Université de Marne la Vallée
Cité Descartes - 5, bd Descartes
Champs-sur-Marne
77454 Marne la Vallee FRANCE

http://www.univ-mlv.fr

Conseil Européen pour la Recherche Nucléaire
1211 Genève 23 SUISSE

http://www.cern.ch
http://choruswww.cern.ch (CHORUS)

Rapport de stage 2001: applications Java pour le contrôle d'un laboratoire de microscopes

Daniele Raffo

En couverture : des traces de particules dans une chambre à bulles. Photo CERN

Table des matières

Chapitre 1

Introduction

Le présent rapport décrit le stage que j'ai effectué au CERN, au cours de mes études en Maîtrise Informatique à l'Université de Marne la Vallée, France.
Le stage a eu comme sujet le développement d'applications Java pour le contrôle du laboratoire de microscopes utilisé dans le cadre du projet CHORUS, et a été effectué sous la supervision de M. Jaap Panman, chercheur et porte-parole du projet.
Il s'est déroulé du 15 Mai au 31 Août 2001 dans les laboratoires du CERN, site de Meyrin, Suisse.

Remerciements

Avant tout, je remercie infiniment Jaap Panman et Biagio Saitta, qui m'ont rendu ce stage possible. Merci beaucoup également à Johan Uiterwijk, aussi pour son aide lors de mon séjour en Suisse. Leur compétence et leur gentillesse ont fait qu'il a été vraiment enrichissant de travailler pour eux et avec eux.

Je veux remercier Sergey Kalinin, Mihail Chizhov, Christian Schmitt, Bart Van De Vyver et Mathieu Doucet pour m'avoir fourni les informations sur le programme Monitor quand j'en avais besoin.

Merci à Roger Åström pour sa disponibilité en tant que sysadmin du groupe CHORUS, et au Java hacker Mark Dönszelmann pour le Workshop JAS/WIRED.

Oliver Gutsche et Risto Paju ont pourvu les photos du Microscope Lab. Tania Rinaldi m'a aidé pour la rédaction en français de ce rapport.

Parmi tous les étudiants d'été que j'ai rencontré au CERN, je tiens à rappeler quelques uns d'entre eux : Hanna, Juha, Satu, Sami, Leo, Mikael et tous les autres Finns ; Petra, Gilda, Amanda, Erin, Alison et Cathleen; et aussi mes collègues de bureau Ronald, Ivana et Argelia.

Un remerciement spécial à ma famille, pour leur constant soutien moral et matériel pendant mes années d'études à l'étranger.

Enfin, derniers mais non moins importants, merci à Marco Perisi et à Jean-François Vassallucci, pour tout.

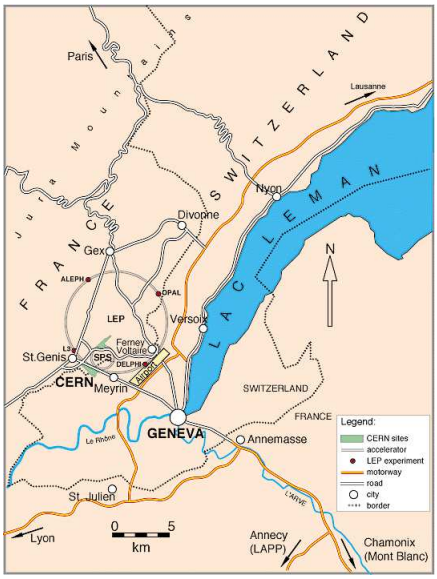
Chapitre 2

Le CERN

Le CERN, Conseil Européen pour la Recherche Nucléaire, est le plus grand centre mondial de recherche en physique des particules.
Fondé en 1954 par 12 Etats Membres de l'Europe (la République Fédérale d'Allemagne, la Belgique, le Danemark, la France, la Grèce, l'Italie, la Norvège, les Pays-Bas, le Royaume-Uni, la Suède, la Suisse et la Yougoslavie), il compte aujourd'hui 20 Etats adhérents (en ayant successivement joint l'organisation l'Autriche, l'Espagne, le Portugal, la Finlande, la Pologne, la Hongrie, les Républiques Tchèque et Slovaque et enfin la Bulgarie en 1999).

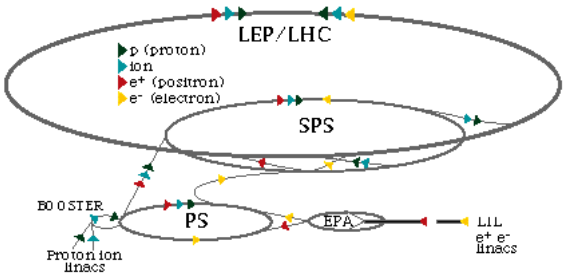
D'autres nations comme les Etats-Unis et le Japon y ont la fonction d'Etat Observateur.
Le CERN emploie environ 3000 personnes et hôte 6500 scientifiques pour leur travail de recherche ; ils représentent 500 universités et plus de 80 nationalités. Plusieurs scientifiques y travaillant ont reçu des distinctions prestigieuses, dont notamment des Prix Nobel (les derniers ont été Rubbia – Van der Meer en 1984, Steinberger – Lederman – Schwartz en 1988, Charpak en 1992), pour leurs découvertes.

Le CERN est situé de part et d'autre de la frontière entre la France et la Suisse, au nord-ouest de Genève. Il est constitué de deux grand sites : le site de Meyrin, à cheval sur la frontière franco-suisse, est l'installation originaire. En 1973, en raison de l'espace à disposition pour les expériences insuffisant, un deuxième site a été construit dans la localité de Prevessin, en territoire français.



L'objectif du CERN est d'étudier la nature de la matière et sa structure, et les interactions entre les composantes fondamentales de la matière.
Pour ces études, on utilise des accélérateurs de particules, qui sont parmi les instruments scientifiques les plus grands et les plus complexes au monde. Dans ces machines, les particules sont accélérées, au moyen de grands aimants, à des vitesses proches de celle de la lumière et mises en collision avec d'autres particules ou de cibles fixes.
Le CERN dispose en tout de dix accélérateurs, circulaires ou linéaires, situés dans le sous-sol. Le plus grand parmi eux est le LEP (Large Electron Positron collider), avec sa circonférence de 27 km ; actuellement il n'est plus en état de marche et va être substitué par le LHC (Large Hadron Collider) qui est en cours de construction dans le tunnel du LEP. Les autres accélérateurs les

plus importants sont le SPS (Super Proton Synchrotron) et le PS (Proton Synchrotron).



La recherche pure effectuée au CERN a de nombreuses retombées dans la vie quotidienne. La thérapie du cancer, l'imagerie médicale et industrielle, des techniques de détection et de résistance aux radiations, l'électronique rapide, l'instrumentation de mesure, de nouveaux matériaux, sont quelques unes des nombreuses applications de technologies développées au CERN. Le CERN étant financé avec des fonds publics, les résultats de ses recherches sont toujours publiés; ils sont libres de droits et librement utilisables par l'humanité.
C'est le cas par exemple du système d'information distribué basé sur l'interconnexion entre hypertextes, inventé en 1990 par Tim Berners-Lee et appelé World Wide Web. Conçu comme aide à la communication au sein de la communauté de la physique des particules, qui avait besoin d'accéder à travers l'Internet à des bases de données communes, échanger et éditer des documents scientifiques, le WWW s'est rapidement répandu à l'ensemble du monde académique pour finalement devenir un outil universel.

Chapitre 3

Le projet CHORUS

Préambule

Les scientifiques ont essayé de mesurer la densité de matière dans l'Univers en observant les étoiles et les galaxies (Univers visible).
Toutefois les conclusions qui en découlent vont à l'encontre d'autre observations faites par les astrophysiciens, qui prédisent une densité de matière plus élevée. La masse visible des étoiles, galaxies et gaz interstellaires semble en effet ne former que le 10% de la masse estimée de l'Univers. Cette énigme est connue sous le nom de "Problème de la masse manquante".

Un candidat possible comme composante de cette partie manquante est le neutrino (ν). Les neutrinos sont des particules élémentaires invisibles qui ont été produites au moment du Big Bang et qui se trouvent partout dans l'Univers. Actuellement, on n'a pas encore réussi à mesurer leur masse, que les scientifiques considèrent comme nulle. Les neutrinos ont en effet une faible interaction avec la matière; ceux qui proviennent du Soleil traversent la Terre entière sans résistance.
La découverte de la masse du neutrino aurait donc pu fournir une réponse au problème de la masse manquante. [1]

La résolution de ce problème permettrait aussi d'en savoir plus sur le destin de l'Univers. En effet, l'Univers est en expansion depuis le Big Bang ; sa future évolution dépend de la force de gravitation, qui est la seule capable de contrebalancer le mouvement d'expansion, et qui dépend de la densité de

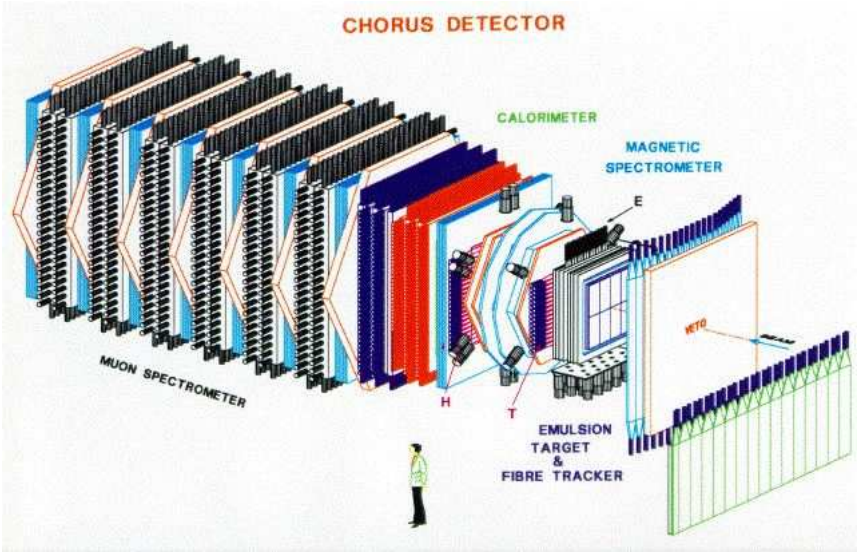
1. C'est dans ce cadre qu'il est né, en 1990, le projet CHORUS. Actuellement, on pense que les neutrinos ne peuvent pas constituer la totalité de la masse manquante.

matière dans l'Univers. Si l'Univers a une densité suffisante il arrêtera donc de s'étendre pour après se comprimer à nouveau. Dans le cas contraire, il continuera son expansion éternellement.

Ainsi donc, la mesure de la masse des neutrinos intéresse aussi bien la science de l'infiniment petit (physique des particules élémentaires) que de l'infiniment grand (cosmologie).

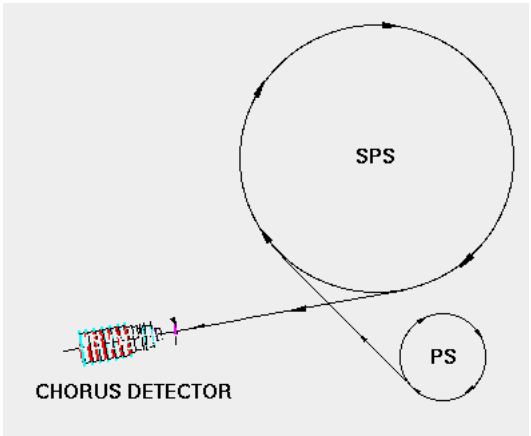
Dans l'expérience CHORUS (CERN Hybrid Oscillation Research apparatUS) on analyse les interactions d'un faisceau pur de neutrinos d'un type particulier (neutrino muonique ν_μ) produit par l'accélérateur SPS. Le dispositif d'analyse présente une sensibilité très élevée pour mettre en évidence la présence de neutrinos d'un autre type (le neutrino tau ν_τ), indiquant l'existence d'oscillations entre les deux types. La détection de ces oscillations permettrait, pour un phénomène typique de la mécanique quantistique, d'en déduire la valeur de la masse du neutrino.

Une telle découverte aurait évidemment un grand impact dans plusieurs domaines de la science.



L'expérience

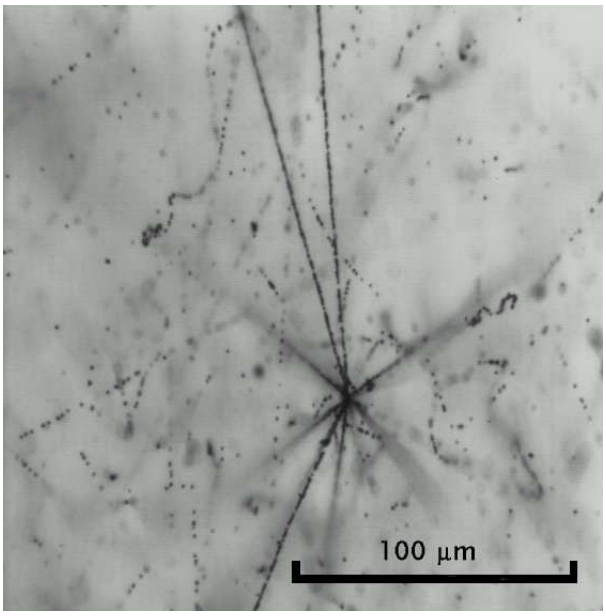
Durant cette expérience, des émulsions nucléaires ont été exposées, de 1994 à 1997, dans un faisceau de neutrinos produit par le SPS.



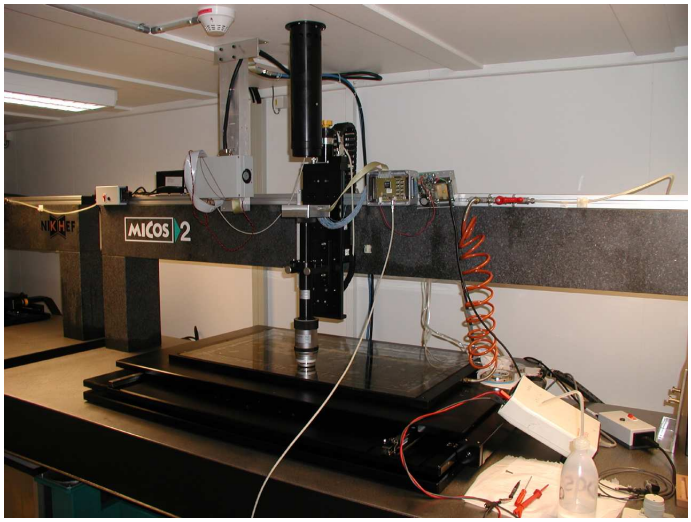
Ces émulsions sont des films photographiques améliorés ayants une sensibilité équivalente à plusieurs dizaines de milliers de ASA. Ils peuvent enregistrer des empreintes de particules avec une précision de 1 micron (un millième de millimètre). Le poids total de l'émulsion utilisée est 800 kg.

Les émulsions ont été utilisées en unités, appelées *modules*, de 36 x 72 cm. Des ensembles de 8 modules étaient regroupés dans une unique plaque (*plate*). 36 plaques étaient empilées pour former un *stack*. Chaque *stack* a ensuite été exposé au faisceau de neutrinos pendant une période variant de 3 mois à 2 années.

Les traces de collisions et de désintégrations des particules sont donc restées enregistrées dans les émulsions (figure en bas). Après le développement, ces dernières sont maintenant en train d'être analysées à l'aide de microscopes assistés par ordinateur, à partir des informations fournies par les détecteurs de trajectoires.

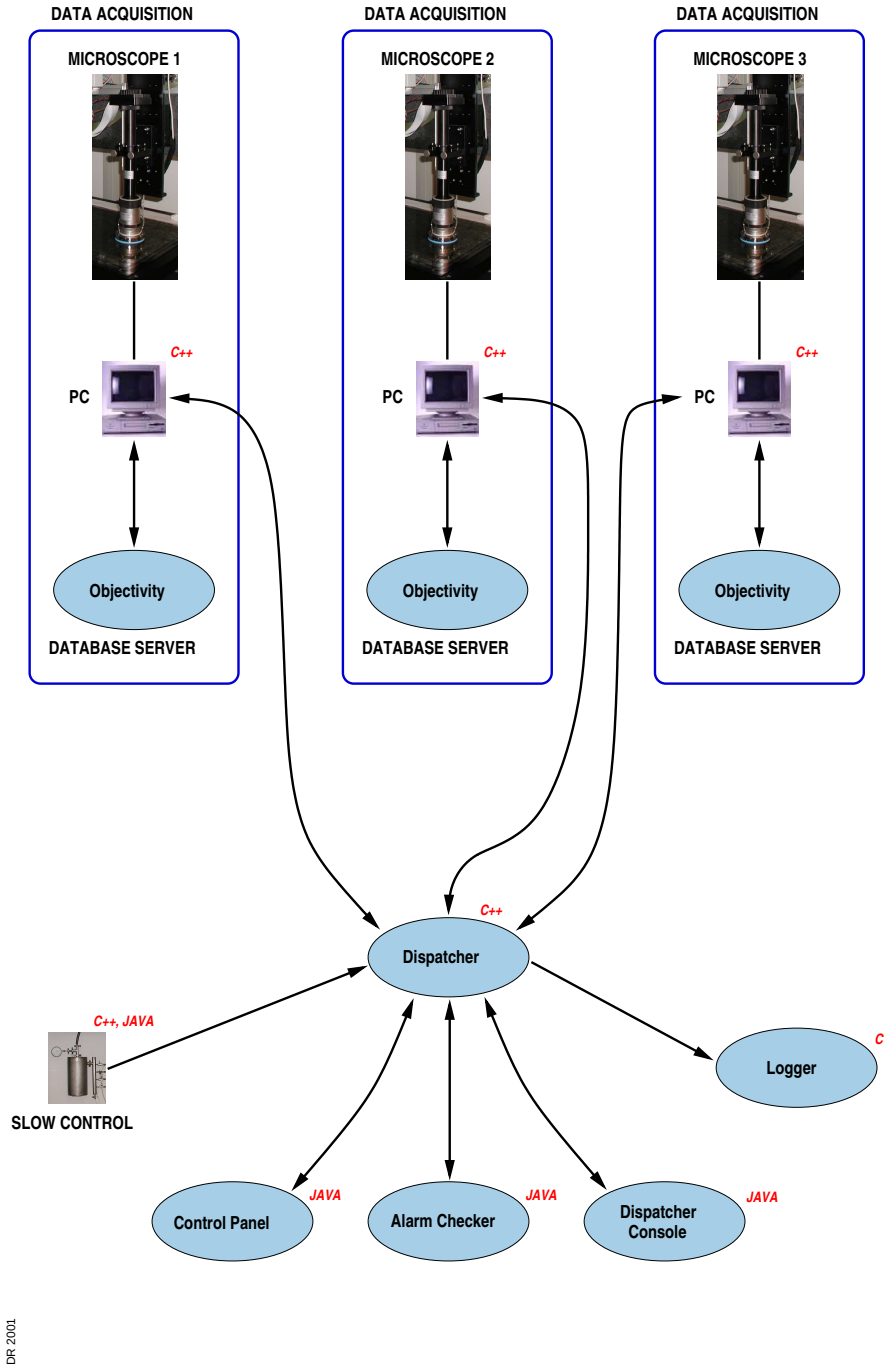


Le laboratoire des microscopes



Le Microscope Lab du CERN, où les émulsions sont scandées, possède trois microscopes connectés à un système d'ordinateurs.
Les microscopes sont équipés d'une caméra CCD pour la scansion. La dimension de la cible de scansion varie, en fonction du niveau de la scansion, de 120 à 2000 microns.
Chaque microscope est connecté à un PC WinNT sur lequel tourne un programme pourvoyant à la lecture et au stockage des données dans une base de données objet (Objectivity). L'analyse des données est ensuite faite au moyen de programmes écrits en C++ et Java, et interactivement avec Mathematica.
Tous les microscopes sont aussi bien connectés en réseau par interface sérielle. Il est donc possible de contrôler le processus entier à partir de n'importe quelle station de travail sur le réseau.
Ce système complexe pourvoit donc au contrôle, à la scansion, à l'acquisition et à l'analyse des données.

Les microscopes et les ordinateurs du Microscope Lab peuvent communiquer grâce à un programme en C++ nommé *Dispatcher* (voir le schéma dans la page suivante). Ce logiciel tourne en permanence sur un serveur et permet l'échange de messages, sous forme de chaînes de caractères, entre entités différentes.
Chaque message est identifié par une balise ; le multiplexage/demultiplexage des messages est fait automatiquement par le Dispatcher lui-même, au moyen d'un mécanisme de souscription (chaque entité, au moment de la connexion au Dispatcher, souscrit aux balises désirées ; ensuite, elle ne recevra que les messages ayant une balise comprise dans la liste de souscription).
Le code pour le Dispatcher fournit des fonctions en C, C++, FORTRAN et Tcl/Tk. En Java, l'échange des données se fait au moyen des méthodes usuelles `java.io.InputStream.read()` et `java.io.OutputStream.write()`, après connexion préalable via un `java.net.Socket`.



CHORUS MICROSCOPE LAB

Mon stage

Le système de scansion des plaques, de contrôle, d'acquisition et d'élaboration des données du Microscope Lab, ainsi que la détection des erreurs éventuelles, sont effectués par un grand nombre de programmes Java. Ces programmes ont été écrits et modifiés successivement par plusieurs générations d'étudiants effectuant un stage d'été au CERN, à l'aide du logiciel CVS (Concurrent Version System). Le projet compte plus de 370 sources `.java`, pour un total d'environ 57000 lignes de code.

Les modules du système de commande des microscopes, du contrôle et de détection des erreurs, étaient réunis à l'intérieur d'un programme unique, nommé *Monitor*. Ce programme souffrait d'un bogue de stabilité ; il allait en crash après un temps qui pouvait varier de quelques minutes à plusieurs semaines.

Le but de mon stage a donc été de corriger et de réécrire ce programme, en y ajoutant d'autres fonctionnalités nécessitées par le projet CHORUS.

En examinant Monitor je me suis aperçu que le problème pouvait être causé, probablement, par une erreur dans l'architecture du programme. En effet, il était composé de plusieurs modules, chacun communiquant avec les autres au moyen des messages échangés par le Dispatcher. N'ayant d'autres liaisons que celles effectuées à travers le Dispatcher, les différents modules n'avaient aucune raison d'être inclus dans le même programme. De plus, une erreur dans le fonctionnement d'un module se serait propagé aussi à travers les autres, en amenant le programme entier au crash. Le problème majeur de ces crashes, en fait, était de comprendre quel pouvait en être la cause et le module qui l'avait engendré. Il s'agissait donc d'une violation de la loi UML de Demeter, qui peut être résumée avec la phrase "Ne pas parler aux inconnus".

J'ai donc décidé de réécrire le Monitor comme une suite de différents programmes distincts, divisant les compétences de chaque module: *Control Panel*, *Dispatcher Console* et *Alarm Checker*. Chaque programme a donc un seul but, et il peut être lancé et arrêté indépendamment des autres. En outre, le module pour la commande du microscope (Control Panel) peut être démarré plusieurs fois pour gérer des microscopes différents. Par contre, le Monitor originel prévoyait à son intérieur trois panneaux de commandes, pour les trois microscopes.

Pour le squelette du projet, je me suis basé partiellement sur le programme

originel et sur les classes déjà existantes, surtout pour ce qui concerne les opérations à bas niveau. Toutefois, presque toute l'architecture et le code du projet ont été développés ex novo.
Les programmes ont été écrits en Java 1.2.2-RC2, au moyen du Standard Development Kit. Pour l'interface graphique, j'ai utilisé les paquetages Swing. Les machines tournaient sous système d'exploitation Linux.
En vue d'un développement successif de ce logiciel au moyen d'un outil de Visual Programming, j'ai utilisé, dans les classes pour lesquelles cela s'avérait utile, le standard JavaBeans.
Etant donné que cette suite de programmes est censée tourner sur les machines du laboratoire pendant un temps très long (plusieurs mois), j'ai prêté une grande attention à sa stabilité. Tous les programmes permettent la configuration d'un grand nombre de paramètres pour pouvoir être adaptés aux besoins de l'utilisateur. Je me suis efforcé, en outre, de développer une interface graphique facile á l'emploi.

La suite de ce document décrit une par une chaque composante du groupe de programmes que j'ai développé. Pour en savoir plus, je suggère de consulter le manuel d'instructions en annexe à ce rapport. Il contient tous les détails de l'utilisation du logiciel, avec la liste des classes.

Chapitre 4

Control Panel

C'est le programme principal, qui permet de commander et de régler les microscopes, toujours au moyen de messages envoyés à travers le Dispatcher. Une session du Control Panel gère un seul microscope. Il n'y a pas de risque d'interférence avec d'autres instances du programme, car les messages de chaque Control Panel et de chaque microscope possèdent des balises différentes : les messages provenant d'un Control Panel ont la balise `PANELxx` (oú `xx` = `01`, `02`, `03`[1]) et les messages provenant du microscope correspondant ont comme balise `MICOSxx`. Chaque balise identifie ainsi d'une façon univoque un seul Control Panel ou microscope.

Un panneau de boutons permet de passer dans les différents états de la scansion. Chaque bouton envoie un message au microscope, en demandant le passage dans l'état correspondant ; le microscope répond ensuite avec un message d'acquittement. Le passage à travers les différents états est donc fait au moyen d'un mécanisme de requête-réponse entre le Control Panel et le microscope. Le diagramme des états est montré à la page 13 du manuel d'instructions en annexe.

Sous le panneau de boutons se trouvent un champ permettant d'envoyer à la main un message au microscope, ainsi qu'une fenêtre enregistrant tous les messages envoyés au cours de cette session.

Un autre panneau permet d'envoyer au microscope un message d'information relatif à la plaque actuellement sous scansion, contenant la valeur

1. Pour la phase de test et les images de ce rapport, j'ai utilisé la balise fictive `PANEL07`.

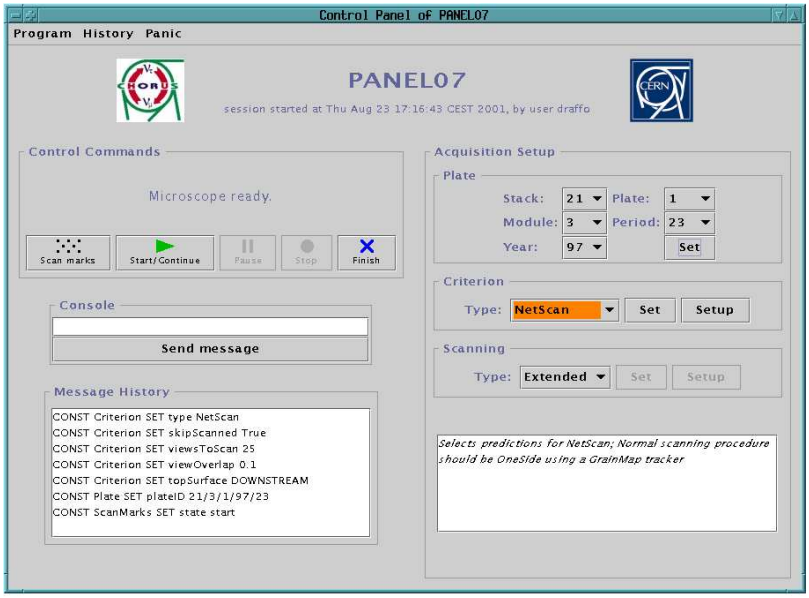


FIG. 4.1 – *Le panneau principal.*

Stack/Module/Plate/Year/Period.
Pour ces valeurs, des contraintes doivent être respectées: pour un *Stack* donné, il y a un choix limité pour la valeur de *Module* ; la valeur de *Year* dépend de la valeur sélectionnée pour *Period*, et tous les deux dépendent de la *Plate* sélectionnée à cet instant là. Les éléments des menus à cascade sont donc créés à la volée.
En outre, pour des changements prévus dans le futur du projet CHORUS, ces contraintes ne doivent pas être fixes, mais sont censées changer avec le temps. Il fallait donc un moyen facile pour configurer les contraintes sans avoir à modifier et à recompiler les sources du programme.
J'ai donc développé un système qui prévoit que les contraintes sont écrites dans un fichier de texte ASCII, selon un pseudo-langage de programmation que j'ai inventé pour l'occasion. On peut voir un exemple d'un tel fichier de configuration á la page 24 du manuel en annexe.

Enfin, au moyen des autres panneaux du Control Panel, on peut spécifier d'autres réglages pour la scansion. Chaque panneau contient un menu à cascade depuis lequel on peut sélectionner le type de réglage. Une fenêtre *Setup* (figure 4.2) permet aussi de préciser la valeur de toute une suite de

paramètres liés à chaque type de réglage.
Un message est envoyé au microscope, relatif à chaque paramètre ; un message est aussi envoyé quand on veut confirmer la sélection du type de réglage. Il est possible de caractériser un certain type de réglage pour qu'il envoie automatiquement, en même temps que le message de confirmation du type de réglage, tout le bloc des messages avec les valeurs des paramètres ; quand cette option est activée, le type correspondant est peint en couleur orange (pour exemple, le type "NetScan" en figure 4.1).
Des messages d'aide sont affichés, soit en bas de la fenêtre Setup quand on glisse avec la souris sur chaque paramètre, soit en bas du Control Panel quand on sélectionne un type de réglage.

Toujours en vue des changements prévus dans CHORUS, toutes ces composantes du Control Panel - types de réglage, fenêtre Setup, paramètres, valeurs par défaut, envoi d'un bloc de messages - ne sont pas prédéfinies dans le programme, mais sont créées dynamiquement à partir des instructions contenues dans un fichier de texte, lu au démarrage. Le format de ce fichier était déjà établi auparavant, étant utilisé pour la configuration des microscopes ; je me suis occupé d'implanter le parser en Java pour ce programme. Á la page 19 du manuel en annexe est montré le début d'un tel fichier; c'est ce dernier qui détermine la configuration montrée dans toutes le images du Control Panel de ce chapitre.

Le Control Panel est donc complètement configurable à bas niveau, dans le but de l'adapter à ses propres exigences, sans devoir modifier et recompiler les sources du programme. Il peut être modifié aisément, même par des personnes sans expérience en Java ou en programmation. Il s'agit d'une des exigences plus importantes du programme.

Un panneau avec un joystick (figure 4.3) permet aussi le mouvement manuel du microscope sur le tableau. Ce panneau est activé et désactivé automatiquement au moyen d'un message provenant du microscope.

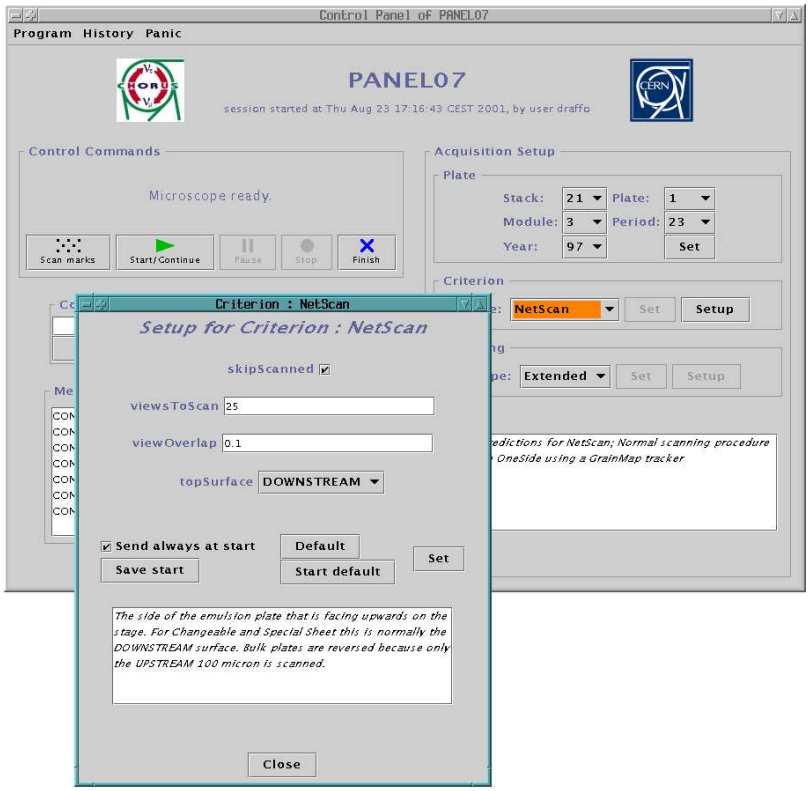


FIG. 4.2 – *La fenêtre Setup pour le type "NetScan".*

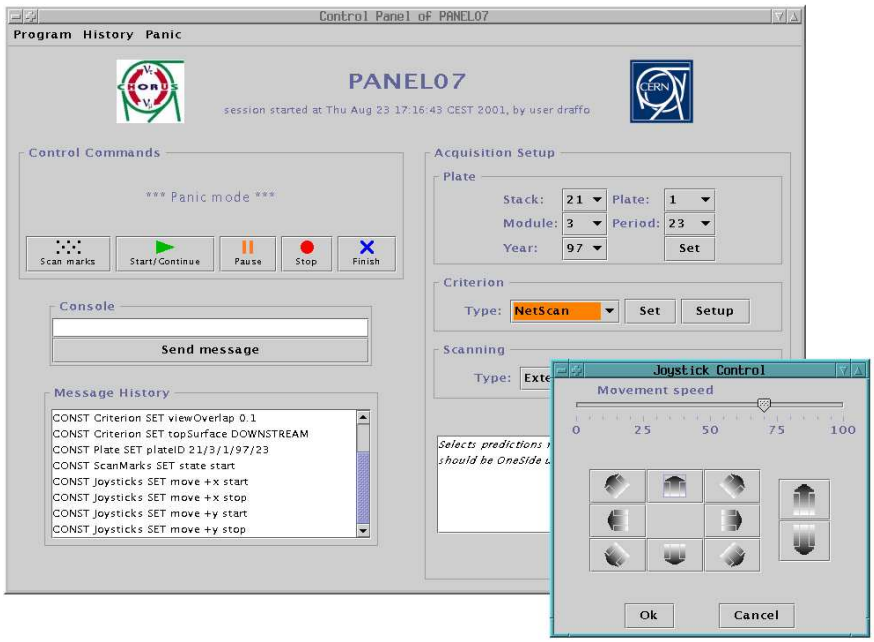


FIG. 4.3 – *Le joystick.*

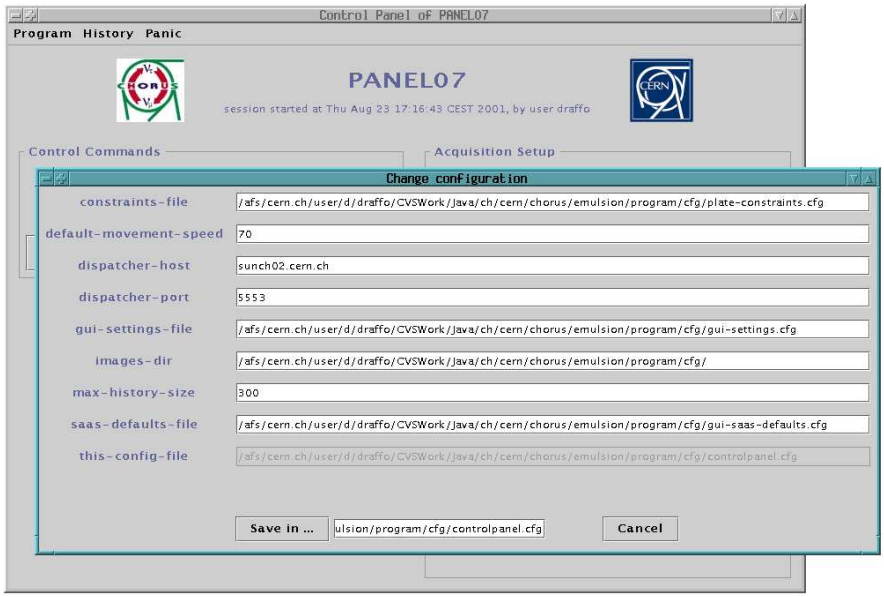


FIG. 4.4 – *On peut configurer de nombreuses options du programme.*

Chapitre 5

Dispatcher Console

Il permet d'intercepter et de visualiser tous les messages ayant des balises au choix passant à travers le Dispatcher.
Bien que non nécessaire, il est très utile au déboguage, afin de détecter les échanges de messages parmi les différents programmes qui tournent dans les machines du laboratoire.

FIG. 5.1 – *Le panneau principal du programme.*

Chapitre 6

Alarm Checker

Ce module s'occupe de la détection des erreurs.
Il vérifie à intervalles réguliers (choisis par l'utilisateur) s'il n'y a pas de message provenant du Dispatcher; l'absence de message peut en effet indiquer des problèmes de nature variée.
Il peut aussi intercepter le démarrage ou la terminaison, par n'importe quel utilisateur du réseau, d'une copie du Control Panel. Cela se fait en écoutant le Dispatcher pour une certaine balise, engendrée automatiquement par le Dispatcher lui-même quand une connexion depuis un client est effectué ou est coupée.
Enfin, il averti l'utilisateur de la fin de la scansion d'une plaque et donc de la nécessité de la changer; cela est aussi perçu par un message spécial que le microscope envoie au Dispatcher.
De plus, le projet originel prévoyait que l'Alarm Checker soit connecté au module *Slow Control* pour le contrôle des valeurs de température et pression dans le laboratoire, avec le déclenchement d'une alarme si les valeurs dépassaient un certain seuil. J'ai prévu cette possibilité dans mon programme, mais elle n'a pas encore été incluse à cause de problèmes avec la connexion des dispositifs hardware.

Si un des événements listés ci-dessus se produit (figure 6.1), le programme déclenche une alarme et la visualise dans la fenêtre du panneau principal; il peut aussi avertir l'utilisateur (ce qui est le but principal du programme) au moyen d'un message qui s'affiche à l'écran et/ou par email. Pour cette caractéristique, j'ai utilisé le JavaMail API (paquetage `javax.mail`) qui permet de créer et d'envoyer du courrier électronique.

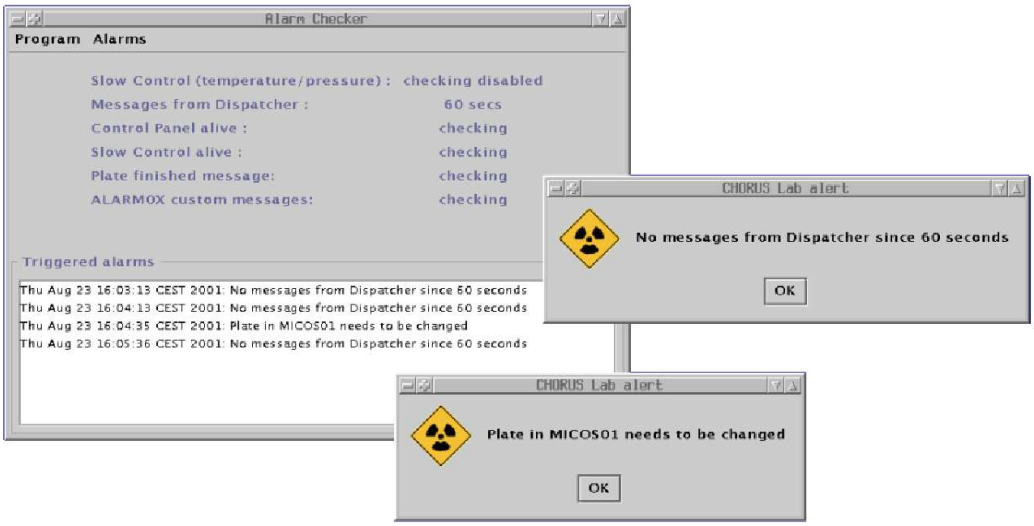


FIG. 6.1 – *Des alarmes ont été déclenchés.*

Par ce même moyen, il est aussi possible d'avertir l'utilisateur par SMS (Message Court), en s'appuyant sur une caractéristique du réseau mobile interne du CERN.

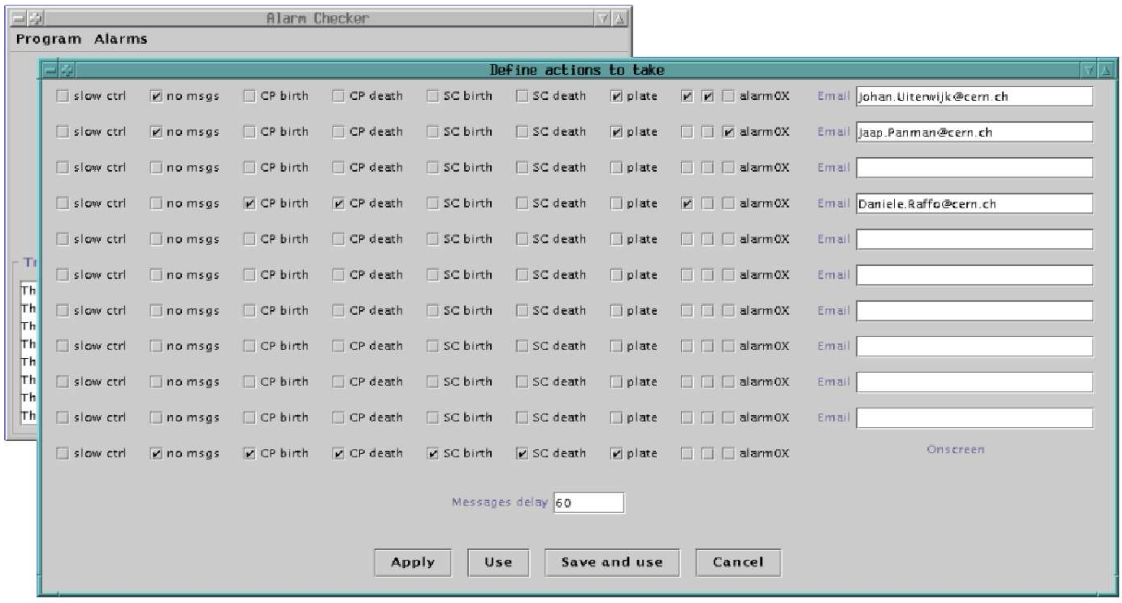


FIG. 6.2 – *La choix des actions à effectuer en cas d'alarme.*

Chapitre 7

Conclusions

Ce stage m'a donné l'opportunité de contribuer à un projet de recherche scientifique de haut niveau. J'ai aussi eu pour la première fois la possibilité de travailler sur un projet informatique de grandes dimensions, développé par plusieurs personnes. Pour toutes ces raisons, ce stage a été une expérience très précieuse.
Le système de contrôle que j'ai développé est complet et en état de marche. J'ai donc pu voir mon programme achevé et prêt à l'utilisation dans le cadre du projet CHORUS ; ce qui est très satisfaisant d'un point de vue personnel. Bien sûr, on pourra par la suite y ajouter d'autres fonctionnalités.
Au moyen de ce stage, j'ai pu étendre mes connaissances de Java, particulièrement dans les domaines de la programmation multithread, du paquetage JavaMail et du développement de l'interface utilisateur.
J'ai énormément apprécié le travail dans un milieu international tel celui du CERN, stimulant autant du point de vue intellectuel que du point de vue humain. Mon superviseur, M. Panman, m'a en outre laissé une grande liberté d'organisation du travail, me permettant aussi de mener à bien ma mission selon mes horaires ; ce qui m'a permis ainsi de tirer le maximum de profit de ce stage.