à une procédure d'établissement et de terminaison des connexions. La signalisation est spécifiée dans le document de l'ATM Forum UNI 4.0 [ATM95] qui s'appuie sur la recommandation ITU Q 2931 [ITU92]. Cette procédure est complexe et le temps d'établissement d'une *Switched Virtual Connection* (SVC – connexion virtuelle commutée) en ATM se chiffre en milli-secondes. Compte tenu des objectifs de performance pour l'échange de messages entre les différents nœuds du T/DAQ d'ATLAS, il n'est pas envisageable d'utiliser l'établissement dynamique des connexions ATM pour cette application. La totalité des connexions virtuelles nécessaires doit être établie par avance et demeurer figée pendant toute la durée de fonctionnement du système. Il est tout à fait possible d'établir des SVCs qui resteront actives pendant de longues périodes. Une autre possibilité est d'utiliser des connexions virtuelles permanentes (*Permanent Virtual Connections* – PVCs) qui peuvent être mises en place sans faire appel à la procédure de signalisation, par programmation directe des commutateurs et des cartes réseau ATM. Dans notre cas, le logiciel pilote de carte réseau utilisé est un développement spécifique optimisé, et aucun mécanisme de signalisation n'a été implanté à ce jour, afin de réduire la tâche de programmation. La solution adoptée est donc d'utiliser des PVCs, ce qui, du point de vue fonctionnel et des performances, est équivalent à l'utilisation de SVCs pré-établies. La seule différence concerne la configuration du système: la programmation et la maintenance des PVCs étant plus lourde et moins facile à automatiser que l'utilisation de SVCs.

7.5. Logiciel pilote

7.5.1. Principe

Afin de prendre en main les cartes réseau basées sur NicStar et évaluer les performances de communication en utilisant un logiciel non optimisé, une première version de logiciel pilote a été développée. Le principe de fonctionnement est présenté Figure 7-2.

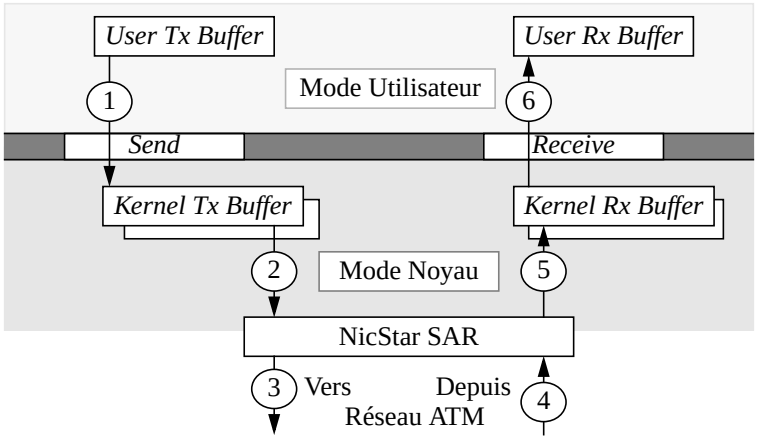


Figure 7-2: Fonctionnement du pilote de carte réseau ATM.

Une opération d'envoi de message se passe comme suit. Le *buffer* contenant le message à envoyer a été au préalable rempli par l'application. Le point d'entrée *Send* du logiciel pilote de la carte réseau est appelé en passant comme paramètres le message à transmettre et le VCI/VPI déterminant le destinataire du message (flèche 1 sur la Figure 7-2). La suite des opérations s'effectue dans le mode sécurisé du système d'exploitation (*kernel mode*). Le message est recopié depuis le

buffer fourni par l'utilisateur (UTB – *User Transmit Buffer*) vers un *buffer* du pilote (KTB – *Kernel Transmit Buffer*); NicStar est notifié de la demande d'envoi d'un message (flèche 2) et la fonction *Send* se termine par le retour à l'application. Le UTB contenant le message peut être immédiatement détruit ou ré-utilisé. L'envoi du message par NicStar est asynchrone et dépend de la bande passante allouée à la connexion virtuelle considérée. Le retour de l'opération *Send* à l'application ne signifie nullement que le message a été envoyé. Il est à l'évidence exclu de n'effectuer le retour de l'opération d'envoi que lorsque le message a été effectivement transmis car le temps d'envoi réel d'un message sur le lien ATM peut être très long selon la taille du bloc transmis et la vitesse de segmentation (flèche 3). Lorsque le message a effectivement été transmis depuis le KTB vers le réseau ATM par NicStar, le KTB peut être utilisé à nouveau. L'opération de copie des données des UTBs vers les KTBs permet de découpler le déroulement de l'application de celui de l'envoi effectif du message. Les KTBs ont des caractéristiques particulières et indispensables au bon fonctionnement des opérations (plages d'adresses mémoire contiguës, *cacheable*, résidents en mémoire centrale, adresses physiques connues...) que n'ont pas les UTBs.

L'opération de réception d'un message est la suivante. Un certain nombre de *Kernel Receive Buffers* (KRBs) ont été pré-alloués. Leurs caractéristiques (adresse physique connue, etc) sont semblables à celles des KTBs. Lors de la réception de cellules ATM (flèche 4), NicStar transfère directement la charge utile dans un ou plusieurs KRBs en effectuant le ré-assemblage des messages selon le champ VPI/VCI de chaque cellule (flèche 5). NicStar place dans une file d'attente accessible par le processeur de la machine hôte la liste des KRBs remplis ou contenant des messages complets. Lors de l'appel au point d'entrée *Receive* du pilote, l'utilisateur fournit un *User Receive Buffer* (URB) de taille suffisante pour contenir le plus grand message attendu. Si un message est disponible, les données sont recopiées du (ou des) KRB(s) vers le URB; le(s) KRB(s) est (sont) recyclé(s) et le retour à l'application est effectué (flèche 6) en précisant de quelle connexion virtuelle le message a été reçu (VPI/VCI). Si aucun message n'est disponible, deux modes de fonctionnement sont possibles. En mode bloquant, le retour n'est effectué que lorsqu'un message est disponible; en mode non bloquant le retour est immédiat avec un code signalant qu'aucun message n'est disponible. Pour des raisons de performance, le mode bloquant est basé sur des interruptions. Le mode non-bloquant est plus simple; le soin d'effectuer l'interrogation périodique de la carte réseau est laissé à l'application (*polling*). La séparation entre URB et KRB permet, comme dans le cas de la transmission, de découpler le fonctionnement de la carte réseau de celui de l'application.

7.5.2. Réalisation et performance

L'implantation du logiciel pilote a été faite pour les systèmes d'exploitation WindowsNT et Lynx OS. Les tests ont été effectués sur un PC et un ordinateur sur carte au format VME dont les caractéristiques principales sont répertoriées Table VII-1.

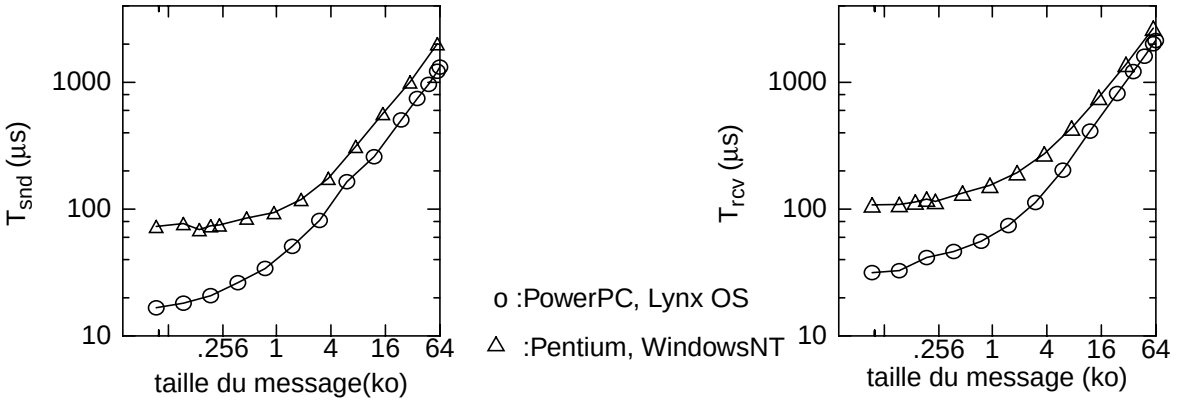
Machine	Fabricant	CPU	Fréquence	Système d'exploitation
VME SBC	CES	PowerPC 604	96 MHz	Lynx OS 2.3
PC	Dell	Pentium	133 MHz	WindowsNT 4.0

Table VII-1: Machines utilisées pour le test du pilote de carte ATM NicStar.

L'opération de NicStar est concurrente avec celle du CPU de la machine hôte et ce composant est entièrement dédié à sa tâche. A l'inverse, l'utilisation du processeur hôte pour les communications est une perte (nécessaire) de ressources de calcul. Les mesures de performance

visent donc en priorité à établir l'impact des opérations de communication sur l'utilisation du processeur de la machine hôte en mesurant le temps d'occupation du CPU pour l'envoi et la réception de messages de taille variable.

Le temps d'utilisation du CPU de la machine hôte pour l'envoi d'un message, T_{snd}, est présenté Figure 7-3a. Ce temps est mesuré depuis l'appel à la fonction *Send* jusqu'à son retour et n'a pas de relation directe avec le temps d'émission effective du message sur le lien ATM. Pour les messages de faible taille, la performance est limitée par le temps de passage dans le logiciel pilote (changement de contexte entre le mode utilisateur et le mode sécurisé du système d'exploitation).



a) T_{snd} vs. taille du message

b) T_{rcv} vs. taille du message

Figure 7-3: Performance du logiciel pilote.

La fréquence maximale d'envoi de messages de faible taille (inférieure à 256 octets) est de l'ordre de 15 kHz pour le PC, et de 60 KHz pour la carte VME. Pour le PC, ce chiffre n'est pas satisfaisant: ~80 µs de temps CPU est consommé à chaque envoi. La performance pour l'envoi de messages de grande taille (supérieure à 4 ko) est déterminée par la vitesse de recopie des données entre les UTBs et KTBs. Celle-ci est de ~28 Mo/s pour le PC et 36 Mo/s pour la carte VME. L'utilisation du CPU est proportionnelle à la taille du message émis et constitue une perte de capacité de calcul importante. La transmission sur le lien ATM se fait à 155 Mbit/s au maximum et la saturation du lien est atteinte pour des messages de taille supérieure à ~2 ko.

L'utilisation du processeur local lors d'une opération de réception, T_{rcv}, est présenté Figure 7-3b. La réception se fait en mode non bloquant (i.e. sans interruptions). Le temps T_{rcv} est mesuré depuis l'appel à la fonction *Receive* jusqu'à son retour. La performance pour la réception de messages de faible taille est limitée par le temps d'appel au logiciel pilote et est déterminée par la vitesse de recopie des données pour les messages de taille plus importante.

Compte tenu de ces résultats jugés médiocres, ce logiciel pilote pour NicStar a été modifié en utilisant les techniques d'optimisation précédemment exposées.

7.6. Pilote et bibliothèque de fonctions optimisées

7.6.1. Principe

7.6.1.1. Gestion de la mémoire minimisant les recopies de données

Cette amélioration vise à réduire l'occupation du CPU local causée par la recopie de données entre les *buffers* utilisés par NicStar et ceux utilisés par l'application. En rendant visibles dans

l'espace de mémoire virtuelle de l'application les *buffers* utilisés par NicStar, cette opération de recopie peut être supprimée. L'application écrit et lit les messages directement dans les *buffers* qui sont manipulés par NicStar. Cette approche introduit un certain nombre de complications. Dans le cas de l'envoi de message, le *buffer* contenant le message n'est pas ré-utilisable dès que la fonction *Send* retourne car la transmission par NicStar est asynchrone. Un mécanisme approprié doit être utilisé pour effectuer le recyclage des *buffers* au moment opportun. Pour la réception de messages, l'application doit accepter les messages dans le format délivré par NicStar, c'est-à-dire sous forme d'un ou de plusieurs *buffers* chaînés et non d'une zone mémoire constituée d'emplacements consécutifs. Lors de la réception d'un paquet AAL 5, sa taille n'est pas connue à l'avance. Il serait peu économique d'utiliser pour chaque message un *buffer* pouvant contenir 64 kilo-octets (taille maximale d'un paquet AAL 5). NicStar utilise un mécanisme plus astucieux faisant appel à des réserves de petits *buffers* et grands *buffers*. Les premières cellules d'un nouveau message sont stockées dans un petit *buffer* (capacité de 240 octets par exemple). Si cela se révèle insuffisant, NicStar puise au fur et à mesure des besoins dans une réserve de grands *buffers* (capacité de 4 ko par exemple). Un message complet peut donc être composé de plusieurs blocs mémoire situés à des adresses (physiques et virtuelles) non consécutives. Ces difficultés supplémentaires sont à prendre en compte pour effectuer les communications sans recopie de données.

7.6.1.2. Contrôle de NicStar par l'application

Cette optimisation vise à améliorer la performance pour la manipulation des messages de faible taille. Afin d'éviter les changements de contexte *user/kernel* à chaque appel au pilote de la carte réseau, les ressources nécessaires sont rendues visibles dans l'espace mémoire de l'application. Ces ressources sont les registres de NicStar, les files d'attente de messages à transmettre, la file d'attente des messages reçus, celle des *buffers* disponibles pour la réception... Un point d'entrée du pilote (*ioctl – Input Output ConTrL*) permet de retourner à l'application la structure contenant toutes ces informations. L'appel à ce point d'entrée doit être fait à chaque démarrage de l'application. A la fin de l'application, un appel symétrique permet de libérer les ressources allouées. Pour des raisons de simplicité lors de l'implémentation, seule une tâche peut avoir le contrôle de la carte NicStar à un instant donné: l'opération de mise à disposition des ressources n'est pas prévue pour les applications multi-tâches. Pour néanmoins permettre une certaine flexibilité, l'opération en *multi-threads* est possible (avec certaines limitations). L'utilisation sur des machines mono et multi-processeurs est possible sans contrainte particulière. Certains registres de NicStar sont utilisés à la fois pour l'émission et la réception. De ce fait l'émission et la réception simultanée par deux *threads* totalement indépendants n'est pas possible. Un mécanisme d'accès exclusif aux registres de NicStar est implanté pour garantir le fonctionnement concurrent de multiples *threads* pour l'émission et d'un *thread* unique pour la réception. En principe, il est envisageable d'avoir plusieurs *threads* pour la réception, mais en pratique, il est plus simple de n'en n'utiliser qu'un seul effectuant la réception de tous les messages venant du réseau et se chargeant de les redistribuer selon leur origine, leur contenu ou toute autre méthode d'aiguillage propre à l'application considérée.

7.6.1.3. Gestion optimisée des interruptions

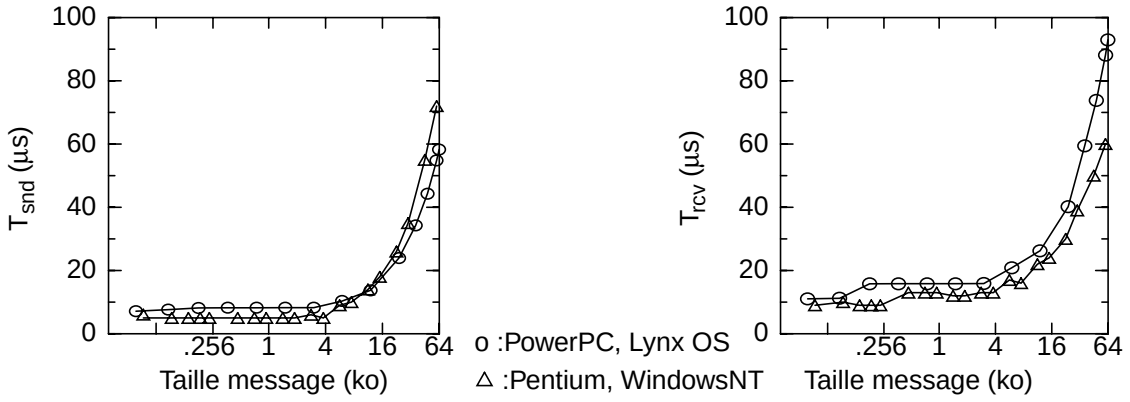
Cette optimisation vise à gérer la réception en mode non bloquant ou en mode bloquant. Dans le mode de réception non bloquant, les interruptions de NicStar signalant la réception de paquets complets sont inhibées de sorte que le processeur de la machine hôte n'est jamais dérangé par cette activité. En mode non bloquant, les opérations sont plus complexes car la gestion des interruptions au niveau d'une tâche utilisateur n'est pas possible avec les systèmes d'exploitation utilisés. La méthode est donc la suivante: les interruptions de NicStar étant inhibées, on effectue une première

opération de réception de message en mode non bloquant. Si un message est disponible, il est pris en compte immédiatement. Si aucun message n'est disponible, les interruptions de NicStar sont autorisées, et un appel au pilote de la carte ATM est effectué. Après le passage en mode noyau, la tâche de réception se met en attente (sémaphore). Lorsqu'un message complet est arrivé, NicStar génère une interruption au processeur de la machine hôte. La routine d'interruption commence par inhiber les interruptions de NicStar puis signale le sémaphore qui bloquait en attente la tâche de réception du pilote. Cette tâche est alors activée et effectue le retour en mode utilisateur. Une nouvelle opération de lecture en mode non bloquant est effectuée pour lire le message reçu. Cette méthode rend possible l'opération en mode bloquant pour une tâche utilisateur en minimisant le nombre d'interruptions générées ainsi que le nombre d'appels au pilote (élément incontournable pour la gestion des interruptions).

7.6.2. Réalisation et performance

L'implantation du logiciel pilote optimisé a été faite pour les machines mentionnées dans la Table VII-1. Le logiciel comporte deux parties: le pilote effectuant l'initialisation de la carte réseau et la gestion des interruptions et la bibliothèque de fonctions optimisées. Les prototypes des fonctions optimisées doivent être inclus lors de la compilation de l'application, et la bibliothèque correspondante doit être ajoutée lors de l'édition de liens pour générer le programme exécutable. Les détails techniques et les spécifications de l'interface de programmation (*Application Programming Interface* – API) sont documentés dans [Cal98].

Le temps d'occupation du CPU de la machine hôte pour l'envoi (la réception) de messages de taille variable est présenté Figure 7-4a (7-4b).



a) T_{snd} vs. taille du message b) T_{rcv} vs. taille du message

Figure 7-4: Performance de la bibliothèque optimisée.

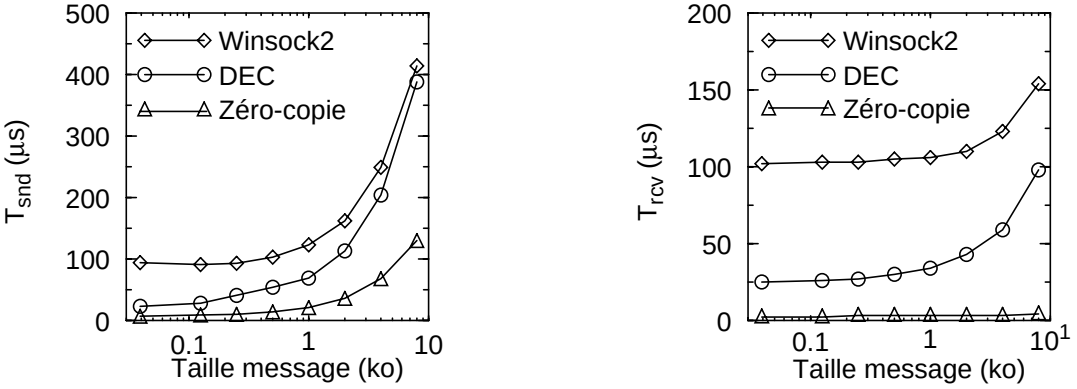
La réception de messages est effectuée en mode non bloquant. Pour une taille de message inférieure à ~4 ko, T_{snd} et T_{rcv} sont approximativement constants. Pour le test sur un PC, le temps d'utilisation du CPU local est réduit d'un facteur ~10 par rapport au pilote non optimisé. Pour les messages de plus grande taille, la manipulation de listes chaînées de blocs de la taille d'une page (4 ko) entraîne une dégradation des performances. Elles demeurent cependant nettement supérieures à celles obtenues avec le pilote non-optimisé: un gain d'un facteur ~20 sur le temps d'utilisation du processeur local pour les communications est observé. D'autres mesures comparant les performances du pilote et de la bibliothèque de fonctions de communication optimisées sont rapportées dans [Cal97a].

Une étude comparative entre divers produits disponibles dans le commerce a également été

effectuée [Cal98b]. Les matériels testés sont les suivants:

- une station de travail DEC Alpha – 333 MHz (Digital Unix) équipée d'une carte réseau ATMWorks 350 et utilisant l'API d'accès direct à la couche ATM AAL 5 fournie par DEC,
- un PC Pentium II – 300 MHz (WindowsNT) équipé d'une carte réseau ATM FORE PCA 200 et utilisant l'interface de programmation Winsock2 pour un accès à la couche ATM AAL 5,
- un PC identique au précédent mais équipé d'une carte ATM basée sur NicStar et utilisant la bibliothèque de fonctions de communication optimisées (zéro-copie).

Les mesures effectuées, T_{snd} et T_{rcv}, sur ces différentes machines sont présentées Figure 7-5.



a) T_{snd} vs. taille du message

b) T_{rcv} vs. taille du message

Figure 7-5: Performance de pilotes du commerce et de la version optimisée.

Pour des raisons techniques, la taille des messages qui peuvent être manipulés par le pilote fourni par DEC est limitée à 8 ko. Les performances des produits de DEC sont relativement satisfaisantes, et nettement supérieures à celles du produit de FORE Systems. Dans tous les cas, la version optimisée du logiciel de contrôle est celle qui permet les communications les plus efficaces.

7.7. Discussion

Malgré un gain de performance significatif, l'adoption de l'approche proposée pour l'optimisation des communications comporte un certain nombre de revers.

7.7.1. Développement spécifique

Le développement effectué est spécifique et non standard. Le logiciel ne peut fonctionner qu'avec le circuit de SAR choisi. En plus de l'effort de développement, la maintenance, l'évolution du logiciel et l'utilisation avec d'autres modèles de carte réseau n'est pas très facile. Compte tenu des performances des cartes réseau et des logiciel pilotes actuels et de l'évolution des performances observée sur les dernières années, je pense que les optimisations proposées demeureront nécessaires pour la réalisation du système de sélection d'événements de l'expérience ATLAS. Les prévisions concernant l'évolution de la technologie à l'échelle de plusieurs années étant toujours fort difficiles, je pense que l'attitude à adopter est de concevoir les applications de sorte qu'elles puissent utiliser aussi bien le modèle de communications optimisées (qui est le plus restrictif et doit donc être celui considéré lors de la conception de l'application) qu'une approche basée sur une solution standard (API de Winsock2 par exemple).

7.7.2. Configuration du système; compatibilité avec les protocoles standards

A présent, seule l'utilisation de connexions virtuelles permanentes est possible avec la bibliothèque de fonctions de communication optimisées. La gestion des connexions permanentes est une tâche assez lourde et difficile à automatiser. Cela pourrait causer des difficultés dans un système à l'échelle du trigger d'ATLAS. Le support de SVCs avec la bibliothèque optimisée demande un certain effort de développement. Pour le moment, je pense qu'il est prématuré de faire cet investissement, et il me semble que l'utilisation de PVCs est une solution acceptable.

Une possibilité intéressante est de pouvoir accéder à la carte réseau non seulement au niveau de la couche ATM AAL 5 mais aussi au travers des couches de protocoles standards (*LAN Emulation*, *Multiprotocol Over ATM* – MPOA [Sch98]) pour permettre l'utilisation simultanée des applications classiques fonctionnant au dessus de TCP-IP en plus de celles, optimisées, fonctionnant en accès direct à la couche d'adaptation ATM. Pour l'application envisagée, cela permet par exemple d'utiliser un réseau unique à la fois pour les transferts de données rapides et les tâches moins prioritaires telle la configuration (accès à une base de données) et la surveillance du système. Ces deux modes d'accès simultanés au réseau sont possibles avec divers produits du commerce (DEC, FORE Systems, et en général l'ensemble des cartes réseau supportant les accès à la couche ATM AAL 5 par l'interface Winsock2). L'utilisation de LAN Emulation ou MPOA avec le bibliothèque de communications optimisées serait en principe possible, mais demande un important effort de développement. Ce travail n'est pas envisagé à ce jour. La solution la plus simple pour faire co-exister sur la même machine une application utilisant la bibliothèque de communication optimisée et les applications classiques basées sur TCP/IP est de disposer d'un réseau secondaire (Ethernet par exemple). Dans notre cas, les performances requises pour ce réseau secondaire sont modestes par rapport à celles demandées pour le réseau principal. En tenant également compte du fait que la plupart des machines du commerce sont équipées en standard d'une interface Ethernet, le surcoût entraîné par la mise en place de ce réseau secondaire me semble acceptable.

7.7.3. Transfert de messages non garanti

Une autre limitation de l'approche proposée pour les communications est de n'offrir aucune garantie pour les transferts de bout-en-bout. Les échanges de messages se faisant au niveau de la couche ATM AAL 5, aucune re-transmission de message n'a lieu en cas d'erreur ou de perte de cellules. La mise en place d'un protocole de transport fiable est envisageable, mais n'est pas facile à réaliser si l'on souhaite conserver la caractéristique de transferts sans re-copie de données.

Le plus simple serait de revenir à l'utilisation des couches TCP/IP classiques qui offrent ces fonctionnalités et d'accepter les performances (médiocres à ce jour) des produits du commerce. Une autre façon de résoudre le problème est de considérer que l'application envisagée peut tolérer les pertes occasionnelles de messages. La perte d'une fraction des données de quelques événements à cause d'erreurs de communication me semble acceptable si ces erreurs sont aléatoires et n'introduisent pas de biais dans la chaîne de mesure de l'expérience. Les erreurs de transmission sont une des nombreuses sources de temps mort du système d'acquisition de données. Le temps mort acceptable pour le trigger de niveau 2 d'ATLAS est fixé arbitrairement à 1% [Abo00]. Une étude pour estimer le temps mort induit par les erreurs de communication serait fort complexe et dépasse le cadre de cette thèse. Dans le cas de la technologie ATM, on peut distinguer au moins deux causes de perte de messages: les erreurs survenant au niveau de la transmission elle-même (bruit dans les composants électroniques et optiques), et les destructions forcées de cellules ATM à cause d'un débordement de capacité de mémoire tampon ou de la non conformité de la cellule par rapport au contrat de qualité de service de la connexion virtuelle correspondante. Les erreurs du premier type

sont inévitables (quantifiées par le *Bit Error Rate* – BER) alors que les destructions forcées de cellules ATM peuvent, en principe, être éliminées. Des méthodes adaptées au type de trafic rencontré dans l'application envisagée pour minimiser (voire éliminer) cette seconde source d'erreurs de transmission seront présentées dans la suite de ce travail. Concernant les erreurs inévitables, le BER des liens ATM sur SDH/SONET est estimé entre 10^{-13} et 10^{-16}, ce qui malgré un débit par lien supérieur à 100 Mbit/s et le nombre de liens utilisé (~1000), ne devrait pas induire de perte d'information inacceptable.

7.8. Résumé

Dans ce chapitre, j'ai montré la nécessité de mise en place d'un mécanisme de communication optimisé pour satisfaire le besoin de performance de notre application. J'ai décrit les techniques permettant de réaliser ces optimisations et les ai mises en œuvre sur du matériel en technologie ATM. J'ai présenté les performances obtenues et les ai comparées à celles de produits disponibles dans le commerce. J'ai décrit les contraintes et les limitations imposées par l'utilisation de ce mécanisme de communication optimisé.

CHAPITRE VIII. DÉMONSTRATEURS DU MODÈLE DU *TECHNICAL PROPOSAL*

8.1. Introduction

Dans ce chapitre, je présente les systèmes de démonstration qui ont été assemblés par différents groupes de la collaboration ATLAS pour étudier les modèles d'architecture et les options technologiques du T/DAQ de cette expérience. Le schéma du *Technical Proposal* étant la base acceptée par la collaboration lors de la construction de ces prototypes, des projets distincts pour le trigger de niveau 2 et celui de niveau 3 ont été mis en place. Je n'ai pas contribué à la construction de ces prototypes et je présenterai dans le chapitre suivant ma contribution relative aux démonstrateurs de l'architecture proposée dans cette thèse.

8.2. Modèle flot de données du trigger de niveau 2

8.2.1. Présentation

Une description complète de ce modèle est rapportée dans [Kug98] et [Kug98a]. L'architecture exploite le concept de traitement parallèle des RoIs par détecteur suivi de la combinaison des résultats pour donner une vue globale sur chaque événement. Comme il n'est pas prouvé que des processeurs séquentiels classiques et matériels réseau standards auront les performances requises pour le système envisagé, les concepteurs de ce modèle proposent une solution de repli dans laquelle la majorité des éléments critiques sont implantés en logique câblée (concept semblable au trigger de niveau 1). Les réseaux pour la collecte des données des RoIs sont réalisés par du matériel spécifique. Un prototype de *"RoI Collector"* est rapporté dans [Kug98a]. Les algorithmes de traitement des données des RoIs sont implantés dans des machines à base de circuits logiques programmables (*Field Programmable Gate Arrays* – FPGAs). L'utilisation de FPGAs permet une certaine flexibilité au niveau des algorithmes mais en limite la complexité compte tenu de la quantité de ressources matérielles disponible dans ce type de composant. Bien que plus difficiles à programmer que les ordinateurs classiques, les machines à base de FPGA sont particulièrement efficaces pour certaines tâches: manipulation de bits, algorithmes basés sur des tables (*Look Up Tables* – LUTs) ou se prêtant au parallélisme massif, tâches nécessitant une grande bande passante d'entrée/sortie. Au niveau de la combinaison des résultats du traitement des RoIs, l'utilisation de processeurs séquentiels classiques est envisagée car ces éléments semblent avoir des performances suffisantes et offrent davantage de flexibilité que de la logique câblée.

La possibilité d'utiliser des machines à base de FPGAs ne se limite pas au modèle du trigger de niveau 2 du *Technical Proposal* d'ATLAS. Une variante est l'utilisation de FPGAs pour un système de pré-sélection (*pre-scaler*) en sortie du trigger de niveau 1. Une vue schématique de ce système est présentée Figure 8-1. Compte tenu de la relative simplicité imposée aux algorithmes de traitement des RoIs réalisables avec des FPGAs, le facteur de rejet attendu avec ce *pre-scaler* est moindre que celui visé avec un système permettant l'implantation d'algorithmes plus évolués. A l'aide d'algorithmes simples exploitant les données des RoIs dans les calorimètres, le spectromètre

à muons et le *Transition Radiation Tracker*, l'utilisation de machines à base de FPGAs permettrait de réduire d'un facteur ~10 le taux d'événements après le trigger de niveau 1. Cette option pourrait diminuer sensiblement les besoins en puissance de calcul et en bande passante de réseau pour les triggers situés en aval (au prix de l'ajout d'un système de pré-sélection intermédiaire).

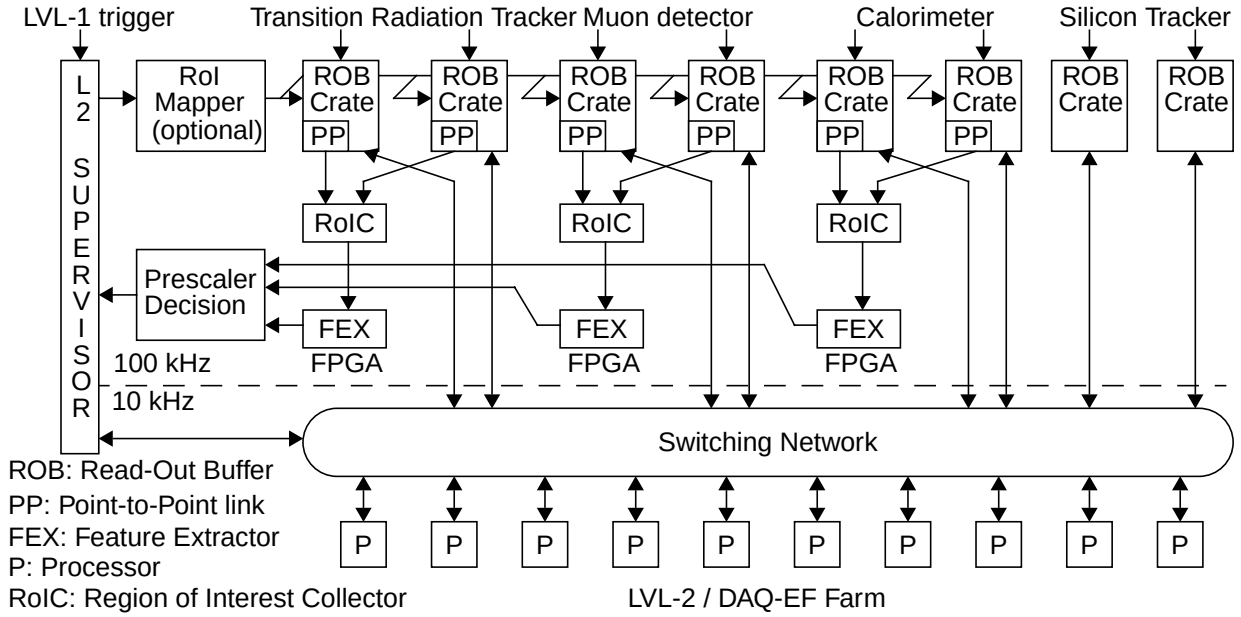


Figure 8-1: Trigger de niveau 2 et 3 avec *pre-scaler* basé sur des FPGAs.

Une seconde variation de l'architecture du trigger de niveau 2 à base de FPGAs est l'utilisation de machines re-programmables pour certains algorithmes nécessitant des ressources importantes. Les FPGAs viennent en complément des processeurs séquentiels classiques. La recherche initiale des traces, de manière exhaustive, dans le TRT (*TRT full scan*) est un algorithme bien adapté à une implantation en logique câblée (volume de données important, parallélisme massif, utilisation de tables) [Hin99]. Par rapport à un processeur classique, un gain d'un facteur ~20-100 sur le temps de traitement a été rapporté en utilisant la machine Enable++ [Kug98a]. Pour les événements relatifs à la physique du méson B, l'utilisation d'un sous-ensemble spécial basé sur des FPGAs effectuant le *TRT Full Scan* est une option qui permettrait de décharger le trigger de niveau 2 de cette tâche particulièrement lourde.

8.2.2. Critique

L'utilisation de machines à base de FPGAs permet d'améliorer sensiblement le temps d'exécution de certains algorithmes. En revanche la sophistication des algorithmes est assez limitée. La programmation de ce type de machine nécessite souvent des connaissances dans des langages de description de matériel (par exemple VHDL) ce qui complique le transcodage des algorithmes relatifs à la physique. Des améliorations sont en cours pour rendre la programmation plus aisée mais un outil permettant l'implantation automatique dans une machine à base de FPGAs d'algorithmes écrits en C++ ou en Fortran n'est pas encore opérationnel.

Du point de vue technique, la réalisation de réseaux pour la collecte de données en logique câblée n'est pas triviale et demande un effort important de conception et de développement. La faisabilité reste à démontrer.

Au niveau technologique, les progrès sont constants aussi bien dans le domaine des FPGAs que dans celui des processeurs classiques. La fréquence de fonctionnement des processeurs

séquentiels d’aujourd’hui est supérieure à 600 MHz, alors que les cartes électroniques à base de FPGAs fonctionnent à ~100 MHz. Il ne me semble pas certain que l’avantage des FPGAs par rapport aux processeurs classiques sera conservé dans le futur. Même si cet avantage demeure, les PCs offrant une puissance de calcul importante pour un coût très faible, il n’est pas certain que du point de vue économique, des machines à base de FPGAs soient plus compétitives que l’ensemble des PCs qu’elles pourraient remplacer.

Concernant la flexibilité, la minimisation de l’effort de développement, la maintenance à long terme et les possibilités d’évolution du système, une architecture à base de matériel dédié est à l’évidence inférieure à une solution utilisant du matériel industriel standard.

Il me paraît clair que les FPGAs seront largement utilisés dans le T/DAQ d’ATLAS comme composants électroniques. Concernant l’utilisation de FPGAs en tant que machines de traitement de données, je suis plus réservé. L’architecture décrite ci-dessus est complexe, relativement figée et nécessite de nombreux développements de matériel et de logiciel (en particulier la construction des réseaux de collecte de données et de contrôle). Néanmoins, le *TRT Full Scan* est une tâche problématique pour le trigger de niveau 2 d’ATLAS compte tenu des ressources nécessaires. Une utilisation possible de machines à base de FPGAs est un système dédié à ce traitement. Il pourrait s’agir d’un bloc séparé du trigger de niveau 2, relié à l’ensemble des *Read-Out-Drivers* du TRT par des liens dédiés (en dupliquant l’information destinée aux ROBs). En sortie, les résultats seraient envoyés vers un ROB particulier pour donner au trigger de niveau 2 l’accès à la liste des traces identifiées par ce système pour chaque événement. La faisabilité et l’intérêt de cette approche est une étude qui dépasse très largement les objectifs de cette thèse.

8.3. Modèle “local-global” du trigger de niveau 2

8.3.1. Présentation

Ce modèle est la réalisation du trigger de niveau 2 tel qu’il est décrit dans le *Technical Proposal* d’ATLAS. Il exploite le parallélisme de traitement au niveau de la multiplicité des RoIs et au niveau des détecteurs pour chaque RoI. Le processus de sélection est partagé entre des processeurs effectuant des traitements locaux et des processeurs combinant les résultats par événement de manière globale. Le mode de contrôle du flot de données est le modèle “données poussées”. Des processeurs de traitement de signal (DSPs), ou des processeurs classiques (avec ou sans système d’exploitation) sont utilisés aussi bien pour les traitements locaux que pour le traitement global. Les réseaux de communication sont basés sur diverses technologies. Une des premières réalisations de ce modèle [Cla94] utilisait les liens de communication du DSP 320C40 de Texas Instruments pour les communications entre les ROBs et les processeurs locaux (également réalisés à l’aide de DSP C40) et la technologie SCI pour le réseau de communication entre les processeurs locaux et globaux (processeurs Alpha de Digital). Compte tenu de la grande diversité des composants utilisés (types de processeurs et systèmes d’exploitation), de problèmes au niveau de l’interface entre les différents réseaux et de la complexité de l’ensemble du système, ce premier démonstrateur a connu des difficultés de fonctionnement [Hor97]. Les versions suivantes comportent un certain nombre de simplifications d’architecture. Une réduction de la diversité des composants rend le système plus homogène. La simplification principale est la réunion de plusieurs réseaux de communication: ceux reliant les ROBs aux processeurs de traitements locaux, celui reliant ces derniers aux processeurs de traitements globaux et celui reliant le superviseur à l’ensemble de ces processeurs. En plus de ce réseau principal subsistent cependant deux autres

réseaux: un premier réseau permet au superviseur de communiquer avec l'ensemble des ROBs, et un réseau local classique est utilisé pour la configuration et la surveillance de l'ensemble du système. Deux technologies sont envisagées pour la réalisation du réseau principal: DS-Link et SCI. Le démonstrateur correspondant est décliné en deux versions, rapportées dans [Mal98] et [Bog97]. La version basée sur la technologie DS-Link est présentée Figure 8-2.

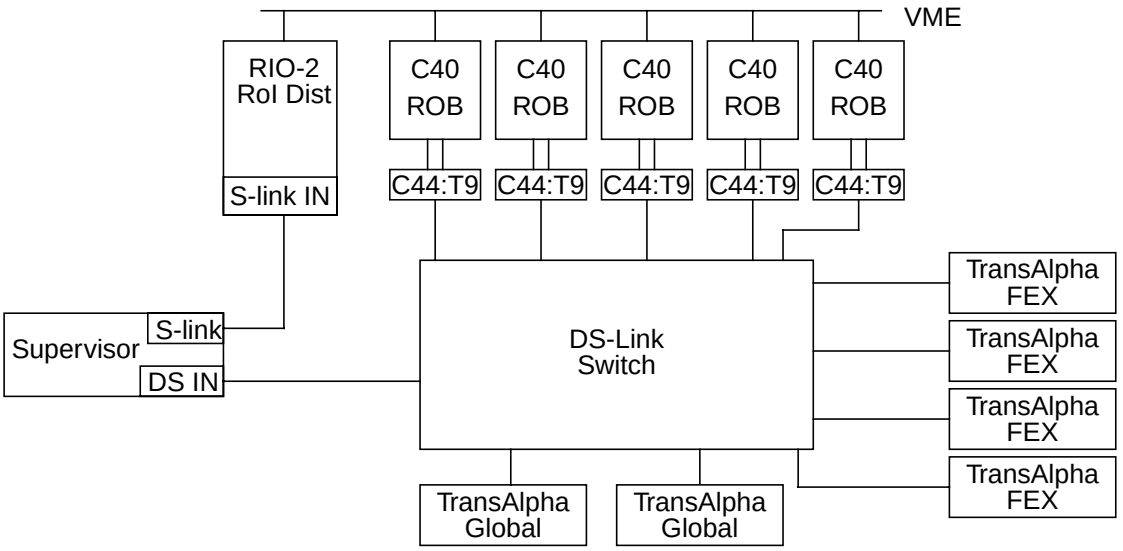


Figure 8-2: Modèle "local-global" du trigger de niveau 2.

Les ROBs sont réalisés avec des DSP TMS320 C40 placés dans un châssis VME. Les processeurs locaux (*Features EXtractors* – FEXs) et globaux utilisent des cartes comportant un Transputer chargé de la communication avec l'extérieur et un processeur Alpha comme unité de calcul principale. Le superviseur est composé de deux ordinateurs sur carte. L'un d'eux communique avec le contrôleur de chaque châssis contenant des ROBs par un lien dédié du type S-Link [Bij97]. Le système final comporterait un réseau de S-Link à sorties multiples (>100) pour diffuser à l'ensemble des contrôleurs de châssis de ROBs l'information fournie par le superviseur. Ce réseau spécialisé est destiné à distribuer les décisions du trigger de niveau 2 à l'ensemble des ROBs (via le bus de fond de panier à l'intérieur de chaque châssis) et à commander l'envoi des données depuis les ROBs vers les processeurs FEX.

Suivant le modèle "données poussées", le superviseur diffuse la liste des régions d'intérêt et l'identificateur du processeur de traitement global alloué pour chaque événement à l'ensemble des contrôleurs de châssis de ROBs. Chaque contrôleur filtre localement ces informations et les transmet aux ROBs concernés par le bus local du châssis. Les ROBs envoient les données des RoIs aux processeurs FEX concernés en utilisant le réseau DS-Link (ou SCI). Ces processeurs sont assignés à une région géographique du détecteur de manière statique ce qui permet aux ROBs d'identifier le destinataire de chaque bloc de données simplement. Les processeurs rassemblent et traitent les données des RoIs puis envoient les résultats à un des processeurs de traitement global en utilisant également le réseau DS-Link (ou SCI). Ce processeur collecte les résultats pour toutes les RoIs à travers tous les détecteurs, effectue des traitements et retourne les résultats au second ordinateur sur carte constituant le superviseur (toujours à travers le réseau DS-Link ou SCI). Le superviseur prend la décision de conserver ou de rejeter l'événement, transmet cette décision via le bus VME au second module du superviseur qui diffuse cette information à tous les ROBs (via S-Link cette fois-ci). Les événements rejetés sont effacés, ceux qui sont acceptés seraient envoyés par un autre réseau au trigger de niveau 3 (non implémenté dans ce démonstrateur).

8.3.2. Critique

Le regroupement des réseaux “locaux” à chaque détecteur et du réseau “global” en un réseau unique est une solution simplificatrice qui s’écarte du modèle du *Technical Proposal*. De même le réseau de diffusion des RoIs et celui de diffusion des décisions de trigger venant du superviseur ont été regroupés. Ce démonstrateur n’illustre donc pas complètement la structure décrite dans le *Technical Proposal* composée de réseaux multiples. Malgré ces simplifications, je pense que le modèle proposé demeure complexe.

Le réseau de distribution liant le superviseur aux ROBs mis en place dans le démonstrateur n’est qu’une liaison point-à-point. La faisabilité d’un réseau de diffusion connectant plusieurs superviseurs à des centaines de ROBs reste à démontrer. L’intérêt d’utiliser un réseau séparé construit de toutes pièces me semble difficile à justifier par rapport à une solution utilisant le réseau de communication principal si ce dernier est bi-directionnel (ce qui est le cas aussi bien pour SCI que DS-Link) et comporte un mécanisme pour les *multi-cast* (envisageable avec SCI selon les produits; mécanisme non prévu dans le commutateur DS-Link).

L’allocation des processeurs locaux de manière géographique est très simpliste. Cette méthode ne conduit pas à une utilisation optimale des ressources compte tenu de la répartition non homogène des RoIs dans les détecteurs. D’autres méthodes d’allocation plus performantes sont possibles mais me semblent difficile à implémenter compte tenu du nombre de processeurs à allouer par événement (>20) à la fréquence des événements (100 kHz).

Les démonstrateurs présentés ne couvrent que le trigger de niveau 2, l’intégration avec le trigger de niveau 3 n’est pas décrite et me semble difficile à réaliser techniquement. D’une manière générale, je pense que les résultats obtenus suggèrent davantage l’adoption de simplifications d’architecture supplémentaires plutôt que la poursuite du travail de démonstration en se basant sur un modèle trop complexe et mal adapté.

8.4. Modèle du trigger de niveau 3 – “DAQ -1”

Le projet “DAQ -1” consiste en la construction d’une tranche verticale complète et opérationnelle du système d’acquisition de données de l’expérience ATLAS [Amb97]. Ce projet destiné à servir de base au système final recouvre tous les aspects indispensables au bon fonctionnement d’une expérience de l’envergure d’ATLAS: interface avec les détecteurs, construction du *DAQ Crate* en incluant un système de surveillance locale (*Local DAQ*), trigger de niveau 3 comprenant le système de reconstruction d’événements (*event builder*) et la ferme de processeurs pour le filtrage des événements (*event filter*), dispositif de stockage des données, base de données de configuration et des paramètres de calibration, contrôle du système, affichage et *monitoring* en ligne (*back-end software*)... Ces domaines sont largement étudiés et rapportés dans les publications des participants au projet “DAQ -1”[1].

8.4.1. Présentation succincte d’une partie du projet

Le principal aspect que recouvre l’étude rapportée dans cette thèse est le démonstrateur du système de reconstruction d’événements. La structure de ce prototype et les résultats obtenus ont été rapportés dans [Amb97a] et [Amb98]. Une vue schématique de cet *event builder* est présentée

1. plus d’une centaine de notes et publications relatives à ce projet sont disponibles sur le site: http://atddoc.cern.ch/Atlas/

Figure 8-3. Chaque source représente un *DAQ Crate*. Les destinations sont les processeurs qui rassemblent les événements complets et les font suivre à l'*event filter* qui est en charge de leur traitement. Le *Data Flow Manager* (DFM) est l'élément contrôlant le flot de données. La reconstruction d'événements est basée sur le protocole suivant. Quand une source reçoit un fragment d'événement, elle utilise un mécanisme du DFM nommé *GetId* pour identifier la destination de ce fragment. La transmission effective utilise une primitive nommée *Transfer*. Lorsque qu'un fragment d'événement a été reçu par une destination, celle-ci informe le DFM par un *End-of-Transfer*. La primitive *End-of-Event* est utilisée par le DFM pour informer une destination que toutes les sources lui ont transmis un fragment d'événement. Si une destination n'est pas en mesure de recevoir des données, un mécanisme *Busy/NotBusy* permet de forcer le DFM à suspendre le flot de fragments d'événements vers cette destination.

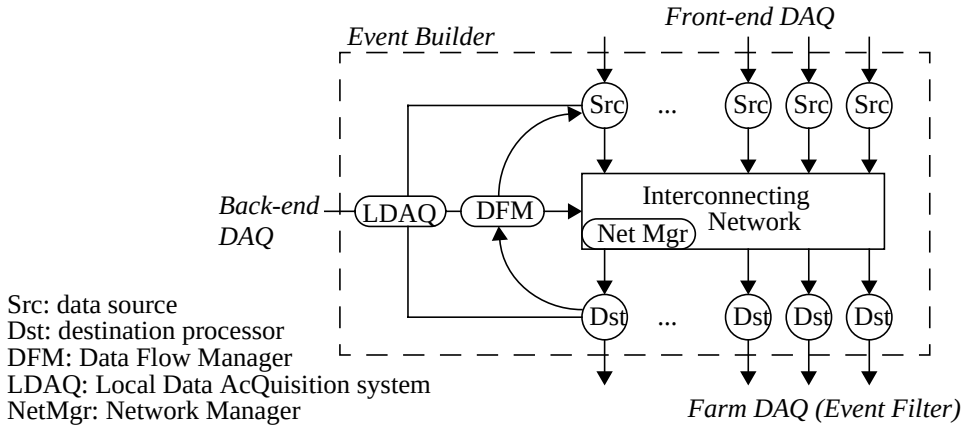


Figure 8-3: Modèle de l'*event builder* du prototype "DAQ -1".

Du point de vue de la réalisation technique, le démonstrateur rapporté dans [Amb98] comporte 2 sources, 2 destinations et 1 DFM (ordinateurs sur carte de type RIO II de CES avec Lynx OS comme système d'exploitation). Les technologies de réseau utilisées sont Fibre Channel, ATM et Fast Ethernet (TCP/IP).

L'intégration de l'ensemble des composants du "DAQ -1" est en cours et les résultats finaux du projet sont attendus pour le début de l'année 2000.

8.4.2. Critique

Ce prototype doit s'intégrer dans un système complet de démonstration fonctionnelle du DAQ d'ATLAS sans l'objectif principal d'atteindre le niveau de performance requis pour le système final ni d'assembler un système d'une certaine échelle. Je pense que les aspects de performance et l'investigation de démonstrateurs de plus grandes dimensions sont néanmoins indispensables pour démontrer la faisabilité de l'*event builder* d'ATLAS. En ce sens, l'étude rapportée dans cette thèse me semble un bon complément au projet très large qu'est le "DAQ -1".

8.5. Résumé

Dans ce chapitre, j'ai présenté les différents démonstrateurs basés sur le modèle du *Technical Proposal* d'ATLAS: modèles en logique re-programmable ou à base de processeurs classiques pour le trigger de niveau 2, et projet "DAQ -1" pour le trigger de niveau 3 et le DAQ.

CHAPITRE IX. DÉMONSTRATEURS DU MODÈLE DE TRIGGER SÉQUENTIEL

9.1. Introduction

Dans ce chapitre, je détaille les systèmes de démonstration associés à l'architecture de la partie du T/DAQ de l'expérience ATLAS défendue dans cette thèse. Les résultats présentés couvrent à la fois le trigger de niveau 2 ainsi que certains aspects du trigger de niveau 3.

9.2. Structure du démonstrateur et motivation

Le système visant à démontrer l'architecture de trigger séquentiel pour ATLAS a été conçu pour offrir une grande évolutivité permettant de réaliser une configuration initiale de taille minimale (investissements modestes) suivie de phases de déploiement à plus grande échelle. Le modèle minimal a servi à valider les principes d'architecture, et au cours du temps, les résultats présentés ont permis d'obtenir le ralliement d'autres collaborateurs acceptant d'investir en commun et avec confiance dans la poursuite du programme pour réaliser un démonstrateur de taille significative. Le système final devant comporter plus d'un millier d'éléments connectés par réseau, un modèle de démonstration n'en comportant qu'une dizaine ne semble pas véritablement crédible. Assembler un démonstrateur représentant de l'ordre de ~1:20 du système final semble un objectif réaliste du point de vue technique et économique.

Un autre aspect dans la conception du démonstrateur est la flexibilité du système. La possibilité d'utiliser différents types de machines et systèmes d'exploitation de manière transparente est nécessaire compte tenu de l'évolution de la technologie. La possibilité de remplacer un élément spécifique du système (par exemple un superviseur ou un ROB) par une machine émulant sa fonctionnalité est également fondamentale. Cela permet de développer le système et le composant spécifique en parallèle. En définissant, lors de la conception, l'interface de programmation avec l'élément spécifique, ce dernier peut être intégré par la suite sans peine au système par remplacement de l'élément qui l'émulait.

Après une phase de construction et de mise au point, un démonstrateur modulaire (de taille modeste et ne comportant pas les éléments spécifiques ROBs) a été rapporté dans [Cal97c]. La structure est reproduite Figure 9-1. Ce démonstrateur comporte 10 éléments principaux connectés par un commutateur ATM ASX 1000 de FORE Systems[1]. Ce commutateur peut être équipé de différents types d'interfaces (25, 155 ou 622 Mbit/s; 64 ports 155 Mbit/s au maximum). Les caractéristiques techniques de ce produit, son évolutivité et le souci de compatibilité avec le matériel acquis auparavant par la collaboration RD-31 justifient ce choix de matériel.

Les sources sont les éléments émulant les groupes de ROBs liés aux détecteurs. Il s'agit d'ordinateurs sur carte de type RIO II utilisant Lynx OS comme système d'exploitation (compatibilité avec le prototype DAQ -1). Les processeurs destination sont les éléments en charge du traitement des données en provenance des modules source. Les machines utilisées sont des PCs

1. http://www.fore.com

Pentium-Pro 200 MHz fonctionnant sous WindowsNT. Les PCs sont la solution la plus économique pour une machine de performance donnée.

Le superviseur est composé d'un RIO II et de matériel spécifique décrit dans [Bla96]. N'importe quelle machine peut jouer le rôle d'une source, d'un processeur destination ou d'un émulateur de superviseur. Cela permet une grande flexibilité dans la configuration. Une machine supplémentaire est utilisée pour le démarrage du système en utilisant un réseau Ethernet classique.

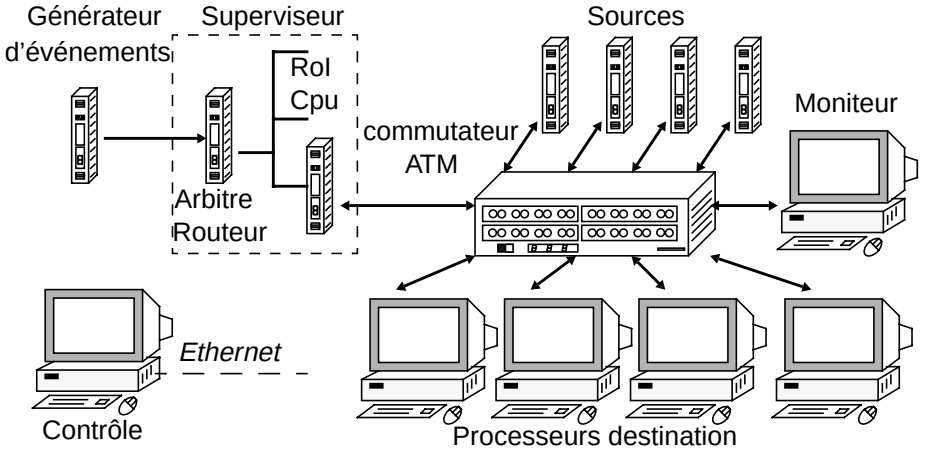


Figure 9-1: Démonstrateur du modèle séquentiel du trigger de niveau 2 et 3.

Suite au succès de ce projet, et avant d'effectuer les investissements pour étendre ce modèle, il m'a été proposé de tester le logiciel développé sur du matériel existant: le système informatique du Research Center for Nuclear Physics (RCNP, université de Osaka) comportant 22 machines interconnectées par ATM (serveurs 4 et 12 processeurs, stations de travail Digital Alpha, et PCs). Cette étude, rapportée dans [Cal98a], représente une avancée d'un facteur ~2 concernant la taille du démonstrateur et démontre la flexibilité, la portabilité et l'évolutivité de l'approche proposée.

Après avoir effectué les investissements nécessaires par étapes, et en conservant la plupart des éléments qui constituaient la configuration initiale, un démonstrateur comportant 32 machines (22 PCs et 10 ordinateurs sur cartes) a été assemblé. Ce modèle intègre une version plus complète du superviseur et le *RoI Builder* ainsi que des prototypes de ROBs. Des résultats sont rapportés dans [Cal99]. Suivant le plan de montée en puissance du démonstrateur et la participation de nouveaux collaborateurs, un système comportant 48 machines a été assemblé récemment[1]. Ce démonstrateur représente en taille de l'ordre de 1:20 du trigger de niveau 2 final; ce qui correspond à l'objectif que nous souhaitions atteindre. Il serait possible de construire un démonstrateur à 64 nœuds sans investir dans un nouveau commutateur ATM, mais cela n'est pas jugé indispensable à ce jour.

9.3. Structure du logiciel

Le logiciel fonctionnant sur les différents nœuds du système a été développé suivant un certain nombre de principes et d'objectifs (décrits en détail dans [Cal98c]):

- maximum d'indépendance vis-à-vis des matériels, machines et systèmes d'exploitation,
- portabilité et évolutivité du logiciel,
- relative indépendance concernant la technologie de réseau,
- structure modulaire et en couches (Figure 9-2),

1. voir les derniers résultats du démonstrateur présentés dans l'Annexe 3 de cette thèse.

- optimisation des performances,
- robustesse du système avec détection des erreurs mais pas nécessairement correction,
- simplification des aspects relatifs à la configuration et la surveillance du système car ce logiciel est destiné à une démonstration et non au système opérationnel final.

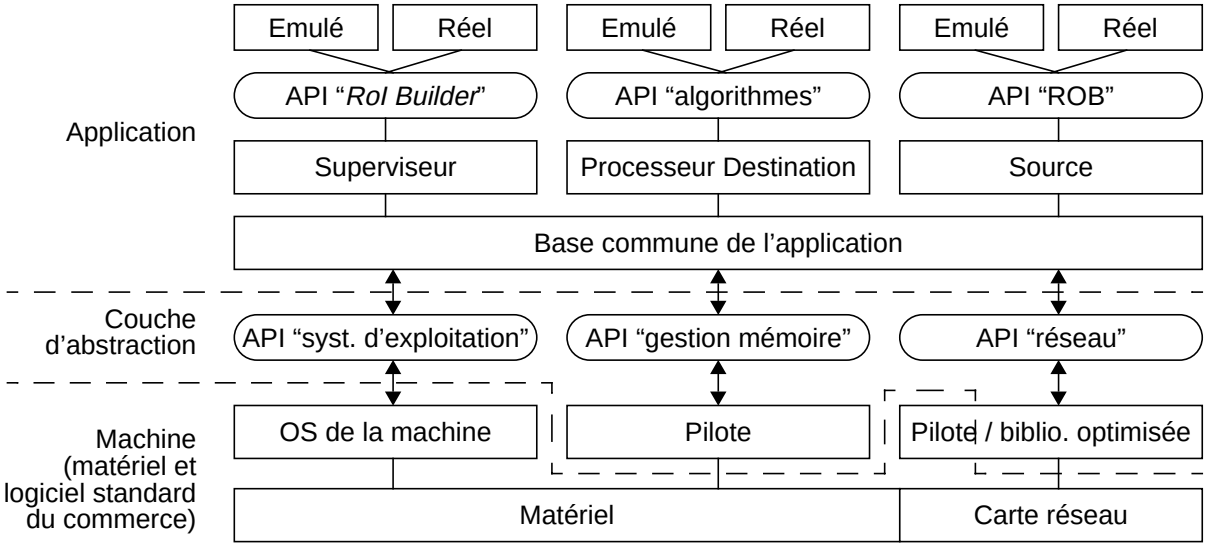


Figure 9-2: .Structure du logiciel.

9.3.1. Couche d'abstraction "système d'exploitation"

Une première couche d'abstraction vise à unifier l'interface aux différents systèmes d'exploitation utilisés. Une interface de programmation regroupant un nombre réduit de primitives a été définie [Cal98d]. Elle fournit des appels pour:

- la gestion du temps (mesure d'intervalles de temps, attente durant une certaine période, utilisation du processeur pendant un temps donné),
- la création de *threads* et la gestion dynamique des priorités,
- un mécanisme de signalisation entre *threads* (sémaphore),
- un mécanisme d'exclusion mutuelle ("mutex") pour la protection des ressources partagées,
- l'utilisation de tubes nommés ("*pipe*") pour la communication entre processus.

D'autres primitives peuvent être définies selon les besoins. L'implémentation de cette couche a été effectuée pour WindowsNT, Lynx OS, Linux et Digital UNIX ce qui ouvre un éventail assez large concernant le choix des machines utilisables pour notre application.

9.3.2. Couche d'abstraction "réseau"

Une autre couche d'abstraction concerne l'accès au réseau de communication. L'interface de programmation correspondante est définie dans [Cal98]. Elle fournit un moyen pour établir et fermer des connexions, envoyer et recevoir des messages. Cette couche a été implémentée en utilisant la bibliothèque de fonctions de communication optimisées décrite au chapitre VII. Une implémentation basée sur UDP/IP (interface *socket*) et deux versions pour ATM AAL 5 (API Winsock2 et API de Digital) ont également été développées.

9.3.3. Couche d'abstraction "gestion de la mémoire"

Un élément de logiciel en charge de la gestion centralisée des ressources mémoire a été développé: le *Buffer Manager* [Cal98e]. Cet élément permet l'allocation, la fourniture et le recyclage

de *buffers* dans une application. Afin de permettre les transferts d'information sans recopie de données entre l'application, la carte réseau et (pour les modules sources) les ROBs, un certain nombre de contraintes doivent être satisfaites:

- les *buffers* doivent être résidants et verrouillés en mémoire, *cacheable* si possible. Chaque *buffer* doit occuper une plage d'adresses contiguës (aussi bien virtuelles que physiques). L'adresse physique de chaque *buffer* doit être connue ainsi que l'adresse virtuelle (dans l'espace de mémoire de l'application). Pour des raisons de simplicité, une seule adresse virtuelle est associée à chaque *buffer*. Seules les applications mono-tâche peuvent donc utiliser le *Buffer Manager* (applications mono ou *multi-threads* cependant).
- l'application doit pouvoir recevoir les messages venant du réseau dans la structure qui est délivrée par la carte réseau. Dans le cas d'une carte ATM basée sur NicStar (et de nombreuses autres cartes réseaux), chaque paquet AAL 5 reçu peut être contenu dans un ou plusieurs *buffers* selon sa taille (par exemple les 240 premiers octets dans un *buffer* et la suite dans une série de *buffers* de 4 ko). La manipulation de listes chaînées de *buffers* par l'application est nécessaire.
- les messages à transmettre peuvent, dans certains cas, être composés de plusieurs blocs qui ne résident pas à des adresses physiques consécutives. La manipulation de listes chaînées de *buffers* par le matériel de la carte réseau pour l'émission de ce type de message (sans recopie de données) est nécessaire.
- les messages transmis/reçus doivent être formatés/interprétés suivant la convention d'ordre des octets sur le réseau (i.e. *Big Endian* ou *Little Endian*).

Pour des raisons de performance, la réserve de *buffers* du *Buffer Manager* est allouée au démarrage de l'application; l'adresse physique et virtuelle de chaque *buffer* est déterminée. Les *buffers* alloués sont de taille fixe (par exemple 4 ko correspondant à la taille d'une page de mémoire sur les machines utilisées). Pour traiter des messages dont la taille excède celle d'une page, ces *buffers* peuvent être chaînés. Il est également possible d'utiliser des *buffers* qui ont été créés à l'extérieur du *Buffer Manager* en fournissant une interface compatible.

Les entités manipulées par le *Buffer Manager* et l'application ne sont pas les *buffers* eux-mêmes, mais des descripteurs contenant diverses informations. La documentation complète sur les descripteurs de *buffers* et leur manipulation est disponible dans [Cal98f]. Le principe de listes chaînées de descripteurs de *buffers* est similaire à la méthode employée dans le système d'exploitation BSD UNIX [Lef89] (structure appelée *"mbuf"*). Néanmoins, le modèle utilisé dans notre cas comporte un certain nombre de fonctionnalités supplémentaires par rapport aux *mbuf*. Ces descripteurs et *buffers* sont présentés Figure 9-3.

Le descripteur associé à chaque *buffer* comporte (entre autre) les champs suivants:

- l'adresse virtuelle de début de la zone contenant les données,
- l'adresse physique correspondante,
- la capacité du *buffer* (en octets),
- un numéro d'identification du descripteur et *buffer* associé,
- un pointeur sur l'objet créateur du *buffer* (i.e. le *buffer manager*),
- un pointeur sur la fonction à appeler pour recycler le *buffer*,

Ces champs ne doivent pas être modifiés par l'application. Les principaux champs modifiables sont les suivants:

- la taille effective du bloc de données dans le *buffer* (toujours inférieure à sa capacité),
- un pointeur sur le descripteur suivant dans la chaîne,
- un pointeur sur le dernier descripteur de la chaîne,
- un pointeur sur le descripteur courant,

- l'offset du bloc de données contenu dans le *buffer* par rapport au début de la chaîne,
- la taille totale des données dans la chaîne,
- la position du pointeur courant pour la prochaine opération de lecture/écriture...

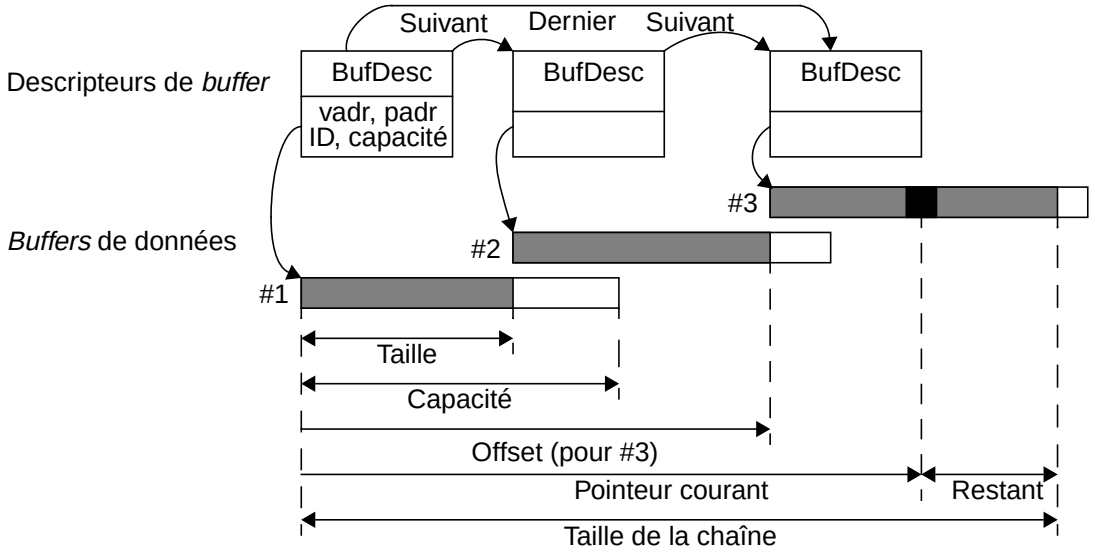


Figure 9-3: Structure d'une chaîne de descripteurs de *buffer.*

Compte tenu de la structure relativement complexe d'un *buffer* formé d'une liste chainée, l'accès aux données se fait au moyen de méthodes particulières cachant pour l'utilisateur la structure interne des *buffers*. La manipulation de données peut être faite avec une conversion optionnelle de l'ordre des octets pour se conformer à l'ordre de transmission sur le réseau qui peut être différent de celui utilisé de manière interne par la machine. Des méthodes sont disponibles pour écrire/lire dans un *buffer* les types de données simples: caractère (*char*, 8 bit), entier court (*short*, 16 bit), entier (*int*, 32 bit), nombre flottant double précision (*double*, 64 bit).

L'écriture et la lecture de structures plus complexes se fait à partir des opérations disponibles pour les types simples. Afin de garantir la portabilité des messages entre différentes machines, une représentation externe doit être définie pour chaque structure échangée à travers le réseau. Une possibilité est d'utiliser l'*Abstract Syntax Notation 1* (ASN.1) [Dub99]; l'*eXternal Data Representation* (XDR) [SUN87] est également utilisable. Les méthodes de passage de la représentation interne vers/depuis la représentation externe doivent être implémentées (couche "présentation" du modèle OSI). Pour l'application envisagée, on peut considérer qu'un récepteur connaît tous les types de données qu'il peut recevoir. Il n'est donc pas utile de coder dans le message de manière explicite le type de chaque donnée (méthode utilisée avec ASN.1). Dans notre cas un codage de type implicite proche de XDR est utilisé avec un certain nombre de variations:

- l'ordre de transmission peut être *Big Endian* ou *Little Endian* (et non uniquement *Big Endian*),
- les quantités 8 (16) bit peuvent être alignées sur des frontières 8 (16) bit (et non uniquement sur des frontières 32 bit),
- les types booléens et les types énumérés ne sont pas supportés (contrairement à XDR).

9.3.4. Application

L'application est également structurée en couches et en modules. La partie commune à tous les types de nœuds (source, superviseur, processeur, moniteur, co-processeur) est le squelette de l'application. Cette partie couvre par exemple l'analyse du fichier de paramètres pour déterminer la

configuration du système, l'établissement des connexions [Cal98g], etc. Une partie spécifique implémente la fonctionnalité particulière du nœud. Un niveau de spécialisation supplémentaire pour certains nœuds est également possible (source du calorimètre, du TRT...).

Une autre couche d'abstraction permet le fonctionnement de certains éléments avec ou sans le matériel spécifique qu'ils intégreraient dans un système plus complet. Ainsi, le superviseur peut fonctionner sans l'interface au *RoI Builder*. L'arrivée des événements en provenance du *RoI Builder* est simulée par logiciel à l'intérieur du superviseur. La même API est utilisée pour lire les événements du *RoI Builder* ou à l'aide du code d'émulation ce qui rend ces deux modules interchangeables. Pour les sources, le fonctionnement est possible avec ou sans cartes ROB. Si aucun ROB n'est connecté à une source, ces derniers sont simulés par logiciel. Une API adaptée permet de passer du mode où les ROBs sont simulés à un fonctionnement avec de véritables ROBs sans avoir à modifier le code du nœud source.

Les différents modules et l'application sont codés en langage C. Ce choix est en partie justifié par le fait que plusieurs *device drivers* ont dû être développés (le C est alors le seul outil), par le souci de performance (JAVA est en ce sens un choix sujet à controverse) et afin de garder la possibilité d'utiliser diverses machines et compilateurs (les spécifications du C++ n'étaient pas encore stables au moment de notre choix). Aujourd'hui, un choix plus judicieux serait d'utiliser C++. Le code développé suit en partie les principes de programmation orientée objet (en associant des fonctions aux structures). Les fonctions virtuelles (permettant le polymorphisme) sont implémentées en langage C par des pointeurs sur des fonctions (i.e. les méthodes de l'objet) placés dans la structure représentant cet objet. Le mécanisme d'héritage n'est pas réalisable en langage C de manière simple. Les aspects de modularité et de structure en couches sont des qualités du code indépendantes du langage de programmation.

9.4. Superviseur et *RoI Builder*

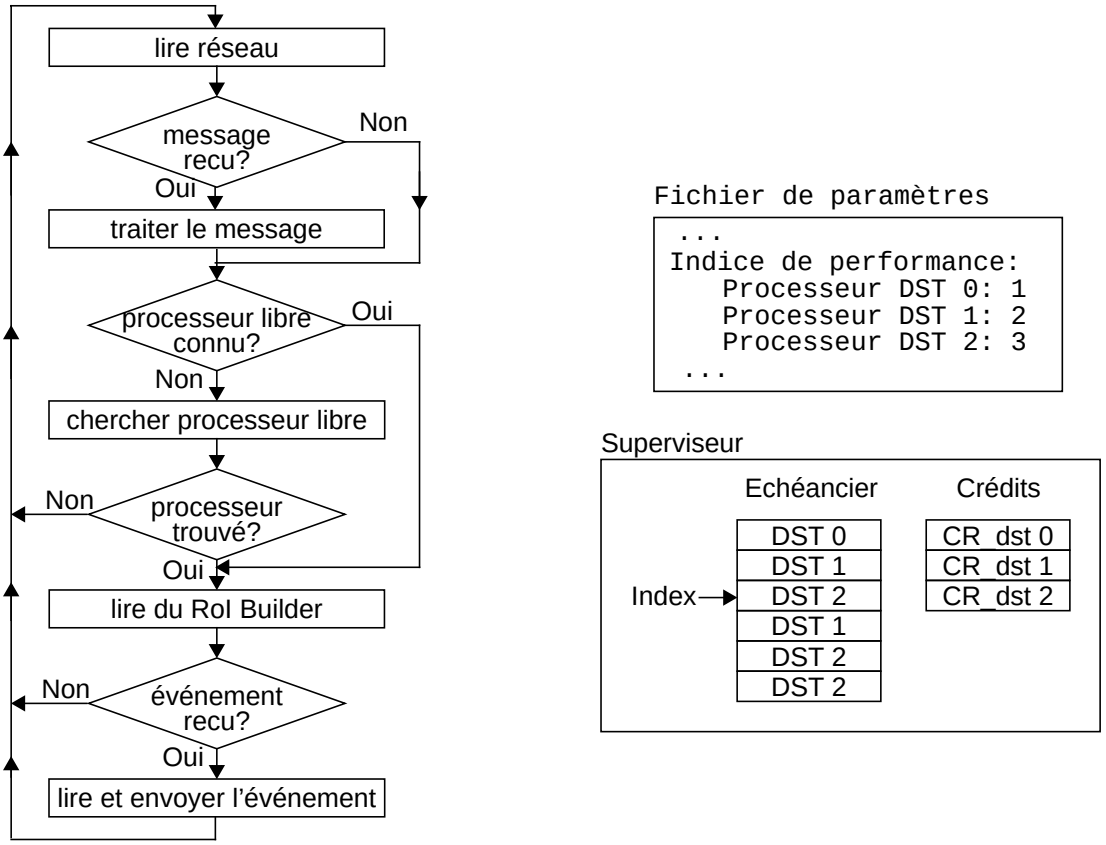
9.4.1. Matériel

Le *RoI Builder* est l'interface entre le trigger de niveau 1 et celui de niveau 2. Pour chaque événement accepté par le trigger de niveau 1, cet élément reçoit les informations relatives aux RoIs de type muon, électron/gamma, jet, énergie totale et énergie transverse manquante. Ces informations arrivent par 7 liaisons point-à-point et doivent être combinées sur la base de chaque événement pour former les blocs transmis au superviseur. Il s'agit d'un petit système de reconstruction d'événements comportant 7 sources et plusieurs destinations (les superviseurs). L'étude et la réalisation matérielle du *RoI Builder* a été faite par nos collaborateurs de Argonne National Laboratory et Michigan State University (USA). Les cartes d'entrée du *RoI Builder* sont basées sur des FPGAs. Chacune comporte trois liaisons avec le trigger de niveau 1. La distribution des fragments d'événements se fait de manière cyclique à 1,2,4 ou 8 processeurs superviseurs reliés par une liaison point-à-point au *RoI Builder*. Chaque superviseur reçoit un flux ordonné de 7 fragments par événement qu'il rassemble en un seul puis l'envoie (via le réseau ATM dans notre cas) au processeur de traitement affecté. Les superviseurs sont des ordinateurs sur carte ou des PCs standards.

9.4.2. Logiciel et principe de fonctionnement

Le superviseur est en charge de la distribution des événements fournis par le *RoI Builder* aux processeurs de traitement, de la réception des décisions de trigger et de leur diffusion aux sources. Dans le modèle d'architecture du T/DAQ comportant des groupes de processeurs séparés, le

superviseur effectue également la distribution des événements acceptés par le trigger de niveau 2 aux processeurs du trigger de niveau 3. Pour des raisons de performance, le programme du superviseur est une tâche ne comportant qu'un seul *thread*. Il effectue l'interrogation cyclique de deux périphériques extérieurs: l'interface au *RoI Builder* et la carte réseau. L'organigramme simplifié du programme du superviseur est présenté Figure 9-4a.



a) tâche principale du superviseur

b) mécanisme de distribution d'événements

Figure 9-4: Organigramme du superviseur.

Le superviseur doit effectuer l'allocation des événements aux destinations en tenant compte des disparités potentielles entre ces dernières (fréquence d'horloge, nombre de processeurs) et en considérant également la charge instantanée de chaque destination. Lors de l'initialisation, les processeurs sont répartis en autant de groupes que le système comporte de superviseurs (partage statique des ressources). En cas de défaut d'un superviseur, il est prévu de répartir ses ressources entre les autres superviseurs.

La détermination du processeur destination pour le prochain événement utilise un mécanisme d'échéancier et de crédits. Chaque superviseur construit une table donnant la séquence d'allocation des destinations (échéancier). A chaque processeur est associé un indice relatif de performance, par exemple un entier entre 1 et 16. Le nombre d'entrées dans l'échéancier pour chaque destination est proportionnel à la puissance relative du processeur considéré. Un exemple d'échéancier est présenté Figure 9-4b. Un certain nombre de crédits sont alloués à chaque destination. Le nombre de crédits d'une destination est décrémenté lorsque un événement lui est assigné et est incrémenté lorsque elle retourne ses résultats au superviseur.

Le mécanisme de crédits permet de limiter le nombre d'événements alloués à chaque destination (contrôle de flux dynamique), et d'isoler les destinations ne fonctionnant plus (si les

crédits d'une destination sont épuisés, elle ne reçoit plus d'événements). Le mécanisme d'échéancier permet la distribution non-homogène des événements en sollicitant plus fréquemment les processeurs de puissance relative plus importante (équilibrage de la charge des destinations). Pour des raisons de robustesse, le nombre initial de crédits de chaque destination doit être rétabli périodiquement afin que les pertes occasionnelles de messages n'entraînent pas la mise à l'écart de certains processeurs. Le choix de la destination pour le prochain événement se fait suivant le pseudo-code suivant:

```
procédure Trouver_prochaine_destination:
  essai:=0;
  tant que (essai <= taille de l'échéancier)
    destination:=echéancier[index];
    incrémenter index modulo(taille de l'échéancier);
    si (crédit de destination > 0 )
      décrémenter crédit de destination;
    retourner destination;
    fin si;
    incrémenter essai;
  fin tant que;
  retourner pas_de_processeur_libre;
fin procédure;
```

Les décisions de trigger reçues par chaque superviseur sont accumulées puis diffusées à l'ensemble des sources. Le facteur de groupement est ~10-100 afin de réduire d'autant la fréquence d'envoi des messages vers les sources. Pour un système fonctionnant à 100 kHz avec 4 superviseurs, grouper les décisions de trigger par paquets de 25 réduit à 4 kHz la fréquence de réception de ce type de message au niveau des sources mais augmente de 1 ms le temps de latence du trigger.

Dans le modèle "fermes séparées" (voir section 4.4.3), le superviseur alloue une destination de la ferme "filtre d'événements" pour chaque événement accepté par le trigger de niveau 2. Le mécanisme de distribution d'événements dans chaque superviseur est dupliqué pour distinguer les processeurs appartenant au trigger de niveau 2 et ceux du filtre d'événements.

Chaque superviseur conserve la trace des événements alloués et détecte les débordements de temps (*time-out*) si une destination n'a pas retourné de résultat dans le temps imparti. Le superviseur accumule localement un certain nombre de statistiques et communique avec le moniteur.

9.4.3. Mesures de performance

La mesure la plus significative sur le *RoI Builder*/superviseur est la fréquence maximale de fonctionnement, F_{sup}. L'objectif à atteindre est 100 kHz.

Chaque superviseur peut fonctionner sans *RoI Builder*, en générant de manière interne des événements. En calculant la différence entre les mesures effectuées avec le *RoI Builder* et sans ce dernier, on peut déterminer le temps mis par le superviseur pour lire un événement venant du *RoI Builder*. Les mesures de F_{sup} pour diverses configurations sont présentées Table IX-1.

On remarquera la nette amélioration de performance obtenue sur un PC 300 MHz en utilisant les fonctions de communications optimisées par rapport à un pilote standard (gain d'un facteur ~10). La performance obtenue avec un PC est ~4 fois supérieure à celle obtenue avec une station de travail Digital de fréquence CPU comparable (toujours grâce à l'optimisation des communications). Les ordinateurs sur cartes VME ont des performances comparables à celles des PCs, et sont légèrement supérieures à fréquence d'horloge égale. Avec une carte PowerPC 300 MHz, le temps moyen de

traitement d'un événement par superviseur est de ~36 µs, temps incluant ~11 µs pour la lecture de l'événement depuis le *RoI Builder*.

F_{sup} (kHz)	sans *RoI Builder*		avec *RoI Builder*		
Type de superviseur / Nombre	1	2	1	2	4
PowerPC 100 MHz[a]	16	–			
PowerPC 200 MHz[a]	35	70	24	48	96
PowerPC 300 MHz[a]	40	80	28	56	
Pentium-Pro 200 MHz[b]	26	52	–		
Pentium II 300 MHz[c]	4	8			
Pentium II 300 MHz[b]	38	76			
Alpha 333 MHz[d]	9	–			
Pentium II 400 MHz[b]	48	96			

Table IX-1: Fréquence de fonctionnement du *RoI Builder*/superviseur.

a. Ordinateur sur carte VME (origine: CES), Lynx OS, bibliothèque ATM zéro-copie
b. PC (origine: DELL), WindowsNT, bibliothèque ATM zéro-copie
c. PC (origine: DELL), WindowsNT, interface ATM FORE + API Winsock 2
d. Station de travail Digital Alpha, Digital UNIX, interface ATM et API Digital

Lors de l'intégration du *RoI Builder*, seules les cartes fonctionnant sous Lynx OS ont pu être testées, le pilote de l'interface S-Link pour WindowsNT n'étant pas encore disponible. La fréquence d'opération requise pour le système final (100 kHz) est pratiquement atteinte en utilisant 4 superviseurs. Bien que la totalité des fonctionnalités du superviseur n'aient pas été intégrées dans le logiciel de ce démonstrateur, on peut considérer que 2 à 4 superviseurs seront suffisants pour ATLAS compte tenu de l'évolution de la performance des machines.

9.5. Source et *Read-Out-Buffers*

9.5.1. Matériel

Chaque source est constituée d'un PC ou d'un ordinateur sur carte (formant un *Rob to Switch Interface* – RSI) relié au réseau de communication principal (ATM) et d'un certain nombre de ROBs connectés par un bus partagé (PCI). Plusieurs options sont envisagées pour les ROBs. La réalisation de cartes spécifiques est en cours, telle celle décrite dans [Gac96]. La version actuelle de cette carte comporte un pont PCI, un processeur Intel I 960 (33 MHz) et sa mémoire de programme. Une version plus complète comportant un lien d'entrée rapide (100 Mo/s), 8 Mo de mémoire d'événements et de la logique de contrôle est en cours de développement et sera intégrée dans le démonstrateur ultérieurement. Une autre option pour le ROB est d'utiliser une carte disponible dans le commerce, telle la carte ASP C10 de Transtech[1]. Cette carte au format PMC comporte un pont PCI, un DSP SHARC (40 MHz) disposant de 6 liens d'entrée (40 Mo/s par lien), 12 Mo de DRAM.

1. http://www.transtech-dsp.co.uk/

9.5.2. Logiciel et principe de fonctionnement

Le rôle du RSI est de servir les requêtes de données provenant des processeurs de traitement et de diffuser aux ROBs les décisions de trigger en provenance du superviseur. Lorsqu'une requête arrive, le RSI la transmet aux ROBs concernés. Les ROBs localisent les données demandées et informent le RSI lorsque celles-ci sont disponibles. Deux modes de fonctionnement sont possibles pour la transmission des données à travers le réseau vers le processeur qui avait émis la requête.

- Mode avec recopie de données dans la mémoire du RSI

Lors de la réception d'une requête:

i) le RSI demande au *Buffer Manager* un *buffer* pour contenir l'en-tête de la réponse ainsi qu'un *buffer* (de capacité suffisante) par ROB concerné,

ii) le RSI poste la requête à chaque ROB en fournissant l'adresse physique du *buffer* alloué au ROB correspondant,

iii) chaque ROB localise puis transmet les données demandées (par DMA de préférence) dans le *buffer* indiqué,

iv) chaque ROB poste un message à destination du RSI pour indiquer que les données ont été recopiées dans sa mémoire,

v) lorsque le RSI a reçu les données de tous les ROBs concernés, ces données peuvent éventuellement être traitées. La taille du bloc total est calculée, placée dans l'en-tête, une chaîne contenant les différents *buffers* est constituée puis postée au contrôleur de la carte réseau,

vi) la carte réseau transmet le message (DMA depuis la mémoire du RSI), puis appelle la fonction de recyclage du premier *buffer* de la chaîne,

v) la fonction de recyclage automatique de chaque *buffer* est appelée par son prédécesseur, tous les *buffers* sont retournés au *Buffer Manager*.

- Mode de transfert direct depuis la mémoire des ROBs

Dans ce mode de fonctionnement, il est nécessaire que la mémoire d'événements des ROBs soit visible sur le bus PCI par le processeur du RSI et par le matériel de la carte réseau. Lors de la réception d'une requête:

i) le RSI demande au *Buffer Manager* un *buffer* pour contenir l'en-tête de la réponse, ainsi qu'un descripteur de *buffer* vide (i.e. sans *buffer* associé) par ROB concerné,

ii) le RSI poste la requête aux ROBs,

iii) les ROBs localisent les données demandées et retournent au RSI l'adresse physique (sur le bus PCI) et la taille du bloc considéré,

iv) lorsque tous les ROBs concernés ont informé le RSI de la disponibilité de leur bloc de données, le RSI calcule la taille totale du bloc, la place dans l'en-tête du message, forme la chaîne de *buffers* complète et poste la demande de transmission au contrôleur de la carte réseau,

v) la carte réseau transmet en un message unique l'en-tête (DMA depuis la mémoire du RSI) et chaque bloc de données des ROBs (DMA depuis la mémoire de chaque ROB), puis appelle la fonction de recyclage du premier *buffer* de la chaîne,

vi) ce *buffer* est retourné au *Buffer Manager*, chacun des *buffers* suivants est retourné à son ROB respectif.

La méthode avec recopie de données est la plus simple et permet de faire des traitements sur les données des ROBs dans le RSI avant transmission. La méthode de transfert direct est plus subtile et permet de réduire la charge du processeur du RSI (ainsi que celle du bus partagé entre les différents éléments) en réduisant les mouvements de données. En revanche, le traitement des données des ROBs par le RSI n'est pas possible avant envoi. Les études sur le ROB et le RSI sont en cours pour décider du meilleur compromis.

On distingue deux types de requêtes de données en provenance des processeurs: celles du type “sélection d’événement” et celles du type “reconstruction de l’événement”. Le premier type correspond le plus souvent à une requête de données de RoI (faible nombre de source concernées) et la réponse est transmise en utilisant la bande passante maximale du lien au réseau (classe UBR en ATM). Pour le second type de requêtes, toutes les sources sont concernées. Afin d’éviter l’engorgement du réseau, les données sont retournées en utilisant une fraction de la bande passante du lien (classe CBR). La fraction allouée par couple source/destination est:

$$B_{SD}' = \beta \times \frac{B_{SD}}{Max(S, D)} \qquad \textbf{(9-1)}$$

Le facteur β permet de faire varier la fraction de bande passante réservée pour le trafic CBR. Lors de nos tests, la valeur 0,99 est utilisée, ce qui signifie que le trafic de l’*event builder* peut utiliser la quasi-totalité de la bande passante disponible. Dans un système réel, une valeur plus faible peut être adoptée afin de garantir un minimum de bande passante pour servir les requêtes du type “sélection d’événements”.

L’application fonctionnant sur le RSI comporte un *thread* principal effectuant le *polling* sur la carte réseau et sur une file d’attente accessible en écriture par les ROBs. Une tâche secondaire vérifie périodiquement si certaines requêtes émises vers les ROBs sont restées sans réponse (détection d’erreurs).

L’application fonctionnant sur le ROB est une tâche unique (processeur fonctionnant sans système d’exploitation). La version actuelle est très simple et ne comporte que la partie de communication avec le RSI. Lorsqu’une requête est reçue par le ROB, celui-ci retourne un bloc de données pris parmi quelques dizaines d’événements possibles. Le fonctionnement avec le lien d’entrée du ROB et la recherche d’un événement parmi quelques centaines à l’aide d’une table de hachage n’ont pas encore été testés dans le démonstrateur.

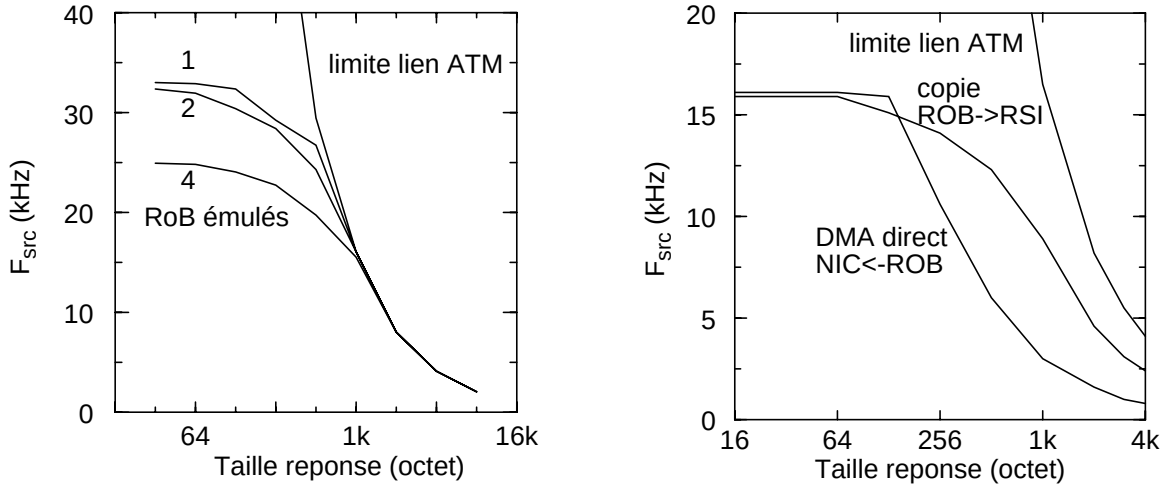
9.5.3. Mesures de performance

De manière comparable au superviseur, l’un des paramètres de performance important pour le RSI et les ROBs est la fréquence maximale, F_{src}, à laquelle les requêtes en provenance des processeurs peuvent être servies (les requêtes du type “sélection d’événement” sont les plus fréquentes). L’objectif à atteindre est une fréquence de service des requêtes de 10-20 kHz pour des fragments de 1-4 ko par ROB.

La Figure 9-5a montre F_{src} en fonction de la taille du bloc retourné en utilisant un ordinateur sur carte (PowerPC 100 MHz) comme RSI et une simulation interne des ROBs. Pour les réponses de faible taille, la fréquence maximale de fonctionnement est ~38 kHz. La limitation due à la bande passante du lien ATM est atteinte pour des paquets de taille supérieure à 1 ko.

La Figure 9-5b montre F_{src} en utilisant un PC Pentium II (400 MHz) comme RSI et une carte SHARC ASP C10 comme ROB. Les performances sont relativement décevantes: ~16 kHz pour les messages courts. Une limitation correspondant à ~77 Mbit/s de la bande passante du lien ATM est observée. Contrairement à toute attente, la performance dans le mode avec recopie de données (noté “copie ROB->RSI” sur le graphe) est supérieure à celle utilisant les transferts directs (“DMA direct NIC<-ROB”). L’observation des signaux sur le bus PCI permet d’apporter une explication. Pour des raisons d’architecture, la DRAM externe au SHARC sur la carte ASP C10 n’est pas accessible au travers du bus PCI. De ce fait, les données à transférer du ROB sont stockées dans la SRAM interne du SHARC. Les transferts par DMA qu’effectue le pont PCI depuis la mémoire du SHARC vers la

mémoire du PC hôte sont limités à ~10 Mo/s. La lecture par un agent PCI (par exemple NicStar) dans la mémoire du SHARC au travers du pont PCI de la carte ASP C10 ne peut pas se faire en rafale (accès mot par mot uniquement) ce qui limite à ~4 Mo/s la bande passante correspondante.



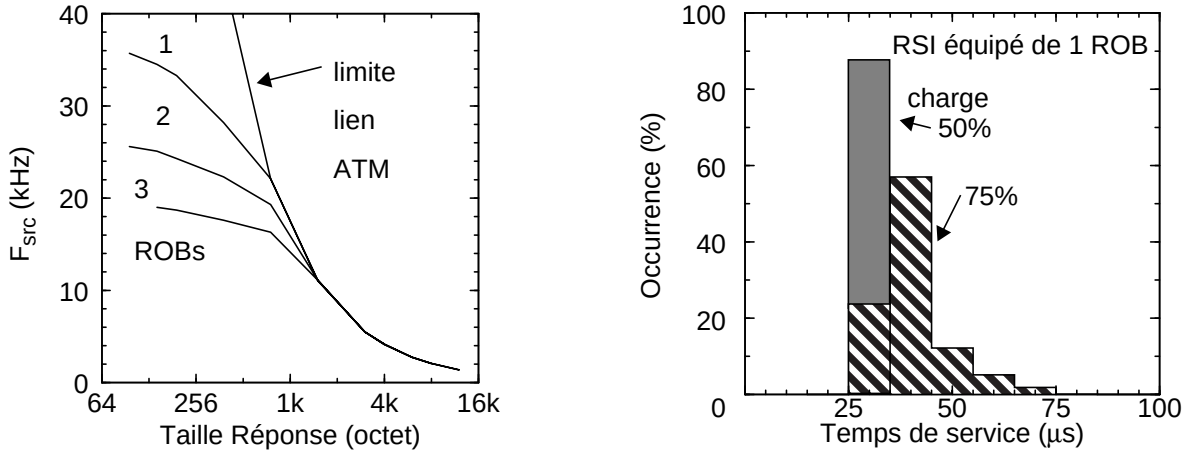
a) RSI et simulation des ROBs

a) RSI équipé d'un ROB (SHARC)

Figure 9-5: Fréquence de service du RSI avec et sans ROB.

Compte tenu des performances médiocres, l'utilisation de la carte ASP C10 en tant que ROB a été abandonné. Je pense qu'il serait néanmoins intéressant de tester une carte plus moderne (des cartes équipées de DSP 200 MHz sont disponibles) et mieux conçue: mémoire externe au DSP accessible par le bus PCI en mode rafales, etc. Cette étude n'est pas abordée dans cette thèse.

La figure 9-6a présente les résultats de performance extraits de [Cal99] pour la version du ROB développée et intégrée par notre groupe.



a) F_{src} vs. taille de la réponse

b) temps de service requête/réponse

Figure 9-6: Fréquence de service du RSI/ROBs.

Le RSI est une carte PowerPC 300 MHz à laquelle sont connectés 1, 2 ou 3 ROBs. L'envoi des messages se fait sans recopie de données dans la mémoire du RSI, par DMA direct du contrôleur de la carte ATM dans la mémoire des différents ROBs. Les performances sont bien supérieures à celles obtenues avec la carte SHARC ASP C10 (facteur 2 environ). La limitation due à la saturation de la liaison ATM est obtenue pour les paquets de taille supérieure à ~1,5 ko. Pour les réponses de faible taille, la formule suivante est établie:

$$F_{src} = 1/(20\mu s + NRobs) \quad \textbf{(9-2)}$$

La Figure 9-6b donne l'histogramme du temps de service d'une requête mesuré entre l'instant de réception de celle-ci et l'instant où l'ordre de transmettre le message est posté à la carte réseau. La fréquence des requêtes est de 18 et 27 kHz, ce qui correspond respectivement à 50% et 75% de la capacité de l'élément testé. Le temps de service moyen est de l'ordre de 40 µs. Les résultats obtenus sont satisfaisants, mais une étude plus détaillée du ROB est nécessaire pour mieux caractériser l'ensemble RSI et ROBs.

Pour la conception du système final, les paramètres à évaluer sont la fréquence maximale des requêtes pour chaque ROB (ou groupe de ROBs) ainsi que le volume de données à transmettre. Cette évaluation est complexe et fait intervenir de nombreux paramètres: caractéristiques des détecteurs, organisation du système de lecture, fréquence des événements, position nombre et taille des régions d'intérêt, séquence type d'algorithmes, taux de rejet aux différentes étapes de la sélection, etc. Les études sont en cours dans le groupe en charge de la modélisation du système[1]. Une estimation grossière de la fréquence maximale de participation des ROBs est 10-20 kHz. Ce chiffre est compatible avec les performances du système actuel. Pour une fréquence de requêtes donnée, la bande passante du lien utilisé fixe la taille maximale du bloc qui peut être envoyé. Actuellement, l'utilisation d'un lien 155 Mbit/s pour chacun des ROBs les plus sollicités semble satisfaire la majorité des besoins. Dans le cas contraire, des liens plus rapides seront nécessaires (par exemple des liens 622 Mbit/s). Les ROBs moins sollicités peuvent être regroupés.

9.6. Processeurs

9.6.1. Matériel

Les processeurs de traitement sont des PCs, des stations de travail mono ou multi-processeurs. En principe, il est également possible d'utiliser des ordinateurs sur carte, mais le rapport coût/performance de ce type de machine est très supérieur à celui d'un PC (facteur ~2 à 4).

9.6.2. Logiciel et principe de fonctionnement

Le rôle des processeurs est d'effectuer les traitements sur les événements alloués par le superviseur pour décider de ne conserver que ceux satisfaisant les critères de sélection du trigger. L'application fonctionnant dans les processeurs assure l'interaction avec le superviseur, la collecte des données des ROBs et les traitements de ces données. La partie de traitement est liée à la physique alors que le reste est l'infrastructure permettant de fournir les données nécessaires. Au niveau du démonstrateur, l'accent a été mis sur le développement de l'infrastructure permettant d'acheminer les données, et non sur la partie relative au traitement des données et à la physique (cette partie étant développée par les groupes de physiciens concernés puis intégrée ultérieurement dans un système opérationnel).

Le principe de l'application fonctionnant sur un processeur est présenté Figure 9-7 sous forme d'un diagramme de classes. L'objet Réseau permet l'émission et la réception des messages par le réseau ATM. Le *Buffer Manager* est en charge de la gestion de la mémoire. Le Thread Réception est destiné à aiguiller les messages reçus. Le traitement des événements (algorithme de

1. information disponible sur le site http://www.nikhef.nl/pub/experiments/atlas/daq/modelling.html

sélection) est effectué par les Threads Traitement. Chaque processeur est capable de traiter plusieurs événements simultanément (un événement par Thread Traitement). Dans la phase initiale, tous les Thread Traitement sont en attente sur une queue destinée à recevoir les nouveaux événements (Queue nouveaux événements) et le Thread Réception est en attente sur le Réseau (mode bloquant).

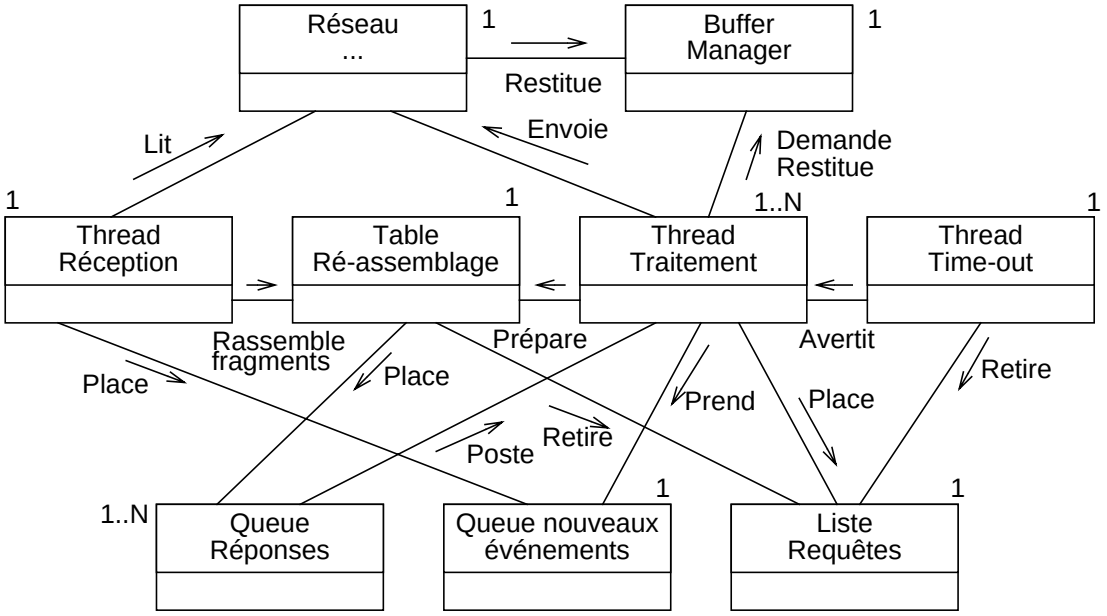


Figure 9-7: Diagramme de classes de l'application du nœud processeur.

Lorsqu'un message est reçu et qu'il s'agit d'un événement transmis par un superviseur, celui-ci est placé dans la queue des nouveaux événements. Le premier Thread Traitement qui était en attente est débloqué (synchronisation par sémaphore) et prend en charge l'événement. La partie en charge de l'algorithme de sélection analyse les informations fournies pour cet événement (relatives aux RoIs par exemple) et décide de demander certaines données. Les sources concernées sont identifiées. Le Thread Traitement prépare alors un message de requête (en demandant au Buffer Manager un *buffer* pour contenir ces messages) et prépare également une structure dans la Table de Ré-assemblage destinée à recueillir les fragments de données qui seront reçus. Le Thread Traitement place la requête dans une liste (Liste Requêtes) contenant toutes les requêtes à servir, poste le message à transmettre par le réseau et se bloque en attente sur la Queue Réponses correspondante.

Le Réseau transmet les requêtes puis recycle le *buffer* correspondant. Selon les cas, le même message de requête est envoyé à chaque source concernée (sans dupliquer physiquement le message à envoyer), ou bien l'envoi se fait en une seule opération en utilisant une connexion point-à-multi-points. Ce dernier mode de transmission est possible pour certains groupes de sources pré-déterminés (par exemple toutes les sources d'un détecteur ou de l'ensemble des détecteurs).

Les messages contenant les fragments d'événements en provenance des sources sont placés dans la Table de Ré-assemblage. Lorsque tous les fragments correspondants à une requête sont disponibles, un message est placé dans la Queue de Réponses du Thread Traitement demandeur. La requête est supprimée de la Liste des Requêtes.

Le Thread Traitement prend en compte la notification, formate les données reçues, retourne les *buffers* qui contenaient les messages d'origine à leur propriétaire (l'objet Réseau), retourne à la Table de Ré-assemblage la ressource utilisée et fournit le bloc de données complet à la couche supérieure en charge de l'exécution de l'algorithme de sélection. Suivant les cas, une ou plusieurs

itérations du cycle de demande de données, d'attente puis de traitement ont lieu jusqu'à ce que la décision de trigger soit prise. Les résultats à retourner au superviseur sont formatés, puis envoyés par l'intermédiaire du Réseau. le Thread Traitement tente alors de lire un nouvel événement dans la Queue des nouveaux événements, et se bloque en attente si celle-ci est vide.

La liste des requêtes à servir (Liste Requêtes) est balayée périodiquement par un Thread Time-out vérifiant que certaines d'entre-elles ne restent pas sans réponse pendant plus d'un certain temps ("durée de vie" de la requête). Si une requête a expiré, le Thread Time-out la retire de la liste correspondante et informe le Thread Traitement demandeur que sa requête n'a pas pu être servie dans le temps imparti. Ce cas d'erreur est traité par le Thread Traitement en abandonnant le traitement de l'événement et en retournant au superviseur un message d'erreur. Ce dernier peut décider de conserver ou de rejeter l'événement.

9.6.3. Mesures de performance

Les mesures sur les processeurs destination visent à caractériser les différentes couches de logiciel de cette application. La couche la plus basse est le protocole de communication avec les différents nœuds du système (requête et collecte des données, communication avec le superviseur). Un niveau intermédiaire est la couche de présentation des blocs de données reçus dans un format adapté pour le traitement. Le niveau supérieur est l'algorithme de traitement des données proprement dit. La mesure du temps d'exécution des différents algorithmes de traitement peut être faite en dehors des démonstrateurs. Il s'agit d'une étude pouvant être menée de manière indépendante. La caractérisation des autres couches est un des objets du démonstrateur.

Dans le démonstrateur, le fichier de paramètre permet la description d'un algorithme fictif exécuté par chaque processeur suivant la syntaxe:

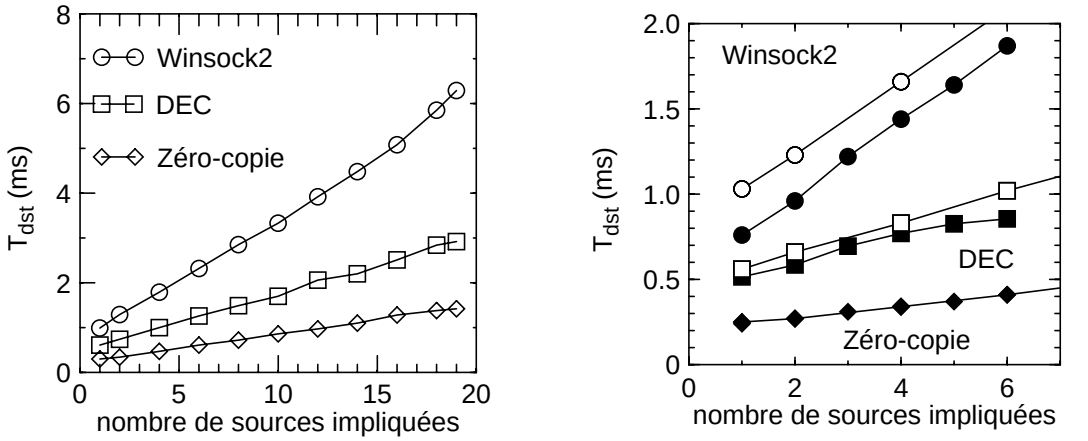
<Etape> <type_algo> <nb_src> <nb_RoI> <durée> <prob_oui> <prob_non> <p_goto E>

- <Etape> est le numéro de l'étape courante de l'algorithme,
- <type_algo> est le type de traitement à effectuer. Il peut s'agir d'un traitement nul retournant une décision de trigger sans demander des données aux sources, d'un algorithme générique demandant des données de <nb_src> sources (prises de manière aléatoire ou non) ou bien d'un véritable algorithme de traitement de données;
- <nb_RoI> donne pour l'algorithme générique le nombre de RoIs à collecter à cette étape;
- <durée> donne le temps de traitement à simuler pour l'algorithme générique;
- <prob_oui> donne la probabilité d'accepter l'événement à cette étape;
- <prob_non> donne la probabilité de rejeter l'événement à cette étape. Les événements qui ne sont ni acceptés ni rejetés passent à l'étape suivante (Etape+1) ou bien à l'étape E avec la probabilité <p_goto> (probabilité de branchement direct à l'étape numéro E).

Cette méthode de description est assez souple et permet de simuler des séquences de traitements relativement complexes sans implémenter les véritables algorithmes correspondants. Pour les mesures de performance du protocole de collecte des données, un algorithme ne comportant qu'une seule étape et ne faisant aucun traitement est utilisé. La Figure 9-8a présente le temps d'occupation du processeur local, T_{dst}, pour la collecte d'un nombre variable de fragments de taille minimale (44 bytes, i.e. 1 cellule ATM). Pour ce test, la configuration du démonstrateur est 19 nœuds sources, 1 superviseur, 1 moniteur et 1 processeur destination [Cal98a]. Les matériels utilisés comme nœud destination sont successivement:

- une station de travail DEC Alpha – 333 MHz (Digital Unix) équipée d'une carte réseau ATMWorks 350 utilisant l'API d'accès direct à la couche ATM AAL 5 fournie par DEC,

- un PC Pentium II – 300 MHz (WindowsNT) équipé d'une carte réseau ATM FORE PCA 200 et utilisant l'interface de programmation Winsock2 pour un accès à la couche ATM AAL 5,
- un PC identique au précédent mais équipé d'une carte ATM basée sur NicStar et utilisant la bibliothèque de fonctions de communication optimisées (zéro-copie).



a) T_{dst} vs. nombre fragments b)T_{dst}, démonstrateur 19 et 6 sources

Figure 9-8: Temps CPU pour la collecte de fragments de N sources.

Le temps d'utilisation du CPU local varie linéairement avec le nombre de fragments à collecter. La configuration la plus performante est le PC avec communications optimisées.

Le système final comportant plusieurs centaines de sources, il est important d'étudier la dépendance du temps de collecte des fragments d'un nombre fixé de sources lorsque le nombre total de sources dans le système varie. Le test correspondant est effectué pour deux configurations du démonstrateur: la première comporte un total de 19 nœuds sources et la seconde n'en comporte que 6. Les résultats sont présentés Figure 9-8b. Les courbes supérieures correspondent au système à 19 sources. On constate une différence pour les deux configurations sur la machine DEC et le PC/ Winsock2. Une explication possible est la dépendance vis-à-vis du nombre de *sockets* du temps requis par l'appel système *select* lors de la réception d'un message (pour déterminer de quelle *socket* un message peut être lu).

D'une manière générale, la gestion concurrente de plusieurs centaines de *sockets* par une application n'est pas une configuration courante; des difficultés n'ont pas été résolues pour pouvoir utiliser plus de 64 *sockets* simultanément avec la version du pilote et de la bibliothèque Winsock 2 fournis par FORE Systems. A l'inverse, T_{dst} est indépendant du nombre de sources dans le système avec l'utilisation de la bibliothèque optimisée. D'après la structure de la carte incorporant NicStar et celle du logiciel de contrôle, on peut émettre l'hypothèse que les performances observées sur le système réduit ne dépendent pas de manière sensible du nombre total de sources.

On peut mettre T_{dst} sous la forme suivante:

$$T_{dst} = T_{dst0} + T_{req} \times N\text{src\_requested} \qquad \textbf{(9-3)}$$

Les valeurs de ces deux paramètres, déduites des mesures sur différentes machines (en utilisant la bibliothèque ATM zéro-copie dans tous les cas), sont présentées Table IX-2.

La différence de performance entre WindowsNT et Linux peut venir des optimisations effectuées par les compilateurs et de temps plus courts, avec Linux, pour la commutation entre

threads et la gestion des interruptions.

Machine	T_{dst0} (µs)	T_{req} (µs)
PowerPC 100 MHz – Lynx OS	362	26
PowerPC 200 MHz – Lynx OS	186	23
PowerPC 300 MHz – Lynx OS	128	18
Pentium Pro 200 MHz – WindowsNT	178	49
Pentium II 300 MHz – WindowsNT	137	34
Pentium II 400 MHz – WindowsNT	114	26
Pentium II 300 MHz – Linux	103	14
Pentium II 400 MHz – Linux	84	11

Table IX-2: Performance du protocole de collecte de fragments.

Lorsque la totalité des fragments demandés est disponible, ces blocs de données doivent être formatés de manière à être utilisables par l'algorithme de traitement. Les opérations à effectuer dépendent du type de détecteur, de l'organisation du système de lecture, format des données brutes et format attendu par l'algorithme de traitement. Dans le démonstrateur, le formatage effectué est très simple et consiste à recopier (éventuellement avec changement d'ordre des octets) dans une zone mémoire contiguë l'ensemble des fragments reçus. En mesurant à basse fréquence le temps de traitement par événement en effectuant puis en n'effectuant pas l'opération de recopie, on peut déterminer la vitesse de formatage B_F. Des résultats de mesures sont présentés dans la Table IX-3.

B_F (Mo/s)	Ordre réseau	
Machine	*Big Endian*	*Little Endian*
PowerPC 300 MHz	30	26
PC Pentium II 400 MHz	50	62

Table IX-3: Performance pour un formatage simple.

Les données transportées sont toutes du type entier 32 bit. La performance est la meilleure lorsque l'ordre de transmission des octets sur le réseau est identique à l'ordre interne de la machine (i.e. *Big Endian* pour le PowerPC et *Little Endian* pour les processeurs Intel). Le PC est dans tous les cas le plus performant: fréquence d'horloge du processeur supérieure de 25% et bus mémoire fonctionnant à 100 MHz (contre 66 MHz pour la carte PowerPC). Bien que l'opération de formatage dans le système final soit plus complexe qu'une simple copie de données, une hypothèse réaliste pour la modélisation est une vitesse de ~50 Mo/s pour la manipulation des données avant traitement.

La dernière couche de logiciel est le traitement effectif des données. Les algorithmes de traitement sont en cours de développement par divers groupes qui évaluent également le temps d'exécution correspondant. Afin de réduire la complexité du démonstrateur, la plupart des mesures sont faites en simulant un temps de traitement identique à celui mesuré pour l'algorithme correspondant. La génération de données issues de simulation par les ROBs et l'exécution de

véritables algorithmes de traitement dans les processeurs est un objectif qui n'a pas été atteint dans les premières versions du démonstrateur. Cette étude est poursuivie dans le cadre d'un projet plus large que ce démonstrateur compte tenu de l'effort à fournir et de la main d'œuvre nécessaire[1]. Un algorithme de traitement simple a néanmoins été intégré dans le démonstrateur (traitement des RoIs électron/gamma dans le calorimètre), le temps d'exécution correspondant est ~80 µs (sur un PC 400 MHz). Davantage de détails sont disponibles dans [Cal99].

Pour cette étude, le temps de traitement des données est pris comme un paramètre déterminé à l'extérieur du démonstrateur. Il est noté T_{algo}. Le temps de traitement de l'étape i d'un algorithme séquentiel est noté $T_{step}(i)$. Lorsque le temps de transmission des données est inférieur au temps de collecte, de formatage et de traitement, on a:

$$T_{step}(i) = T_{dst0} + T_{req} \times Nrequests(i) + \frac{eventsize(i)}{B_F} + T_{algo}(i) \qquad \textbf{(9-4)}$$

Dans ce cas le système est limité par le nœud processeur; $T_{step}(i)$ est le temps d'utilisation du CPU local. Lorsque le temps de transfert des données est dominant, on a simplement:

$$T_{step}(i) = \frac{eventsize(i)}{B_D} \qquad \textbf{(9-5)}$$

En fonction des différents paramètres, on peut déterminer la limite de fonctionnement entre ces deux régimes. Par exemple, en utilisant un lien ATM 155 Mbit/s avec un PC 400 MHz, la détermination du régime de fonctionnement se fait à l'aide du graphe de la Figure 9-9. Considérons par exemple la collecte de 1 ko depuis 8 sources. Le point de fonctionnement est à la limite des deux zones, mais avec un temps de traitement T_{algo} non nul, T_{step} est déterminé par le temps d'utilisation du CPU du nœud destination.

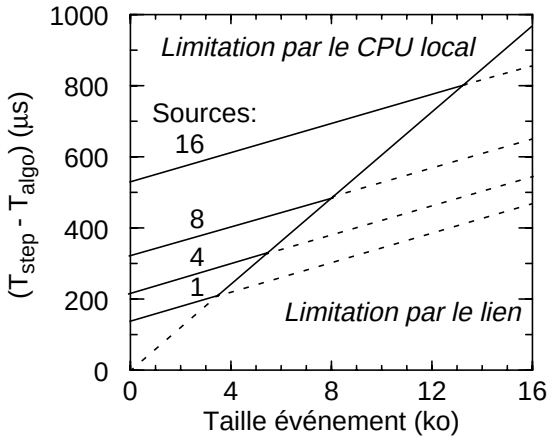


Figure 9-9: Régime de fonctionnement du nœud destination.

Si le volume de données à collecter est de 2 ko par source (depuis 8 sources), T_{step} est déterminé par le temps de transmission des données (950 µs) tant que la durée de traitement est inférieure à ~300 µs.

Le temps de traitement total pour une séquence de K étapes est:

1. voir la description du projet pilote du trigger de niveau 2 d'ATLAS présentée au chapitre suivant.

$$T_{\text{selection}} = \sum_{i=1}^{K} A(i) \cdot T_{step}(i) \quad \textbf{(9-6)}$$

où A(i) est la probabilité d'exécution de chaque étape. Le nombre de nœuds processeurs requis pour traiter un flux d'événement arrivant à intervalles T_a est:

$$D = \frac{1}{\alpha} \cdot \frac{T_{\text{selection}}}{T_a} \quad \textbf{(9-7)}$$

où α est la charge maximale admissible sur le système. Ce paramètre peut être choisi de manière à disposer d'une marge de sécurité suffisante (0.75 par exemple). Ces formules sont exploitées lors de la modélisation du système pour évaluer le nombre de processeurs requis.

9.7. Co-processeur

Le nœud nommé co-processeur est présent dans une variation de l'architecture étudiée. Compte tenu de la puissance de calcul nécessaire pour effectuer le *TRT Full Scan*, une option d'architecture consiste à ajouter un ou plusieurs co-processeur(s) dédié(s) à cette tâche. Il peut s'agir d'une carte spécifique ajoutée à chaque nœud processeur ou bien d'un bloc commun à l'ensemble des processeurs communiquant à travers le réseau liant les différents nœuds. Seule cette dernière option a été étudiée dans le démonstrateur (configuration correspondant au schéma de la Figure 4-2).

9.7.1. Matériel

Le co-processeur comporte deux parties:

- un PC (Pentium II 400 MHz, WindowsNT) connecté au réseau ATM ainsi qu'à une machine dédiée par une liaison haut débit (S-link dans cette implémentation),
- la machine dédiée au traitement des données (Enable++ [Kug98]).

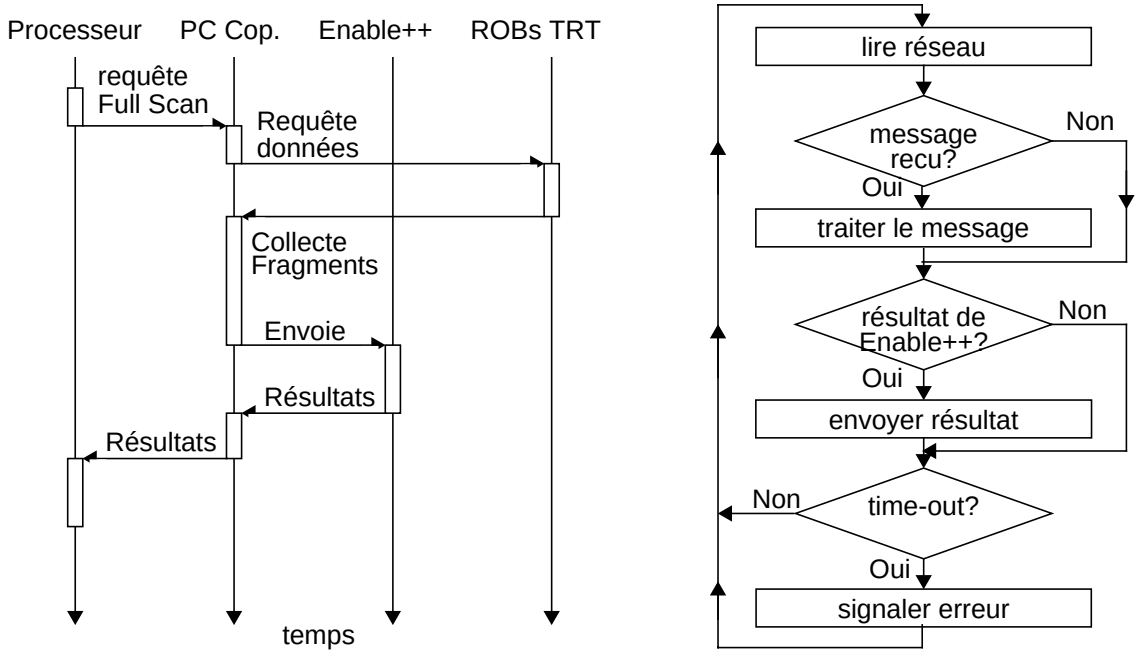
Le rôle du PC est de recevoir les requêtes des processeurs de traitement, de les transmettre à toutes les sources du TRT et de retourner les résultats des calculs de la machine dédiée aux processeurs demandeurs. Une possibilité pour la collecte des données des ROBs vers la machine Enable++ est la mise en place d'un réseau spécifique réalisé en logique câblée (élément du modèle de l'architecture flot de données décrite au chapitre précédent). Compte tenu de la difficulté de réalisation, une approche simplificatrice a été adoptée. Elle consiste à utiliser le réseau ATM pour ces transferts de données. Une tâche supplémentaire du PC est donc de collecter les données de tous les ROBs du TRT et de les transmettre à la machine Enable++.

9.7.2. Logiciel

La séquence des opérations pour l'exécution du *TRT Full Scan* dans un système incluant le co-processeur est présentée Figure 9-10a. C'est un système de reconstruction d'événements constitué de toutes les sources du TRT et d'une destination unique (le co-processeur).

L'application fonctionnant sur le PC du nœud co-processeur est une boucle infinie qui effectue un *polling* sur la carte réseau et sur le lien connecté à la machine spécifique. L'organigramme est présenté Figure 9-10b. Lorsqu'un message de requête en provenance d'un

processeur est reçu, le PC co-processeur poste cette requête à tous les ROBs du TRT (*multi-cast*) et reçoit par la suite les fragments demandés de manière asynchrone. Lorsque tous ont été reçus, une liste chaînée est constituée et transmise à la machine Enable++ via S-link. Ce transfert peut se faire par DMA sans recopie de données. En fin de transfert, les *buffers* sont retournés au contrôleur de la carte réseau. Le PC alloue également un *buffer* destiné à recevoir le résultat des traitements. Lorsque la machine Enable++ a fini sa tâche, les résultats sont placés dans ce *buffer* (également sans recopie de données) et transmis au demandeur par le réseau ATM. Le PC co-processeur vérifie également en permanence qu'aucun débordement de temps ne se produit (demande de données des ROBs ou attente de résultats venant de Enable++) et retourne au demandeur un message d'erreur en cas de problème. Ce mécanisme assure la robustesse de fonctionnement.



a) diagramme temporel b) tâche principale du PC co-processeur

Figure 9-10: Principe de fonctionnement du Co-processeur.

La partie relative au transfert des données vers la machine Enable++ à travers le S-Link et la programmation de la machine Enable++ n'est pas considérée dans cette thèse; il s'agit de la partie prise en charge par nos collaborateurs de l'université de Mannheim.

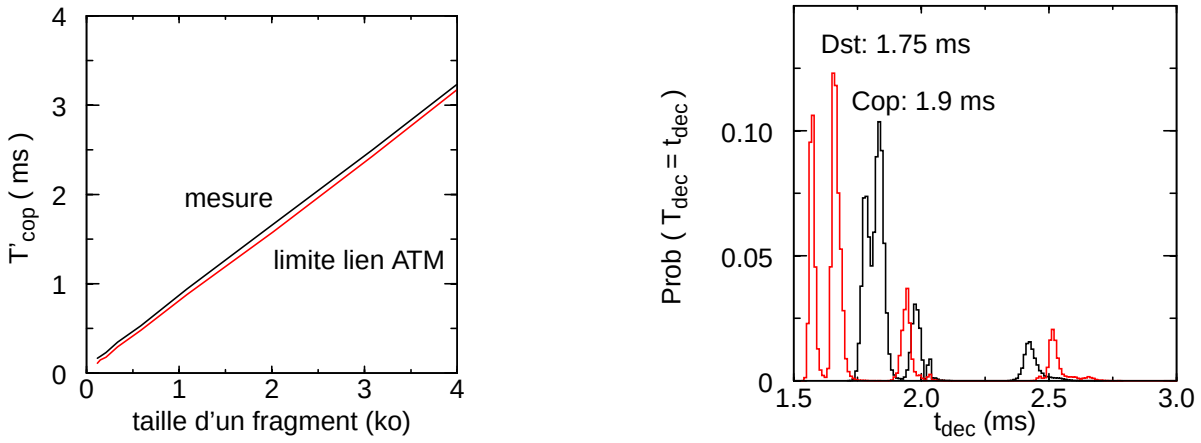
9.7.3. Mesures de performance

Le nœud co-processeur comporte deux parties distinctes pouvant fonctionner de manière indépendante. La mise au point du protocole de communication a été faite sans la connexion à la machine Enable++. Dans ce mode de fonctionnement, l'opération consistant à envoyer les données à la machine Enable++ ne fait que recycler les *buffers* correspondants; la lecture des résultats est immédiate (on retourne une valeur constante).

Le temps d'utilisation du CPU pour la tâche considérée est noté T_{cop} (excluant le temps de communication avec Enable++). Pour la collecte de fragments de taille minimale, sur un Pentium II 400 MHz, on mesure:

$$T_{cop} = 140\mu s + 2\mu s \times nbofRobs \qquad \textbf{(9-8)}$$

Ces chiffres sont proches de ceux obtenus lors du test du nœud processeur (fonctionnalité proche mais plus simple dans le cas présent). Lorsque la taille des fragments collectés augmente, le temps de manipulation d'un événement T'_{cop} est déterminé par la bande passante du lien ATM comme le montre la Figure 9-11a. Lors de ce test, le démonstrateur comporte 13 sources TRT, 12 processeurs destinations (de type différent), 1 superviseur, 1 moniteur et 1 co-processeur.



a) collecte des données

b) délai du cycle de traitement

Figure 9-11: Performance du PC co-processeur.

La Figure 9-11b montre le délai du cycle complet, T_{dec}, pour un événement dans deux configurations:

- Dst: cycle superviseur -> processeur -> requête à 13 sources d'un fragment de taille minimale -> collecte des 13 fragments -> retour au superviseur;
- Cop: cycle superviseur -> processeur -> requête au co-processeur -> requête à 13 sources d'un fragment de taille minimale -> collecte des 13 fragments -> retour au processeur -> retour au superviseur.

Dans les deux cas, le système fonctionne à faible charge. Les pics observés correspondent aux différents temps de réponse des processeurs destination utilisés (système hétérogène). Le temps de latence ajouté par la mise en place du co-processeur est ~0,15 ms. Ce paramètre n'est pas véritablement significatif, mais la mesure permet d'illustrer le fonctionnement du système.

L'étape suivante dans l'évaluation du co-processeur est la connexion de la machine Enable++ au PC du nœud co-processeur. Cette étude n'a pas pu être menée avec succès. Pour des raisons techniques (utilisation de deux cartes µEnable avec des interfaces S-Link 3,3V et 5V), des transferts de manière stable entre un PC et la machine Enable++ n'ont pas pu être réalisés par nos collaborateurs. Compte tenu du manque de main d'œuvre et de l'obsolescence de la machine Enable++, l'objectif d'intégration des deux blocs a été abandonné. Je pense que cette décision est regrettable dans la mesure où la partie liée au démonstrateur ATM et à la collecte des données est opérationnelle. Les résultats obtenus permettent néanmoins de poursuivre l'étude de modélisation du système. Dans le principe, cette solution comporte néanmoins divers points délicats. L'ensemble du trafic pour le *TRT Full Scan* est concentré vers un point unique qui risque fort de constituer un goulet d'étranglement (même en équipant ce nœud avec une liaison de bande passante plus élevée). En cas de défaut du co-processeur, l'ensemble du système risque d'être affecté (on peut envisager plusieurs co-processeurs, mais cela complique l'architecture). Une autre option d'architecture est d'intégrer à chaque processeur destination une carte accélératrice (à base de FPGAs par exemple). Cette étude en cours n'est pas détaillée dans cette thèse.

9.8. Contrôle et Moniteur

9.8.1. Matériel

Le *run control*/moniteur est volontairement très simplifié dans le démonstrateur. Le moniteur est constitué d'une machine (PC ou autre) connectée au réseau ATM. Cette machine peut également être utilisée pour commander le démarrage de tous les nœuds (ordre envoyé par le réseau Ethernet standard). Au démarrage, chaque nœud accède à un fichier de configuration commun (via Ethernet), et démarre l'exécution de la tâche correspondant à son rôle: superviseur, source, destination, co-processeur ou moniteur. Les communications s'effectuent dès lors uniquement par le réseau ATM, sous le contrôle du moniteur. L'interface avec l'utilisateur se fait par une console ASCII pour des raisons de simplicité. Les commandes disponibles permettent à l'utilisateur de démarrer, suspendre et arrêter les différents nœuds du système, collecter et sauvegarder les statistiques sur disque et changer divers paramètres sans qu'il soit nécessaire de stopper et re-démarrer le système (nombre d'événements à générer, fréquence, taille des données des sources...).

9.8.2. Logiciel

Le nœud moniteur est en charge de la surveillance du fonctionnement du système et de l'interface avec l'utilisateur. L'application moniteur comporte plusieurs *threads* en charge de:

- la réception des messages venant du réseau (ATM),
- l'émission de messages sur le réseau (ATM),
- la détection des *time-out*,
- l'affichage d'informations sur la console,
- l'interface avec le programme communiquant avec l'utilisateur (on utilise un tube nommé, "*pipe*").

Périodiquement, le moniteur transmet à chaque nœud du système un message "*Echo Request*" et moins fréquemment un message "*Statistics Request*" (rapport 1:100 par exemple). Les communications avec le moniteur utilisent des connexions CBR à très faible débit (0,1% de la bande passante disponible) ce qui ne perturbe pas les autres communications. L'état des différents nœuds est affiché périodiquement sur la console. En cas de mauvais fonctionnement d'un nœud, aucune action particulière n'est tentée, l'utilisateur doit agir en conséquence. Cette méthode est suffisante dans le cas d'un démonstrateur.

L'interface utilisateur est découplée de l'application moniteur et communique avec ce dernier en utilisant un tube nommé. Les commandes de l'utilisateur sont interprétées puis déposées dans ce tube nommé, les résultats des actions sont relues depuis le tube nommé puis présentées à l'utilisateur. Cette séparation permet de conserver inchangée l'application moniteur quelle que soit l'interface utilisateur adoptée (graphique ou non). La seule restriction est de faire fonctionner l'application moniteur et l'interface utilisateur sur la même machine. Le remplacement du tube nommé par une *socket* TCP/IP pourrait permettre davantage de flexibilité, mais ceci n'a pas été jugé nécessaire.

9.8.3. Mesures de performance

Le moniteur n'est pas un élément critique dans le système et aucune mesure de performance relative à ce dernier n'a été faite. Le moniteur développé pour le démonstrateur n'est qu'un élément permettant d'observer et de contrôler le système étudié et ne comporte pas les fonctionnalités requises pour le système final.

9.9. Performance du système complet

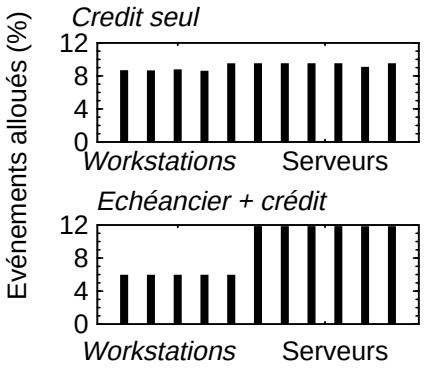
Après avoir présenté le principe de fonctionnement et la performance de chaque élément pris de manière isolée, les résultats obtenus avec le système complet sont décrits.

9.9.1. Equilibrage de la charge des processeurs destination

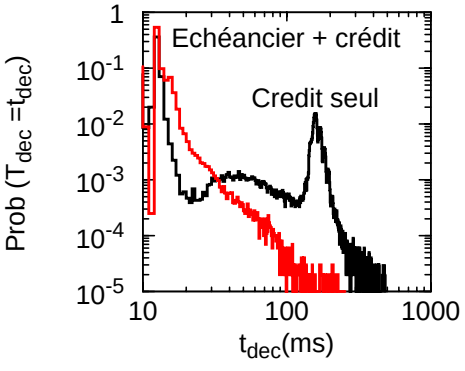
Compte tenu de la diversité possible des processeurs de traitement des données, il est important que le superviseur distribue les événements en fonction de la capacité de chaque processeur. A cet effet, un mécanisme d'échéancier et de crédit a été développé (voir section 9.4.2).

Le test suivant est effectué sur le démonstrateur à 24 noeuds rapporté dans [Cal98a]. Le système comporte 1 superviseur, 1 moniteur, 11 sources et 11 processeurs destination dont 5 sont des stations de travail mono-processeur et les 6 autres des serveurs multi-processeurs (4 serveurs quadri-processeurs et 2 serveurs dodéca-processeurs). La tâche des processeurs pour chaque événement consiste à collecter 1 ko de 4 sources prises de manière aléatoire parmi les 11, formater les données reçues, effectuer un traitement fictif durant 10 ms et retourner une décision de trigger aléatoire au superviseur. On mesure la fréquence maximale de fonctionnement du système, puis on fait fonctionner le système à 50% de cette valeur (soit 1 kHz).

Dans un premier test, la distribution des événements est faite par le superviseur en mode *round-robin* (le mécanisme de crédit est néanmoins conservé). La répartition des événements sur les processeurs qui est obtenue est présentée Figure 9-12a (Crédit seul). On remarque que toutes les destinations reçoivent approximativement la même fraction du nombre total d'événements. Le temps du cycle par événement depuis sa création par le superviseur jusqu'au retour de la décision de trigger (T_{dec}) est présenté Figure 9-12b (Crédit seul). La valeur moyenne est 51 ms. Le pic à 200 ms correspond aux événements alloués aux stations de travail qui fonctionnent en régime saturé. Le mécanisme de crédit seul ne permet pas de distribuer les événements en tenant compte de la disparité des processeurs destination. Ce n'est que lorsque le crédit d'une destination est épuisé que l'allocation de cette destination est suspendue. Lors du fonctionnement du système à 50% de charge, cette situation survient rarement.



a) repartition des événements



b) distribution de T_{dec}

Figure 9-12: Equilibrage de charge sur les destinations.

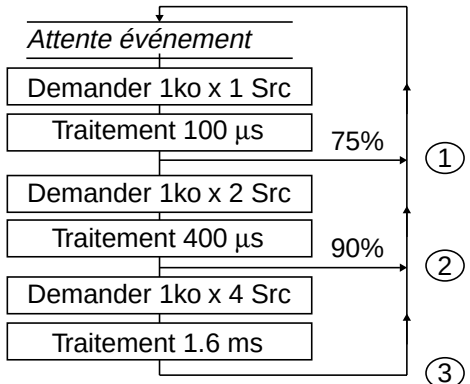
Dans un second test, le mécanisme d'échéancier est utilisé. La table d'allocation comporte deux fois plus d'entrées pour les serveurs multi-processeurs que pour les machines mono-processeur. La répartition des événements sur les destinations observée est présentée Figure 9-12a (Echéancier + crédit). Dans ce cas, les serveurs traitent ~2 fois plus d'événements que les stations de travail. La distribution de T_{dec} est présentée Figure 9-12b (Echéancier + crédit). La valeur moyenne

est réduite à 14 ms. La distribution non uniforme des événements aux destinations permet un meilleur équilibrage de la charge des divers processeurs de traitement.

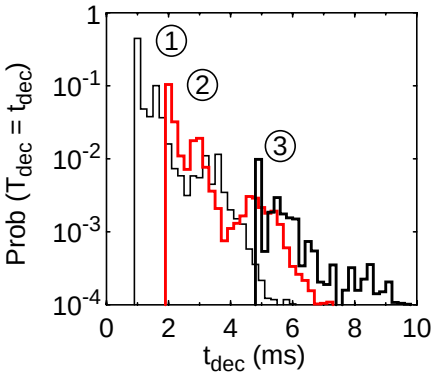
Il est important d'effectuer la distribution des événements de manière à tirer profit des machines plus performantes que d'autres à condition que la limitation du système provienne des processeurs de traitement. Dans le cas où la limitation est due à la saturation du lien au réseau de certaines destinations, il est à l'évidence néfaste de tenter d'allouer davantage d'événements à ces destinations (quelle que soit leur puissance de calcul). Dans le système final, les paramètres pour la distribution des événements devront être ajustés en tenant compte à la fois de la puissance relative des processeurs et de la bande passante disponible au niveau des liens au réseau.

9.9.2. Transfert et traitement séquentiel

Ce mode de fonctionnement est le mode d'opération normal pour le trigger de niveau 2. Les processeurs effectuent un algorithme séquentiel alternant les phases de requête de données et de traitement. La Figure 9-13a présente un algorithme séquentiel de test.



a) traitement séquentiel



b) histogramme de T_{dec} pour chaque étape

Figure 9-13: Transferts et traitements séquentiels.

Les mesures de T_{dec} effectuées sur le démonstrateur de la Figure 9-1 sont présentées Figure 9-13b. Le système comporte 4 sources, 4 destinations, 1 superviseur et 1 moniteur. La fréquence des événements est 3,35 kHz (50% de la capacité du système). La première étape du traitement est effectuée avec une priorité supérieure à celle des autres étapes. La dernière étape est la reconstruction complète de l'événement. Ce test permet de valider le principe de fonctionnement du système (protocole requête/réponse, alternance des phases d'attente de données et de traitement, prise en charge simultanée de plusieurs événements dans chaque processeur, mélange de différents types de trafic dans le réseau, etc.). Davantage de détails sont rapportés dans [Cal97c].

Une autre évaluation avec un algorithme de test composé de 1 à 16 étapes identiques est décrit dans [Cal98b].

9.9.3. Mode de fonctionnement *event builder*

Dans ce mode de fonctionnement, les processeurs destination collectent l'ensemble des données de toutes les sources pour chaque événement. Cette configuration est celle du trigger de niveau 3 (hypothèse de reconstruction complète de l'événement après le trigger de niveau 2).

9.9.3.1. Comportement du réseau

Le test suivant est effectué sur le démonstrateur à 24 nœuds rapporté dans [Cal98a]. Les 11

destinations collectent pour chaque événement un bloc de 4 ko des 11 sources (soit 44 ko au total), puis retournent une décision de trigger aléatoire au superviseur sans effectuer aucun formatage ni traitement (le but de ce test étant d'étudier le comportement du réseau). On mesure la fréquence maximale de reconstruction d'événements observée, F_{eb}, en fonction de la fréquence tentée, F_{ar} (i.e. la fréquence d'événements fixée par le superviseur tant que le mécanisme de contrôle de flux ne force pas de ralentissement). On mesure également le nombre d'événements qui n'ont pas pu être reconstitués complètement (e.g. à cause de la perte d'un ou de plusieurs fragments).

Dans un premier cas, toutes les connexions virtuelles utilisées sont du type UBR, les sources peuvent utiliser la pleine bande passante de chaque liaison. La Figure 9-14a (UBR) présente F_{eb} en fonction de F_{ar}.

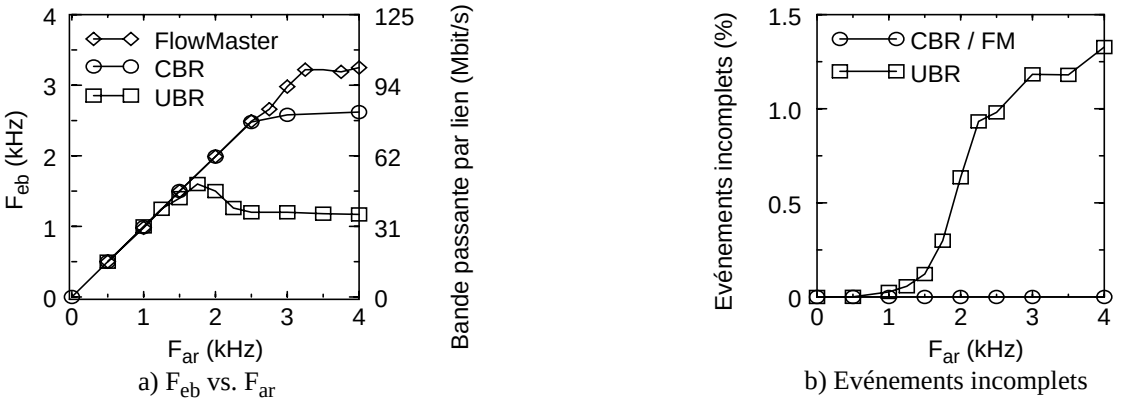


Figure 9-14: Comportement du réseau avec le trafic de type *event builder.*

Tant que F_{ar} est inférieure à ~1,5 kHz, le système se comporte de manière satisfaisante. Lorsque F_{ar} augmente, la concentration du trafic vers chaque destination entraîne le débordement de capacité des éléments des commutateurs ATM et la perte de fragments d'événements (Figure 9-14b). Tenter d'augmenter davantage F_{ar} ne fait qu'aggraver la situation en créant davantage de congestion dans le réseau. On observe un effondrement des performances du réseau. Le mécanisme de crédit du superviseur permet de limiter la fréquence des événements injectés dans le système et d'éviter la perte croissante d'un nombre de messages. Dans le meilleur des cas, l'utilisation des liens ATM est de ~50 Mbit/s (soit 40% de la bande passante disponible).

Dans un second test, les connexions virtuelles utilisent le mécanisme de contrôle de flux propriétaire de Digital, "*FlowMaster*" [Sur97]. Le système atteint un régime de saturation autour de 3 kHz, aucune perte d'événement n'est observée (Figure 9-14a et 9-14b, *Flow Master*). L'utilisation des liens ATM est ~100 Mbit/s. Ce mécanisme de contrôle de flux permet effectivement d'éviter les pertes de cellules ATM. L'inconvénient est que le *Flow Master* est un mécanisme non standard qui n'est disponible que sur les produits de Digital.

Le troisième test est effectué en utilisant des connexions virtuelles du type CBR. Pour des raisons techniques, la fraction de la bande passante allouée par les cartes réseau de Digital aux connexions CBR est limitée à 90% de la bande passante totale. La bande passante allouée à chaque couple source/destination est 1:11 de cette fraction (soit ~12 Mbit/s par connexion virtuelle). Le système atteint un régime de saturation autour de 2,5 kHz, et comme dans le cas d'utilisation du *Flow Master*, aucune perte d'événement n'est observée (Figure 9-14a et 9-14b, CBR). Les performances observées sont inférieures à celle obtenues avec le *Flow Master* car seulement 90% de la bande passante est affectée aux connexions CBR alors que le *Flow Master* peut utiliser 100% de celle-ci. Le système a été testé pendant plus d'une heure à pleine charge (~400 Go de données

transmis) sans observer de pertes.

Ces tests montrent la nécessité d'utiliser un mécanisme de contrôle de flux adapté pour le bon fonctionnement d'un système de reconstruction d'événements. L'utilisation de connexions CBR est une solution. Il s'agit d'une méthode standard en ATM supportée par de nombreux produits.

9.9.3.2. Performance de l'*event builder*

Les mesures de performance sont présentées pour le démonstrateur à 32 nœuds décrit dans [Cal99]. Le système comporte 14 sources et 14 destinations, 2 superviseurs, 1 moniteur et 1 co-processeur (inutilisé lors de ce test). Toutes les machines utilisent des cartes ATM NicStar et les fonctions de communication zéro-copie. La fréquence maximale de fonctionnement F_{eb} en fonction du nombre de destinations est présentée Figure 9-15.

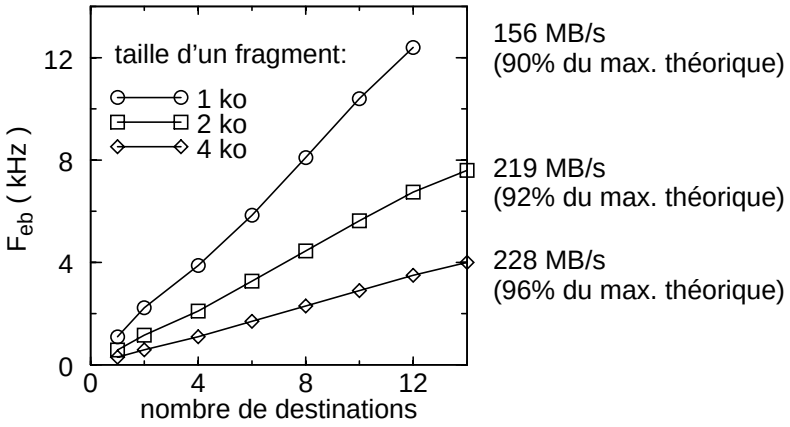


Figure 9-15: Performance en mode *event builder.*

Les connexions virtuelles sont du type CBR; la totalité de la bande passante de chaque lien peut être affectée à ces connexions. La performance augmente linéairement avec le nombre de destinations. Pour des fragments d'événements de 4 ko, le débit total est de 228 Mo/s, soit 96% de la bande passante totale des liens. Le système fonctionne de manière stable, des pertes de données occasionnelles sont observées (erreurs de transmission dues à la qualité des câbles, alimentations...).

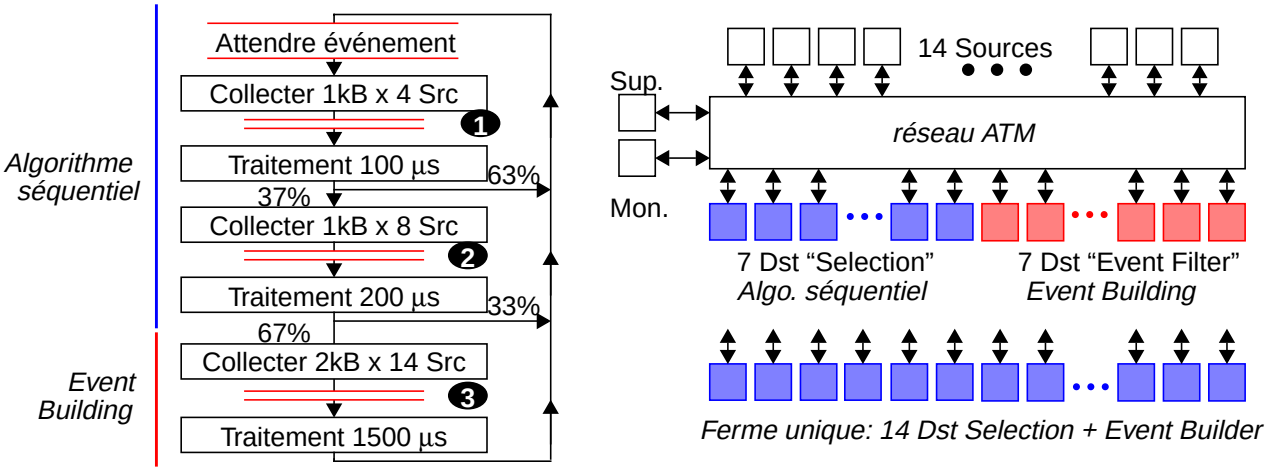
En vue du système final, la performance requise pour l'*event builder* est de ~1000 Mo/s. Ce démonstrateur a ~1/4 de la performance requise pour un système représentant ~1:32 du système final. Du point de vue de la transmission et du ré-assemblage des événements, l'objectif de performance à atteindre semble donc réaliste. La capacité des processeurs à effectuer les algorithmes complexes du filtre d'événements à la fréquence requise est un problème différent qui n'est pas considéré dans cette étude.

9.9.4. Modes fermes séparées / ferme commune

Ce mode de fonctionnement vise à tester l'intégration du trigger de niveau 2 et de niveau 3 (*event builder* seulement) en utilisant un réseau unique pour les transferts des données. Deux variations d'architecture sont considérées: la première comporte des processeurs distincts pour le trigger de niveau 2 et celui de niveau 3, la seconde est basée sur un groupe unique.

Un algorithme de traitement test est présenté Figure 9-16a. Les deux premières étapes sont un algorithme séquentiel (figurant le trigger de niveau 2), la dernière étape est la reconstruction complète de l'événement suivie d'un traitement fictif (figurant le traitement du trigger de niveau 3). La fréquence des événements générés (8 kHz), les volumes de données transmises et les différents

temps de traitement sont choisis de sorte que l'utilisation de tous les liens du réseau soit de 50% et que l'utilisation du CPU des destinations soit également de 50%. Le trafic généré par chaque source représente 8 Mo/s: 4 Mo/s pour les besoins de l'algorithme séquentiel et 4 Mo/s pour l'*event builder*. La dernière étape est effectuée pour un événement sur 4 (c'est à dire à 2 kHz). Dans le système final, en faisant l'hypothèse d'un *event builder* comportant 512 sources, fonctionnant à 1 kHz et rassemblant des événements de 1 Mo, le débit correspondant par source est de 2 Mo/s, soit la moitié de ce qui est généré lors de ce test. Le transfert des données pour les deux premières étapes utilise des connexions UBR, la dernière étape utilise des connexions CBR (1:14 de la bande passante totale du lien pour chaque paire source/destination).



a) séquence de test

b) configurations du démonstrateur

Figure 9-16: Test fermes séparées / ferme unique.

Dans un premier test, les 14 processeurs sont séparés en deux groupes (Figure 9-16b haut): 7 processeurs sont consacrés aux deux premières étapes de l'algorithme (ferme "sélection"), les 7 autres n'effectuent que la dernière étape de la séquence (ferme "filtre d'événements"). Lorsque le superviseur reçoit un événement accepté par un processeur de la ferme "sélection", il alloue et fait suivre l'information à une destination de la ferme "filtre d'événements". Cette destination envoie alors une requête à toutes les sources, rassemble 14 fragments, effectue un traitement fictif durant 1.5 ms et retourne une décision de trigger aléatoire au superviseur.

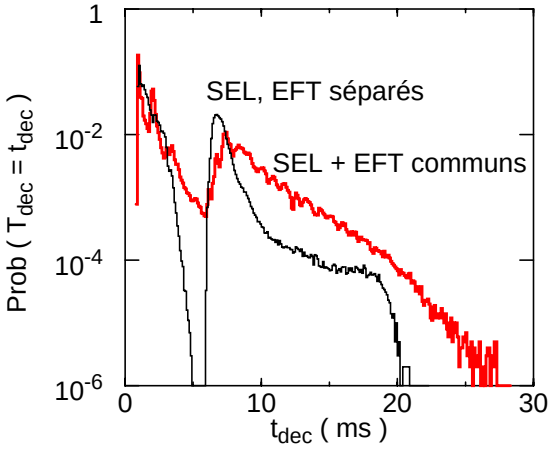


Figure 9-17: T_{dec} modèle ferme séparées / ferme unique.

On mesure au niveau du superviseur T_{dec}, le temps nécessaire pour effectuer le traitement

complet pour chaque événement (Figure 9-17, graphe SEL + EFT séparés). La valeur moyenne de T_{dec} est de 3 ms. La distribution comporte deux pics disjoints. Le premier pic correspond aux événements ayant subi la première ou les deux premières étapes (dans un premier groupe de processeurs), le second pic représente ceux ayant passé la totalité de la séquence (dont la dernière dans un des processeurs du second groupe).

Dans une autre configuration (Figure 9-16b bas), les 14 destinations sont rassemblées en une ferme unique. Chaque processeur effectue une, deux ou les trois étapes de traitement pour les événements qui lui ont été assignés. La distribution de T_{dec} est présentée Figure 9-17 (graphe SEL + EFT communs). La valeur moyenne de T_{dec} est 3,7 ms. Ce paramètre n'est pas véritablement significatif pour influencer le choix d'une configuration plutôt qu'une autre, mais le test illustre le fonctionnement des deux options.

Ce test permet de montrer qu'un réseau unique est capable de transférer différents types de trafic simultanément: allocations du superviseur, requêtes aux sources, blocs de données pour le trigger de niveau 2 et l'*event builder*, diffusion des décisions de trigger aux sources, messages de surveillance, statistiques,... Différents types de connexions et classes de service disponibles en ATM sont utilisés à cet effet.

9.9.5. Autres tests

D'autres tests sont prévus sur le démonstrateur mais n'ont pas encore été effectués à ce jour. Une première série consiste à effectuer des séquences de traitement plus complexes et plus plausibles. Une amélioration importante serait de remplacer les boucles de calcul simulant l'exécution des traitements par de véritables algorithmes, et de transporter depuis les ROBs des données issues de simulation et non des données aléatoires. Ce travail est en cours dans le projet pilote du trigger de niveau 2 qui sera décrit au chapitre suivant.

Concernant le réseau, la mise en cascade de plusieurs commutateurs est en cours de réalisation et de test (topologie en étoile)[1]. Un arrangement composé d'un coeur de commutation et de commutateurs d'accès a été en partie étudiée dans le démonstrateur décrit dans [Cal98c]. Le test plus approfondi de liaisons ATM 622 Mbit/s n'est pas envisagé dans l'immédiat. L'assemblage d'un réseau de Clos n'est pas considéré car afin de conserver la caractéristique de réseau non bloquant, le chemin de routage des messages doit être déterminé de manière dynamique, ce qui est incompatible avec l'utilisation de connexions virtuelles permanentes utilisant des chemins figés.

9.10. Résumé

Dans ce chapitre, j'ai présenté les démonstrateurs de la partie du T/DAQ d'ATLAS étudiée dans cette thèse. J'ai décrit les différents matériels utilisés et j'ai donné le principe de fonctionnement de chacun des composants. J'ai présenté les mesures de performance relatives aux différents éléments et aux démonstrateurs complets.

1. voir les résultats présentés dans l'Annexe 3 de cette thèse.

CHAPITRE X. PROJET PILOTE DU TRIGGER DE NIVEAU 2 D'ATLAS

10.1. Introduction

Dans ce chapitre je présente brièvement le projet pilote du trigger de niveau 2 d'ATLAS et je donne des résultats relatifs à l'activité de modélisation. En me basant sur ces résultats, je propose une réalisation technique pour le réseau du T/DAQ d'ATLAS. J'expose la flexibilité de cette proposition et donne les possibilités d'évolution de ce réseau.

10.2. Présentation du projet pilote

Ce projet fait suite au programme de démonstrateurs du trigger de niveau 2 d'ATLAS. Après une revue des résultats de ce programme et des conclusions qui ont été tirées, un projet commun aux différents groupes impliqués dans le trigger de niveau 2 d'ATLAS a été établi. Les motivations, l'organisation et les objectifs de ce projet pilote sont décrits dans [ATL98a]. Ce projet est en phase d'achèvement et les résultats contribuent à la rédaction du "*Technical Proposal on DAQ, High Level Triggers and Detector Control System*" en cours de rédaction (publication le 31 Mars 2000), et qui sera le document de référence définitif dans ce domaine pour l'expérience ATLAS.

Le projet pilote comporte plusieurs domaines d'activité[1]:

- ROB: étude et démonstration de performance du ROB et d'un regroupement de plusieurs ROBs,
- Superviseur: réalisation du *RoI Builder* et du superviseur,
- Réseau et processeurs: étude des solutions technologiques pour ces composants,
- Banc de test: réalisation et mesure sur des démonstrateurs composés de ROBs, superviseurs et processeurs connectés par un réseau,
- Logiciel de référence: ensemble du logiciel fonctionnant sur les éléments des bancs de test,
- Modélisation: évaluation des besoins pour le système final, extrapolation des résultats des bancs de test, simulation du système,
- Intégration: documents de référence, étude de l'intégration avec le reste de l'expérience, coût.

L'architecture du système de trigger étudiée dans ce projet repose en grande partie sur les concepts exposés dans cette thèse. Le processus de sélection des événements est un algorithme séquentiel; le traitement en parallèle des RoIs n'est plus un principe retenu. La séparation des traitements entre processeurs locaux et un processeur global est abandonnée au profit du traitement de chaque événement dans un seul processeur. Le mode de contrôle du flux de données suit le modèle "données tirées". Utiliser un réseau dédié pour la distribution des informations du superviseur vers les ROBs est une option rejetée; le réseau principal est utilisé à cet effet. La structure en sous-réseaux multiples de certains modèles est une option remplacée par un réseau unique permettant la communication de n'importe quel ROB, ou groupe de ROBs, avec n'importe quel processeur de traitement. La possibilité d'utiliser ce réseau commun pour transporter également

1. L'ensemble des informations sur ce projet sont disponibles sur le site: http://atlasinfo.cern.ch/Atlas/GROUPS/DAQTRIG/L2PILOT/L2pilot.html

le trafic de l'*event builder* n'est pas étudiée dans ce projet limité au trigger de niveau 2. La possibilité d'utiliser les mêmes processeurs pour le trigger de niveau 2 et le filtre d'événements n'est pas non plus étudiée. Ces deux derniers points seront discutés lors de l'intégration du trigger de niveau 2 et du filtre d'événements qui sera une étape postérieure à l'accomplissement de ce projet pilote et du projet "DAQ -1".

La structure des bancs de test est identique à celle du démonstrateur de l'architecture de trigger séquentiel rapportée dans cette thèse. Trois options de technologies de réseau ont été retenues: ATM, Fast/Gigabit Ethernet et SCI. Le réseau est un élément interchangeable dans les bancs de test ce qui devrait permettre des comparaisons. L'utilisation et le développement du logiciel utilisé sur le démonstrateur rapporté dans cette thèse pour l'ensemble des bancs de test du projet pilote n'a pas été la voie retenue par la collaboration. Un autre logiciel a été développé à cet effet: le "*Reference Software*" basé sur des méthodes modernes d'ingénierie du logiciel (conception orientée objet). Ma contribution personnelle à ce projet est pour le moment limitée au portage du *Reference Software* pour le fonctionnement avec un réseau en technologie ATM, et à l'exploitation de ce logiciel sur le banc de test ATM. La description de cette activité dépasse le cadre de cette thèse[1].

10.3. Modélisation du T/DAQ d'ATLAS

10.3.1. Méthode

L'activité de modélisation a pour but de comprendre la dimension du problème et d'évaluer les besoins: nombre de ROBs, groupements possibles, volumes de données, durée des traitements, bande passante des liens... Le "modèle papier" permet d'évaluer les caractéristiques du système en se basant sur les valeurs moyennes de ces divers paramètres. Le tableur Excel est l'outil utilisé pour ce type d'investigation. Un modèle informatique est également proposé pour évaluer les caractéristiques dynamiques du système. Il s'agit d'un programme de simulation par événements discrets permettant d'évaluer la taille des files d'attente en divers points du système, de révéler d'éventuels goulets d'étranglement, etc. Deux versions du modèle informatique sont développées avec des outils différents: Ptolemy[2], et un environnement nommé SIMDAQ [Hun95] basé sur le langage C++ (et non plus le langage MODSIM comme dans la première version de SIMDAQ). L'ensemble de l'information sur l'activité de modélisation est disponible sur Internet[3], je ne rapporte ici qu'une partie des hypothèses et certains résultats des modèles papier.

10.3.2. Hypothèses principales d'un des modèles papier

10.3.2.1. Organisation des *Read-Out-Buffers*

La lecture des canaux des détecteurs d'ATLAS se fait par les *Read-Out Drivers* connectés par des liaisons point-à-point aux *Read-Out-Buffers*. L'organisation des RODs et ROBs par détecteur est décrite dans [Boc97a]. Ce schéma n'est à ce jour pas complètement figé et le modèle reproduit dans la Table X-1 est celui de [Ama99]. Le nombre total de ROBs dans le modèle [Ver99] est 1530 ce qui n'est que légèrement différent du chiffre précédent (1444). Le nombre de sources est plus éloigné

1. Les résultats obtenus à ce jour sont inclus dans l'Annexe 3 de cette thèse.
2. Ptolemy est un environnement logiciel développé à l'Université de Californie, Berkeley permettant (entre autre) la simulation par événements discrets. Pour référence, consulter le site: http://ptolemy.eecs.berkely.edu
3. http://www.nikhef.nl/pub/experiments/atlas/daq/modelling.html

entre ces deux modèles: 921 dans [Ver99] et 588 à 812 dans l'autre modèle. La principale différence provient du facteur de regroupement des ROBs du TRT.

Détecteur	Muon Trigger	Muon Précision	Calorimètre Hadronique	Calorimètre Electro-M.	TRT	Tracker Silicium	Détecteur Pixel	Total
# de ROBs	32	192	112	712	256	92	48	1444
données[a] (ko)	0,38	0,85	1,8	1,8	(0,6)[b], 0,8	1,6	1	-
ROBs par source	4	8	4	2	(1) 8	1	1	-
# de sources	8	24	28	356	(256) 32	92	48	(812) 588

Table X-1: Nombre de ROBs et Sources.

a. volume de données brutes moyen par événement dans chaque ROB.
b. les chiffres entre parenthèses correspondent au fonctionnement à basse luminosité.

10.3.2.2. Menus du trigger

Les menus du trigger donnent la liste des différentes signatures des processus physiques que l'on souhaite observer et leur fréquence d'apparition, estimée par les simulations Monte-Carlo de la physique et des détecteurs de l'expérience. La Table X-2 présente les 5 signatures les plus fréquentes que l'on devrait observer dans le flux d'événements en sortie du trigger de niveau 1, pour le fonctionnement à basse et à haute luminosité (adapté de [Geo99]).

Basse Luminosité		Haute Luminosité	
Signature	Fréquence (kHz)	Signature	Fréquence (kHz)
MU6[a]	23	EM30 I	24,3
EM20 I[b]	11,5	2 * EM20 I	4,9
2 * EM15 I	1,6	2 * MU6	4
TAU20[c] + XE30[d]	1,3	MU6	3,9
2 * MU6	1	TAU20 + XE30	0,9
Autres (50 signatures)	1,7	Autres (37 signatures)	1,4
Total	40	Total	39,4

Table X-2: Evénements en sortie du trigger de niveau 1.

a. désigne 1 candidat muon d'énergie supérieure ou égale à 6 GeV.
b. désigne 1 candidat électron/photon d'énergie supérieure ou égale à 20 GeV, isolé.
c. désigne 1 candidat tauon d'énergie supérieure ou égale à 20 GeV.
d. désigne une énergie manquante supérieure ou égale à 30 GeV.

10.3.2.3. Séquences de traitement

Les traitements varient selon les candidats particules identifiés et la stratégie de sélection des événements. Le facteur de rejet à chaque étape est estimé par les études sur les algorithmes de traitement des données des détecteurs. Les hypothèses sur la séquence de traitement à effectuer pour le trigger de niveau 2 et les facteurs de rejets attendus sont décrits dans [Ver99]. Une séquence

simplifiée pour le fonctionnement à haute luminosité est reproduite dans la Table X-3.

Etape	#1		#2		#3	
Candidat(s)	Détecteur utilisé	Facteur de rejet[a]	Détecteur utilisé	Facteur de rejet[a]	Détecteur utilisé	Facteur de rejet[a]
Muon(s)	muon	25%	tracker	47%	calorimètre	90%
Electron(s) / photon(s)	calorimètre EM	-	calorimètre HAC	86%	tracker	86%
Jet(s)	calorimètre	50%	-	-	-	-
Hadron(s)	calorimètre	80%	tracker	60%	-	-

Table X-3: Séquence de traitement suivant les types de candidats.

a. fraction par rapport au nombre initial pour la première étape et par rapport à l'étape précédente dans les autres cas.

10.3.2.4. Quelques résultats

Les résultats présentés dans la Table X-4 sont extraits de [Ama99]. La fréquence des événements en sortie du trigger de niveau 1 est de 75 kHz. La reconstruction finale d'événements est effectuée à 2 kHz. Les liens au réseau ont une bande passante de 15 Mo/s. Les temps de traitement des données sont 2 fois inférieurs au temps mesuré sur des machines actuelles.

	Utilisation moyenne des liens des sources (Mo/s)							Total sources (Go/s)	Destinations	
	mu trig.	mu prec.	Cal. Had	Cal. Em	TRT[a]	SCT	Pixel		# proc[b]	Lien[c] (Mo/s)
Basse luminosité + *TRT Scan*	2	13	5,5	9,3	3,1	4,3	6,3	5,3	663	8
Basse luminosité sans *TRT Scan*	2	13	5,5	9,3	3,2	4,3	6,3	4,6	178	26
Haute luminosité	1	10	6	10	8,1	2,7	1,1	4,5	41	110

Table X-4: Quelques résultats de la modélisation.

a. 1 ROB par source pour le cas Basse luminosité + *TRT Scan*; 8 ROBs par source dans les autres cas.
b. nombre théorique minimum de processeurs (occupés à 100%) pour effectuer l'ensemble du processus de sélection du trigger de niveau 2.
c. bande passante théorique de la liaison de chaque processeur au réseau (rapport du flux total des sources sur le nombre minimum de processeurs).

On remarque que le *TRT Full Scan* a une grande influence sur le dimensionnement du système. Le regroupement de 8 ROBs par source du TRT n'est plus possible si l'on considère des liens à 15 Mo/s au réseau, cela conduit à passer de 32 sources à 256 pour le TRT. De plus, le nombre de processeurs nécessaire est multiplié par ~3,7 dans le fonctionnement à basse luminosité avec *TRT Full Scan* par rapport à celui sans *TRT Full Scan*. Des études sont en cours pour réduire cette distorsion (réduction du temps d'exécution de cet algorithme, autres hypothèses sur le volume de données, taux d'événements concernés...).

En considérant des liens ATM OC-3, le réseau à utiliser doit comporter de 950 à 1500 ports selon les modes de fonctionnement. Le modèle de [Ver99] donne des résultats du même ordre de grandeur concernant le nombre de ports du réseau et le nombre de processeurs de traitement.

10.4. Proposition pour le réseau du T/DAQ d'ATLAS

Compte tenu des diverses hypothèses plus ou moins certaines et des valeurs de nombreux paramètres qui demeurent à ce jour très fluctuantes, le dimensionnement précis du réseau n'est pas possible. Les études de modélisation se poursuivront jusqu'à la phase de conception finale du système qui devrait intervenir en 2002 environ. Dans l'immédiat, seul l'ordre de grandeur des besoins est connu et l'évaluation de la faisabilité du réseau doit couvrir une certaine gamme de valeurs. D'autre part, la flexibilité et l'extensibilité du système commandent également de proposer une solution de réseau relativement souple. Le scénario envisagé ici est la réalisation d'un réseau comportant l'équivalent de 800 à 1600 ports ATM OC-3 (i.e. 125-250 Gbit/s) ce qui permet de satisfaire les besoins extrêmes établis par les études de modélisation citées.

La solution la plus simple consiste à utiliser un commutateur unique de capacité suffisante. A ce jour, seuls les produits destinés à équiper les réseaux de télécommunications offrent les caractéristiques requises. La Table X-5 répertorie certains de ces produits.

Fournisseur	Produit	Capacité
Alcatel	1000 BBS	20 - 160 Gbit/s, >1 Tbit/s annoncé
Ericsson	AXD 301	10 - 160 Gbit/s, 2,5 Tbit/s prévu
Fujitsu	FETEX-150	2,5 - 160 Gbit/s
NEC	ATOMNET M20	jusqu'à 160 Gbit/s
Nortel Networks	Passport 15000	40 - 160 Gbit/s
Siemens / Newbridge	MainStreetXpress 36190	20 - 160 Gbit/s, >1 Tbit/s annoncé

Table X-5: Commutateurs ATM pour les réseaux publics.

Techniquement et en terme de performance, ces matériels semblent adéquats, mais du point de vue économique leur utilisation ne me paraît pas réaliste. Il est particulièrement difficile d'obtenir des informations sur ces produits car notre application ne correspond pas au marché visé par les constructeurs et parce que les perspectives commerciales de notre projet sont très maigres par rapport aux investissements habituels des clients de ces sociétés. Le prix par port OC-3 annoncé par Nortel Networks pour le Passport 15000 est de ~6000 US$. Il est difficile de prévoir l'évolution du prix de ce type de matériel, mais en se basant sur une baisse de 10% par an, on peut estimer le coût d'un commutateur à ~4300 US$ par port en 2003. Un commutateur à 1000 ports OC-3 serait donc facturé ~4,3 MUS$. Je pense que cette solution sera très difficile à justifier du point de vue économique auprès de la collaboration ATLAS par rapport au coût d'une infrastructure basée sur des produits destinés aux réseaux d'entreprises.

Les commutateurs destinés aux réseaux d'entreprises disponibles sur le marché offrent une capacité maximale de ~40-80 Gbit/s et un nombre maximal de ~210-224 ports ATM OC-3[1]. Des

1. SmartCore 9500 de Cabletron, Cajun M770 de Lucent, modèle 8390 de IBM, ASX 4000 de FORE,...

produits de plus forte capacité et comportant un nombre plus élevé de ports pourraient apparaître dans les prochaines années, mais cela est davantage guidé par le marché que par des considérations techniques. Je prends pour hypothèse que des produits comportant ~1000 ports ne seront pas disponibles pour notre application et qu'il sera nécessaire d'interconnecter plusieurs commutateurs. La Table X-6 donne différentes configurations de réseau selon le nombre et la capacité des commutateurs inter-connectés suivant une topologie en étoile (cf. Figure 6-9b).

Commutateur de base	# de commutateurs	# liens entre 2 commutateurs	# ports total	# ports disponibles	Coût[a] (MUS$)
224 ports	4	21	896	644	0,8
	6	13	1344	954	1,3
	8	10	1792	1232	1,7
	10	8	2240	1520	2,1
384 ports	3	32	1152	960	1,1
	4	35	1536	1116	1,4
	5	27	1920	1380	1,8
512 ports	2	102	1024	820	1
	3	64	1536	1152	1,4
	4	46	2048	1496	1,9
768 ports	2	192	1536	1152	1.4

Table X-6: Assemblages de commutateurs.

a. estimé en 2003 en considérant une baisse annuelle de 15% par rapport au prix actuel de ~1850 US$ par port OC-3 (soit 950 US$ par port).

Un réseau comportant ~1000 ports pourrait être réalisé par l'interconnexion de 6 commutateurs 224 ports pour un coût de ~1,3 MUS$. Il me paraît indéniable qu'un arrangement équivalent réalisé avec des éléments Fast/Gigabit Ethernet sera nettement moins cher compte tenu des prix actuels et de leur évolution prévue (facteur 2 à 4 par rapport à ATM). Néanmoins, la solution basée sur ATM demeure largement compatible avec les moyens prévus par la collaboration qui se montent à ~3,2 MUS$[1]. Je pense donc qu'un réseau ATM pour le T/DAQ d'ATLAS est une solution techniquement réalisable et d'un coût abordable.

Les résultats des évaluations sur Fast/Gigabit Ethernet faites par nos collaborateurs sont attendues pour explorer la faisabilité technique de cette option (je pense que cette technologie devrait pouvoir convenir, mais la preuve n'a pas encore été apportée). L'évolution du marché des réseaux

1. Cette estimation est basée sur les chiffres mentionnés dans [ATL98]. Dans ce document, l'évaluation du coût du système de T/DAQ est basée sur le modèle du *Technical Proposal* d'ATLAS (le trigger de niveau 2 et de niveau 3 sont physiquement séparés). Le coût prévu pour l'*event builder* est de 3,7 MSFr. Je considère que 90% de cette somme est affectée à du matériel de réseau soit 3,3 MSFr. Le coût prévu pour le trigger de niveau 2 (à l'exclusion du superviseur et *RoI Builder*) est de 9,3 MSFr. Je considère que 20% de ce budget est affecté à l'achat de matériel de réseau soit 1,9 MSFr. Le coût possible de l'ensemble du matériel de réseau pour le trigger de niveau 2 et de niveau 3 est donc de 5,2 MSFr soit 3,2 MUS$ (au taux actuel de 4,1 FF pour 1 SFr, et 6,5 FF pour 1 US$ ou 1 Euro).

dans les prochaines années devrait permettre d'effectuer le meilleur compromis technique et économique entre ces deux solutions au moment opportun (2001-2002).

La Figure 10-1 montre un arrangement possible pour le réseau principal du T/DAQ d'ATLAS dans la version de l'architecture utilisant des processeurs distincts pour le trigger de niveau 2 et le filtre d'événements.

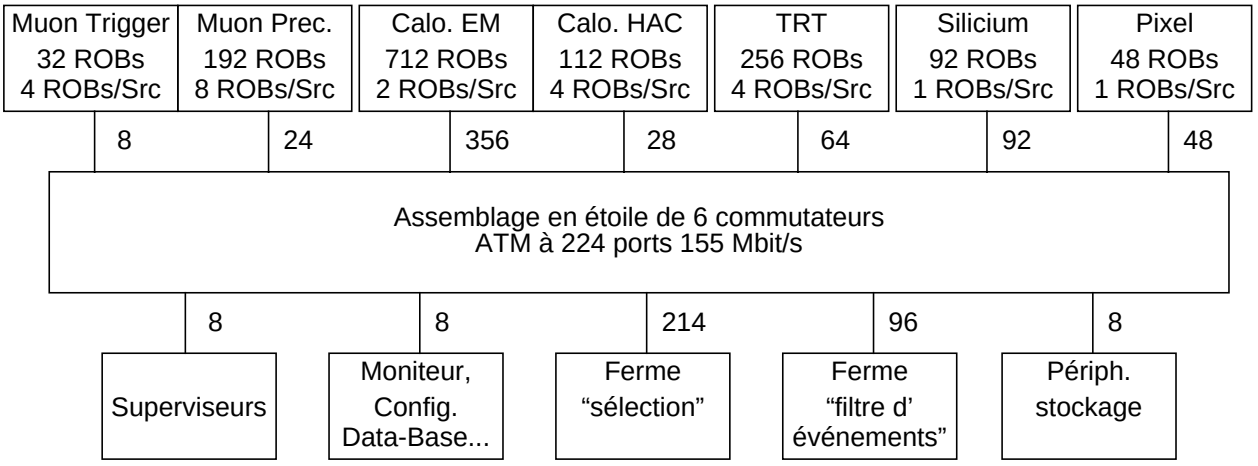


Figure 10-1: Configuration de départ proposée pour le réseau du T/DAQ d'ATLAS.

Je pense que la mise en place du système de T/DAQ dans ATLAS se fera par étapes et qu'au démarrage de l'expérience, seule une configuration réduite sera installée (réduction du nombre de processeurs et des flux de données). Cette étape permettra de véritablement évaluer les besoins, et d'effectuer une montée en puissance optimale. Je propose donc une configuration initiale de performance réduite pouvant être mise à niveau.

Les hypothèses sur le regroupement des ROBs sont celles de [Ama99] sauf pour le TRT où le regroupement est effectué par 4 et non par 8 afin d'offrir un supplément de bande passante pour ce détecteur. Le réseau réalisé comporte 954 ports utilisables, dont 620 sont affectés aux sources. Les superviseurs peuvent être au nombre de 8 ainsi que les moniteurs, liaisons à la base de données de configuration, etc... Dans ce modèle comportant des fermes de processeurs séparées pour la sélection et le filtrage des événements, 96 ports sont affectés à l'*event builder*. La bande passante théorique globale de ces liens est de ~1,5 Go/s (elle est en pratique garantie par l'utilisation de connexions de type CBR). Ces ressources permettent d'effectuer la reconstruction d'événements de 1 Mo à 1 kHz (soit 1 Go/s) avec 50% de bande passante en réserve. En supposant que chaque lien affecté au filtre d'événements est connecté à une machine à 4 processeurs, on dispose à ce niveau de 384 processeurs. Les périphériques de stockage sont connectés par 8 liens offrant une bande passante de 128 Mo/s, ce qui est suffisant pour véhiculer des événements de 1 Mo/s à ~50 Hz. Les 214 ports restant (~3,4 Go/s de bande passante théorique) sont affectés aux processeurs de la ferme "sélection". Selon le type de machines utilisées (1 à 4 processeurs), on peut disposer de 214 à 856 processeurs ce qui est suffisant (cf. Table X-4). En revanche, la bande passante disponible est insuffisante pour un fonctionnement à 75 kHz (limite de saturation théorique à ~60 kHz). On peut changer l'affectation des ressources entre les deux fermes de processeurs pour trouver la répartition optimale en fonction des besoins. Je pense que la simulation du système pour étudier son comportement dynamique est intéressante. La topologie de réseau en étoile est adaptée au transport du trafic correspondant à la reconstruction complète d'événements, mais pour le profil de trafic spécifique au trigger de niveau 2 d'ATLAS, seule la simulation peut permettre de déterminer les limites. Cet important effort dans l'amélioration de la définition des besoins et la mise au point des

modèles de simulation est un travail pris en charge par le groupe concerné et dépasse le cadre de cette thèse.

Pour accroître les performances du système, on peut ajouter un ou plusieurs commutateurs ainsi que des processeurs. Cette opération nécessite le re-câblage de la structure en étoile et la re-configuration du système. En ajoutant un commutateur à 224 ports, le réseau en étoile à 7 branches obtenu comporte 152 ports utilisables de plus que la version à 6 branches. Avec une autre branche supplémentaire, on ajoute 126 ports. En principe, cette structure peut encore croître (la limite théorique est une étoile à P/2 branches composées de commutateurs à P ports) mais est en pratique, la première limitation incontournable avec la technologie ATM devrait venir du nombre de connexions virtuelles à établir. Le champ VCI des cellules ATM autorise 64 K connexions, mais la plupart des cartes réseau ne gèrent que 4-16 K connexions simultanées. Le nombre de connexions à établir est le plus élevé au niveau des nœuds sources. Une source établit avec chaque nœud destination: 1 connexion point-à-point VBR, 1 CBR, plusieurs *multi-cast* (2 au minimum par exemple) soit un total de 4 connexions virtuelles. Le nombre de connexions avec les autres éléments est faible et l'on peut situer la limite sur le nombre de destinations est ~1000 avec des cartes réseau courantes. Une autre limitation peut venir du nombre maximal de connexions gérées par les commutateurs (à vérifier selon les produits).

Avec la technologie Fast/Gigabit Ethernet, une limitation du même ordre peut être observée (nombre maximal d'adresses MAC des commutateurs: 4-64 K selon les modèles). Cela limiterait à 4000-64000 le nombre d'éléments connectés au réseau (ce qui est largement suffisant dans notre cas). Un autre problème pourrait se situer au niveau de l'équilibrage de la charge de chaque lien lors de l'agrégation de plusieurs liens pour relier des commutateurs. Ce problème est facilement résolu en ATM lorsque l'utilisateur décide du chemin physique à établir pour chaque connexion virtuelle (utilisation de PVCs). Avec Ethernet, cette question est à étudier (une norme sur l'agrégation de liens Gigabit Ethernet est en cours d'établissement).

10.5. Perspectives

A ce jour, seules les liaisons ATM 155 Mbit/s sont répandues et relativement bon marché. Des liaisons 622 Mbit/s (ainsi que 2,4 Gbit/s) existent mais sont d'un coût très élevé par rapport à l'Ethernet 1 Gbit/s. Dans la limite actuelle des produits disponibles en ATM, il est possible de réaliser à l'aide d'un seul commutateur, un système de reconstruction d'événements comportant une centaine de sources/destinations offrant ~1,5 Go/s de débit global (suffisant pour le T/DAQ de nombreuses expériences telles Babar, CDF, L3, Phenix...). Des configurations plus importantes sont réalisables par inter-connexion de commutateurs, jusqu'à ~1000 paires sources et destinations soit 15 Go/s de débit global. Des configurations offrant un débit de ~60 Go/s (besoins de l'expérience CMS) me semblent très ardues du point de vue technique aussi bien avec ATM que Gigabit Ethernet, et irréalistes aujourd'hui du point de vue du coût.

10.6. Résumé

Dans ce chapitre, j'ai présenté le projet pilote du trigger de niveau 2 d'ATLAS et j'ai donné certains résultats relatifs à la modélisation. J'ai proposé une réalisation pour le réseau du T/DAQ de l'expérience ATLAS et j'ai montré comment la faire évoluer. J'ai donné les limites de performance des systèmes de reconstruction d'événements qui me semblent réalisables.

CONCLUSIONS

Cette thèse a présenté le projet de grand collisionneur de hadrons LHC et les futures expériences de physique des hautes énergies associées. J'ai exposé le principe de sélection d'événements en ligne pour l'acquisition de données de ces expériences. L'importance que prennent les réseaux informatique à haut débit dans ces systèmes a été montrée. L'architecture des systèmes de T/DAQ initialement proposée pour les expériences ATLAS et CMS a été décrite.

Les fonctionnalités du réseau de communication pour le T/DAQ d'ATLAS ont été établies et j'ai décrit différents mécanismes de contrôle des flux de données dans ce type de système. Une analyse critique des propositions initiales d'architecture des systèmes de T/DAQ d'ATLAS et de CMS a été présentée pour justifier le modèle défendu dans cette thèse. Cette architecture est basée sur un réseau de communication unique reliant la plupart des éléments du système. Le traitement de chaque événement se fait dans un processeur exécutant un algorithme de sélection séquentiel. L'objectif principal est de minimaliser les transferts de données ainsi que la puissance de calcul nécessaire pour absorber le flux d'événements. Un autre objectif est d'avoir un système homogène et flexible.

J'ai présenté une revue des différentes technologies de réseau utilisées dans les expériences de physique des hautes énergies ainsi que celles envisagées pour ATLAS. Les deux candidats à retenir sont, à mon avis, ATM et Fast/Gigabit Ethernet.

Plusieurs types de réseau adaptés au problème de reconstruction d'événements ont été décrits. Leurs performances ont été étudiées du point de vue théorique et par simulation. J'ai montré la limite théorique d'une topologie en anneau. L'utilisation d'un mécanisme d'allocation de bande passante permet d'éliminer les problèmes éventuels de congestion dans les commutateurs d'un système de reconstruction d'événements. J'ai étudié diverses topologies visant à assembler de manière optimale des matrices de commutation afin de créer un réseau de plus grande dimension. La structure en étoile est celle conduisant à la réalisation la plus économique.

Le besoin de performance justifie l'utilisation d'un protocole de communication non standard pour l'application envisagée. La manière d'optimiser un logiciel pilote de carte réseau pour effectuer des communications efficaces a été décrite: contrôle de la carte réseau au niveau de l'application et élimination des opérations de recopie de données. J'ai implémenté ces principes sur des cartes réseau ATM. Les mesures de performance correspondantes ont été rapportées.

Différents démonstrateurs du T/DAQ d'ATLAS ont été décrits. J'ai détaillé celui de l'architecture défendue dans cette thèse. Le fonctionnement de chacun des éléments du système a été présenté ainsi que les mesures de performance qui ont été faites. Le démonstrateur de plus grande taille assemblé à ce jour comporte 48 machines connectées par un commutateur ATM. Cela correspond environ à 1:20 du trigger de niveau 2 final. Lors des tests en mode "système de reconstruction d'événements", la performance obtenue est de l'ordre de 1:3 de celle requise pour le système final, ce qui, compte tenu de la taille réduite du démonstrateur, est une bonne indication de faisabilité.

Quelques résultats sur la modélisation du T/DAQ d'ATLAS ont été présentés et j'ai fait une

proposition pour la configuration initiale du réseau de ce système. Une solution possible basée sur un commutateur ATM unique à 1000 ports 155 Mbit/s a été provisoirement écartée car ce matériel, bien que disponible dès aujourd'hui chez une demi-douzaine de fournisseurs de matériel pour les télécommunications, sera probablement d'un coût prohibitif. Une interconnexion en étoile de ~6 commutateurs ATM à 224 ports (également disponibles aujourd'hui) me semble adaptée à une configuration initiale. Le coût de cette solution est estimé à ~1,3 MUS$, soit moins de la moitié du budget initialement prévu par la collaboration pour ces équipements. Le moyen de faire évoluer cette configuration pour en accroître les performances a été décrit.

Considérant une solution alternative à la technologie ATM pour ATLAS, je pense que les candidats doivent offrir:

- des cartes réseau ayant un logiciel pilote optimisé pour des communications rapides et efficaces,
- la possibilité d'allocation de la bande passante ou un mécanisme de contrôle de flux efficace pour éviter les engorgements du réseau,
- la gestion des priorités pour éviter l'influence mutuelle des différents types de trafic,
- la gestion des communications point-à-point et point-à-multi-points,
- des commutateurs de forte capacité et pouvant comporter un grand nombre de ports,
- la possibilité de cascader les commutateurs et éventuellement d'effectuer l'agrégation de liens,
- des produits de grande qualité et fiabilité compte tenu de la taille du système,
- un coût compétitif,
- des garanties sur la pérennité des produits, de multiples sources d'approvisionnement, etc.

Dans cette thèse, j'ai exposé les résultats de mes recherches sur l'architecture du système de T/DAQ pour l'expérience ATLAS et plus particulièrement sur les aspects de réseau. J'ai essayé de contribuer de manière objective à l'évaluation de la technologie ATM qui me semble être une solution satisfaisante pour l'application envisagée aussi bien du point de vue technique que du point de vue économique. Je pense que nous n'avons pas rencontré à ce jour d'obstacle insurmontable dans l'utilisation de l'ATM pour le réseau du T/DAQ d'ATLAS. L'évaluation d'une alternative basée sur la technologie Fast/Gigabit Ethernet est étudiée par nos collaborateurs. Les choix définitifs seront effectués par la collaboration ATLAS en 2001-2002.

RÉFÉRENCES

[Abo00] M. Abolins et al., “ATLAS Level 2 Trigger User Requirements Document”, ATLAS COM DAQ Note 2000-001.

[Ahm94] H. Ahmadi et W.E. Denzel, “A Survey of Modern High Performance Switching“, IEEE Journal on Selected Areas in Communication, vol. 7 No 7, Septembre 1994, pp. 1091-1103.

[ALI95] Collaboration ALICE, “Technical Proposal for A Large Ion Collider Experiment at the CERN LHC”, CERN/LHCC/95-71, Décembre 1995 [1].

[Ama99] A. Amadon, “ATLAS LVL2 Trigger Modelling”, communication interne Mars 1999 [2].

[Amb97] G. Ambrosini et al., “The ATLAS DAQ and Event Filter Prototype “-1” Project”, actes de la conférence Computing in High Energy Physics (CHEP’ 97), Berlin, Allemagne, 7-11 Avril 1997 [3].

[Amb97a] G. Ambrosini et al., “Event Building in the ATLAS DAQ Prototype”, actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 185-190.

[Amb98] G. Ambrosini et al., “Event Building in the ATLAS DAQ Prototype “-1””, actes de la conférence Computing in High Energy Physics (CHEP’ 98), Chicago Illinois, USA, 31 Août - 4 Septembre 1998 [4].

[And91] J. Andresen et al., “Scalable Parallel Open Architecture (SPOA) Data Acquisition System Design Report”, Fermi National Accelerator Laboratory Internal Report, Chicago, Illinois, USA, August 1991.

[Anv97] S. Anvar et al., “The Charged Trigger System of the CERN -SPS NA48 Experiment”, actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 37-41.

[Arc86] J. Archibald et J.L. Baer, “Cache Coherence Protocols: Evaluation using a Multiprocessor Simulation Model”, ACM Transactions on Computer Systems vol. 4 No 4, Novembre 1986, pp. 273-298.

[ATL94] Collaboration ATLAS, “Technical Proposal for a General Purpose pp Experiment at the Large Hadron Collider at CERN”, CERN/LHCC/94-43 LHCC/P2, 15 Décembre 1994 [5].

[ATL98] Collaboration ATLAS, “ATLAS DAQ, EF, LVL2 and DCS Technical Progress Report”, CERN/LHCC 98-16, 30 Juin 1998 [5].

[ATL98a] Collaboration ATLAS, Groupes Trigger LVL2, “ Level-2 Pilot Project”, ATLAS Internal Note DAQ-98-118, 8 Octobre 1998 [5].

[ATL98b] ATLAS Level-1 Trigger Group, “Level-1 Trigger Technical Design Report”, ATLAS TDR-12, 28 Juin 1998[5].

1. disponible sur le site http://www.cern.ch/ALICE/documents.html
2. disponible sur le site http://www.nikhef.nl/pub/experiments/atlas/daq/modelling.html
3. disponibles sur le site http://www.ifh.de/CHEP97/chep97.htm
4. disponibles sur le site http://www.hepnet/chep98/
5. disponible sur le site http://atlasinfo.cern.ch/Atlas

[ATM95] ATM Forum Technical Committee, “ATM User-Network Interface (UNI) Signalling Specification Version 4.0”, document af-sig-0061.000, Juillet 1996 [1].

[Bar64] P. Baran et al., “On Distributed Communications”, série de 12 rapports RAND Corp., USA, 1964 [2].

[Bar90] E. Barsotti et al., “A Proposed Scalable Parallel Open Architecture Data Acquisition System for Low to High Experiment Rates, Test Beam and All SSC Detectors”, IEEE Transactions on Nuclear Science, vol. 37, No. 3, Juin 1990, pp.1216-1221.

[Bar96] G. Barazza et al., “Fibre Channel Based Network for the Euroball Experiment”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 151-157.

[Bat68] K.E. Batcher, “Sorting Networks and their Applications”, Proceedings of the Spring Joint Computer Conference, AFIPS, 1968, pp. 307-314 [3].

[Bij97] H.C. van der Bij et al., “S-Link, a Data Link Interface Specification for the LHC era”, actes de la Xème conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 199-203.

[Bla96] R. Blair et al., “The ATLAS Level-2 Trigger Supervisor”, actes du Second Workshop on Electronics for LHC experiments, Balatonfured, Hongrie, 23-27 Septembre 1996, pp. 191-193.

[Bla97] J.J. Blaising et al., “Performance of the L3 second level Trigger implemented for the LEP II with the SGS Thomson C104 packet Switch”, actes de la Xème conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 45-50.

[Bla97a] G.C. Blazey, “The D0 Run II Trigger”, actes de la Xème conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 83-87.

[Boc97] R.K. Bock et al., “A Hybrid Approach for the ATLAS Level-2 Trigger”, ATLAS Internal Note DAQ-97-78, 23 Octobre 1997 [4].

[Boc97a] R.K. Bock et P. Le Dû, “Detector and Readout Specifications and Buffer RoI Relations for the Level-2 Trigger Demonstrator Program”, ATLAS Internal Note DAQ-97-62, 27 Janvier 1997 [4].

[Bod95] N. Boden et al., “Myrinet: A Gigabit-per-second Local Area Network”, IEEE Micro, Février 1995, pp. 29-36.

[Bog96] A. Bogaerts et al., “Application of the Scalable Coherent Interface to Data Acquisition at LHC”, RD-24 Status Report, CERN / LHCC 96-33, 2 Octobre 1996 [5].

[Bog97] A. Bogaerts et al., “Studies of SCI for the ATLAS Level-2 Trigger System”, actes de la Xème conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 141-145.

[Boy94] O. Boyle et al., “A Data Merger for the NA48 Experiment at CERN”, actes du IEEE Nuclear Symposium & Medical Imaging Conference, Norfolk, Virginie, USA, Octobre 1994, vol. 2 pp. 991-993.

[Bys97] J. Bystricky et al., “A Sequential Processing Strategy for the ATLAS event Selection”, Actes du IEEE Nuclear Science Symposium, Annaheim Californie, 3-5 Novembre 1996, également publié dans IEEE

1. disponible sur le site http://atmforum.com
2. disponibles sur le site http://www.rand.org/publications/RM/baran.list.html
3. disponible sur le site http://intrepid.mcs.kent.edu/~batcher/conf.html
4. disponible sur le site http://atlasinfo.cern.ch/Atlas/GROUPS/notes.html
5. disponible sur le site http://www.cern.ch/RD24/

Transactions on Nuclear Science, vol. 44 No 3, Juin 1997, pp. 342-347.

[Cal95] D. Calvet et al., “A Study of Performance Issues of the ATLAS Event Selection System Based on an ATM Switching Network”, actes de la IX^{ème} conférence internationale IEEE Real Time Systems, East Lansing Michigan, USA, 24-28 Mai 1995, pp. 337-346; également publié dans IEEE Transactions on Nuclear Science, vol. 43 No. 1 Février 1996, pp 90-98 [1].

[Cal97] D. Calvet et F. Servaz, “Development of a PCI Mezzanine Card ATM Interface”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 218-224.

[Cal97a] D. Calvet et al., “Performance Analysis of ATM Network Interfaces for Data Acquisition Applications”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 73-80 [1].

[Cal97b] D. Calvet et al., “Evaluation of a Congestion Avoidance Scheme and Implementation on ATM Network based Event Builders”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 96-107 [1].

[Cal97c] D. Calvet et al., “Operation and Performance of an ATM based Demonstrator for the Sequential Option of the ATLAS Trigger”, actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, 22-26 Septembre 1997, pp. 101-105; également publié dans IEEE Transactions on Nuclear Science, vol 45, pp 1793-1794, Août 1998 [1].

[Cal98] D. Calvet et al., “Network Application Programming Interface for Low Level Communication”, note interne ATLAS Level-2 Testbed Software No 6, Avril 1998 [1].

[Cal98a] D. Calvet et al., “Emulation of an Event Builder using the ATM Local Area Network of the RCNP Institute”, actes de IEEE Conference on Computing in High Energy Physics (CHEP’ 98), Chicago, Illinois, USA, 31 Août - 6 Septembre 1998 [1].

[Cal98b] D. Calvet et al., “Emulation of the Sequential Option of the ATLAS Trigger and Event Builder using the ATM Local Area Network of the RCNP Institute”, ATLAS Internal Note DAQ-130, 31 Août 1998 [1].

[Cal98c] D. Calvet et al., “Testbed Software Structure”, note interne ATLAS Level-2 Testbed Software No 1, Avril 1998 [1].

[Cal98d] D. Calvet et al., “Operating System Specific Modules”, note interne ATLAS Level-2 Testbed Software No 2, Avril 1998 [1].

[Cal98e] D. Calvet et al., “Buffer Manager”, note interne ATLAS Level-2 Testbed Software No 3, Avril 1998 [1].

[Cal98f] D. Calvet et I. Mandjavidze, “Buffer Descriptors and Low Level Data Manipulation”, note interne ATLAS Level-2 Testbed Software No 4, Avril 1998 [1].

[Cal98g] D. Calvet et I. Mandjavidze, “Communication over the Network”, note interne ATLAS Level-2 Testbed Software No 5, Avril 1998 [1].

[Cal99] D. Calvet et al., “The ATLAS High Level Trigger ATM Testbed”, actes de la XI^{ème} conférence internationale IEEE Real Time Systems, Santa Fe, USA, 14-18 Juin 1999 [1].

[Cam97] M. Campanella et al., “TCP-UDP/IP Performance at High Speed over ATM”, actes du 2nd International

1. disponible sur le site http://www-dapnia.cea.fr/Phys/Sei/externe/sommaire/sommaire_sei.htm

Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 67-72.

[Cla89] D. Clark et al.,"An analysis of TCP processing overhead", IEEE Communication Magazine, vol. 27 No. 6, Juin 1989, pp. 23-29.

[Cla94] P.E.L. Clarke et al., "Test of Components for an Asynchronous Level-2 Trigger for ATLAS", ATLAS Internal Note DAQ No 36, 16 Décembre 1994 [1].

[Clo53] C. Clos, "A Study of Non-Blocking Switching Networks", Bell System Technical Journal Vol. 32, Mars 1953, pp. 1227-1238.

[CMS94] Collaboration CMS, "CMS, The Compact Muon Solenoïd Technical Proposal", CERN LHCC 94-38 LHCC/P1 15 Décembre 1994 [2].

[Com91] D.E. Comer, "Internetworking with TCP/IP, Volume I; Principles, Protocols and Architecture", Prentice-Hall, New Jersey, USA, 1991.

[Cos95] M. Costa et al., "NEBULAS: a high performance data-driven event building architecture based on asynchronous self-routing packet switching network", RD-31 Status Report, CERN / LHCC / 95-14, 6 Mars 1995[3].

[Cot98] W.N. Cottingham et D.A. Greenwood, "An Introduction to the Standard Model of Particle Physics", Cambridge University Press, 1998.

[Cul97] D.E. Culler et al., "Parallel Computing on the Berkeley NOW", actes du 9th Joint Symposium on Parallel Processing, Kobe, Japon, 1997.

[Cut93] L.G. Cuthbert et J-C. Sapanel, "ATM: The broadband Telecommunications Solution", IEE Telecommunications series 29, Londres, Royaume-Uni, 1993.

[Dam97] M. Dam et al., "Higher Level Trigger Systems for the HERA-B experiment", actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, Septembre 1997, pp. 77-81.

[Dem98] Demonstrator "C" Group, "Overview of the Sequential Single Farm option for the ATLAS High Level Trigger", ATLAS DAQ Note 109, June 1998 [4].

[Div92] R.Divia et al., "Scalable Coherent Interface and LHC: a good marriage?", actes de la X^{ième} conférence Computing in High Energy Physics, Annecy, France, 21-25 Septembre 1992, pp. 685-688.

[Dob98] R.W. Dobinson et al., "Ethernet for the ATLAS Second Level Trigger", actes de la conférence Syscomms 98, CERN, Genève, 25-26 Mars 1998.

[Dub99] O. Dubuisson, "ASN.1 – Communication between heterogeneous systems", Springer Verlag, 1999.

[Dum97] G.W.A. Dummer, "Electronic Inventions and Discoveries", Institute of Physics Publishing Bristol and Philadelphia, 1997, pp. 88.

[Fas83] "FastBus: modular high speed data acquisition and control system for high energy physics and other applications", publié par le Department of Energy, USA, 1983.

1. disponible sur le site http://atlasinfo.cern.ch/Atlas/GROUPS/notes.html
2. disponible sur le site http://cmsinfo.cern.ch/Welcome.html
3. disponible sur le site http://pcvlsi5.cern.ch/MicDig/rd31/lhcc_95_1.html
4. disponible sur le site http://weblib.cern.ch

[Fib94] Fibre Channel Association, “Fibre Channel: Connection to the Future”, ISBN 1-878707-19-1, 1994.

[Fro98] J. Fromm et al. “ATM Based Event Building and PC based Level 3 Trigger at CDF”, actes de la conférence Computing in High Energy Physics [1], Chicago, Illinois, USA, 31 Août - 4 Septembre 1998.

[Gac96] O. Gachelin et al., “ROBIN: A Functional Demonstrator for the ATLAS Trigger/DAQ Read-Out Buffer”, actes du Second Workshop on Electronics for LHC Experiments, Balatonfüred, Hungary, 23-27 September, 1996, pp. 204-207.

[Geo99] S. George et al., “RoI based event description for modelling the ATLAS second level trigger”, ATLAS Communication Note 99-010, Avril 1999 [2].

[Gia97] A. Gianoli, “the NA48 Data Acquisition System”, actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, 22-26 Septembre 1997, pp. 265-268.

[Gun90] J.F Gunion et al., “The Higgs Hunter’s Guide”, Frontiers in Physics, Addison-Wesley, 1990.

[Gus90] D.B. Gustavson, “The Scalable Coherent Interface, IEEE P1596, Status and Possible Application to Data Acquisition and Physics”, IEEE Transacrions on Nuclear Science vol. 37 No 2, Avril 1990, pp. 365-368.

[Gus92] D.B. Gustavson, “The Scalable Coherent Interface and other Related Standard Projects”, IEEE Micro vol. 12 No 1, Février 1992, pp. 10-22.

[Haa97] S. Haas et al., “Results from the Macramé 1024 Node IEEE 1355 Switching Network”, actes de la conférence European Multimedia, Microprocessors and Electronics, Florence, 3-5 Novembre 1997.

[Hin99] C. Hinkelbein et al., “Pattern Recognition in the TRT for the ATLAS B-Physics Trigger”, ATLAS Internal Note DAQ-99-007, 14 Juillet 1999 [2].

[Hor97] C. Hortnagl, “Evaluation of Technologies of Parallel Computers’ Communication Networks for a Real-Time Triggering Application in a High-Energy Physics Experiment at CERN”, thèse de doctorat, Innsbruck, Autriche, Décembre 1997.

[Hug98] R.E. Hughes Jones et al., “Using SCI to implement the Local-Global Architecture for the ATLAS Level 2 Trigger”, ATLAS Internal Note DAQ-98-111, 15 Juin 1998 [2].

[Hun95] S. Hunt et al., “SIMDAQ, A System for Modelling DAQ/Trigger Systems”, IEEE Transactions on Nuclear Science, Vol. 43 No 1, Février 1996, pp 69-73.

[IDT97] IDT 77211 NicStar™ User's manual, version 1.0, Integrated Device Technology Inc., Santa Clara, Californie,USA, Février 1997 [3].

[IEE92] IEEE Standard 1596, “SCI: Scalable Coherent Interface”, IEEE, Inc. USA, 1992.

[IEE95] IEEE Standard 1355, “Standard for Heterogeneous InterConnect (HIC). Low Cost, Low Latency, Scalable Serial Interconnect for Parallel System Construction”, IEEE, Inc., USA, 1995.

[ITU92] International Telecommunication Union, “B-ISDN – Digital Suscriber Signalling System No. 2 (DSS 2) – User Network Interface (UNI) – Layer 3 specification for basic call/connection control”, Recommendation ITU Q.2931, Février 1995 [4].

1. disponible sur le site http://www.hep.net/chep98/PDF/sessions.html
2. disponible sur le site http://weblib.cern.ch
3. disponible sur le site http://www.idt.com
4. à acheter à l’ITU, renseignements sur le site http://www.itu.int/

[Kad98] J. Kadambi et al., “Gigabit Ethernet”, Prentice Hall Series in Computer Networking and Distributed Systems, Prentice Hall, 1998.

[Kug98] A. Kugel et al., “ATLAS Level-2 Trigger Demonstrator A, Activity Report, Part 1: Overview and Summary”, ATLAS Internal Note DAQ-98-085, 26 Mars 1998 [1].

[Kug98a] A. Kugel et al., “ATLAS Level-2 Trigger Demonstrator A, Activity Report, Part 2: Demonstrator Results”, ATLAS Internal Note DAQ-98-084, 26 Mars 1998 [1].

[Lef89] S. Leffler et al., “The Design and Implementation of the 4.3 BSD UNIX Operating System”, Addison Wesley, 1989.

[Let97] M. Letheren, “An overview of Switching Technologies for Event Building at the Large Hadron Collider Experiments”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 47-66.

[Mal98] P. Maley et al., “A Local-Global Implementation of a Vertical Slice of the ATLAS Second Level Trigger”, ATLAS Internal Note DAQ-98-081, 27 Janvier 1998 [1].

[Man93] I. Mandjavidze, “Modelling and Performance Evaluation for Event Builders based on ATM Switches”, Note Interne RD-31 93-06, Décembre 1993.

[Man94] I. Mandjavidze, “A New Traffic Shaping Scheme: the True Barrel Shifter”, Note Interne RD-31 94-03, Février 1994.

[McD94] D. E. McDysan et D.L. Spohn, “ATM, Theory and Applications”, McGraw-Hill Series on Computer Communications, USA, 1994.

[Mej00] F. Meijers et al., “The CMS Event Builder Demonstrator based on Myrinet”, Actes de la conférence Computing in High Energy Physics (CHEP) 2000, Padova, Italie, Février 2000.

[Mul97] P.A. Muller, “Modélisation objet avec UML”, Eyrolles, 1997.

[Nag97] Y. Nagasaka et al., “Performance of an Event Builder System with SBus G-Link Modules”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 146-150.

[Nar88] M.J. Narasimha, “The Batcher-Banyan Self Routing Network: Universality and Simplification”, IEEE Transactions on Communications, vol. 36 No. 10, Octobre 1988.

[Nat98] National Research Council, “Elementary Particle Physics”, National Academic Press, Washington D.C. 1998.

[Nom93] M. Nomachi, “Event Builder Queue Occupancy”, SDC Note 93-566, August 1993.

[Pav98] T.J. Pavel et al., “Network Performance Testing for the BABAR Event Builder”, actes de la conférence Computing in High Energy Physics, Chicago, Illinois, USA, 31 Août - 4 Septembre 1998.

[PMC97] PM7375 Lasar-155 long form data sheet, PMC-Sierra Inc., Canada, Juillet 1997 [2].

[Puj95] Guy Pujolle, Les Réseaux”, Eyrolles, Paris, 1995.

1. disponible sur le site http://weblib.cern.ch
2. disponible sur le site http://www.pmc-sierra.com

[Rob97] J. Robinson, “RACEway Interlink’s Adoption and Growth”, VITA Journal Septembre 1997, pp. 12-16.

[Sas93] O. Sasaki et al., “A VME Barrel Shifter System for Event Reconstruction for up to 3 Gbps Signal Trains”, IEEE Transactions on Nuclear Science, vol. 40 No. 4, Août 1993, pp. 603-606.

[Sch98] A.G. Schmidt et D. Minoli, “Multiprotocol over ATM”, Manning Publications Co., Greenwich, USA, 1998.

[SGS94] ST C-104 Asynchronous Packet Switch data sheet, SGS Thomson, June 1994.

[Sha48] C.E. Shannon, “A mathematical Theory of communication”, Bell System Technical Journal, vol. 27, Juillet et Octobre 1948, pp. 379-423 et 623-656 [1].

[Sha93] A. Shah et al., “FDDI: a High Speed Network”, Prentice Hall, USA, 1993.

[Sim95] R.J. Simcoe et T.B. Pei, “Perspectives on ATM Switch Architecture and the Influence of Traffic Pattern Assumptions on Switch Design”, ACM Computer Communication Review vol. 25 No 2, Avril 1995, pp. 94-105.

[Ska97] B. Skaali et al., “A Prototype DAQ System for the Alice Experiment based on SCI”, actes de la X^{ème} conférence internationale IEEE Real Time Systems, Beaune, France, 22-26 Septembre 1997, pp. 293-298.

[SUN87] SUN Microsystems Inc. Network Working Group, “XDR: External Data Representation”, Request For Comment (RFC) 1014 [2].

[Sur97] K. Suruga et K. Hayakawa, “No Cell Loss: Digital’s ATM Flow Control”, actes du 2nd International Data Acquisition Workshop on Networked Data Acquisition Systems, 13-15 Novembre 1996, Osaka, Japon, World Scientific Publishing, Singapour, 1997, pp. 81-85.

[Tra92] SARA Chipset Technical Manual, TranSwitch Corp., Connecticut, USA, Novembre 1992.

[Ver99] J. Vermeulen, “Modelling parameters”, communication interne, 6 Juillet 1999 [3].

[Wel97] M. Welsh et al., “ATM and Fast Ethernet Network Interfaces for User-level Communications”, actes du 3rd IEEE Symposium on High Performance Computer Architecture, San Antonio, Texas, USA, 1-5 Février 1997.

[Wu94] B. Wu et al., “Applications of the Scalable Coherent Interface in Multistage Networks”, actes de la conférence IEEE TENCON’94, Frontiers of Computer Technology, 22-26 Août 1994, Singapour.

[X3T93] X3T9.T Task Group of ANSI, “Fibre Channel Physical and Signalling Interface (FC-PH)”, Revision 4.2 October 1993.

1. disponible sur le site http://cm.bell-labs.com/ms/what/shannonday/paper.html
2. disponible sur le site http://library.kitten.org.uk/rfc/
3. disponible sur le site http://www.nikhef.nl/pub/experiments/atlas/daq/modelling.html

ANNEXE 1: MÉTHODE POUR ÉVALUER LA TAILLE MÉMOIRE DES ROBS

A1.1. Modèle

La capacité mémoire d'un ROB doit être suffisante pour que les éventuels débordements dans cet élément n'induisent pas un temps mort supérieur à une valeur jugée acceptable (1% suivant le cahier des charges du T/DAQ d'ATLAS). Il s'agit de déterminer la capacité d'une file d'attente. Le modèle étudié est présenté Figure A-1-1.

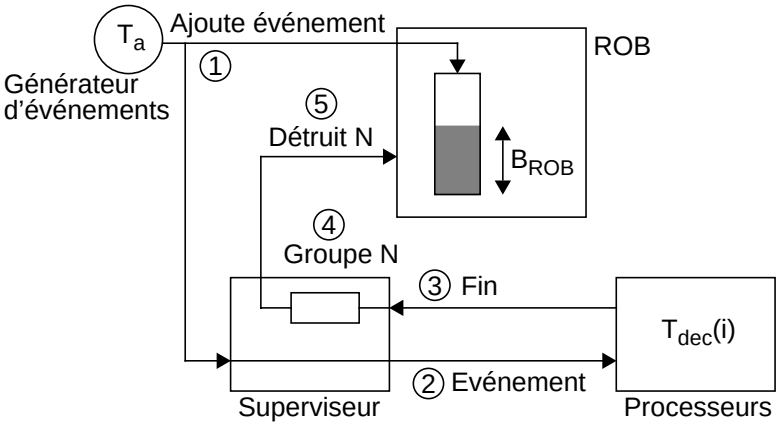


Figure A-1-1: Modèle de la file d'attente dans un ROB.

Les opérations sont les suivantes:

- (1) le générateur d'événements produit à intervalle régulier, T_a, un nouvel événement et incrémente le compteur du nombre d'événements dans le ROB;
- (2) l'événement est passé au superviseur puis aux processeurs; le temps de traitement total, T_{dec}, pour cet événement est tiré de manière aléatoire suivant la distribution du temps de traitement que l'on a fixée;
- (3) la fin de traitement est communiquée au superviseur;
- (4) le superviseur accumule N indications de fin de traitement d'événement;
- (5) N événements sont retirés du ROB.

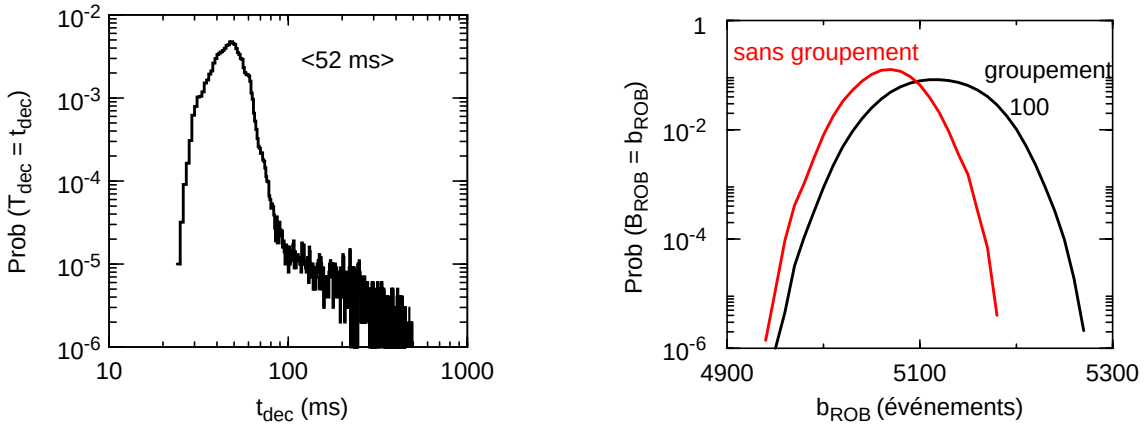
La simulation vise à déterminer la distribution du nombre d'événements dans le ROB, B_{ROB}, en faisant varier divers paramètres du modèle.

A1.2. Résultat de simulation

Pour la simulation les paramètres suivants sont choisis:

- T_a vaut 10 μs (trigger de niveau 1 fonctionnant à 100 kHz).
- Le facteur de regroupement des décisions de trigger effectué par le superviseur est 1 ou 100.
- La distribution de T_{dec} est celle que l'on mesure sur un démonstrateur (voir détails dans [Cal98b]). La distribution de T_{dec} est présentée Figure A-1-2a. La valeur moyenne de T_{dec} vaut

52 ms et le maximum vaut plusieurs centaines de milli-secondes.



a) distribution de T_{dec} (mesure)

b) distribution de B_{ROB} (simulation)

Figure A-1-2: Mesure de T_{dec} et simulation de B_{ROB}.

Les résultats de simulation sont présentés Figure A-1-2b. Une capacité de 6000 événements dans le ROB est suffisante. Considérant que chaque événement fait 1-2 ko par ROB; la taille mémoire nécessaire est de 6-12 Mo ce qui ne pose pas de problème particulier.

Examinons maintenant la situation relative aux processeurs destination. Pour chaque événement, le traitement effectué par un processeur consiste en 16 étapes. Chaque étape consiste en la collecte de 1 ko de données provenant de 4 ROBs suivi d'un traitement d'une durée de 100 µs. Le système fonctionne à 50% de sa capacité. Le temps moyen d'occupation d'un processeur par événement est de ~15 ms (16 fois le temps d'envoi de 4 requêtes, collecte de 4 fragments, formatage et traitement de 100 µs). Le nombre total de processeurs qu'il faudrait pour absorber un flux d'événements de 100 kHz dans ces conditions est de 3000 unités (fonctionnement à 50% de charge) ce qui dépasse le nombre envisagé. La réduction nécessaire du temps de traitement moyen par événement conduira à une baisse du temps de latence du trigger.

D'autres simulations peuvent être effectuées en adoptant une distribution plus réaliste pour T_{dec} (bien que le modèle adopté ici soit à mon sens plutôt pessimiste). On observe que:

- la valeur moyenne de B_{ROB} est déterminée par la valeur moyenne de T_{dec} et le facteur de regroupement des décisions de trigger,
- la queue de la distribution de B_{ROB} est peu dépendante de la distribution de T_{dec}, un temps de latence élevé pour une faible fraction des événements n'induit pas d'augmentation sensible de la capacité de mémorisation requise au niveau des ROBs.

ANNEXE 2: ARTICLE SUR LA RECONSTRUCTION D'ÉVÉNEMENTS

ATLAS Internal Note

DAQ-No-63

28 January 1997

Evaluation of a Congestion Avoidance Scheme and Implementation on ATM Network based Event Builders

D. Calvet, P. Le Dû, I. Mandjavidze

CEA Saclay, 91191 Gif-sur-Yvette CEDEX, France

M. Costa, J.-P. Dufey, M. Letheren, C. Paillard

CERN, 1211 Geneva 23, Switzerland

Abstract

This paper addresses the design of event builders based on switching networks for data acquisition systems in the field of High Energy Physics. We show that the use of a simple congestion avoidance scheme based on a connection permutation algorithm can eliminate the potential contention in the switching network and make most popular network topologies adequate to carry event building type of traffic. The performance of this scheme is presented and some techniques to improve its efficiency are outlined. We describe implementation issues and show how to adapt our scheme to build a system based on standard ATM components. We present simulation results that show the expected performance of this implementation. We detail the additional benefits of using ATM technology for event builders and outline some possible applications to future physics experiments.

I. Introduction

The next generation of High Energy Physics experiments, ATLAS [1] and CMS [2], proposed at the CERN Large Hadron Collider (LHC), will place heavy demands on the data acquisition and on-line filtering systems. Sophisticated multi-level selection systems will reduce the raw data flow from a few tens of TBytes/s down to the several tens of Mbyte/s that will then be recorded on permanent storage for subsequent off-line analysis. The use of large switching networks is envisaged to handle the several tens of Gbit/s of data bandwidth that remains after a first level of selection. The RD-31 project [3] aims at evaluating an approach to data acquisition based on the use of standard Asynchronous Transfer Mode (ATM) packet switching technology [4]. The main activities of the collaboration are to propose event builder architectures, evaluate their performance by means of simulation and implement small scale demonstrators to validate the concepts that were developed.

This paper deals with event builder architecture and performance evaluation. It is organized as follows. The problem of event building and some requirements for the design of an event builder are presented in Section II. The problem of congestion in event builders is briefly described in Section III. Some network topologies suited to event builders as well as a congestion avoidance scheme are described in Section IV. The performance of this scheme is expounded in Section V. Some possible improvements are presented in Section VI. General implementation issues are discussed in Section VII and a solution based on ATM is presented in Section VIII. The additional benefits of using ATM for event builders along with possible applications to future physics experiments are outlined in Section IX. Section X summarizes and concludes the paper.

II. The Problem Of Event Building

A *Basics*

For LHC experiments, it is commonly admitted that the last level of on-line selection of events will be based on sophisticated algorithms that will process a large part of the available event data. It is envisaged that the processing of a given event will be done by a single processing node. For each event, the data that has to be transferred to the selected processing node is spread across a variable number of sources (~1000 at most). Furthermore, a single processor will certainly not have enough computing power to cope with the rate of incoming events. Therefore, a farm of processors will be needed. It is foreseen that as many as ~1000 processing nodes might be needed for each of the ATLAS and CMS trigger systems to process in real-time the incoming events at the targeted rate of ~1-10 KHz. A large switching network will be required to link all data sources to the farm of processors. The system that collects data from many sources and delivers complete events to a given processor is called an event builder [3]. A very simplified view of an event builder is shown in Figure 1.

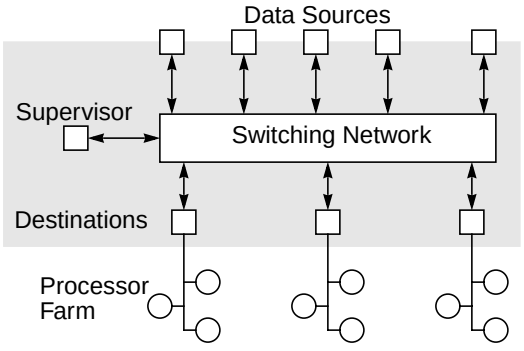


Figure 1. Event Builder

An event builder includes a switching network with its appropriate control, a number of data sources, a number of des-

Presented at the 2nd International DAQ Workshop on Networked Data Acquisition Systems, RCNP, Osaka, 13 - 15 November 1996

tination modules that collect event fragments and re-assemble the corresponding events. The software that controls each part of the system is beyond the scope of this paper.

We shall now present some of the most desirable properties of the switching network.

B. Requirements for the network of an event builder

For LHC experiments, one of the most important requirements of the network of the event builder is that it has to offer an aggregate bandwidth of the order of ~10-100 Gbit/s to interconnect several hundreds of nodes. The network has to work in a reliable way under the event building type of traffic (i.e. concentration of event data from many sources towards one destination, correlated sources...). The traffic is almost unidirectional with most of the flow going from the data sources to the destination processor farm. The packet routing latency should be low enough to allow for the routing of several thousands of packets at a few kilo-Hertz rate.

In order to reduce the cost of development and ease maintenance, the use of industrial standard equipment is strongly recommended. Technical support during the lifetime of the experiment is essential (~10-15 years). To facilitate installation and test, it is also desirable that the network can be partitioned. The system should be scalable to accommodate future upgrades. Efficient tools for network management, monitoring, fault diagnosis and isolation are needed. Obviously the cost of the system must fit within the budget of the physics experiments.

III. Congestion In Event Builders

As previously mentioned, the event building type of traffic is very specific. A potentially large number of sources can simultaneously send data to the same destination. If no care is taken, this continuous concentration of traffic can exceed the available bandwidth of the output link. This can induce long transmission latencies and even data losses due to buffer overflow in some elements of the network. Several techniques have been proposed by the RD-31 collaboration to reduce the contention in event builders based on ATM switches. These techniques fall into two main categories. Rate division techniques are used to prevent the sources from exceeding the available bandwidth of an output link [5]. Traffic shaping techniques are used to artificially break the time correlation between different sources in order to avoid contention within the switching fabric [3]. In this paper, we will study in more details a particular traffic shaping technique, and more specifically how a simple congestion avoidance scheme can eliminate contention within the switching network of an event builder.

IV. Non Blocking Event Builders

A. Network topologies for event building

At first, we shall recall some of the requirements for the network of an event builder with regard to the connectivity. It is required that the network can provide a path from any data source to any destination processor. It may or may not be required that every destination has a path to every source. A separate path can be envisaged to carry the relatively small amount of traffic flowing from the destination processors to the data sources. It may not be required that a path can be established between sources. A path between destination processors might not be needed either. It is acceptable that a given destination has its connection to each source established one at a time in a sequential manner. Reciprocally, it is acceptable that each source is connected successively to all destinations. Ideally, all connections between *any* pair of source/destination can be established at a given instant in a non-blocking fashion. Nonetheless, it is sufficient that all connections between *selected* pairs of source/destination can be established at a given instant in a non-blocking way. We shall now describe how popular network topologies can fulfill those requirements.

Since the dawn of the telephony era, there has been an enormous amount of research to design non blocking switches with a number of crosspoints less than that of a single-stage crossbar matrix. Many different multistage interconnection networks have been proposed: Banyan, Delta, Clos [6]. The performance of those networks under various types of traffic has been studied extensively [7], [8]. It is well known that a Banyan switch is blocking: even when every input is assigned to a different output, as many as $\sqrt{N}$ connections may be contending for the use of the same internal link. Several techniques have been proposed to render Banyan networks non-blocking. One possibility is to add a sorting network in front of the Banyan network (Batcher-Banyan network) [9]. Another possibility is to run internal links faster than input and output links. For example, the 8-port AT&T Phoenix switch is fully non-blocking because it has an internal bandwidth expansion of 2 [10]. For the problem of event building, the key point is that the use of a non-blocking network is not mandatory. A non-blocking network provides a path between any pair of source/destination, but, as previously stated, it is sufficient that a path between selected pairs of source/destination can be established in a non-blocking way. When every input is assigned to a different output correctly selected, Banyan switches, Delta switches and others are non-blocking. This is illustrated below on an Omega network.

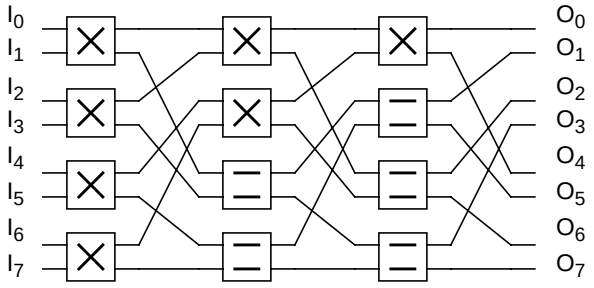


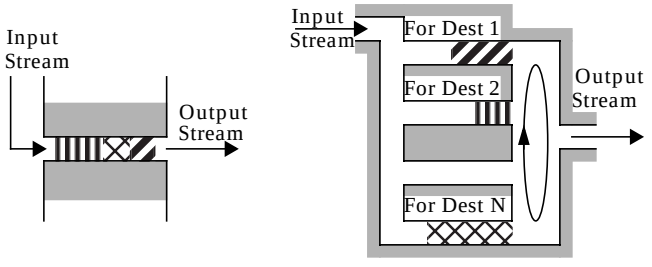
Figure 2. Omega Network

By successive shuffling and exchange operations, the Omega network can perform not all but some important permutations of its inputs in $\log_2 N$ steps [11]. In particular, it can perform cyclic shifts and permutations that connect input i to output (t+i) Mod N. We show in Figure 2 the configuration where t=1, that is input i is connected to output (i+1) Mod 8.

When t is increased, all sources are connected to all destinations following a cyclic sequence of period 8. At any given instant all sources are connected to a different destination without contending for a common internal link. After a complete cycle, any particular source has been connected to all destinations. This is sufficient for an event builder. The network is used in a fully non-blocking way. There is no contention in the network. A circuit switched network where a physical path is established between end-point pairs during a given time slot can be used. A packet switched network where data packets are routed on the basis of the information contained in the packet header can also be used. Each cross-connect need not to include buffering since there is no queuing in the network at any load. Consequently no data losses can occur due to internal buffer overflow. All data is buffered in the sources and queuing only occurs in the sources.

B. Source models

In what follows, it is assumed that the switching network is used as described previously. In this case, the sources determine the overall performance of the system and must be carefully designed. We present in Figure 3.a a possible design of a source. In this model, all the messages are placed in the same queue, regardless of the destination of each individual message. Messages are serviced on a First In First Out basis. The message at the head of the queue is sent when the configuration of the switching network permits its transmission to the correct destination. It is obvious that all messages in the queue are blocked until the message at the head of the queue has been removed. This problem known as "head of line blocking" is the main limitation of systems based on input queuing rather than output queueing. With various assumptions, it has been demonstrated that a N x N space division packet switch based on input queuing saturates at a utilization of $\sim(2-\sqrt{2}) = 0.586$ for large N [12].



a) Single queue model b) Multiple queues model

Figure 3. Logical model of a source

Though very simple, a system based on this type of source has a poor throughput. We present in Figure 3.b a design of a source that does not suffer from the limitation previously described. In this arrangement, the source comprises a distinct logical queue for each destination. All messages in a given queue have the same destination. Hence messages in this queue cannot be blocked by other messages at the head of the queue that have a different destination. Those logical queues in each source are serviced in a periodical way to give the opportunity to each source to send some messages to each destination. At a particular time slot, a different queue in each source is serviced. The queues that are serviced correspond to the source/destination connection pattern of the switching network.

We will now investigate the performance of an event builder that employs this type of source queuing. Some results on queue occupancy have been reported in [13].

V. Non Blocking Event Builder Performance

If the switching network is used as described in section IV, its operation is transparent and need not to be modeled or analyzed. The switching network simply introduces a constant delay to the data packets transmitted. Only the behavior of the sources has to be investigated.

We shall use the following notations. Let N be the number of sources. We assume that the number of destinations is also N (i.e. "square" event builder). Let $S_i(t)$ be the size of the event fragment in source i for the current event at time t. Let $f_i(x)$ be the distribution of S_i and $F_i(x)$ its integral. We assume that $f_i(x)$ is independent of i. All sources have the same characteristics. Let S_{av} and S_{max} be the average size of an event fragment and its maximum size respectively. Let L be the amount of data that each input link can transfer per time unit (i.e. link speed). Let C be the number of time units between two successive changes of the connection pattern between sources and destinations. We assume that the arrival of events is Poisson distributed. Let R be the average amount of time units between two successive events. We also assume that events are assigned to destinations in a uniform periodical way.

The event building latency, T_B is the amount of time required to gather the data from all the sources participating to a given event into a destination processor. When all fragments (especially the largest one) have been sent by the sources, we shall consider that the event has been built. We show that T_B can be expressed as follows:

$$T_B = T_Q + T_S \qquad (5.1)$$

The amount of time due to queuing in the source is T_Q. The amount of time required to send the last event fragment is T_S. We will now study some of the properties of T_Q and T_S.

A source can send data to a given destination within a timeslot of C time units every NC time units. Therefore the amount of time needed for a given source to send its fragment is:

$$T_s(i) = \frac{S_i}{L} + (N-1)\ C\ \mathrm{Int}\left(\frac{S_i}{L\ C}\right) \qquad (5.2)$$

We consider two cases:

- case a. $C << \mathrm{Max}_i(S_i)/L$.

This case correspond to a short interval between the changes of the source/destination connection pattern compared to the time required to send a fragment at full speed. It will be referred to as the "cell based barrel shifter scheme" [3]. In this case, the event building latency depends on the time required to send the largest event fragment. This can be approximated by:

This shows that the available bandwidth of a link is equally

$$T_S = \frac{Max_{i \in [0, N-1]}(S_i)}{\frac{L}{N}} \qquad (5.3)$$

shared between all destinations. T_B is proportional to N. The distribution of the largest event fragment among N is:

$$f_{Max_i(S_i)}(x) = N \cdot F(x)^{N-1} \cdot f(x) \qquad (5.4)$$

- case b. $C >> Max_i(S_i)/L$

This case will be referred to as the "event based barrel shifter" [3]. We have to consider now the amount of time required to send the last (and not the largest) event fragment. That is:

$$T_S \equiv \frac{S_{last}}{L} \qquad (5.5)$$

Where S_{last} is the size of the last fragment required to complete the event. In fact, S_{last} can be any fragment, hence the distribution of S_{last} is f(x).

We will now discuss the second term of the event building latency, T_Q. If the load on each link is low, the amount of time that a non-zero size event fragment is queued in a source is a fraction of (N-1)C time units. It is the amount of time required for the system to change from its current connection configuration pattern to the configuration that allows the transmission of a particular fragment. The distribution of T_Q is uniform in [0,(N-1)C].

At least one complete cycle of connection permutations is needed for the event building if all event fragments are of non null-size. Otherwise, a fraction of the complete cycle might be sufficient for case a), and is always sufficient in case b). For the latter case, if we consider that the k-last event fragments are of null-size, T_Q is in the interval [(N-1-k)C,(N-1)C]. The probability that the k-last fragment are of null-size is $f(0)^k$.

The distribution of the event building latency T_B can be obtained by linear convolution of the distribution of T_Q and T_S. It can also be obtained by means of simulation. We present in Figure 4.a the results obtained by simulation with N=64, L=1, C=1, S uniformly distributed in [0,31] and R arbitrarily large.

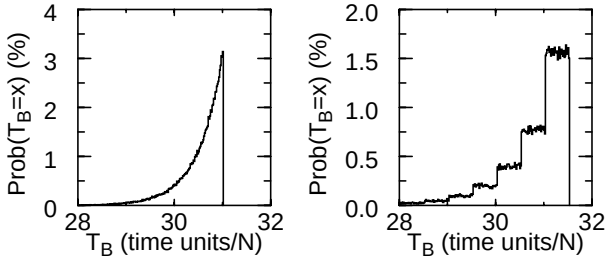


Figure 4: Event Building latency distribution

This corresponds to an operation in mode a). It can be seen that T_Q is negligible compared to T_S. Hence the distribution of T_B follows a law in x^{N-1} as stated in (5.4). We also present the results obtained by simulation for operation in mode b) with the same parameters but C=32 and S in [0,1]. It can be seen that T_S is negligible compared to T_Q. The highest peak in the distribution correspond to the case where the last fragment is not of null size. Other peaks correspond to an increasing number of null size event fragments. It can be seen that the height of the peaks follows a law in 2^{-k} since the probability that an event fragment is of null size is 1/2.

When the load on source links is increased, the probability that several event fragments to be routed to the same destination are present in a particular source at a given instant becomes non null. Hence an additional queuing delay is introduced. This delay is null if the time required to send an event fragment to a destination is always shorter than the amount of time between two successive events assigned to this destination. With the approximation made for case a), this happens when:

$$R \geq \frac{Max_i(S_i)}{\frac{L}{N}} \qquad (5.6)$$

This correspond to ~50% load if f(x) is uniform and if the time interval between events is constant. This condition is almost satisfied if R is Poisson distributed for large N. For case b), it can be estimated that no additional queuing is introduced if R>C. When queuing occurs, the performance is rapidly degraded while the load is increased. The limitation does not come from the saturation of the links of the switching network but from that of the sources. We present in Figure 5.a the average event building latency versus load for case a) and b) with R constant, N=64, L=1 and S in [0,31] for various values of C.

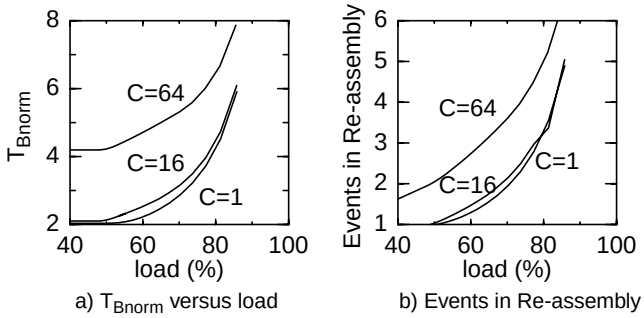


Figure 5. Performance versus load

The normalized event building latency, T_{Bnorm} equals T_B divided by NS/L which represents the time needed to transfer an event of average size. It can be seen that the minimum normalized event building latency equals 2. This comes from the fact that S follows a uniform distribution, hence S_{max} is twice S_{av}. The system does not saturate for a load up to ~75%. However, as soon as several event fragments for the same destination are queued in a source (condition stated in 5.6), the corresponding destination can have more than one event in reassembly phase at a given moment. The average number of events in re-assembly phase per destination is plotted in Figure 5.b.

VI. Increasing Efficiency

As was previously explained, the switching network with an appropriate synchronization can be viewed as a non-blocking

network providing a bandwidth L/N from each source to each destination. It is therefore important that all sources have more or less the same needs in terms of bandwidth. Furthermore, there should not be any privileged destination in the system. A correct load balancing is important. If the amount of data generated by a source has a large dispersion, long queuing delays will be introduced. It is therefore desirable to create groups of sources connected to a single network node in a coherent way.

VII. Implementation Issues

The implementation of the proposed scheme can be done in several ways. It can be envisaged to develop all components with specific hardware. A connection switching network is adequate. Each switching element can be very simple because it does not need to include buffers. All sources and the switch have to be synchronized and specific hardware has to be implemented for this purpose. The obvious advantage of this method is that the network can be tailored to the exact needs of the users. It is rather simple and cost effective. The main disadvantage of this approach is that all components are specific. The entire hardware and software has to be designed, documented and maintained. For LHC experiments, it is strongly recommended that industrial technologies are used whenever possible. Hence packet switching technologies can be considered. The implementation of this scheme on a circuit switched or packet switched network requires that:

- **r1)** all sources can be synchronized,
- **r2)** each source has the capability to transmit to a given destination within a given time slot.

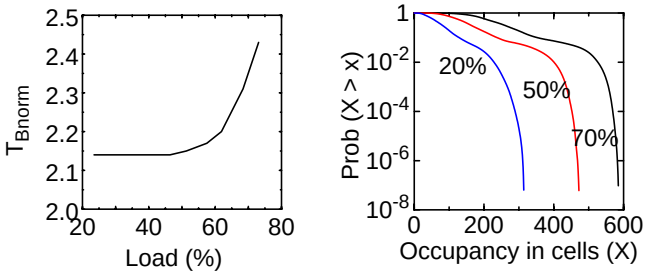
If none of those requirements are met, the operation of the event builder is possible only at very low load. This is obviously the most inefficient solution.

Both requirements must be met if the network has little or no buffering capabilities. In most cases, it should be possible to relax one of those constraints. This is particularly true for the first requirement which might be difficult to meet if the time slot C is of the order of a few µs. A dead-time between time slots cannot be avoided. The results we presented remain valid if we consider that the fraction of the link bandwidth that corresponds to this dead-time is wasted. If an operation with a larger period C is acceptable, software implementation of source synchronization can be envisaged. This is certainly the case in ALICE [14] since the event fragments are of relatively large size. The second requirement is difficult to meet with most networking technologies (e.g. Ethernet) that only allows to send packets on a first in first out basis, at link speed (i.e. they implement the source model described in Figure 3.a). We will show in the next paragraph that this limitation does not exist in the case of ATM. If it is acceptable that the system operates in the event based barrel shifter mode and at low load, the second requirement need not to be satisfied if the first one is met.

VIII. Implementation With ATM

The proposed scheme needs to be adapted in order to facilitate its implementation using standard ATM components. There is no simple way to achieve a perfect synchronization of all sources if the time slot is of the order of a few µs. Hence requirement r1) can be difficult to meet especially if a software synchronization mechanism is used. On the other hand, requirement r2) is very easy to meet using ATM. One of the main features of ATM is that it allows to share the bandwidth of a link between users. When several Virtual Connections (VC) share the same physical link, the packets transmitted on different VCs are interleaved at the level of ATM cells. A possibility for the implementation of requirement r2) is to open a permanent VC with a constant bit rate of L/N between each source and each of the N destinations. In this way, each source will send to a given destination during a time slot proportional to 1/N. Another possibility is to open variable bit rate channels and specify that the average rate of each VC is L/N, and that the maximum size of a burst is 1 cell (that is the peak bit rate is L/N). Although the ATM standards do not specify the exact method for partitioning bandwidth among applications, designers of Segmentation And Re-assembly (SAR) chips use variations of the same mechanisms in their implementations. For example, the transmit scheduler of the IDT NicStar [14] uses a transmit schedule table that specifies explicitly how to interleave the segmentation of packets to be transmitted on different VCs. More sophisticated algorithms are used in Siemens' SARE [16] or in PMC-Sierra's Lazar [17]. Future chips will implement the Available Bit Rate service (ABR).

We have shown in section IV that the network of an event builder does not need any buffering when all sources are correctly synchronized. Reciprocally, the buffering capability of the network can be used to absorb the contention induced by the phase difference of de-synchronized sources. The network should include enough buffering so that contention does not create data losses due to internal buffer overflow. We have simulated the operation of an ATM based event builder where sources implement a NicStar-like interface with no specific synchronization. This corresponds to r2) satisfied but not r1). The model for the ATM switching fabric is a Banyan network connecting 256 sources to 256 destinations. Each cross-point uses a common buffer for its 16 output queues. All sources are identical and generate event fragments of 4 Kbytes (Erlang distribution). The size of the system and the amount of event data correspond roughly to what is expected for the ATLAS experiment.



a) T_{Bnorm} versus load b) Buffer occupancy for various loads

Figure 6. ATM based event builder operation

We present in Figure 6.a the average normalized event

building latency T_{Bnorm}. At low load, T_{Bnorm} is determined by the ratio S_{max}/S_{av}. It can be seen that the operation of the event builder is satisfactory for loads up to ~70%. We plot in Figure 6.b the probability that the number of cells buffered in a given switching element exceeds a certain value for different loads. It is sufficient that each switching element can buffer ~600 cells as shown in Figure 6.b. This is a reasonable value since existing ATM switches like Fore Runner ASX-100 include 4096 cells of buffering per output, that is 64K cells, for a 16 port configuration [18]We can therefore conclude that the operation of the ATM event builder is possible without source synchronization, but with the partitioning of bandwidth supported by ATM technology.

IX. Additional Benefits of using ATM

There are several advantages of using ATM components in event builders. A first set benefit come from the fact that ATM seems to gain acceptance in the telecommunication market and for LAN backbone switches. Prices become affordable and more and more vendors offer a large choice of ATM components, switches, interfaces, testers, etc... The large investments and support from the industry offer a good guarantee for the availability of ATM hardware and software in the long term.

Other advantages of using ATM to implement event builders for data acquisition applications are the capability of sharing the bandwidth (rate division) so as to avoid congestion in the network as previously explained. We have obtained encouraging results with an 8 x 8 event builder demonstrator as reported in [19]. We showed in [5] the benefits of the rate division technique and explained how the same physical ATM link can accommodate various types of traffic. The proposed architecture “C”, for the ATLAS experiment, implements phased event building where only a subset of sources participate in the event building process [20]. The processor analyzing the event decides on the fly which data is needed and sends requests to the corresponding sources. A single ATM network to carry both data and protocol traffic has been proposed.

This paper focused on the transfer of data from sources to destination processors. However, in a real system, some information (run control, performance monitoring, error recovery, etc...) is exchanged between all nodes. All these messages could use the same ATM network that is used to transfer data. If a separate network is considered, all network interfaces, links and switches have to be duplicated. This increases system complexity and cost but may not improve its robustness and reliability.

We believe that ATM technology is a sensible choice for the design of event builders and that it can solve a number of issues in an elegant way.

X. Summary

This paper addressed the design of event builders based on switching networks for data acquisition systems in the field of High Energy Physics. We first recalled the basic principles of event building. The requirements for the network of an event builder with regard to the connectivity were analyzed. We showed that the use of a simple congestion avoidance scheme based on a connection permutation algorithm can eliminate the potential contention in the switching network and make most popular network topologies adequate to carry event building type of traffic. We presented both analytical and simulation results that show the performance of this scheme. Some techniques to improve its efficiency were outlined. We described implementation issues and presented a design based on standard ATM components. Simulation results showed the expected performance of this implementation. We detailed the additional benefits of using ATM technology in the design of event builders. Some possible applications to future physics experiments were outlined.

XI. References

[1] ATLAS collaboration, “Technical Proposal for a General-Purpose pp Experiment at the Large Hadron Collider at CERN”, CERN/LHCC/94-43, December 1994.

[2] CMS collaboration, “The Compact Muon Solenoid Technical Proposal”, CERN/LHCC/94-39, December 1994.

[3] M. Costa et al., “NEBULAS - A high performance data-driven event building architecture based on an asynchronous self-routing packet-switching network”, CERN / LHCC/95-14.

[4] L. G. Cuthbert, J-C. Sapanel, ATM - the Broadband Telecommunications Solution, IEE Telecommunications Series 29, ISBN 0 85296 815 9, London, 1993.

[5] D. Calvet et al., “A Study of Performance Issues of the ATLAS Event Selection System Based on an ATM Switching Network”, IEEE Transactions on Nuclear Science, Vol. 43. No 1. February 1996, pp. 90-98.

[6] H. Ahmadi et al., “A Survey of Modern High-Performance Switching Techniques”, IEEE Journal on Selected Areas in Communication, Vol. 7. No 7. September 1994, pp. 1091-1103.

[7] Y-C. Jenq, “Performance Analysis of a Packet Switch Based on Single-Buffered Banyan Network”, IEEE Journal on Selected Areas in Communication, Vol. Sac-1. No 6. December 1983, pp. 1014-1021.

[8] D.M. Dias et al., “Analysis and Simulation of Buffered Delta Networks”, IEEE Transactions on Computers, Vol. C-30, No 4, April 1981, pp. 273-282.

[9] K. E. Batcher, “Sorting Networks and Their Applications”, in Proc. Spring Joint Computer Conference, AFIPS, 1968, pp. 307-314.

[10] V. P. Kumar et al., “PHOENIX: A Building Block for Fault Tolerant Broadband Packet Switches”, IEEE Global Telecommunications conference, December 1991, pp. 228-233.

[11] T. Lang, “Interconnections Between Processors and Memory Modules Using the Shuffle-Exchange Network”,

IEEE Transactions on Computers Vol. C-25, No 5, May 1976, pp. 496-503.

[12] M. J. Karol et al., “Input Versus Output Queueing on a Space-Division Packet Switch”, IEEE Transactions on Communications, Vol. COM-35, No 12, December 1987, pp. 1347-1356.

[13] M. Nomachi, “Event Builder Queue Occupancy”, SDC Note 93-566, August 1993.

[14] ALICE Collaboration, “Technical Proposal for A Large Ion Collider Experiment at the CERN LHC”, CERN/LHCC/95-71, December 1995.

[15] “IDT 77201 NicStar”, User Manual, IDT Inc., Santa Clara, November 1995.

[16] “SARE PXB 4110”, IC for Communications Data Book, Siemens, August 1995.

[17] “PM 7375 Lazar 155”, Programmer’s guide, PMC-Sierra Inc., March 1996.

[18] S. Bush et al., “Switching to ATM”, Network World Magazine, February 1994, pp. 37-39.

[19] M. Costa et al., “Lessons from ATM-based event builder demonstrators and challenges for LHC-scale systems”, Proceedings of the 2nd Workshop on Electronics for LHC Experiments, Balatonfuered, Hungary, September 23-27 1996.

[20] J. Bistricky et al., “Single Processing Farm Option for the ATLAS Event Selection”, Proceedings of the 2nd Workshop on Electronics for LHC Experiments, Balatonfuered, Hungary, September 23-27 1996.

ANNEXE 3: COMPLÉMENT DE RÉSULTATS DU PROJET PILOTE

ATLAS DAQ Note 2000-011
10 March 2000

An integrated system for the ATLAS High Level Triggers: Concept, General Conclusions on Architecture Studies, Final Results of Prototyping with ATM

J. Bystricky, D. Calvet, O. Gachelin,
M. Huet, P. Le Dû, I. Mandjavidze
CEA Saclay DAPNIA, 91191 Gif-sur-Yvette Cedex, France

Abstract

This paper recalls the concept of a proposed integrated system for the "ATLAS High Level Triggers"[1] that has been investigated over the last 5 years of our involvement in the ATLAS collaboration. We give the rationale that justifies our proposed model and list a number of architecture principles that could be a basis for the final system. We detail the major results of implementation studies made during the "LVL2 demonstrator program" and the "LVL2 Pilot Project". We present the final results of our investigations on ATM networking technology.

I. INTRODUCTION

This paper is organized as follows. Section II describes the concept of the proposed scheme for ATLAS High Level Triggers (HLT) and gives some elements of its justification. Section III describes our proposal for some aspects of the HLT architecture and different implementation options. Section IV introduces the various ATM testbeds. The operation and performance of individual testbed components, Read-Out Buffers, the RoI Builder/ supervisors and the processor nodes are detailed in sections V, VI and VII respectively. Section VIII describes the operation of complete testbeds. Section IX gives the final conclusions of investigations on ATM technology.

II. TOWARD AN INTEGRATED SYSTEM FOR THE ATLAS HLT

In this section, we first recall the principles of the scheme for the LVL2 and LVL3 triggers described in the ATLAS Technical Proposal published in 1994 [1]. We then introduce our alternative view and the main elements of its justification that come from some of the knowledge acquired by the Trigger/DAQ community over the last few years [2].

A. Principles of the '94 ATLAS Trigger/DAQ Model

A detailed view of the overall Trigger/DAQ architecture proposed in 1994 for ATLAS is given in [1]. The system consists of two physically and logically independent parts: the LVL3/DAQ and the LVL2 trigger which is not on the main dataflow. The LVL2 trigger aims to achieve a rejection factor of 100 using the so-called "local/global" scheme. For each event, data from ~5 RoIs are pushed via LVL2 links toward "Feature Extractors" (data-driven hardware or farm of processors) working in parallel. Extracted "features" are sent to a processor within a farm of "global processors" that combines them and issues the LVL2 trigger decision. These are distributed to all "DAQ crates". Rejected events are discarded. For accepted events, the full event data is sent via LVL3 links to the event builder that assembles full events. These are passed to one of the processors within the Event Filter (EF) farm that achieves an event rate reduction of 10 using complicated algorithms similar to those used off-line.

B. Evolutions (accepted, debated or just proposed)

B.1. LVL2 trigger

In the '94 scheme for the LVL2 trigger, shortening decision latency to save on event buffer memories was a major goal, justifying the use of a "push" dataflow, the introduction of parallel processing and the motivation for low latency networks (such as SCI). Hardware based "feature extraction" was considered as well as DSPs or general purpose processors (excluding the use of a standard operating system). Segmenting the systems between LVL2 and LVL3 (networks and processors) as well as segmenting networks inside LVL2 (local network per detector, global network, network for RoI distribution...) was envisaged given the prospects of general purpose processors and networking products for the scheduled date of construction of the system. The '94 architecture was mainly defined for high luminosity RoI guided operation, and the way to implement a B-physics trigger compatible with that scheme was being studied.

1. formerly ATLAS Level 2 and Level 3 Triggers.

A first evolution comes from physics studies indicating that primary and secondary RoIs can be distinguished, and that rejection is possible after processing only primary RoI(s) [2]. Because there are only one or two primary RoI(s) per event, parallel processing at the RoI level cannot bring much advantage. In addition, rejection is also possible after processing RoIs in only one detector, therefore parallel processing at the sub-detector level consumes more network and computing resources than a sequential scheme. A possible sequential scheme for LVL2 was introduced in [3].

Since 94, technology has evolved rapidly and several important factors are now apparent:

- Memory at the Read-Out Buffer (ROB) level can be several tens of MBytes so that cutting on latency is not any more a major design constraint.
- General purpose processors running an operating system (e.g. PCs) are now sufficiently powerful to be considered as the main computing engine for the LVL2 trigger. PCs provide flexible and cost effective platforms for data processing, and their prospects of evolution are excellent.
- Large switching fabrics with a few 100 ports are available now, and link speed is moving from 100 Mbit/s range to 1 Gbit/s.

These factors are also responsible for the evolution of the LVL2 trigger. The use of parallel processing for RoIs is being abandoned because of its relative complexity and because the latency issue is not the most stringent constraint. The “push” data flow for LVL2 was not pursued because it does not allow for sequential data collection and processing in an easy way. Although early prototypes of the LVL2 trigger used different physical networks and processors for local feature extraction and global processing [4], merging these networks in the same physical network was found more practical and became a realistic option given the evolution of networking products. Similarly, using a separate network for the distribution of RoIs to all ROBs brings significant extra complication to the system and offers no real advantage. Hence the “pull” protocol described in [5] has been adopted for data collection at LVL2. All processors in LVL2 play the same role; the control of the processing for a particular event is entirely given to a single processor.

Another factor of evolution is the understanding of the B-physics trigger and the fact that additional algorithms not originally foreseen at the second level trigger are now envisaged at LVL2 (e.g. inner detector Full Scan, b-jet tagging). The current scheme envisaged by the community for B-physics is based on the analysis of the complete TRT data to search for low p_T tracks after an initial LVL1 RoI guided step. The track candidates identified at LVL2 define new RoIs that are analyzed using the silicon and pixel detector data. In average, ~2 particles candidates are identified at this stage. By the analysis of the corresponding data in the calorimeters, an event rejection of ~10 is expected for these events. This scheme for B-physics has several major impacts on the architecture:

- Sequential processing is needed. This scheme does not fit easily in the original ‘94 local/global scheme but can be mapped on a revised version where local and global networks are merged; and local and global processing are combined in the same physical processor. Sequential data transfer using the “pull” mechanism now considered for RoI data collection is adequate. The same architecture can, in principle, be used for low and for high luminosity running.
- The subset of event data defined by RoIs at LVL1 is not sufficient for LVL2. This system needs also to gather data from one or even several complete detectors (e.g. the TRT, or the complete inner detector). The LVL2 system can define new RoIs (not identified at LVL1) and need a mechanism to obtain the relevant data from the ROBs. A physical copy (triggered by LVL1 RoI information) of data corresponding to LVL1 RoIs from the main memory of the ROBs into a secondary buffer for use by the LVL2 trigger is not any more sufficient to preserve the integrity of the main DAQ/EF dataflow chain because the ROBs should support asynchronous data request coming from LVL2. We think that the proposal to copy data of LVL1 RoIs into a secondary buffer brings an unnecessary complication to the system without preserving what is claimed to be its advantage.
- Running an inner detector scan at LVL2 requires to transfer data from all the corresponding ROBs toward one of the processors affected to LVL2. The traffic pattern generated is very similar to that of event building. It is foreseen that this traffic will be handled in the same network that transports RoI data because using a custom made network for that purpose is an option that has now been abandoned. Although the concept of merging different types of traffic on the same network (RoI data collection and event building like) was rather controversial when introduced in [5], this principle is progressively becoming a requirement for a B-physics trigger at LVL2. For an inner detector scan, ~100 kB of data per event have to be transferred at a rate that increased from ~4 kHz to ~10 kHz (today’s estimate). This leads to the necessity of handling 1 GByte/s of additional event building type of traffic in the LVL2 network. This requirement is identical to that of the event builder following ‘94 estimates (1 MByte events to build at 1 kHz).
- the TRT scan is a compute intensive algorithm. Recent estimates show that ~60 and ~600 processors would be required for LVL2 without B-physics and with B-physics respectively (low-luminosity running; 40 kHz LVL1 rate) [6]. For, high luminosity running, ~60 processors would be required. FPGA accelerators could bring a

significant speed-up when integrated in each processor running LVL2. A proof of principle of using a FPGA accelerator in the context of a testbed has been shown, and quantitative measurements that could demonstrate the real benefits of the approach are starting to appear [7].

B.2. Event Builder/Event Filter

Over the last six years, the knowledge of rates and data volumes has evolved. The size of full events in ATLAS is now estimated to ~2 MByte [6]. Current LVL2 rejection is estimated to ~20, leading to a 2 kHz output rate after LVL2. Compared to initial figures (1 MByte events at 1 kHz), a four fold increase of the event builder throughput is needed. Doubling the rejection factor of the Event Filter algorithm and halving its average execution time per event are also required if the available CPU power and rate of data recording are unchanged.

As previously explained, the B-physics trigger puts a major additional load on the LVL2 system. The algorithm split between the LVL2 trigger and the event filter is not yet fixed, and the possibility to run part of B-physics trigger algorithms either at LVL2 or in the Event Filter has to be considered. It is therefore possible that the TRT Scan and subsequent steps will be performed by the Event Filter. The output rate of LVL2 would become ~10 kHz. If this hypothesis is correct, it has several major impacts on the architecture of the system:

- If the dataflow of the ‘94 model is kept, the full event data will be pushed to the event builder at the LVL2 accept rate. The requirements for the event builder becomes ~20 GByte/s. This is a 20-fold increase of the requirements compared to the ‘94 estimate. The impact on strategy, technology choices, cost, etc., have to be evaluated.
- If the dataflow is modified so that data transfers for the Event Filter can proceed in several steps, network requirements for the event builder could be significantly reduced. We think that the possibility of sequential data transfers for the EF should not be ruled out until more studies are made. The “pull” protocol proposed for data collection at LVL2 supports all types and combinations of partial, phased, and full event building. Because the “push” scheme does not offer the same level of flexibility, we propose the use of the “pull” scheme for both LVL2 and EF.
- Compared to initial figures (input rate of 1 kHz, rejection of 10), the event rejection done by the Event Filter has to be increased by an order of magnitude, and simultaneously the average processing time per event has to be divided by 10. Although most of the reduction in the average execution time can be expected to come from the additional rejection that will be achieved, we think that selection strategies and algorithm choices may need to be adapted to take into account these considerations.

B.3. System level

Because there are a lot of uncertainties in rates, data volumes, algorithm choices, and selection strategy, flexibility at the global system level is becoming a key point. The frontier between LVL2 and EF can evolve (e.g. all rejection made by LVL2 or part of the rejection also done by EF). The network capacity and processing power needed for LVL2 and EF systems are tightly coupled: any rejection that is not performed at LVL2 has to be done in the EF. Depending on the level of flexibility that is aimed, the possibility to move computing power and network bandwidth between the two systems could be needed to meet the requirements with available resources. If moving part of the B-physics trigger between LVL2 and EF is envisaged, we think that the capability of moving resources is needed (following today’s hypothesis on rates, algorithm execution times, etc.).

III. A Proposal for the Architecture of ATLAS HLT

A. Some Proposed Principles

This section lists a number of principles that we consider important for ATLAS HLT.

- Event selection is sequential. It can be split between LVL2 and EF (flexible boundary). Network bandwidth and computing resources can be exchanged between LVL2 and EF.
- Every ROB is accessible by every processor. A ROB is an event data server: processors can request data from ROBs as many time as needed. This supports a scheme where all data for the EF is obtained in one step and leaves the possibility of having several steps.
- Data collection from the ROBs is initiated by the processors using a request/response protocol (so-called “pull”). The same data collection mechanism is used for LVL2 and EF.
- Collecting partial events for EF is possible. Sequential processing in EF with phased event data collection is possible. This does not exclude that data collection for EF is made in only one step, but leaves more possibilities opened.

B. Dataflow

The principle of the dataflow is shown in Figure 1.

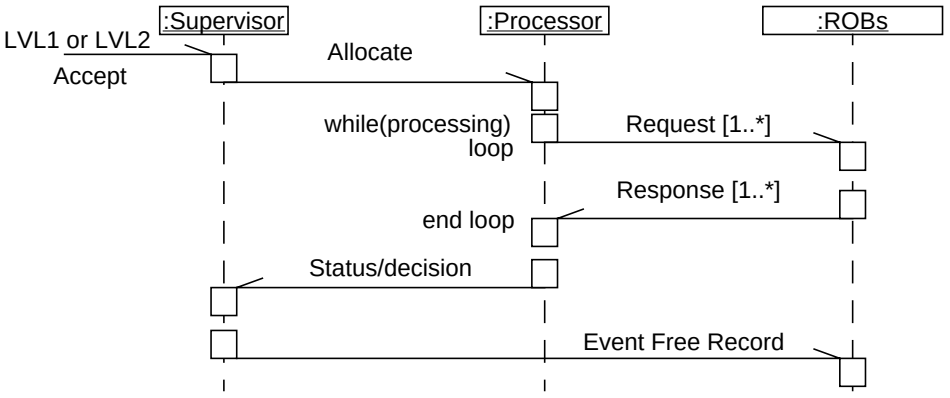


Figure 1. Proposed dataflow for data collection at LVL2 and EF.

The initial guidance for the selection of interesting events will be provided by the Regions of Interest (RoIs) identified at the LVL1 trigger. The selection algorithm consists of a series of steps. A given step is executed only after validation by the previous steps. Each event accepted by the LVL1 trigger is assigned to a processor by a supervisor. The processor "pulls" the information it needs from the sources containing detector data, processes the event fragments received, then decides to fetch more data if needed to make the final decision to accept or reject the event. Accepted events are forwarded to the event filter stage for further selection and analysis while rejected events are discarded. The event filter process may or may not be executed in the same processor that performed LVL2 selection.

C. Implementation

A simplified logical view of the proposed architecture is presented in Figure 2. The main components are:

- the RoI Builder that provides the interface between LVL1 and higher level triggers,
- the supervisors whose main task is to distribute the events accepted by LVL1 to the processors in charge of the selection; these also distribute events accepted by LVL2 if event selection and event filtering are performed in different processors,
- the processors running the selection and/or event filter algorithms. Some of them can include an optional hardware based co-processor to speed-up the execution of some compute-intensive algorithms. Each processor node can be a mono- or a multi-CPU machine, a small cluster of computers or even a remote computing fabric (these two latter implementations would probably require that, for the events assigned to these processors, all the relevant event data is gathered in only one step).
- the Read-Out Buffers (ROBs) grouped into data sources that respond to requests for information from the processors,
- the monitor, configuration and run-control system,
- a common communication network to transfer allocation and protocol messages as well as event data.

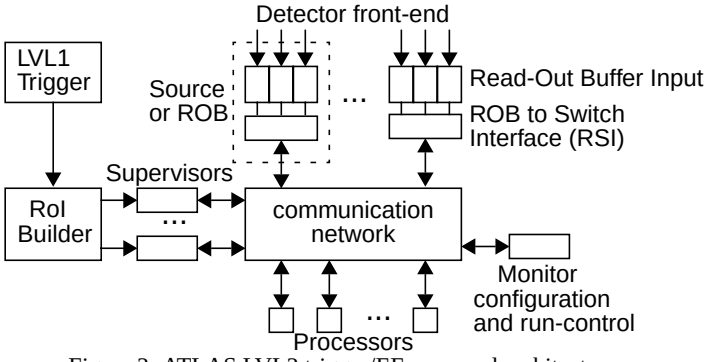


Figure 2: ATLAS LVL2 trigger/EF proposed architecture.

This architecture covers LVL2 and event builder/EF aspects as described in the original proposal [8]. Issues related to LVL2 have been extensively studied in the LVL2 Pilot Project, and our investigations also covered event building. In particular we note that the network used to transfer data to the second level trigger processors can also

be used for event building. We propose to extend to EF aspects the studies on this architecture.

We identify the following options for implementing the proposed architecture.

C.1. Common network, split processor farms option

- A single network for data collection common to LVL2 and EF is used to exchange data and protocol messages between the supervisors, ROBs and processors. A secondary network is used for configuration database access, run control and monitoring...
- Processors are split into two farms: the LVL2 farm performs event selection; the EF farm may perform additional selection before event analysis.
- Transferring computing power and network bandwidth between LVL2 and EF involves re-configuration at the software level only, without hardware changes. Using the same type of processors for both farms would further simplify exchange of computing resources.
- The supervisor is common to LVL2 and EF.
- The EF Farm gets results from the LVL2 farm via the common network.

This model aims to achieve a high level of flexibility while keeping a separation of the processing tasks for LVL2 and EF. The resulting system is uniform at many levels of hardware and software, only the application executed on the various processors differ. We think that this option is a good trade-off between a strict separation of LVL2 and EF into two independent sub-systems and an integrated LVL2/EF system.

This option has been investigated on various testbeds based on ATM [8], [9], [10].

C.2. Common network and common processor farm option

- A single farm of processors is used to perform LVL2 and EF. After accepting an event by running LVL2 algorithms, the same machine executes the EF task. LVL2 and EF tasks may execute in the same or separate processes on one or several CPU's (e.g. SMP's). Network and processor resources sharing between LVL2 and EF is transparent.
- Sending LVL2 results to EF involves only local communication at the processor level.

This model aims to achieve a complete convergence of the LVL2 and EF system at the level of hardware, software and application. It can be implemented on the same hardware as the previous option by software re-configuration. We think that this scheme has the highest potential in terms of flexibility, performance, ease of development and maintenance, as well as future evolution.

This option has been investigated on various testbeds based on ATM [9], [10].

C.3. Split networks, split processor farms option

- Two networks are used for data collection: one to provide data to the LVL2 processor farm, another one for the EF farm. This option would be justified if different networking protocols and/or technologies have to be used for data collection at LVL2 and EF. Exchanging computing resources between LVL2 and EF requires moving hardware. Transferring network resources between the two systems is difficult if not impossible.
- Separate supervisors are used for LVL2 and EF.
- A mechanism has to be defined to provide the EF with results of LVL2. An additional component may be required.

This scheme follows the concept described in the '94 ATLAS Technical Proposal. It cannot be implemented on the hardware configuration suited to the two previous options. We think that this scheme is certainly viable from the technical point of view, but is less attractive than other options in terms of efficiency and flexibility. It leads to a duplication of many hardware and software components, roughly doubles the number of experts and developers needed with implications on global cost, ease of maintenance and possibilities of evolution. Unless a physical separation between LVL2 and EF appears to be mandatory for technical reasons, we think that this option does not compare favorably with the others.

So far, no integrated prototype based on this architecture design have been reported.

IV. Prototype Studies

We have conducted several prototype studies of the common network and split/common processor options, focused on LVL2 aspects and covering aspects related to event building. Because of the importance of addressing all issues relevant to the integration of the LVL2 trigger and DAQ / Event Filter prior to the definition of the final architecture, we have retained the flexible design and functionality of the testbed reported in [8] and, moreover, have extended its capabilities and integrated a number of new components.

A. Modes of operation

The ATM Testbed Software that we designed for the LVL2 Demonstrator Program has been thought for flexibility, performance, modularity and portability. With the same hardware configuration and the same software framework, it demonstrates the operation of a testbed in the following modes of operation:

- A LVL2 like system with a common network for data collection and protocol messages, and a farm of processors emulating a sequential selection process.
- A stand-alone event builder system where full events are assembled.
- A LVL2-EF like system with a common network for data collection for LVL2 and event building; a common supervisor group to allocate events accepted by LVL1 and LVL2; a logical split of the processors into a farm executing a LVL2 like sequential selection, and a farm assembling full events after LVL2 accept and emulating some processing sequence.
- A LVL2-EF like system with a common network for data collection for LVL2 and event building; a common supervisor group to allocate events accepted by LVL1; a single processor farm executing a LVL2-like sequential selection where the last step is the gathering of full event data followed by some processing.

In addition to the ATM Testbed Software, our latest testbed also support the execution of the Reference Software [11] (which covers only LVL2 aspects so far).

B. Description of the latest ATM testbed

Starting from the modest 8-node system assembled during the LVL2 Demonstrator Program, the testbed has grown over the last 4 years and all equipment is now installed at CERN. Putting in common the resources of several of the groups involved in the LVL2 Pilot Project, a system composed of up to 38 commodity PCs, 10 VME and 1 CompactPCI single board computers is being operated. The PCs are a mix of 4 generations of Pentiums with clock speeds from 200 MHz to 450 MHz (mono and dual CPU's). Almost all PCs are dual boot WindowsNT / Linux. The single board computers are PowerPC based (100 MHz, 200 MHz, 300 MHz) and run LynxOS. These different nodes are connected to an ATM switch equipped with 48 ports at 155 Mbit/s (three quarters of the total port capacity of that model of switch). Any machine can act as a processor, data source or supervisor emulator. A typical testbed configuration is a system with 22 sources, 22 destinations, 3 supervisors and 1 monitor.

We implemented an Ethernet based start-up program to start all nodes from a single workstation. For simplicity, instead of a configuration database, we use a shared parameter file that is accessed by each node at start-up. In our implementation, all communications for the run-control and monitoring are handled via ATM once the system has been started. The Reference Software uses a more conservative approach based on a low speed Ethernet path separated from the main data path. We use a simple ASCII terminal based user interface to monitor the operation of the system from a central point. A few basic commands allow to start, suspend, resume or terminate the operation of individual nodes or group of nodes, modify some run parameters dynamically, clear, collect then save online statistics and histograms, etc.

The ATM Testbed Software has been successfully tested in heterogeneous systems composed of LynxOS machines, WindowsNT and Linux PCs, Digital Unix workstations and Symmetric Multi-Processor servers. The system can run on legacy LANs using the standard UDP/IP stack (e.g. for development purposes), and on ATM using the socket interface or optimized true zero-copy device drivers and libraries [12]. The Reference Software runs on all machines excepts the LynxOS platforms and relies on the same optimized ATM libraries and drivers that are used by the Testbed Software.

V. The Read-out Buffers Input and Sources

After a LVL1 trigger accept, all event data are digitized and sent from each of the ~1600 detector front-end cards to Read-Out Buffers via high speed (~1 Gbit/s) point-to-point links. The ROB input card (ROBin) buffers data during the event selection process; only a small fraction of the total event data (few percent) is used by the selection algorithms. In order to reduce the number of connections to the communication network and better match the output bandwidth of ROBins to that of network links, ROBins are grouped in so-called "ROB complexes" or simply ROBs or sources.

A. Description of the hardware

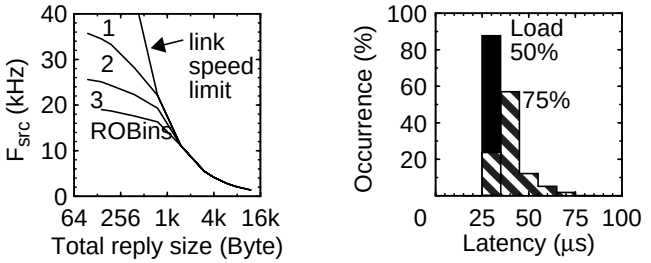
Each source module comprises a variable number of ROBins (typically 1-8) connected via a backplane bus to a RoB to Switch Interface (RSI) attached to the communication network. We developed a prototype of the ROBin on a PCI Mezzanine Card (PMC) [13]. This form factor gives a lot of flexibility for plugging ROBins in VME single board computers equipped with PMC sites, or inside desktop PCs or CompactPCI crates using the appropriate

passive adaptor card. The present version of the ROBin includes a PCI bridge, a 33 MHz Intel I960 processor, its program memory and some glue logic. A more elaborate version of the ROBin that includes a 100 MHz Intel I960 processor, 8 MB of event data memory, a high speed parallel input port and a global control logic has been assembled and is being tested. The ROB to Switch Interface (RSI) is either a standard PC, VME or CompactPCI single board computer equipped with an ATM interface. With these three types of platforms, we have assembled and tested sources comprising 1, 2 and 3 ROBins.

B. Operation and performance

The RSI is in charge of servicing requests coming via the ATM network and distributing to the RoBs the trigger decisions received from the supervisors. When a request for event data is processed by the RSI, it posts a request to each of the RoBs concerned. The memory of the RSI and that of each ROBin is made visible on the PCI bus. The DMA engine of the ATM card uses its chaining capability to send in a single true-zero copy operation event data replies composed of a message header (stored in the RSI memory) and detector data (stored in each of the ROBIN concerned). This scheme minimizes data movement between the ROBins and the RSI, leading to high performance. Alternatively, it is possible that the ROBins will first copy their data in the RSI memory prior to sending them over the network. Though less efficient, this scheme allows the possibility of performing some data manipulation at the RSI level while the zero-copy scheme permits only preprocessing to be done by the ROBins. The trade-off between these two options is being investigated and no choice has been made yet. Both modes of operation are supported in our implementation and can be used simultaneously.

We present in Figure 3a the maximum rate of data requests, F_{src}, that can be serviced by a source versus the total data fragment size of the response message (using the zero-copy scheme).



a) F_{src} vs. fragment size b) request/response latency

Figure 3: Performance of a multi ROBin source.

The RSI is a 300 MHz PowerPC VME single board computer equipped with a PMC extender so that up to 3 ROBins and 1 ATM PMC can be attached to it. The limitation due to the saturation of the output link is reached for packets larger than 1.5 kB. For short packets, we derive the formula:

$$F_{src} = 1 / (20\ \mu s + 10\ \mu s * nb\_of\_RoB) \quad (1)$$

The amount of time needed to process a request in a source composed of 1 RoB (measured from the time of reception of the request message via the network until the reply message is posted to the ATM interface) is presented in Figure 3b. The data request rate is 18 and 27 kHz (i.e. 50% and 75% of source capability) and the reply message size is 128 bytes. The typical request servicing average latency is ~40 μs. The performance figures that were measured on CompactPCI and on a PC are close to those presented.

Final system requirements mandate that sources shall be able to service requests at a maximum rate of ~10-20 kHz depending on sub-detectors. Current results give confidence that this goal can be met although more work is needed to reach a conclusion (operation of the ROBin with its input link, more ROBins per source, data preprocessing in the RSI...).

VI. The RoI Builder and the Supervisors

A. Description of the hardware

A complete description of the Region of Interest Builder (RoI Builder) designed, built, and tested by our collaborators from Argonne National Laboratory and Michigan State University is given in [14]. The RoI Builder is the interface between the LVL1 trigger and higher level triggers. For events accepted by LVL1, it receives information on regions of interest involving muons, jets, electrons/gammas as well as total and missing transverse energies. This information arrives over 7 separate links, which has to be combined event-by-event to form an event summary to be used by the level-2 trigger processors in refining the choice made at LVL1.

The hardware is based on large Field Programmable Gate Arrays (FPGAs) deployed on 12U cards. Each RoI Builder card accepts input from 3 LVL1 trigger partitions. The RoI Builder for the testbed consists of 2 input cards corresponding to 6 of the 7 partitions in the final system. The output cards are mounted on the rear of the 12U crate from where they are connected to Supervisor Processors (1, 2, 4 or 8) by flat cables. The RoI records from each partition may be distributed to the supervisors either in a round-robin fashion or in a more complex manner determined event by event.

B. Operation and performance

From the stream of records it gets from the RoI Builder, each supervisor forms a summary record and sends it over ATM to a destination processor. A combination of a static schedule table and credit based flow-control mechanism is used to distribute events to the processors according to their relative CPU power and instantaneous load. Another task of the supervisors is to collect trigger decisions (over ATM), pack and multi-cast them (also over ATM) to all ROBs. In the mode of operation where the LVL2 trigger and event filter use different processors, the supervisors also schedule a processor of the event filter farm for each event accepted by the LVL2 trigger. The supervisors can run in emulation mode without the RoI Builder, generating events internally at the desired rate.

The maximum event handling rate, F_{sup}, that can be achieved in our setup is presented in Table 1.

Table 1. Supervisor and RoI Builder performance

F_{sup} (kHz)	no RoI Builder		with RoI Builder		
# of supervisors	1	2	1	2	4
PowerPC 200 MHz	35	70	24	48	96
PowerPC 300 MHz	40	80	28	56	
Pentium II 400 MHz	48	96	-		

For the ATLAS experiment, the maximum LVL1 trigger rate will be 75 kHz (upgradable to 100 kHz). These results show that the desired performance is achievable.

Tests of the RoI Builder with the Reference Software have been made to demonstrate functionality. Although current performance figures are lower than that obtained with the Testbed Software, more investigations are needed and various optimizations are possible.

VII. Destination Processor

A. Data collection and algorithms

The destination processor is in charge of making the selection of the events assigned by the supervisors. At each step of the selection, the processor issues data requests to the relevant sources to get the data it needs for the execution of the algorithm. While waiting for the data of a given event to be delivered, it can handle previous events, issue data requests for other events or become idle if no task can be executed.

The local CPU usage to perform this data collection task on a 400 MHz Pentium II PC running WindowsNT is:

$$T_{dst} = 110\ \mu s + 22\ \mu s * nb\_of\_request \quad (2)$$

On the same platform running Linux it is:

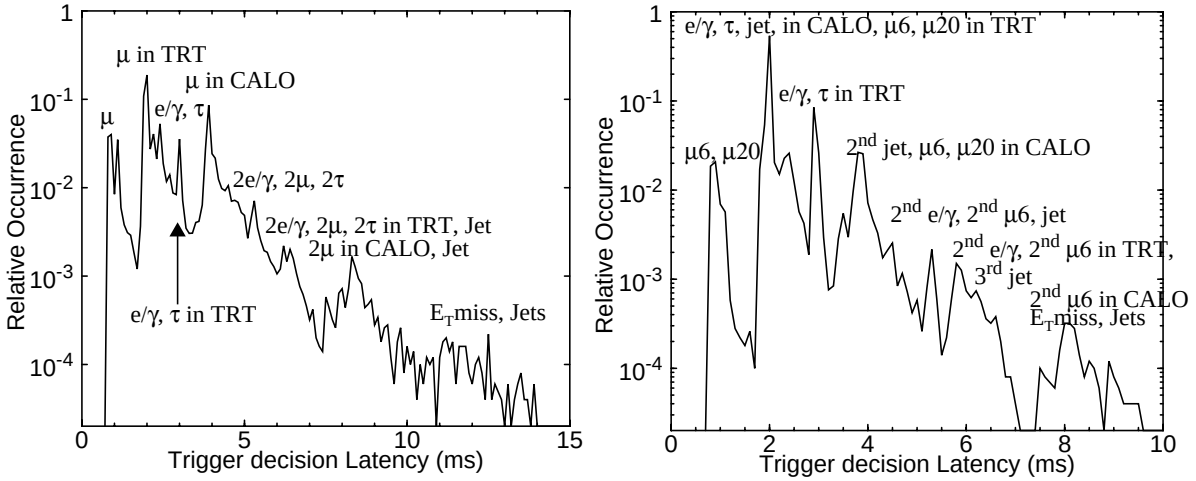
$$T_{dst} = 85\ \mu s + 11\ \mu s * nb\_of\_request \quad (3)$$

Although for supervisor emulation (single thread polling loop application) results are slightly better with WindowsNT (maybe optimizations of Visual C++ compiler are better than that of GNU compiler), shorter context switching and interrupt handling times in Linux make the data collection part more efficient in the processor code (multi-thread interrupt-driven application). Once the desired data fragments have been received, it is necessary to format them in the appropriate structure prior to the execution of the algorithm. Our measurement show that reformatting can be done at ~20 MB/s. So far we have only implemented the algorithm for processing electron/gamma type of RoI, but more complex algorithms and a minimal trigger menu is being studied. This electron/gamma algorithm executes in ~80 μs. In total, a 400 MHz Pentium II, WindowsNT PC is able to handle events composed of 1 electron/gamma RoI at ~2.8 kHz rate (~3 kHz with Linux). For a 14 processor testbed, the total event processing rate is ~22 kHz corresponding effectively to the sum of the capabilities of each processor (not all machines are identical). This type of result provides an input to modeling studies where the combination of expected rates, rejection factors, data volumes, network capabilities and algorithms sequence and timings are used to estimate the number of processors that will be needed to cope with the event rate in the final system.

B. LVL2-like selection sequences

Although no real selection algorithms acting on simulated data were run with the ATM Testbed Software, we tried to run some complex sequences of data collection and dummy processing. Because trying to mimic the complete list of items composing a trigger menu is rather difficult, we made a number of simplifications and approximations. Starting from the trigger menus given in [6], we took the few most significant items and combined the remaining ones in new "trigger items" that do not have any meaning from the point of view of physics, but have a comparable contribution in terms of data movement and processing involved. With these approximations, simplified test trigger menus for low and high luminosity have been derived (see Table 2 and Table 3 in the appendix). The sequences of processing simulated for each type of event are given in Table 4 and Table 5 for the low and high luminosity menu respectively. Data volumes for each detector and RoI types as well as simulated processing times are given in Table 6.

The system is configured with 15 sources, 1 destination (PC 400 MHz), 1 supervisor and 1 monitor. The maximum rate that can be handled by the destination processor is ~950 Hz for both the high luminosity menu and the low luminosity menu without B-physics. When B-physics is added, a rate of 75 Hz is measured. The corresponding execution time is ~13 ms which is close to the sum of the execution time of the low luminosity menu sequence (1 ms) plus the execution time of the inner detector scan performed on one event out of four (45 / 4 =11 ms for algorithm emulation only; the time spent to copy the 120 kB of data received has to be added). With all the hypothesis made, ~80 processors (utilized at 100%) would be needed to cope with a 75 kHz LVL1 rate for the high and low luminosity menus. About ~1000 processors would be needed for the B-physics menu.



a) Low luminosity without B-physics b) High luminosity

Figure 4. Processor operation for the low and high luminosity test menus.

The system is run at 475 Hz event rate (i.e. half of the capacity of the processor node for the no B-physics case) and the distribution of trigger decision latency is measured (Figure 4a and Figure 4b). The peaks observed correspond to the different trigger menu items. The average decision latency is 2.8 ms and 2.4 ms for the low and high luminosity menu respectively.

VIII. Operation of the System

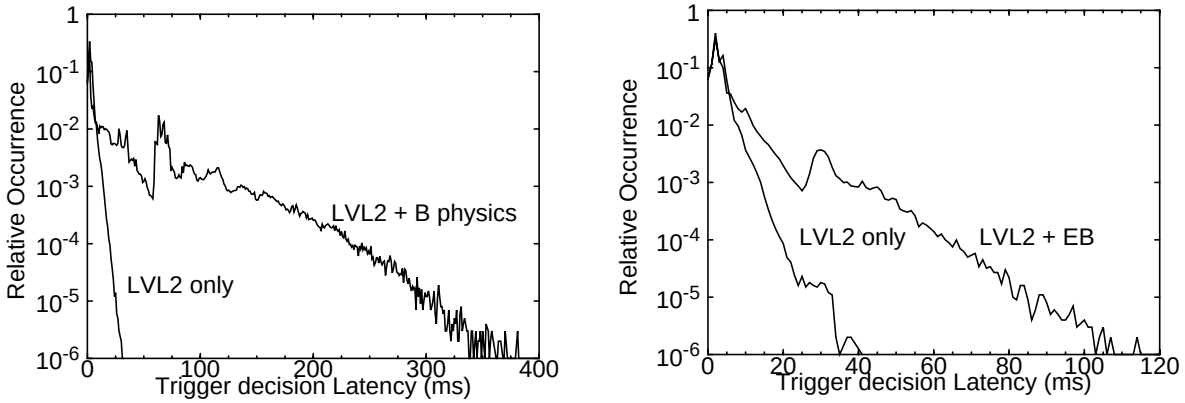
A. LVL2-like selection sequences followed by inner detector scan or event building

The system is now configured with 15 sources, 9 destinations, 1 supervisor and 1 monitor.

The maximum event processing rate for the system is 7.4 kHz and 675 Hz for the low luminosity test menu without and with B-physics respectively. These rates are ~9 times the rate that can be achieved by a single destination processor (9 x 0.95 = 8.55 kHz ≈ 7.4 kHz and 9 x 75 = 675 Hz). The system is run at 50% of its maximum capacity (determined by the processors). The trigger decision latency distribution is shown on Figure 5a. The average is 3.2 ms and 33.4 ms for the low luminosity menu without and with B-physics respectively.

For the low luminosity menu without B-physics, a test is made with executing full event building after a LVL2 accept (in the same processor that accepted the event). The LVL1 event rate is 2.2 kHz when full event building is enabled (this operation is made at 72 Hz given the rejection factor of 30 achieved at LVL2; for our set of parameters the size of each event is 240 kB). The distribution of trigger decision latency is shown in Figure 5b. The average is 3.2 ms and 5.5 ms for the low luminosity menu without and with event building respectively.

A test is made with full event building done in a separate farm of processors after LVL2 accept. No increase of the LVL2 trigger decision latency is observed compared to the case where no event building is performed.

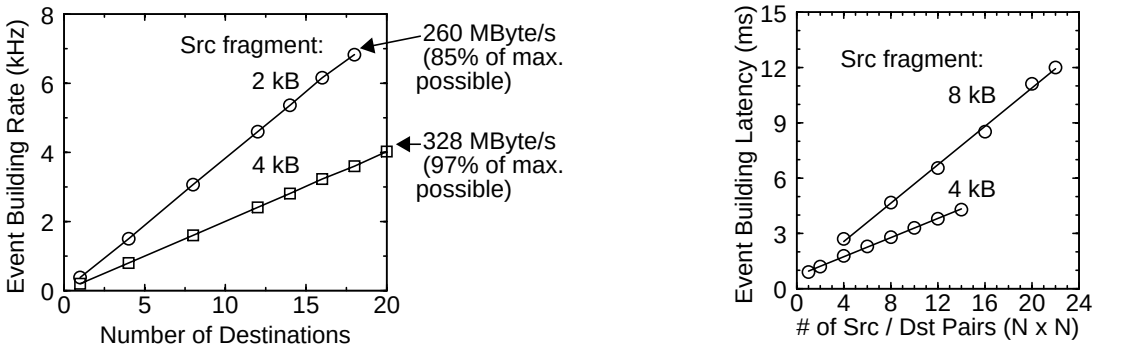


a) Low luminosity with/without B-physics b) Low Luminosity with/without event building

Figure 5.System operation, low luminosity test menu without/with B-physics or event building.

B. Stand-alone Event Builder

The 48 node testbed was operated in event builder mode with the processors requesting complete event data from up to 22 sources. Because we have only assembled 3 ROBins at present, most sources were not equipped with physical ROBins but ran an emulation of them internally.



a) 20 x N Event Builder Performance b) Scalability of N x N Event Builder

Figure 6. Stand-alone event builder mode of operation.

As shown in Figure 6a, the aggregate event builder throughput scales linearly with the number of destination processors. With 20 processors and event fragments of 4 kB per source, the total maximum average throughput is 328 MB/s, corresponding to 97% of the theoretical network capacity and ~1/3 of the initial target performance for the ATLAS event builder (1 GB/s). Because the final system will have ~20 times as many nodes, we think that this performance goal is reachable (even with a 2-4 fold increase of the initial requirement). Tests of event builder scalability were made on a N x N event builder, varying N. Results of measurements are shown in Figure 6b. As expected, the event building latency increases linearly with the size of the system.

C. Combined LVL2/ Event Builder tests

The system was operated in the split LVL2 / event builder and single farm modes. A test algorithm is shown in Figure 7a. The first two steps emulate a trigger LVL2 like selection algorithm. The last step is event building followed by some event filter emulation. Event rate, data volumes and processing times are chosen in a way that network links occupancy and processor CPU utilization are ~50%. For each source, the event selection and event builder traffic is ~4 MB/s (each). Assuming a final system with 512 sources and 2 GB/s global throughput, the requirement for each link of the event builder is ~4 MB/s, identical to the traffic load generated in our tests.

In a first setup, the 14 processors are split into two groups: 7 processors run the trigger LVL2 like process, the 7 others are dedicated to the event building of selected events. In this test, slower machines are assigned to the last step of the sequence which is less demanding in terms of rate and real time response. In a second test, the 14 processors form a single farm with each processor running the complete algorithm sequence. We show the distributions of trigger decision latency, T_{dec}, in 7b. The average of T_{dec} is 3 ms and 3.7 ms for the split farm and single

farm setup respectively.The slow machines used for the event filter emulation in the split farm setup also perform the high rate part of the sequence in the single farm setup. This explains the difference in the latency profiles. Many other considerations need to be taken into account before choosing the processor farm configuration best suited for ATLAS.

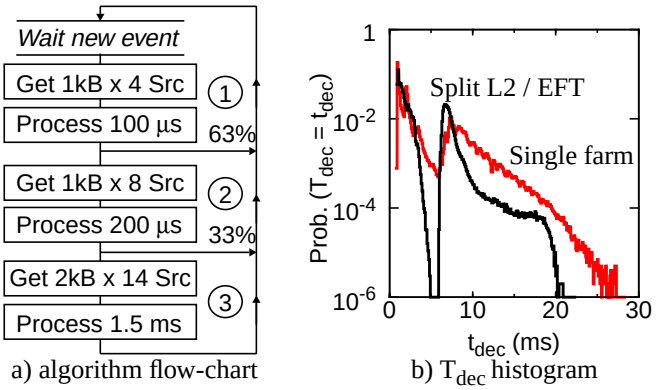


Figure 7: Single farm versus split farm mode of operation.

Considering the flow of data, the various tests suggest that a common network could be used for both the LVL2 trigger (with and without B-physics) and the event builder.

D. System Partitioning

The possibility of partitioning the system in several sub-systems operating independently has been tested. The configuration is shown in Figure 8.

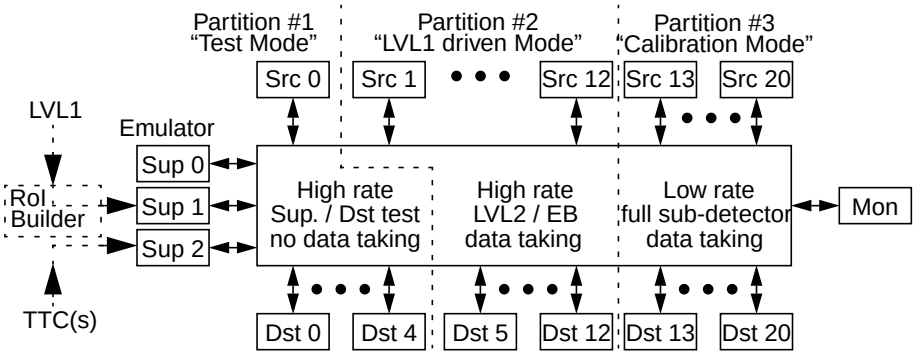


Figure 8. System partition test.

The testbed is split into 3 logical partitions; a supervisor controls the distribution of events for each partition. The operation is as follows:

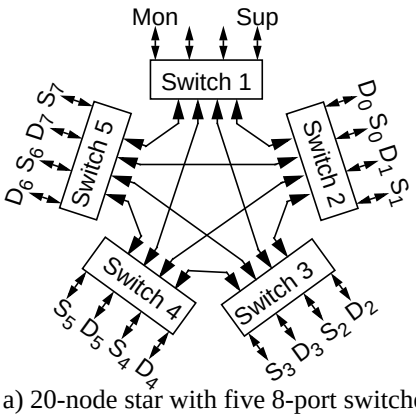
- partition 1: 1 supervisor, 1 source, 5 destinations. The supervisor distributes internally generated events to the destinations that return a random trigger decision. The supervisor accumulates decisions and sends them to the source. This allows to test communications between these elements.
- partition 2: 1 supervisor, 12 sources, 8 destinations. The supervisor dispatches events to processors that perform a sequential data collection and processing. In a real system, events could come from the RoI Builder. This partition is doing normal event selection and data taking.
- partition 3: 1 supervisor, 8 sources, 8 destinations. The supervisor sends events to processors that collect data from all the sources in that partition. In a real system, the trigger could originate from a Trigger Timing and Control (TTC) domain, the sources could correspond to a particular sub-detector where test/calibration are performed at low rate while the rest of the system is running.

We could start/suspend/resume/stop each partition independently. Although a lot of functionality is not present in our software to have a usable system (it is not the purpose of this development), we think that the concept of logical partitioning is compatible with the architecture that we propose. We have exposed the principle of using multiple supervisors (with different types of input) to achieve it.

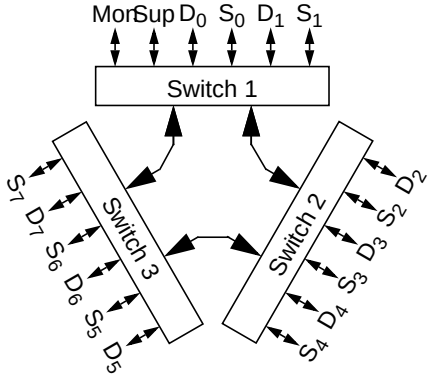
E. Switch cascading

One of the critical aspects of networking for LVL2 and EF in ATLAS (and several other experiments) is to

build networks with a large number of ports. Switches with thousands of ports may or may not be affordable or available. Cascading switches to build larger networks is likely to be required. On the testbed, we have studied the star topology which seems one of the most attractive topology to reduce the number of switch ports used to interconnect the different switches.



a) 20-node star with five 8-port switches

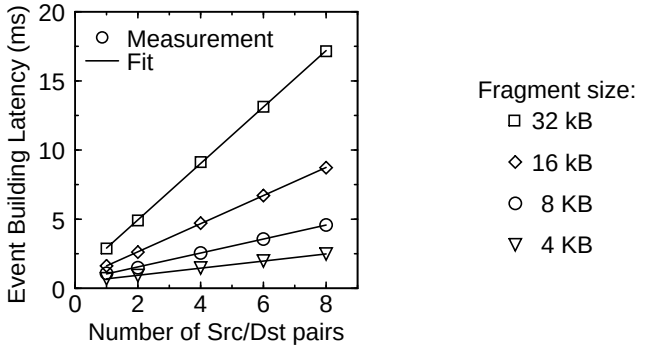


b) 18-node star with three 8-port switches

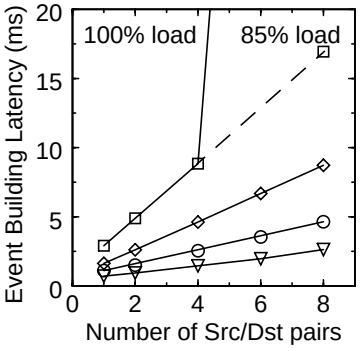
Figure 9. Cascaded switch test.

The arrangement on Figure 9a is a configuration where sufficient bandwidth is provided by inter-switch links to cope with a balanced event building type of traffic without reaching the saturation of these links first. The arrangement on Figure 9b is a configuration where inter-switch links reach saturation before other links (with a balanced event building type of traffic).

Performance measurements of event building latency are shown in Figure 10a and Figure 10b.



b) Event building 20-node star network



a) Event building 18-node star network

Figure 10. Measurements on Interconnected switches.

Linear scaling of performance is observed with the 20-node star topology. As expected, for the 18-node star network, internal link switch saturation is reached when the load on external links is ~86%. Using more than one link between any two switches (link aggregation) could be needed to build larger networks economically.

Although this has not been investigated in our demonstrator, we do not foresee major difficulties related to the routing of traffic in larger multi-path networks and difficulties to achieve a correct load balancing in link aggregations because we use only Permanent Virtual Connections (PVCs). Similar network arrangements can be constructed with other technologies (Ethernet) and we think that these should be studied (simulation and prototyping).

IX. Final Conclusions of Studies on ATM Technology

Over the last 5 years, we think that we have sufficiently studied ATM technology in view of the ATLAS T/DAQ to draw the final conclusions on these investigations, stop further research and developments in this field until the final choices for ATLAS are made, maintain the level of competence acquired on ATM and support the existing system as long as the proof of adequacy and cost effectiveness of another technology does not reach a comparable level of maturity.

A. What we achieved with ATM

- Establish and demonstrate methods to avoid network congestion in event builders.

- Demonstrate communication performance boost with low level zero-copy driver/library for ATM network interface cards.
- Validate a “pull” protocol for data collection for LVL2 and EF using point-to-point and point to multi-point communications.
- Show the use of multi-cast of LVL2 decisions in an integrated network for data and protocol messages.
- Prove the principle of data collection for LVL2 on a system with up to 48 nodes.
- Demonstrate full event building on the same system.
- Show the principle of sequential data collection and processing using simplified trigger menus.
- Show the principle of merging protocol traffic with data traffic on the same network.
- Test a combined network for LVL2 and event builder with split/combined processors.
- Test logical partitioning of the system and network.
- Investigate multi-switch arrangements.

B. Pending studies, limitations, unsolved issues with ATM

- Only PVCs have been used. Connection setup and maintenance would be difficult in a large system.
- The limit on the number of connections to set up is an intrinsic limitation of a connection-oriented technology. Networks composed of more than a couple of thousands nodes may hit that limit.
- Link aggregation has not been tested but could be needed to interconnect switches.
- Running multiple protocols over the same network interface was done at Osaka University (running TCP/UDP/IP and ATM AAL 5 simultaneously) but could be investigated in more details.
- Using “standard” device drivers would be preferable to a custom development, though we think that the performance requirement cannot be met with a standard approach.
- End-to-end data transfer integrity is not guaranteed. So far, no attempt is made to recover lost messages. This strategy is acceptable if losses are rare and do not introduce a bias in the trigger.
- Cost of ATM remains high compared to Ethernet.
- Network interface cards at a speed higher than 155 Mbit/s are not there. This bandwidth limit is a problem that Gigabit Ethernet may solve in a more practical and economical way than 622 Mbit/s ATM.

C. ATLAS HLT based on ATM

Today’s proposal for an initial deployment of an ATM network for ATLAS HLT, could be an interconnection of 8 Cabletron SmartSwitch 9500 ATM switches, each with 224 ports at 155 Mbit/s. A network with 1232 usable 155 Mbit/s ports (560 ports would be used for inter-switch connections) could be built for a cost of ~3.6 M$ (based on today’s public price for this product – 1.8 k$ per port). Assuming that about half of the usable ports (i.e. 600 ports) are connected to sources, the raw bandwidth in the source to processor direction is 93 Gbit/s (10 GB/s net after subtracting the overhead of SONET and ATM). Assuming that this type of network is used for full event building of event of 1 MByte (2 MByte) equally spread across all sources, the saturation rate is 10 kHz (5 kHz). If this network is used to transfer LVL2 traffic only, or LVL2 and full event building traffic, limits have to be evaluated for the different scenarios of LVL2 processing sequences, data volumes, etc.

Another network configuration, could be an interconnection of 25 FORE ASX 1000 ATM switches, each with 64 ports at 155 Mbit/s (1600 ports in total). A network with 1000 usable ports (600 ports used for inter-switch links) could be built for ~1.6 M$ (based on today’s public price – 1 k$ per port).

Evolution of technology, emergence of new products, global cost, requirements for ATLAS, results of investigations on Fast/Gigabit Ethernet are among the many factors that will determine the final choices that will be made at the appropriate time.

D. Past, present and Future of ATM

In the early ‘90, ATM was seen as the best candidate to unify the LAN/WAN world, integrate voice and data over the same network and bring multi-media applications to the desktop. Divergence of interests and a slow standardization process eroded the confidence of potential customer’s who found Fast Ethernet available, easy to learn, compatible with legacy LANs and sufficient for their needs. Though seldom deployed up to the desktop, ATM has been mainly used at the core of large corporate and enterprises networks. As large capacity switches are also becoming available in Ethernet and because Gigabit Ethernet links offer a double fold increase of throughput compared to 622 Mbit/s ATM links for half of their cost, the supremacy of the Ethernet family on this market is acquired. For wide area networking, ATM has been more successful and is now a mature and well established technology. Telecom operators and Internet Service providers are deploying multi-million dollars ATM networks for insatiable customers. It is clear that SDH/SONET is today, and will remain in the foreseeable future, the tech-

nology of choice to transport information over medium and long distances. Whether ATM cells, IP datagrams or other type of packets are the pieces carried over SONET is still subject to an intense debate. Technology developments, strategic directions of major equipment suppliers and choices of operators are pushing in directions that can equally lead to a stagnation or an expansion of the market share taken by ATM, or even to a progressive abandon of this technology in favor of IP for example.

X. REFERENCES

[1] ATLAS Collaboration, “ATLAS Technical Proposal for a general purpose pp experiment at the Large Hadron Collider at CERN”, CERN/LHCC 94-43, 15 December 1994.

[2] ATLAS Collaboration, “ATLAS DAQ, EF, LVL2 and DCS Technical Progress Report”, CERN/LHCC 98-16, 30 June 1998.

[3] J. Bystricky et al., “A Sequential Processing Strategy for the ATLAS Event Selection”, *IEEE Trans. on Nuclear Science*, vol. 44, No 3, pp. 342-347, June 1997.

[4] P. Clarke et al., “SCI with DSPs and RISC Processors for LHC 2nd Level Triggering”, *ATLAS Internal note DAQ-19*, 1 December 1994.

[5] D. Calvet et al., “A study of Performance Issues of the ATLAS Event Selection System based on an ATM Switching Network”, *IEEE Transactions on Nuclear Science*, vol 43, No 1 February 1996, pp. 90-98.

[6] J. Bystricky and J.C. Vermeulen, “Paper Modeling of the ATLAS LVL2 Trigger System”, *ATLAS Internal Note COM-DAQ-2000-022*, 16 March 2000.

[7] C. Hinkelbein et al., “Prospects of FPGAs for the ATLAS LVL2 Trigger”, *ATLAS Internal Note DAQ-2000-006*, 27 February 2000.

[8] D. Calvet et al., “Operation and Performance of an ATM based Demonstrator for the Sequential Option of the ATLAS Trigger”, *IEEE Transactions on Nuclear Science*, vol 45, pp. 1793-1794, August 1998.

[9] D. Calvet et al, “Emulation of the Sequential Option of the ATLAS Trigger an using the ATM Local Area Network of the RCNP Institute”, *ATLAS Internal Note DAQ-130*, 31 August 1998.

[10] D. Calvet et al, “The ATLAS High Level Trigger ATM Testbed”, *Proc. XIth International IEEE Conference on Real Time Systems*, Santa Fe, USA, 14-18 June 1999, pp. 101-105.

[11] R. Hauser, “The Atlas Level 2 Reference Software”, *ATLAS Internal Note COM-DAQ-2000-032*, 17 March 2000.

[12] D. Calvet et al.,”Performance Analysis of ATM Network Interfaces for Data Acquisition Applications”, in *Proc. Second International Data Acquisition Workshop on Networked Data Acquisition Systems*, Osaka, Japan, 13-15 November 1996, World Scientific Publishing 1997, pp. 73-80.

[13] O. Gachelin et al.,”ROBIN: A Functional Demonstrator for the ATLAS Trigger/DAQ Read-Out Buffer”, *Proc. 2nd Workshop on Electronics for LHC Experiments*, Balatonfüred, Hungary, 23-27 Sept. 1996, pp. 204-207.

[14] R.E. Blair et al.,”A Prototype ROI Builder for the Second Level Trigger of ATLAS Implemented in FPGAs”, *ATLAS Internal Note DAQ-99*-016, 7 December 1999.

XI. APPENDIX

Table 2. Test trigger menu for low luminosity running.

Item(s)# [a]	Description	LVL1 Rate (kHz)	Fraction of total rate (%)
1	1 MU	23	57
9	1 EM	11.5	29
10	2 EM	1.6	4
30	1 TAU + XE	1.34	3
2 to 8	2 MU	1.093	3
11-29	1 EM + 4 JET	0.721	2
31-55	2 TAU + XE + 2 JET	0.894	2
TOTAL	-	40.148	100

a. item or list of items with numbering used in [6]

Table 3. Test trigger menu for high luminosity running.

Item(s)# [a]	Description	LVL1 Rate (kHz)	Fraction of total rate (%)
4	1 EM	24.3	62
5	2 EM	4.9	12
2-3	2 MU	4.3	11
1	1 MU	3.9	10
22 to 24	1 TAU + XE	0.961	3
6 to 21	3 JET	0.502	1
25-55	1 TAU + XE + 2 JET	0.532	1
TOTAL	-	39.395	100

a. item or list of items with numbering used in [6]

Table 4. Test sequences for low luminosity running (without B-physics).

Event type	Processing	Reject fraction[a] (%)	Accept ratio[b](%)	LVL2 output rate (kHz)
1 MU	Muon	25		
	Tracker	47		
	Calorimeter[c]	90	4	0.92
1 EM	Calorimeter	86		
	Tracker	86	2	0.23
2 EM	repeat 1 EM twice		0.1	0.001
1 TAU + XE	Calorimeter	80		
	Tracker	60		
	Missing Energy	0	8	0.107
2 MU[d]	Muon	0		
	Tracker	0		
	Calorimeter[e]	0	4	
	Muon	25		
	Tracker	47		
	Calorimeter[c]	90	8	0.087
1 EM + 4 JET[f]	Calorimeter	86		
	Tracker	86		
	Calorimeter Jet 1	25		
	Calorimeter Jet 2	25		
	Calorimeter Jet 3	25		
	Calorimeter Jet 4	25	0.2	0.001
2 TAU + 2 J + XE[g]	Calorimeter Tau 1	0		
	Tracker Tau 1	0		
	Calorimeter Tau 2	73		
	Tracker Tau 2	42		
	Calorimeter Jet 1	25		
	Calorimeter Jet 2	25		
	Calorimeter XE	0	0.6	0.005
TOTAL			3.3	1.351

a. expressed as a fraction of what comes from the previous step.
b. fraction of LVL1 events that passes LVL2 selection.
c. for the B-physics trigger, this step is not run. It is replaced by a full scan of the inner detector (no rejection), followed by the analysis of 2 muons in the calorimeters (rejection 90%)
d. event is accepted if either the first or the second muon passes the selection criteria.
e. this step is skipped for the B-physics trigger.
f. event is accepted if EM is confirmed and all jets are confirmed.
g. event is accepted if at least one of the tau is confirmed and the 2 jets are confirmed.

Table 5. Test sequences for high luminosity running.

Event type	Processing	Reject fraction[a] (%)	Accept ratio[b](%)	LVL2 output rate (kHz)
1 EM	Calorimeter	83	0	
	Tracker	82	3	0.729
2 EM	Calorimeter	83	0	
	Tracker	82	0	
	Calorimeter	83	0	
	Tracker	82	0.09	0.044
2 MU	repeat 1 MU twice		0.2	0.008
1 MU	Muon	25		
	Tracker	47		
	Calorimeter	90	4	0.156
1 TAU + XE	Calorimeter	80		
	Tracker	60		
	Calorimeter XE	0	8	0.077
3 JET	Calorimeter Jet 1	25		
	Calorimeter Jet 2	25		
	Calorimeter Jet 3	25	42	0.211
1 TAU + 2 J + XE	Calorimeter Tau	80		
	Tracker Tau 1	60		
	Calorimeter Jet 1	25		
	Calorimeter Jet 2	25		
	Calorimeter XE	0	4.5	0.024
TOTAL			3.3	1.249

a. expressed as a fraction of what comes from the previous step.
b. fraction of LVL1 events that passes LVL2 selection.

Table 6. Data volumes per source (# of ROBins x size in Bytes) / source multiplicity / execution time[a] (μs).

	Sub-detector		
Processing	Calorimeter	Muon	Inner Detector
RoI MU	4x1k / 4 / 100	4x128 / 4 / 100	4x256 / 4 / 300
RoI EM	4x1k / 4 / 100	-	4x256 / 4 / 200
RoI TAU	4x1k / 4 / 100	-	4x256 / 4 / 200
RoI JET	4x1k / 4 / 100	-	-
XE	4x32 / 7 / 100	-	-
Inner Detector Scan	4x2k / all[b] (15) / 45000		
Event Building	4x4k / all (15) / 0		

a. the processor makes a data copy of all the received fragments before spending the specified amount of CPU time.
b. in the test, all sources are participating (not only those of the inner detector).

ANNEXE 4: PRÉSENTATION LORS DE LA SOUTENANCE DE LA THÈSE

RÉSEAU À MULTIPLEXAGE STATISTIQUE POUR LES SYSTÈMES DE SÉLECTION ET DE RECONSTRUCTION D'ÉVÉNEMENTS DANS LES EXPÉRIENCES DE PHYSIQUE DES HAUTES ENERGIES

Les systèmes de tri en ligne d'événements dans les futures expériences de physique des hautes énergies devront faire appel aux avancées les plus récentes dans le domaine de l'informatique distribuée et des réseaux à haut débit pour traiter en temps réel les flux de données produits.

Après une brève présentation des projets au *Large Hadron Collider*, je décris puis critique l'architecture initiale des systèmes de sélection et de reconstruction d'événements des expériences ATLAS et CMS. Une nouvelle architecture destinée au système de tri en ligne des événements dans ATLAS est définie et justifiée. Je présente les technologies de réseau envisagées pour ce système et plus particulièrement l'ATM. Je décris et étudie de manière analytique et/ou par simulation différents types de réseau adaptés à la reconstruction partielle ou totale d'événements: réseau à permutation synchronisé, anneau à commutation de paquets, réseau à multiplexage statistique.

Je justifie le besoin d'un mécanisme performant pour l'échange de messages par réseau. Les optimisations suivantes sont décrites puis implémentées en ATM: un protocole de communication simplifié, un couplage direct entre une application et sa carte réseau associée, une minimalisation du nombre de recopies des messages. La structure et le fonctionnement des démonstrateurs relatifs au modèle d'architecture proposée sont détaillés. Je rapporte les performances d'un prototype à échelle réduite comportant jusqu'à 48 ordinateurs (environ 1:20 du trigger de niveau 2 complet) reliés par ATM. Par extrapolation de ces résultats et estimation des besoins, une proposition de réalisation pour le réseau principal du système de tri en ligne d'événements de l'expérience ATLAS est décrite.

NETWORK BASED ON STATISTICAL MULTIPLEXING FOR EVENT SELECTION AND EVENT BUILDER SYSTEMS IN HIGH ENERGY PHYSICS EXPERIMENTS

Systems for on-line event selection in future high energy physics experiments will use advanced distributed computing techniques and will need high speed networks. After a brief description of projects at the Large Hadron Collider, the architectures initially proposed for the Trigger and Data AcQuisition (T/DAQ) systems of ATLAS and CMS experiments are presented and analyzed. A new architecture for the ATLAS T/DAQ is introduced.

Candidate network technologies for this system are described. This thesis focuses on ATM. A variety of network structures and topologies suited to partial and full event building are investigated. The need for efficient networking is shown. Optimization techniques for high speed messaging and their implementation on ATM components are described. Small scale demonstrator systems consisting of up to 48 computers (~1:20 of the final level 2 trigger) connected via ATM are described. Performance results are presented. Extrapolation of measurements and evaluation of needs lead to a proposal of implementation for the main network of the ATLAS T/DAQ system.

Discipline: Sciences – Electronique

Mots-Clés:

Système temps réel, informatique distribuée, acquisition de données, tri en ligne d'événements, *trigger*, *event builder*, réseau haut débit, multiplexage statistique, ATM, ATLAS.

Thèse préparée au: DAPNIA/SEI, CEA SACLAY, 91191 GIF-SUR-YVETTE CEDEX, FRANCE