NOTES TECHNIQUES

SCIENCES DE LA TERRE

GÉOLOGIE-GÉOPHYSIQUE

N° 33

2005

Station accélérométrique du barrage ÉNERCAL de Yaté

Système d'alerte
Gestion des données

Pierre LEBELLEGARD
François MASIA
Jean-Michel DEVAUX

IRD
Institut de recherche pour le développement

NOTES TECHNIQUES

SCIENCES DE LA TERRE

GÉOLOGIE-GÉOPHYSIQUE

N° 33

2005

Station accélérométrique du barrage ÉNERCAL de Yaté

Système d'alerte
Gestion des données

Pierre LEBELLEGARD
François MASIA
Jean-Michel DEVAUX

IRD
Institut de recherche
pour le développement

/Lebellegard, P.
/Masia, F.
/Devaux, J.-M.

Station accélérométrique du barrage Énercal de Yaté. Système d'alerte, gestion des données.
Nouméa : IRD. décembre 2005. p. 50
Notes tech. : Sci. Terre ; Géol.-Géophys. ; 33

SISMOLOGIE ; SURVEILLANCE ; RISQUE NATUREL ; LOGICIEL ; ACQUISITION DE DONNEES ; ARCHIVES / NOUVELLE CALEDONIE ; PROVINCE SUD ; YATÉ

Imprimé par la Service de Reprographie
Centre IRD de Nouméa
Décembre 2005

Station accélérométrique du barrage ENERCAL de Yaté

Système d'alerte
Gestion des données

Pierre Lebellegard
François Masia
Jean-Michel Devaux

Table des matières

Annexes

1 – Introduction

L'IRD et la société Enercal ont signé une convention pour l'installation d'un accéléromètre au barrage hydroélectrique de Yaté, afin de permettre une surveillance en temps réel de l'ouvrage. L'IRD ayant à charge de gérer les données continues et de fournir un signal d'alerte à la centrale de surveillance en fonction des seuils d'accélération détectés :

« L'ordinateur contrôlera en permanence, par dépassement de seuils prédéfinis, les accélérations du sol et enverra des messages d'alerte de différents niveaux selon l'amplitude du signal aux services responsables de la sécurité du barrage. Celui-ci aura pour mission de mettre en oeuvre une inspection du barrage en fonction des accélérations enregistrées.

Parallèlement, cet ordinateur enregistrera en continu le signal sur une mémoire de masse (DDA), qui sera traité dans une optique scientifique par le centre IRD de Nouméa. »

La présente note technique a pour objet de présenter les différents aspects matériels et logiciels du système mis en place, et les différentes procédures mises en oeuvre pour la gestion des alertes et des données. Enfin, sont détaillées les procédures à mettre en oeuvre — éventuellement par un opérateur novice — en cas de « plantage » du système d'acquisition, ainsi que pour la gravure périodique des DVD de données.

Contacts :

- Pierre Lebellegard, centre IRD de Nouméa, 26.07.70 ou 91.80.47 (Pierre.Lebellegard@ird.nc);
- Philippe Nething, Enercal, 25.02.72 (p.nething@enercal.nc);
- Alain Manauté, Enercal (à Yaté) (pour la carte PCI 1760), 46.41.03;

2 – Architecture matérielle

Le système est composé d'un accéléromètre Episensor couplé à une station Titan 3DG développée par la société Agecodagis (http://www.agecodagis.com). Cette dernière est reliée à un PC linux (la version installée, à partir du DVD d'origine, est la SuSe 9.0) à travers une interface Keyspan qui convertit le port série en sortie de la Titan 3DG en USB à l'entrée de la station linux. C'est la solution retenue par Agecodagis pour éviter les pertes de données qui se produisent lorsque l'on relie directement la Titan 3DG au port série de la station linux. La station linux pilote d'une part un graveur de DVD, et d'autre part une carte PCI 1760 de huit voies (huit relais), reliée à un ensemble de 7 LEDs et un buzzer :


Episensor
Titan 3DG
*
Station linux
Graveur DVD
Carte PCI 1760 et relais
LEDs et buzzer (test)
Episensor
Keyspan (*)
LEDs et buzzer
Titan 3DG
Relais

3 – Architecture logicielle

3.1 – Principes généraux

Il y a trois ensembles logiciels implantés sur cet ensemble (quatre si l'on compte la gravure de DVD) :

- le logiciel qui pilote l'acquisition, Irae, développé par la société Agecodagis ;
- le logiciel qui pilote la carte relais PCI 1760 ; il s'agit de l'ensemble générique comedi (pour linux), pour **CO**ntrol and **ME**asurement **D**evice **I**nterface ;
- enfin, le logiciel qui effectue le traitement des données Titan de manière à fournir un dispositif d'alerte en fonction des critères retenus.

Nous avons implanté le logiciel Irae et l'ensemble comedi (plus spécialement le driver pci1760), et nous avons développé (en C) les programmes de traitement et d'archivage des données fournies par l'ensemble EpiSensor+Titan3DG.

Ainsi, l'ensemble qui fait l'objet de cette note technique :

- procède à l'acquisition des données accélérométriques ;
- effectue le traitement de ces données pour effectuer une classification des accélérations mesurées et en déduire un niveau d'accélération (an sens intensité de Mercalli) ;
- en fonction des accélérations détectées, envoie via la carte PCI 1760 une information d'alerte à la centrale de surveillance : il s'agit du niveau d'accélération courante sur 7 LEDs + une alerte sonore ;
- range les données une fois traitées et permet leur archivage sur DVD.

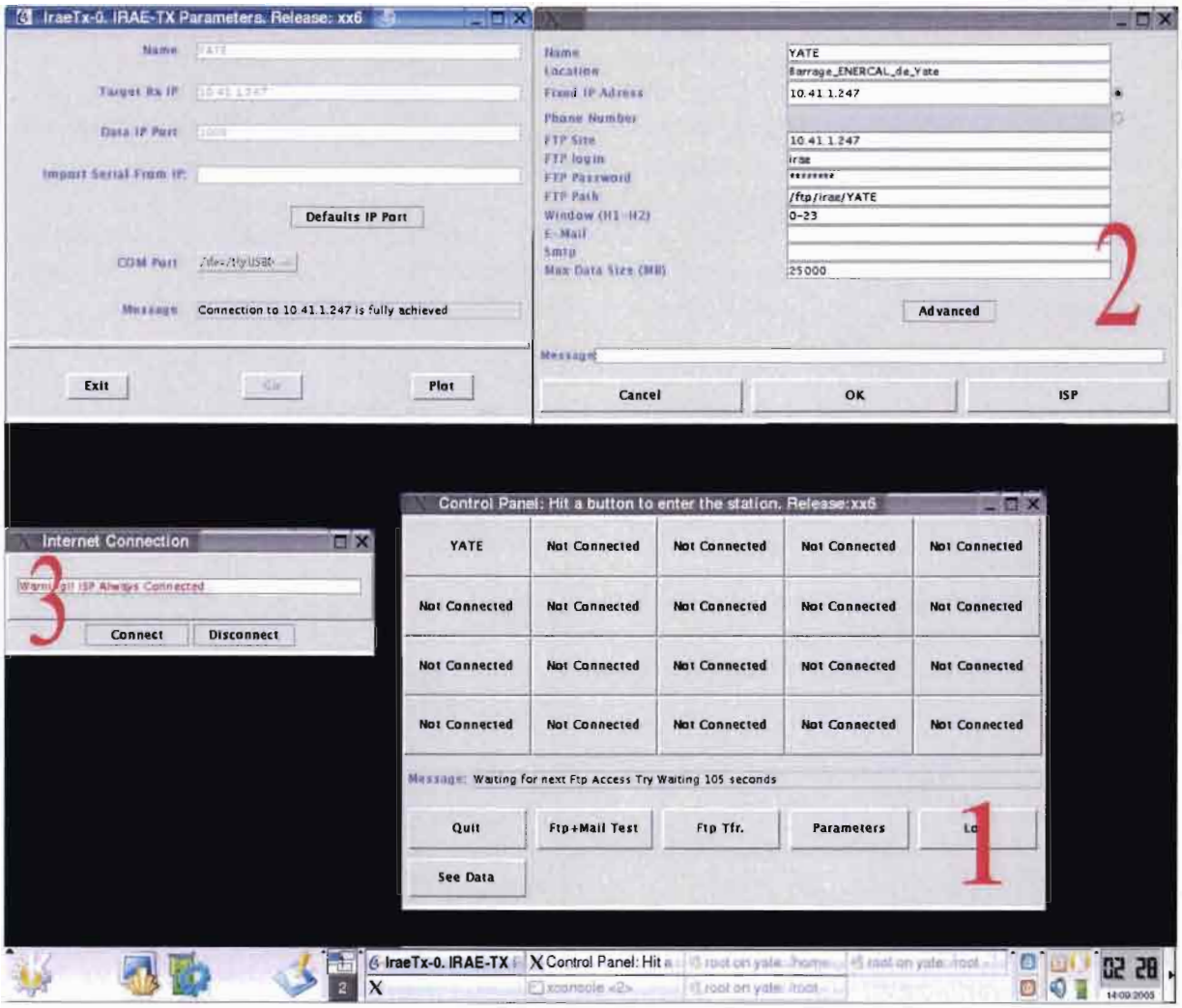
L'ensemble capteur+acquisition est géré par les logiciels Irae développés par la société Agecodagis. L'annexe 3 détaille l'implantation des logiciels Irae sous Linux. Les données numérisées sont stockées sur le disque dans un répertoire ftp ~irae/IraeRx/ftp, ~irae étant le répertoire par défaut du pseudo-utilisateur irae, soit /home/geosci/irae. Le répertoire ftp contient donc un sous-répertoire par jour, nommé selon la convention aaaammjj, qui contient lui-même un sous-répertoire horaire hh, où sont rangés les fichiers de données par tranche de deux minutes. La syntaxe générale pour accéder à un fichier de données est donc : /home/geosci/irae/IraeRx/ftp/aaaammjj/hh/mm. Par exemple, les données du 13 décembre 2005, à 09 heures 03 minutes (donc dans la tranche 02 à 04) sont enregistrées dans /home/geosci/irae/IraeRx/ftp/20051213/09/02.dat…

3.2 – Les logiciels Irae

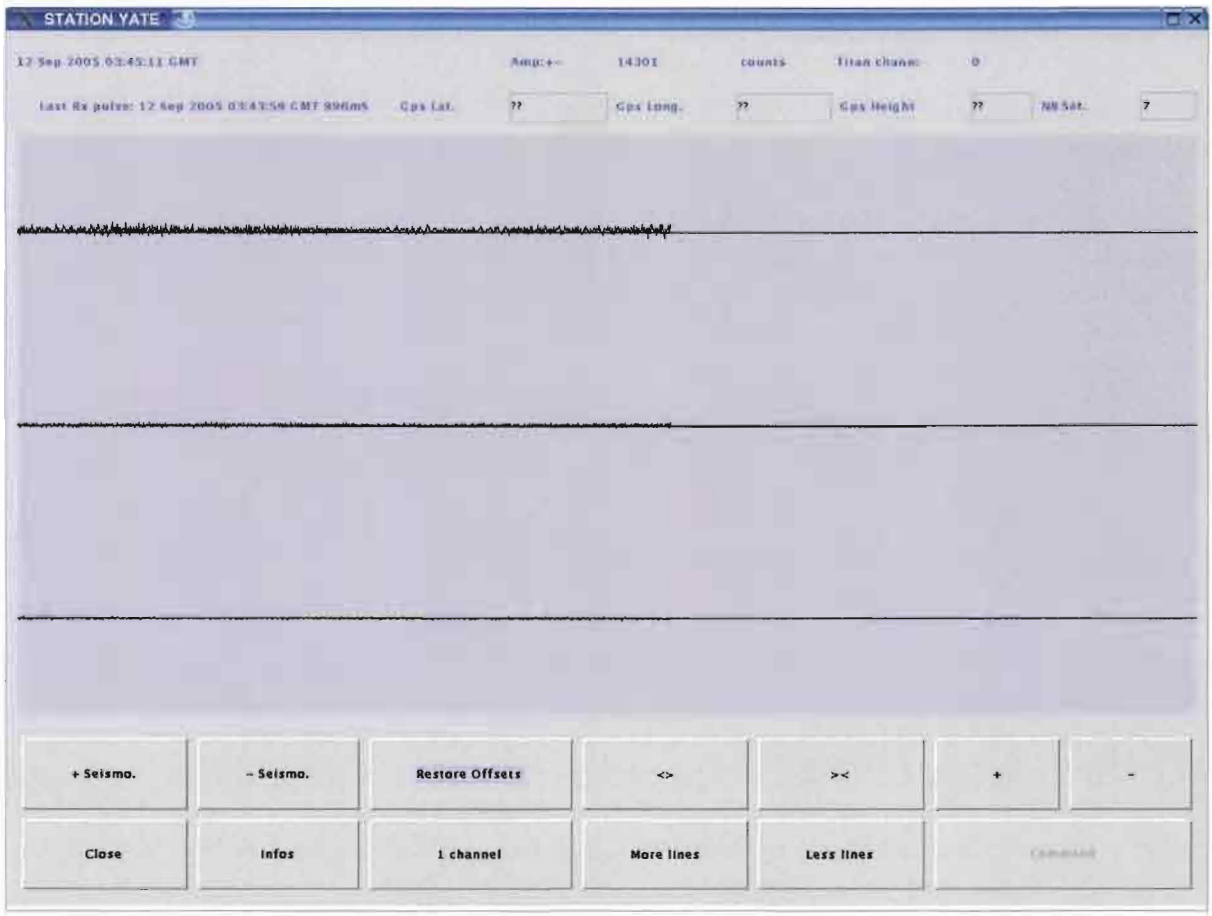
On ne rentrera pas ici dans le détail de l'implantation du logiciel Irae ; en effet, toute la documentation correspondante est disponible sur le site d'Agecodagis (http://www.agecodagis.com). Pour mémoire, on trouvera en annexe 3 la notice d'installation sous linux. Il est à noter toutefois que la transmission des données se fait en mode série à travers un convertisseur RS 232-USB, ceci afin d'éviter une perte de données qui se produit lorsque l'on utilise tel quel le port RS 232. On utilise donc le port USB et un boîtier convertisseur appelé Keyspan, automatiquement reconnu par linux, en tout cas par SuSe 9.0.

Avant de poursuivre, il est fondamental de s'assurer du bon fonctionnement des ports USB et du boîtier Keyspan.

Irae est lancé automatiquement au démarrage du système (avant même le login) ; ne pas tenir compte des éventuelles fenêtres qui s'affichent en même temps que celle de login, et se connecter sous le login root. Une fois l'environnement graphique chargé, on obtient les fenêtres suivantes :



Cliquer sur le bouton « plot » de la fenêtre Irae. Ne pas toucher à la fenêtre 1, cliquer sur « OK » dans la fenêtre 2, et sur « connect » dans la fenêtre 3. On doit obtenir ceci :



Pour obtenir un affichage optimal, cliquer trois fois sur « <> », et trois fois sur « + » ; maximiser également la fenêtre,

3.3 – La gestion de la carte PCI 1760 : comedi

La carte PCI 1760 (cf. architecture matérielle) est une carte qui pilote un ensemble de 8 relais (interrupteurs). En fonction de la valeur sur huit bits envoyée sur cette carte, les relais sont positionnées en conséquence (ouvert/fermé) et peuvent actionner tout dispositif électrique. L'option que nous avons retenue pour coder le signal d'alerte est la suivante : l'intensité de Mercalli courante est codée sur 7 LEDs (de 2 à 7). Si l'intensité courante est de I, alors une LED sur 2 est allumée, et cet affichage est changé au changement de fichier, c'est-à-dire toutes les deux minutes. Cela permet à l'opérateur de s'assurer que le système est toujours opérationnel : une affichage ne changeant pas pendant une période de plus de deux minutes indique un blocage du système, qu'il faut alors redémarrer (cf. plus loin). Le huitième relais est réservé au déclenchement d'une alarme sonore (buzzer), lorsque l'intensité courante dépasse une valeur prédéfinie (IV par défaut).

Ici non plus, on ne détaillera pas la gestion linux de la carte PCI 1760, qui est réalisée à travers l'ensemble comedi. On trouvera toutes les informations nécessaires, et la dernière version de cet ensemble sur le site web http://www.comedi.org, dont voici la page d'introduction :

Introduction

The Comedi project develops open-source drivers, tools, and libraries for data acquisition.

Comedi is a collection of drivers for a variety of common data acquisition plug-in boards. The drivers are implemented as a core Linux kernel module providing common functionality and individual low-level driver modules.

Comedilib is a user-space library that provides a developer-friendly interface to Comedi devices. Included in the Comedilib distribution is documentation, configuration and calibration utilities, and demonstration programs.

Kcomedilib is a Linux kernel module (distributed with Comedi) that provides the same interface as Comedilib in kernel space, suitable for real-time tasks. It is effectively a "kernel library" for using Comedi from real-time tasks.

Features

- Integrated real-time support for most hardware
- High-level library (comedilib)
- Application-level device independence
- Works with Linux 2.0, 2.2, 2.4, 2.6 kernels

Une fois implanté comedi et le driver pci1760, le périphérique associé à la carte est /dev/comedi0. Il est nécessaire de procéder à une réinitialisation en cas de redémarrage du système (cf. annexe 2, initialisation) :

```
#
#       Initialisation de comedi. A faire s'il y a reboot.
#
/sbin/modprobe comedi
/sbin/modprobe pci1760
/usr/sbin/comedi_config /dev/comedi0 pci1760
```

3.4 – Les logiciels développés

Le logiciel de traitement des données, développé en C, est contenu dans le fichier examine.c (sous /home/geosci/src/examine/). S'y ajoutent quelques utilitaires d'initialisation (acquisition, initialisation) et de gestion des répertoires quotidiens pour la gravure des DVD (majsis, ecart).

Le logiciel examine l'arborescence des fichiers irae et pour la tranche courante de deux minutes, en examine les valeurs numériques échantillon par échantillon, pour en déterminer le maximum, et convertir cette valeur en intensité de Mercalli entre I et VIII. Les données de la Titan sont au format constructeur et pour les dépouiller, il est nécessaire d'avoir installé les utilitaires de dépouillement, en particulier cvtit, développés par J.-F. Fels. On trouve ces utilitaires à ftp://renass.u-strasbg.fr/pub/people/fels/. Par commodité, les exécutables ont été installées sous /home/geosci/bin. Dépouiller les valeurs numériques revient à créer un ficher temporaire dans lequel sont écrites les valeurs « en clair » (mais en binaire), fichier que l'on balaie ensuite pour en extraire la valeur maximum. L'appel de cvtit correspondant (cf. source de examine.c (annexe 2) est le suivant : ***cvtit <fichier origine> -bin -shortblk sta=yate | grep \"Output: \" > fichier (binaire) de valeurs numériques"***. Un fichier quotidien de log est mis à jour avec pour chaque ligne, la tranche de deux minutes, et l'intensité mesurée (cad date,

valeur numérique de l'accélération et codage correspondant de I à VIII sur l'échelle de Mercalli). La correspondance intensité/accélération est tirée de *Hough & Martin, 2002, BSSA, 92, 8, 3259-3268* :

Intensité	Accélération en % de g	Accélération en $m.s^{-2}$
I	<0,17% de 1 g	$< 0,017\ m.s^{-2}$
II-III	0,17% - 0,014g	$0,017\ m.s^{-2}$ - $0,137\ m.s^{-2}$
IV	0,014g – 0,039g	$0,137\ m.s^{-2}$ - $0,383\ m.s^{-2}$
V	0,039g – 0,092g	$0,383\ m.s^{-2}$ - $0,903\ m.s^{-2}$
VI	0,092g – 0,18g	$0,903\ m.s^{-2}$ - $1,766\ m.s^{-2}$
VII	0,18g – 0,34g	$1,766\ m.s^{-2}$ - $3,335\ m.s^{-2}$
VIII	0,34g – 0,65g	$3,335\ m.s^{-2}$ - $6,377\ m.s^{-2}$
IX	0,65g – 1,24g	$6,377\ m.s^{-2}$ - $12,164\ m.s^{-2}$
X	> 1,24g	$> 12,164\ m.s^{-2}$

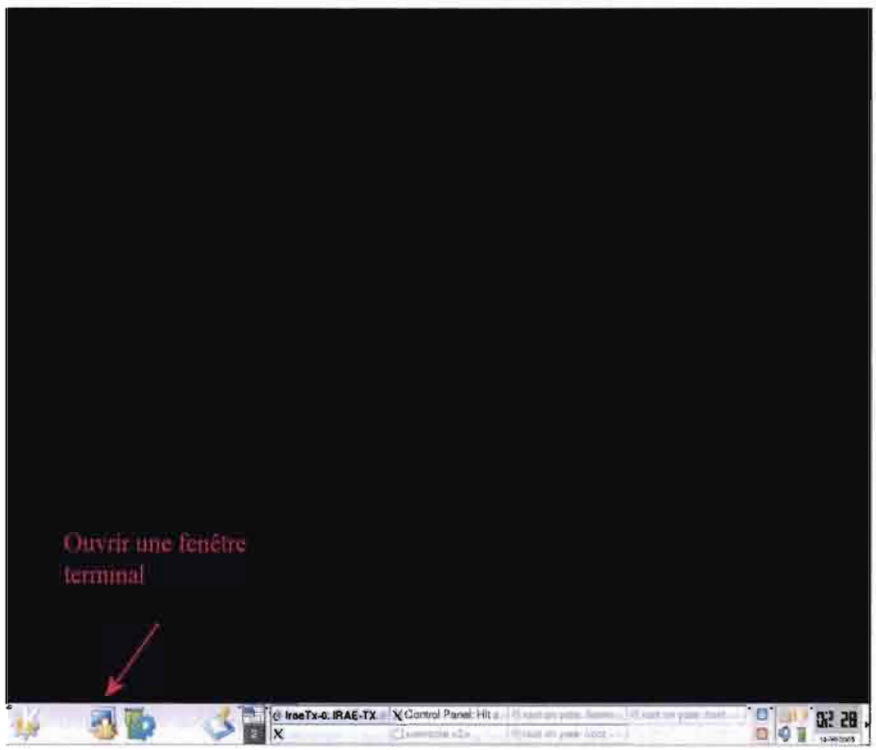
La correspondance valeur numérique/intensité de Mercalli est donnée par le tableau suivant (cf. annexe 2, fichier constants.h) :

Intensité	Valeurs numériques de l'échantillon
I	de 0 à 9.000
II	de 9.001 à 26.000
III	de 26.001 à 51.000
IV	de 51.001 à 102.000
V	de 102.001 à 210.000
VI	de 210.001 à 420.000
VII	de 420.001 à 840.000
VII	> 840.001

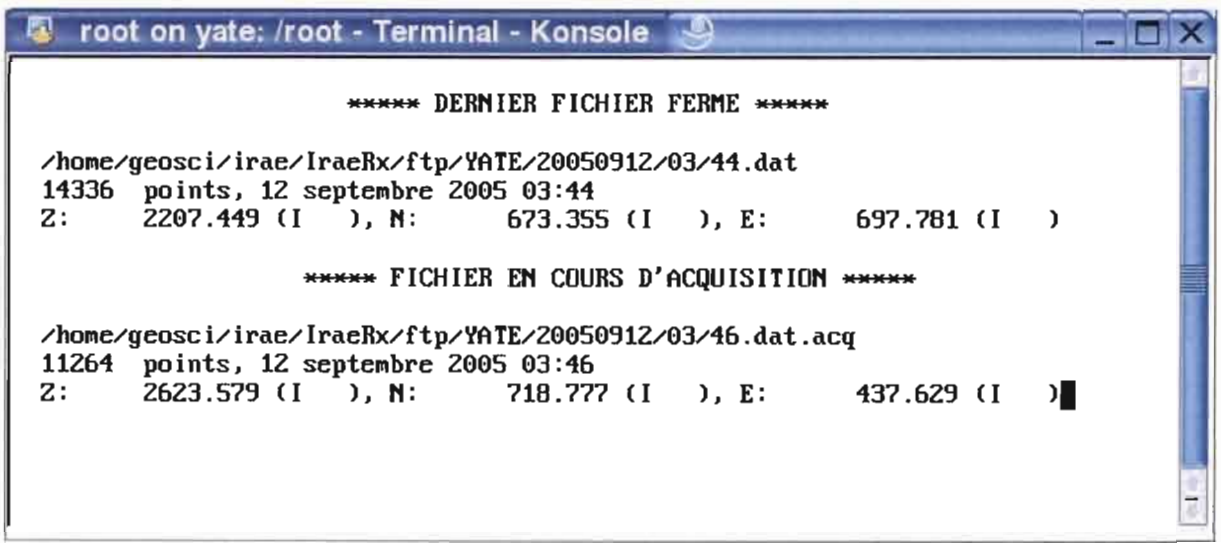
Remarque sur le dépouillement : après avoir récupéré le tableau de valeurs numériques correspondant à une tranche de deux minutes, on calcule tout d'abord la valeur moyenne, que l'on retranche ensuite de l'ensemble des échantillons. La valeur maximale de l'intensité obtenue sur ce fichier est une valeur signée (elle peut être négative). On mémorise également dans le fichier log quelle composante (Z/N/E) a permis d'atteindre cette valeur maximale, généralement il s'agit de la composante verticale (Z). Une fois cette valeur maximale déterminée et convertie en une valeur entre I et VIII, elle est envoyée aux LEDs selon la convention exposée plus haut. La détection étant effectuée sur le fichier courant et mémorisée pour l'affichage du fichier précédent (cf. plus bas), l'affichage d'un état donné dure donc entre deux minutes au minimum et quatre minutes.

Le répertoire balayé par défaut étant ~irae/ftp/IraeRx, on économise du temps de calcul pour les itérations suivantes en déplaçant les fichiers déjà examinés sous le répertoire de gravure /home/geosci/data. De cette manière ne restent présents sous ~irae/IraeRx/ftp que les fichiers les plus récents.

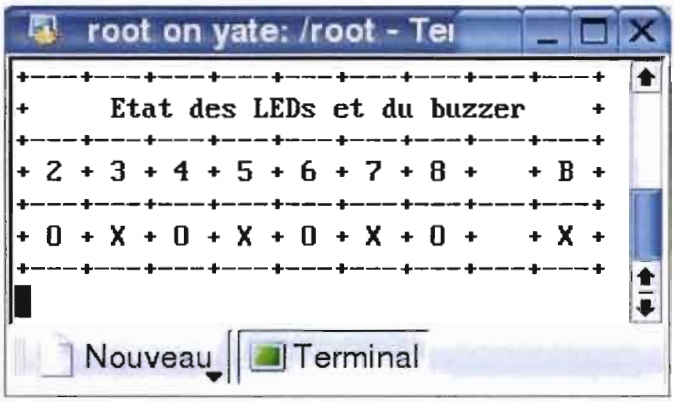
Initialisation : ouvrir deux fenêtres terminal (les fenêtres ont été masquées):



Dans la première fenêtre, taper : ***initialisation***. Cette fenêtre servira ensuite à afficher l'état des LEDs (qui elles, sont affichées à la centrale de surveillance qui n'est normalement pas dans le même local). Cette commande effectue également l'initialisation de la carte PCI 1760 évoquée plus haut. Ensuite, lancer le programme dans la seconde fenêtre en tapant : ***acquisition***. L'affichage doit être le suivant



Pour le contrôle de l'acquisition (actualisé toutes les cinq secondes, il s'agit d'une boucle sans fin sur le programme examine) ; et pour le contrôle de l'état des LEDs:

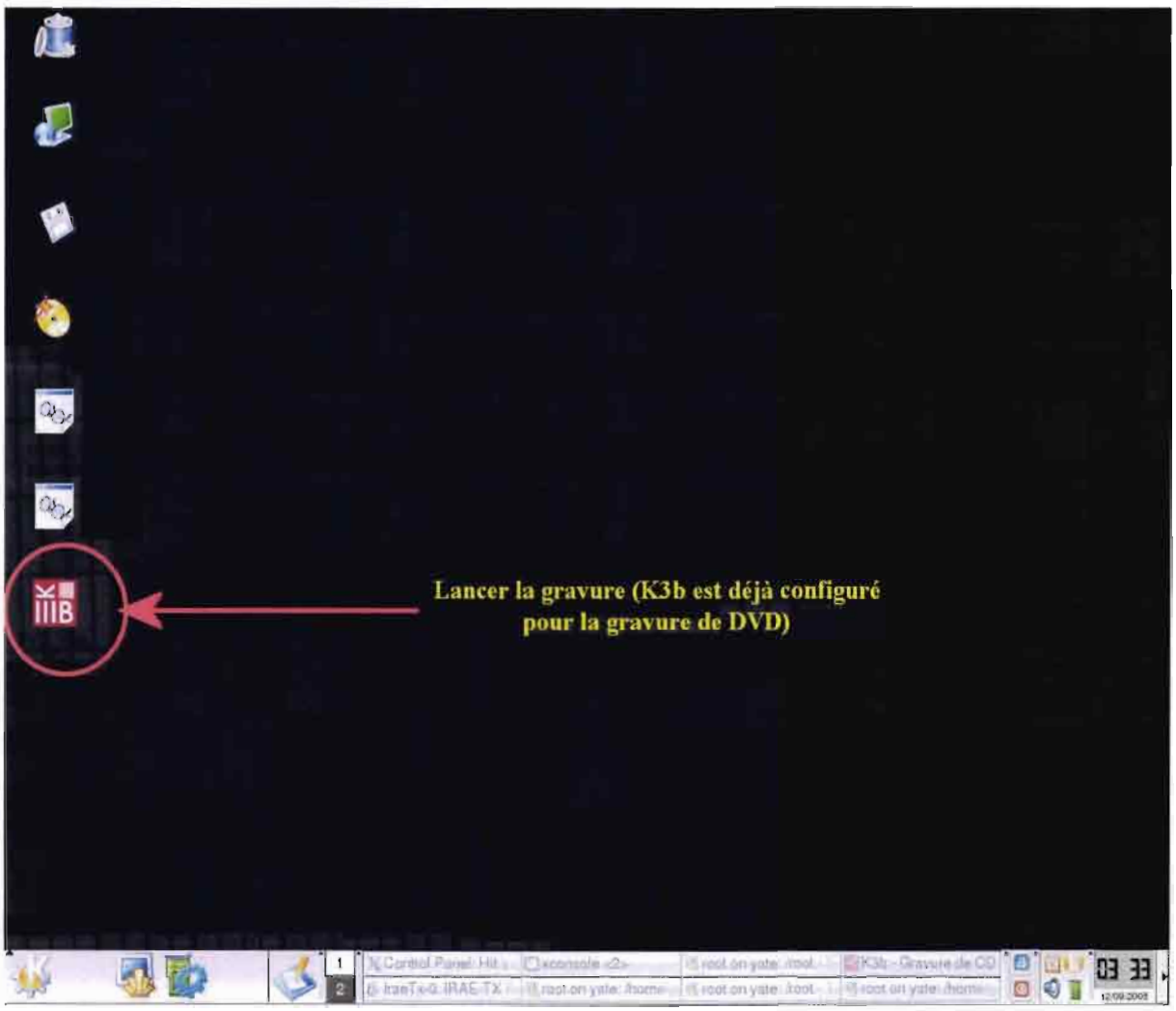


Cette fenêtre est actualisée toutes les deux minutes.

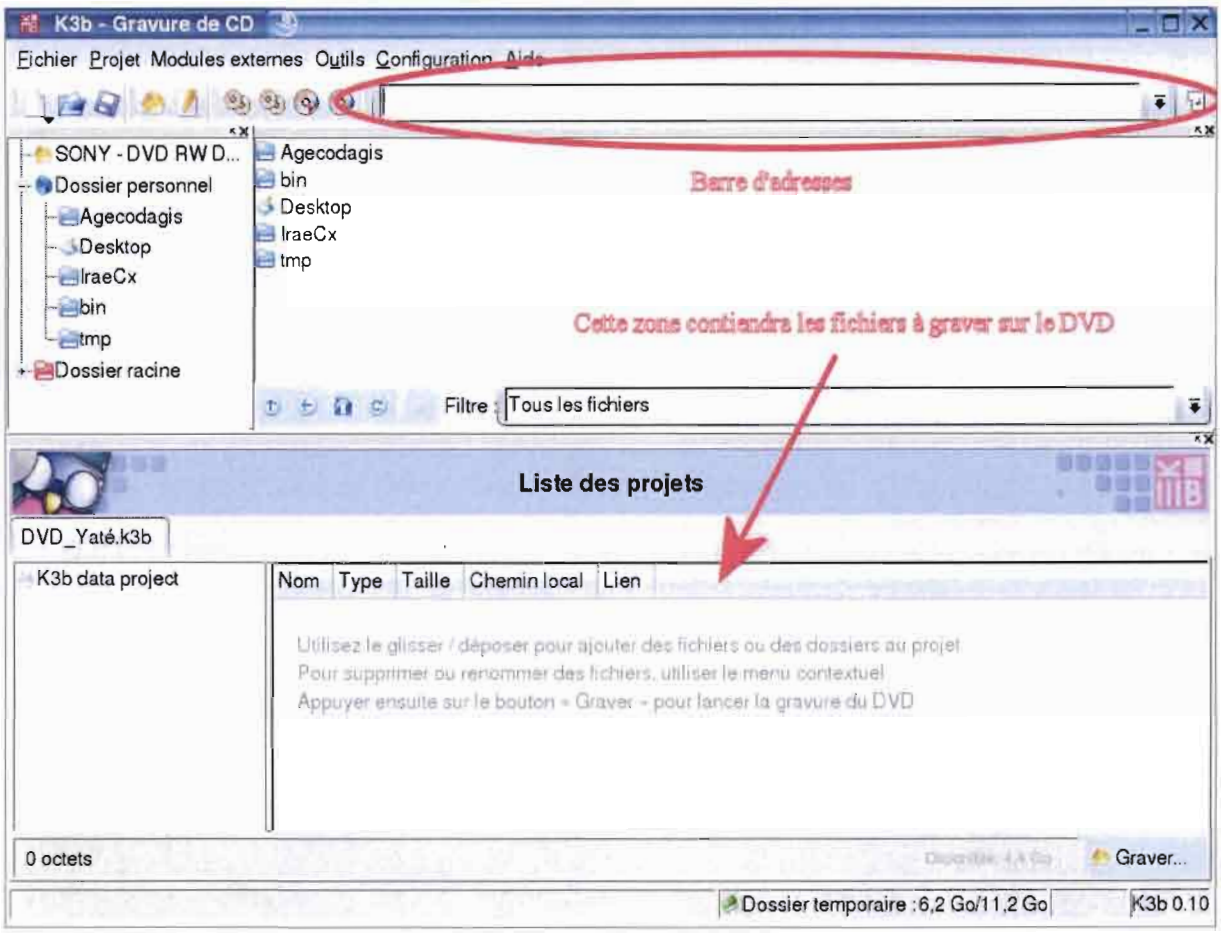
Si l'on souhaite modifier des variables du programmes (chemins, seuils, etc.), tous les paramètres sont regroupés dans le fichier constants.h. Si l'on souhaite modifier le programme lui-même, c'est le fichier examine.c qu'il faut modifier. Dans les deux cas, on met à jour l'exécutable en se plaçant dans /home/geosci/src/examine et en tapant la commande make.

4 – Comment graver un DVD de données

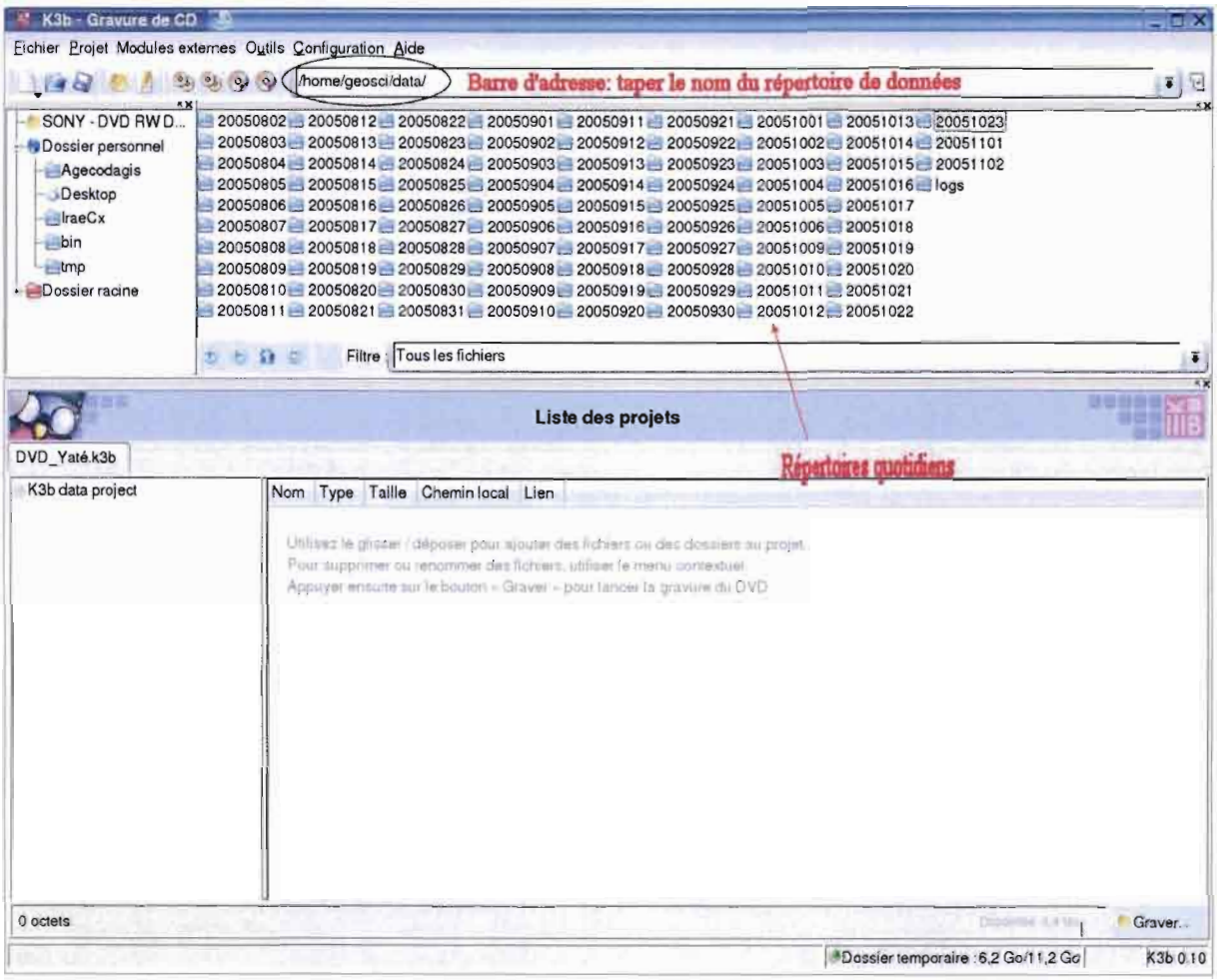
Le programme utilisé pour graver des DVD est l'utilitaire habituel de linux, K3b :



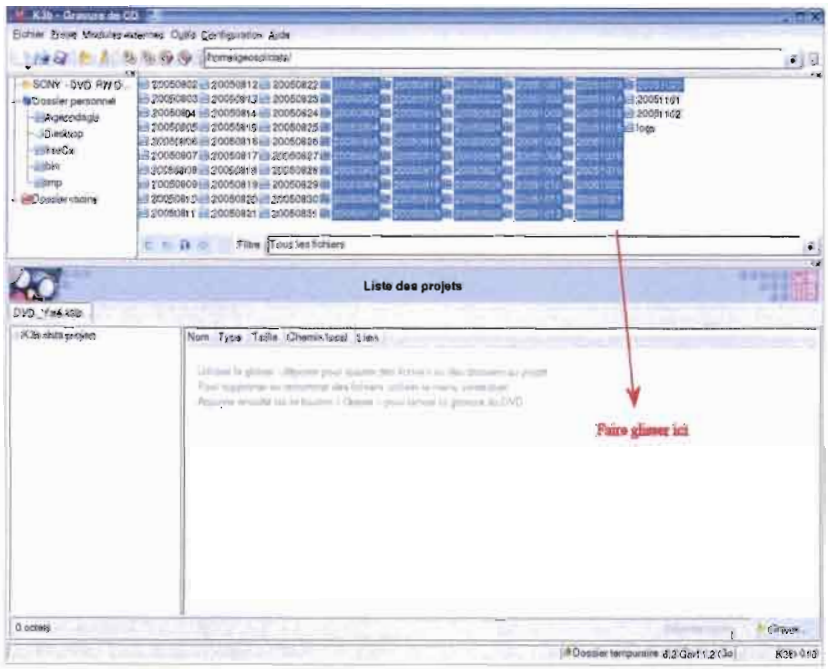
Si l'icône K3b n'est pas présente sur le bureau, lancer K3b en cliquant sur l'icône « K » en bas et à gauche de l'écran, puis « Exécuter » et donner ensuite le nom k3b. On doit obtenir la fenêtre de démarrage de k3b :



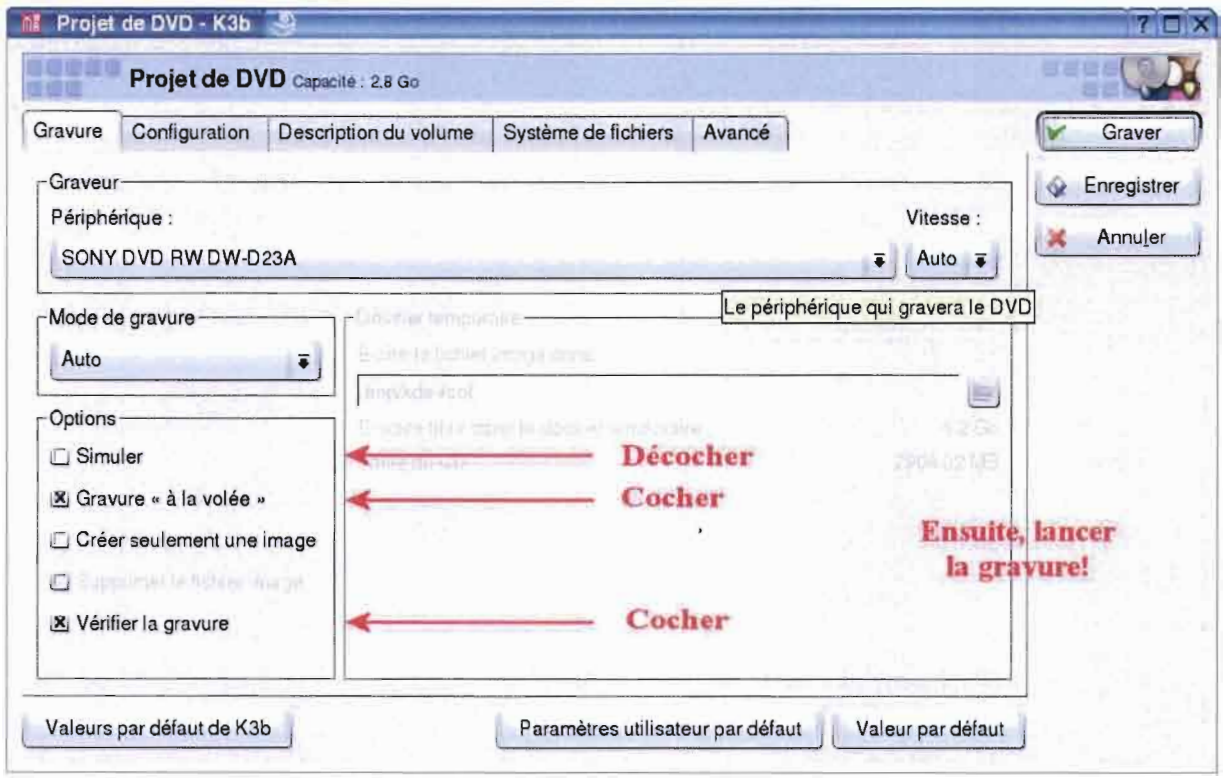
La quantité des données produites quotidiennement limite à deux mois d'enregistrement continu la capacité d'un DVD. Il s'agira donc, tous les deux mois, de graver un DVD, sachant que les fichiers sont conservés six mois sur le disque dur, puis effacés automatiquement (script majsis, voir annexe 2. Ce script est lancé automatiquement chaque jour par la crontab, ce que l'on peut vérifier en tapant ***crontab -l***). Il faut dans un premier temps, faire apparaître les icônes des répertoires quotidiens et pour cela, taper le nom du répertoire de données (/home/geosci/data) dans la barre d'adresses de k3b :



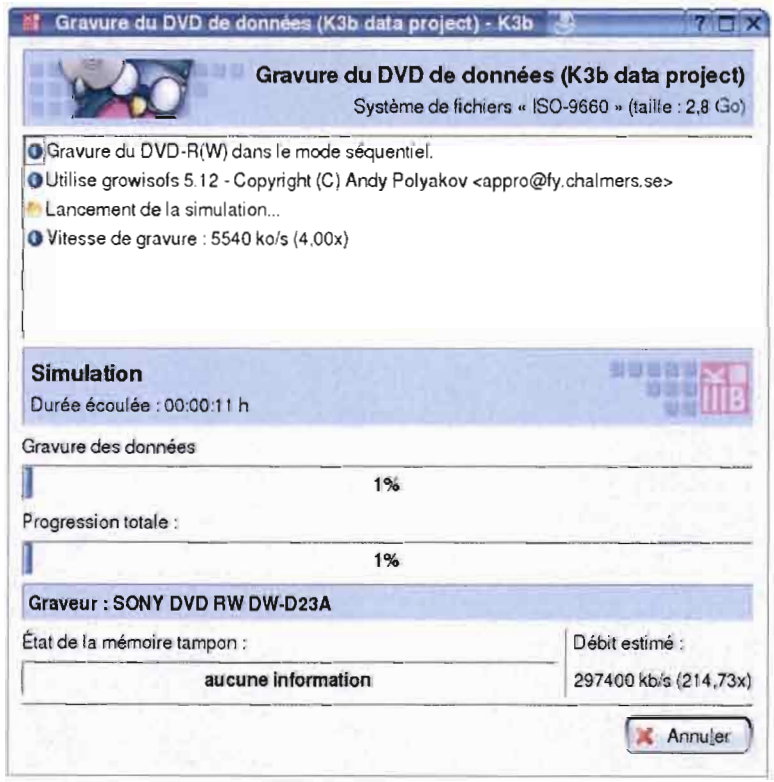
Il faut ensuite faire glisser dans la fenêtre des fichiers à graver tous les dossiers correspondant aux deux mois à graver. Dans l'exemple, il s'agira de septembre et octobre 2005, il faudra donc faire glisser tous les dossiers dont le nom commence par 200509 ou 200510 :



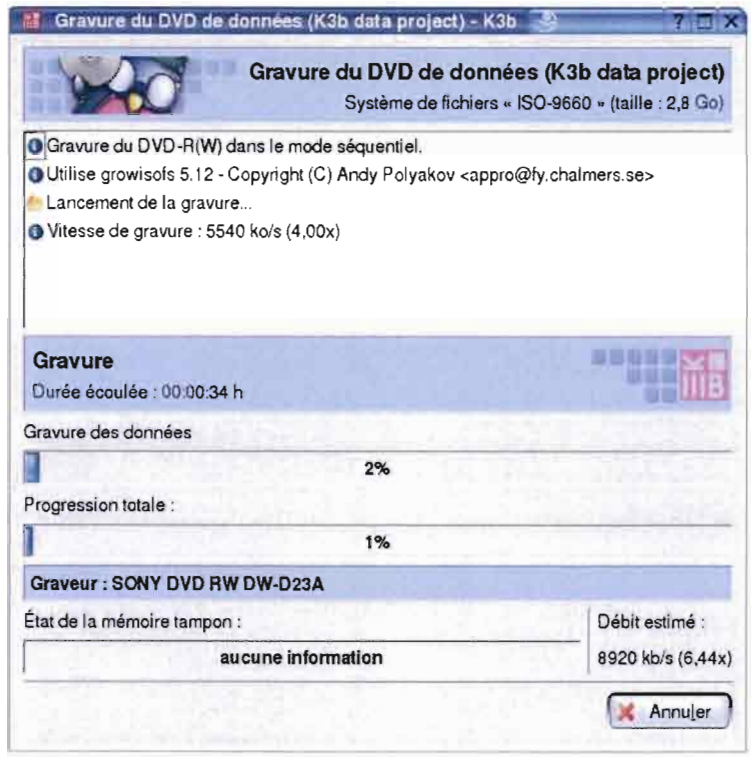
Une fois que l'on a fait glisser les dossiers dans la fenêtre inférieure, lancer la gravure en cliquant sur le bouton « graver » (en bas à droite). On obtient la fenêtre suivante :



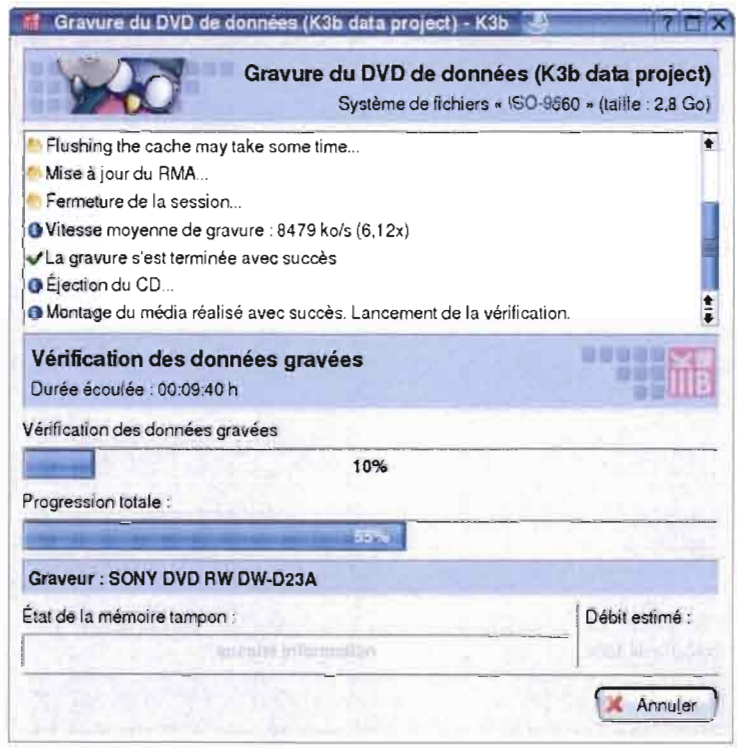
Vérifier les paramètres comme indiqué et lancer la gravure après avoir inséré un DVD vierge. Si l'on a choisi (ce n'est pas indispensable) de cocher « Simuler », la gravure démarre alors par une simulation :



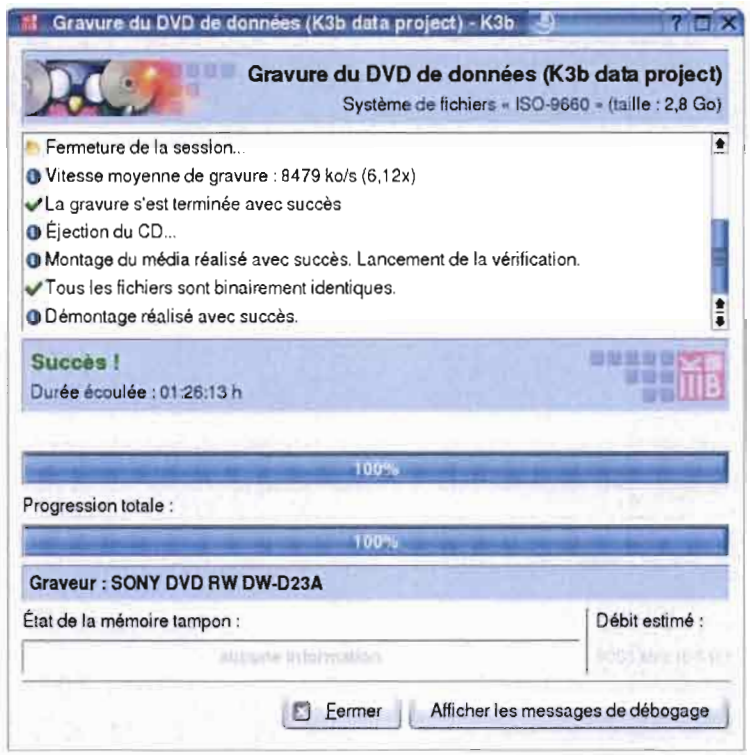
Si la simulation s'est passée sans erreurs, la gravure proprement dite est lancée :



Si la gravure s'est effectuée sans erreurs, k3b va effectuer une vérification du DVD après l'avoir éjecté et rechargé :



Si enfin la vérification est positive, on obtient la fenêtre suivante :



Pour terminer la gravure, il n'y a plus qu'à fermer cette fenêtre et à quitter k3b. On peut ensuite éjecter le DVD. Inscrire au feutre sur le dessus les deux mois gravés sur le DVD.

5 – Comment redémarrer en cas de plantage

Comme expliqué plus haut, si l'affichage des LEDs ne change pas toutes les deux minutes, c'est que le système pour une raison ou une autre est bloqué. Selon le niveau de blocage, il faut alors le redémarrer de manière plus ou moins brutale. Si la souris répond, et que l'on peut ouvrir une fenêtre terminal (cf. § 3.2, première illustration) alors dans cette fenêtre, taper trois fois de suite la commande ***sync***. Ensuite, taper la commande ***halt*** et une fois que le système est arrêté, tout éteindre. Rallumer ensuite, appuyer sur l'interrupteur du PC et redémarrer l'acquisition comme indiqué au § 3.2.

Si la souris ne répond pas, il n'y a pas d'autre solution que d'arrêter le système en appuyant sur l'interrupteur (bouton « power »). Cette solution brutale est à proscrire autant que possible et à n'utiliser qu'en dernière extrémité. Procéder ensuite comme au précédent paragraphe.

En cas de difficultés, contacter l'IRD au 26.07.70 ou au 91.80.47.

6 – Conclusion

Après plusieurs semaines de tests dans les locaux de l'IRD, le système ne s'est bloqué que deux fois, apparemment en raison d'un problème d'acquisition Irae. La procédure indiquée au paragraphe précédent permet de redémarrer sans trop de difficultés. Bien vérifier que l'affichage est régulièrement actualisé (fenêtres alphanumériques et LEDs) pour s'assurer que l'acquisition fonctionne bien.

ANNEXES

Annexe 1 :
Convention ENERCAL / IRD

Convention relative à la station accélérométrique du site du barrage hydro-électrique de Yaté

entre :

L'Institut de Recherche pour le Développement (IRD), Centre de Nouméa, représenté par son Directeur, Monsieur *Christian COLIN*

d'une part,

et :

La Société Néo-Calédonienne d'Energie (Enercal) représentée par

d'autre part,

Il est convenu et arrêté ce qui suit :

Préambule :

Les très forts séismes de l'Arc des Nouvelles-Hébrides et les séismes plus modestes que l'on observe parfois sur la Grande-Terre et dans sa périphérie génèrent des accélérations qui peuvent être préjudiciables à l'édifice hydro-électrique installé sur la rivière Yaté.

La société Néo-Calédonienne d'énergie Enercal et le laboratoire de sismologie de l'IRD de Nouméa (l'Unité de Service des Observatoires, US 127 et l'Unité Mixte de Recherche UMR GéoSciences Azur, UR082), souhaitent participer ensemble à la surveillance sismique du barrage de Yaté.

Article 1 : Objet de la convention

Il est convenu d'installer un accéléromètre à proximité du barrage de Yaté et de surveiller en continu les accélérations du sol.

Article 2 : Apports des parties : infrastructures et équipements

A la charge d'Enercal :

- la réalisation (dalle sismique et petit local) ou la mise à disposition d'un local à proximité du barrage pour recevoir les appareils de mesure. Ce local ne doit pas être perturbé par d'éventuelles machines qui généreraient du bruit sismique susceptible de nuire à l'enregistrement ;

- l'installation d'une alimentation électrique continue 24 V garantissant la continuité de l'enregistrement même en cas de panne de courant prolongée, constituée d'un chargeur 2 x 12 V et de 2 batteries d'environ 70 Ah ;

- d'un capteur accélérométrique, modèle EpiSensor ES-T de chez Kinemetrics (USA) ;

- d'une acquisition numérique Titan3DG (3 voies, 24 bits) de chez Agecodagis (France) ;

- d'une antenne et d'un mini-récepteur GPS qui serviront de base horaire (Agecodagis, France) ;

- d'un ordinateur type PC pour la gestion et l'enregistrement des données (Agecodagis, France) ;

- d'un module d'alimentation continu-continu pour l'ordinateur ;

- d'un modem ou d'une carte N/A pour la sortie de l'information (reste à définir).

A la charge de l'IRD :

- d'un disque dur amovible (DDA) et de son support qui permettra l'enregistrement en continu des données accélérométriques.

L'ordinateur contrôlera en permanence, par dépassement de seuils prédéfinis, les accélérations du sol et enverra des messages d'alerte de différents niveaux selon l'amplitude du signal aux services responsables de la sécurité du barrage. Celui-ci aura pour mission de mettre en oeuvre une inspection du barrage en fonction des accélérations enregistrées.

Parallèlement, cet ordinateur enregistrera en continu le signal sur une mémoire de masse (DDA), qui sera traité dans une optique scientifique par le centre IRD de Nouméa.

Article 3 : Diffusion des données

L'IRD est autorisé à utiliser et à diffuser les données accélérométriques à des fins scientifiques.

Tout commentaire accompagnant l'utilisation et la diffusion de ces données et faisant référence au barrage hydro-électrique de Yaté devra être soumis à l'approbation préalable d'Enercal.

Enercal sera destinataire d'un exemplaire de tout rapport ou publication réalisés à des fins scientifiques à partir de ces données accélérométriques afin de parfaire sa connaissance

du site et pourra, dans le cadre d'études relatives au barrage, les transmettre aux bureaux d'études concernés.

Article 4 : Rôle des différentes parties

Enercal est chargé de la surveillance bi-hebdomadaire de la station qui consiste à la vérification :

- de l'alimentation électrique ;
- du bon fonctionnement des logiciels d'acquisition de données, de réception de l'heure et de l'ordinateur ;
- de l'état d'avancement du remplissage du disque dur amovible (DDA).

Enercal se chargera du remplacement du disque dur amovible plein par un autre vide (fourni par l'IRD) chaque mois environ et de l'expédition du DDA à l'IRD-Nouméa.

En cas de problème technique :

- Enercal tentera d'y remédier selon les compétences et les connaissances du personnel de surveillance et éventuellement avec les conseils téléphoniques de l'IRD ;
- Si le problème persiste, l'IRD, prévenu, interviendra au plus tôt selon ces disponibilités. L'IRD ne pourra intervenir que dans la limite des jours ouvrables.

Les frais de réparation et de remplacement du matériel seront à la charge d'Enercal hormis le DDA qui reste une spécificité de l'IRD.

Les frais de déplacement seront à la charge de l'IRD dans la mesure ou le nombre de déplacements annuels reste limité à quelques interventions. Dans le cas contraire, une participation aux frais de déplacement sera demandée à Enercal après accord des deux parties.

L'IRD aura en charge au moins une visite annuelle de maintenance.

L'IRD aura soin d'instruire le personnel d'Enercal au fonctionnement de la station accélérométrique au moment de l'installation.

Article 5 : Durée de la convention

La présente convention est établie pour une durée d'un an renouvelable par tacite reconduction. Le non renouvellement par l'une ou l'autre des deux parties devra être stipulé par courrier avec un préavis de trois mois.

Article 6 : Personnel impliqué par cette convention au moment de la signature

Philippe Nething	SME	Enercal

Jacques Nething	Yaté	Enercal
Didier Manauté	Yaté	Enercal
Jean-Michel Devaux	Assistant Ingénieur	IRD Nouméa
Jean-Louis Laurent	Technicien	IRD Nouméa
Pierre Lebellegard	Ingénieur de Recherche	IRD Nouméa
Catherine Baldassari	Assistant Ingénieur	IRD Nouméa
Robert Pillet	Chargé de Recherche	IRD Nouméa

Nouméa, le

Pour ENERCAL

Pour l'IRD

Christian COLIN
Directeur du Centre
IRD de Nouméa

Annexe 2 :
Sources des programmes C

Constants.h : Constantes utilisées dans le programme principal (fichier suivant examine.c)

```
/*
 *-------------------------------------------
 *      Longueur maxi des chaines de caracteres
 *-------------------------------------------
 */
#define MAXLEN  1024

/*
 *-----------------------------
 *      Nom de la station Titan
 *-----------------------------
 */
#define STATION "YATE"

/*
 *--------------------
 *      3 composantes
 *--------------------
 */
#define NBCOMPS 3

/*
 *-----------------------------------------------------------
 *      Emplacement des repertoires quotidiens (a l'origine)
 *-----------------------------------------------------------
 */
#define FTPDIR  "/home/geosci/irae/IraeRx/ftp"

/*
 *-----------------------------------------------------
 *      Emplacement des donnees (pour gravure sur DVD)
 *-----------------------------------------------------
 */
#define DATADIR "/home/geosci/data"

/*
 *----------------------------------
 *      Nom du repertoire temporaire
 *----------------------------------
 */
#define TMPDIR  "/tmp"

/*
 *--------------------------------
 *      Nom du fichier temporaire
 *--------------------------------
 */
#define TMPFILE "tagada"

/*
 *-------------------------------------------------------------------------
 *      Intervalle (en secondes) d'examen du fichier en cours d'acquisition
 *-------------------------------------------------------------------------
 */
#define DELAY   5

/*
 *--------------------------
 *      Le fichier d'erreur
 *--------------------------
 */
#define ERR_FILE "/tmp/examine.err"

/*
 *-----------------------------------------------------------------
 *      Le nom du peripherique correspondant a la carte Advantech
 *-----------------------------------------------------------------
 */
#define DEVICE "/dev/comedi0"

/*
 *----------------------------------
 * Pour parametrer la carte Advantech
 *----------------------------------
 */
comedi_t        *dev;
comedi_insn     insn;
int             subdev = 2;
int             chan = 0;

/*
```

```
 *--------------------------------
 *      L'etat individuel des leds
 *--------------------------------
 */
#define LED1    0x01    /* 0000 0001 */
#define LED2    0x02    /* 0000 0010 */
#define LED3    0x04    /* 0000 0100 */
#define LED4    0x08    /* 0000 1000 */
#define LED5    0x10    /* 0001 0000 */
#define LED6    0x20    /* 0010 0000 */
#define LED7    0x40    /* 0100 0000 */
#define BUZZON 0x80     /* 1000 0000 */
#define BUZZOFF         0x7f    /* 0111 1111 */

#define NOIR    0x2a    /* 0101 0101 buzzer eteint */
#define BLANC   0x55    /* 0010 1010 buzzer eteint */

/*
 *---------------------------------------
 *      Seuil de declenchement du buzzer
 *---------------------------------------
 */
#define SEUILBUZZ       4

/*
 *-------------------------------------------------------------------------------
 *      La valeur qui permet de memoriser 1'etat des 7 leds et de la sonnerie
 *-------------------------------------------------------------------------------
 */
int     mask, ancienmask;

typedef int     booleen;

struct
{
        char    mercalli[10];
        float   maxi;
        char    mercalli2[10];
        int     nbpoints;
        char    fichier[MAXLEN];
} comp[NBCOMPS];

int mercalli[8] = {
        0,      9001,   26001,  51001,
        102001, 210001, 420001, 840001
};

char *mercalli_ch[8] = {
        "I", "II", "III", "IV", "V", "VI", "VII", "VIII"
};

char *mercalli_ch2[8] = {
        "1", "2", "3", "4", "5", "6", "7", "8"
};

char
        *jour[] =
        {
                "Dimanche",
                "Lundi",
                "Mardi",
                "Mercredi",
                "Jeudi",
                "Vendredi",
                "Samedi"
        };

char
        *mois[] =
        {
                "janvier",
                "fevrier",
                "mars",
                "avril",
                "mai",
                "juin",
                "juillet",
                "aout",
                "septembre",
                "octobre",
                "novembre",
                "decembre"
        };
```

```
char    veille[MAXLEN];

char    lastopened[MAXLEN];
char    lastclosed[MAXLEN];
booleen lastopenedOK;
booleen lastclosedOK;
```

Examine.c : Programme principal. En fait, celui-ci n'est pas lancé directement, mais au travers du shell script acquisition qui n'est rien d'autre qu'une boucle sans fin lançant l'exécutable /home/geosci/bin/examine

```
#include <stdio.h>
#include <unistd.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <dirent.h>
#include <string.h>
#include <curses.h>
#include <errno.h>
#include <comedilib.h>

#include "constants.h"

/*
 * Message d'erreur dans un fichier d'erreur (/tmp/examine.err)
 *
 */
void erreur(char *message)
{
        char    ch[MAXLEN];
        FILE    *err;

        sprintf(ch, "echo \"----------\" >> %s", ERR_FILE); system(ch);
        sprintf(ch, "date >> %s", ERR_FILE); system(ch);

        err = fopen(ERR_FILE, "a");
        if (err == NULL)
        {
                fprintf(stderr, "%s\n", message);
                return;
        }
        else
        {
                fprintf(err, "%s", message);
                fflush(err);
                fclose(err);
                return;
        }
}

/*
 * Retourne le nombre de jours de l'annee
 */
int dysize(int year)
{
        if (year % 4 != 0)
                        return (365);
        if (year % 400 == 0)
                        return (366);
        if (year % 100 == 0)
                        return (365);
        /*
         * Sinon, annee bissextile
         */
        return (366);
}

/*
 * Jour julien de l'annee (exprimee sur 4 chiffres)
 */
int
qt(int a, int m, int d)
{
        int     i;
        int     quantieme = 0;

        int     nbj[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

        if (dysize(a) == 366)
                nbj[1] = 29;

        for (i = 0; i < m - 1; i++)
                quantieme += nbj[i];
        quantieme += d;

        return (quantieme);
}

/*
 * Affiche l'etat des leds (cad la valeur de mask) sur une fenetre passee en parametre
 * de l'appel (argv[1]). Si pas de parametres, cet etat est affiche sur /dev/console.
```

```
 */
void AfficherLeds(char *device)
{
	FILE *fenetre;
	char ch[MAXLEN];

	fenetre = fopen(device, "w");
	if (fenetre == NULL)
		return;

	sprintf(ch, "clear > %s", device); system(ch);

	fprintf(fenetre, "+---+---+---+---+---+---+---+---+---+\n");
	fprintf(fenetre, "+     Etat des LEDs et du buzzer    +\n");
	fprintf(fenetre, "+---+---+---+---+---+---+---+---+---+\n");
	fprintf(fenetre, "+ 2 + 3 + 4 + 5 + 6 + 7 + 8 +   + B +\n");
	fprintf(fenetre, "+---+---+---+---+---+---+---+---+---+\n");
	fprintf(fenetre, "+ ");
	if ((LED1 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED2 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED3 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED4 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED5 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED6 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");
	if ((LED7 & mask) == 0)
		fprintf(fenetre, "X + ");
	else
		fprintf(fenetre, "O + ");

	fprintf(fenetre, "  + ");

	if ((BUZZON & mask) == 0)
		fprintf(fenetre, "X +");
	else
		fprintf(fenetre, "O +");
	fprintf(fenetre, "\n");
	fprintf(fenetre, "+---+---+---+---+---+---+---+---+---+\n");

}

/*
 * Positionner les huit relais. Les valeurs (ouvert/ferme) sont positionnees par les huit bits
 * correspondants de la valeur mask.
 */
int EcrireRelais(comedi_insn *insn, int mask)
{
	lsampl_t omb[4] =
	{
		0x0,	0x0,	0x1,	0x0
	};
	int	resul;
	char	ch[MAXLEN];

	insn->data	= omb;
	omb[0]		= mask;

	/* Ouverture du peripherique de la carte Advantech */
	dev = comedi_open(DEVICE);
	if (dev == NULL)
	{
		sprintf(ch, "Erreur de comedi_open(\"%s\"): %s\n",
			DEVICE,
			strerror(errno));
		erreur(ch);
		exit(1);
	}
```

```
        resul = comedi_do_insn(dev, insn);

        /* Fermeture du device */
        comedi_close(dev);

        if (resul < 0)
                return (-1);
        else
                return(0);
}

/*
 * Recuperation des infos du fichier de donnees <fichier>
 * On lance cvtit sur ce fichier, et on regarde le listing de l'execution.
 * Comme on fait un cd dans un repertoire temporaire, il faut que le nom
 * du fichier passe en parametre soit le nom complet (path en absolu).
 *
 */
booleen depouille(char *fichier)
{
        int     i, j, resul, nocomposante, status, nbpoints, mercallimax;
        char    ch[MAXLEN];
        char    *pattern;
        char    *pt, *pt2;
        float   *tableau, *ptfloat;
        float   maxi, moyenne;
        struct stat   buf;
        FILE    *desc;

        /* Il faut que le fichier soit de longueur non nulle */
        resul = stat(fichier, &buf);
        if (resul != 0)
                return(FALSE);
        if (buf.st_size == 0)
                return(FALSE);

        resul = chdir(TMPDIR);
        if (resul != 0)
        {
                sprintf (ch, "Impossible d'acceder au repertoire %s.\n", TMPDIR); erreur(ch);
                return(FALSE);
        }

        sprintf(ch, "cvtit %s -bin -shortblk sta=%s | grep \"Output: \" > %s",
                        fichier, STATION, TMPFILE);
        resul = system(ch);
        if (resul != 0)
        {
                sprintf(ch, "Erreur a l'execution de cvtit sur %s.\n", fichier); erreur(ch);
                sprintf(ch, "ls -l %s", fichier); system(ch);
                sprintf(ch, "ls -l %s", TMPFILE); system(ch);
                return(FALSE);
        }

        desc = fopen(TMPFILE, "r");
        if (desc == NULL)
        {
                sprintf(ch, "Impossible d'ouvrir le fichier log.\n"); erreur(ch);
                return(FALSE);
        }
        else
        {
                status = 0;
                for(;;)
                {
                        if (fgets(ch, MAXLEN, desc) == NULL)
                        {
                                fclose(desc);
                                resul = unlink (TMPFILE);
                                if (resul != 0)
                                {
                                        sprintf(ch, "Impossible d'effacer le fichier temporaire.\n");
                                        erreur(ch);
                                        return(FALSE);
                                }
                                break;
                        }
                        pattern = "comp "; pt = strstr(ch, pattern); pt += strlen(pattern);
                                nocomposante = atoi(pt); pt += 2;
                        if ( (nocomposante >= 0) && (nocomposante < NBCOMPS) )
                        {
                                strcpy(comp[nocomposante].fichier, pt);
                                /* Prochain nom de fichier */
```

```
                        pt = strstr(pt, " "); ++pt;
                        /* Nombre de points pour ce fichier */
                        pt = strstr(pt, " "); ++pt;
                        comp[nocomposante].nbpoints = -1;
                        resul = atoi(pt);
                        if (resul <= 0)
                        (
                                sprintf(ch, "Nombre de points incorrect pour %s.\n",
                                                comp[nocomposante].fichier); erreur(ch);
                                status = -1;
                        }
                        else
                        (
                                comp[nocomposante].nbpoints = resul;
                        }
                        /* On remplace le premier blanc du nom par une fin de chaine */
                        pt2 = strstr(comp[nocomposante].fichier, " "); *pt2 = '\0';
                }
                else
                {
                        sprintf(ch, "Numero de composante incorrect.\n", fichier); erreur(ch);
                        status = -1;
                }
        }
}

/*
 * Lecture des fichiers
 */
maxi = 0.0;
for (i=0; i<NBCOMPS; i++)
{
        comp[i].maxi = 0.0;
        /* Allocation du tableau de valeurs pour la composante courante */
        tableau = (float *) malloc(sizeof(float) * comp[i].nbpoints);
        if (tableau == NULL)
        {
                sprintf(ch, "Erreur d'allocation.\n"); erreur(ch);
                return(FALSE);
        }

        sprintf(ch, "%s.data", comp[i].fichier);
        desc = fopen(ch, "r");
        if (desc == NULL)
        {
                sprintf(ch, "Impossible de lire %s.data.\n", comp[i].fichier); erreur(ch);
                return(FALSE);
        }

        /* Lecture des valeurs numeriques */
        nbpoints = fread(tableau, sizeof(float), comp[i].nbpoints, desc);
        if (nbpoints != comp[i].nbpoints)
        {
                sprintf(ch, "Pas assez de valeurs lues dans %s.data.\n",
                        comp[i].fichier); erreur(ch);
                free(tableau);
                fclose(desc);
                return(FALSE);
        }
        else
        {
                fclose(desc);
        }

        /* Calculer la moyenne des valeurs */
        moyenne = 0.0; ptfloat = tableau;
        for (j=0; j<nbpoints; j++)
                moyenne += *ptfloat++;
        moyenne /= nbpoints;

        /* Soustraire la valeur moyenne */
        ptfloat = tableau;
        for (j=0; j<nbpoints; j++)
                *ptfloat++ -= moyenne;

        for (j=0; j<nbpoints; j++)
        {
                if (tableau[j] > comp[i].maxi)
                        comp[i].maxi = tableau[j];
        }
        if (comp[i].maxi > maxi)
                maxi = comp[i].maxi;

        /* Liberation du tableau de valeurs pour la composante courante */
```

```
                free(tableau);
        }

        for (i=0; i<NBCOMPS; i++)
        {
                mercallimax = 0;
                for (j=0; j<8; j++)
                {
                        if (comp[i].maxi > mercalli[j])
                                mercallimax = j;
                }
                strcpy(comp[i].mercalli, mercalli_ch[mercallimax]);
                strcpy(comp[i].mercalli2, mercalli_ch2[mercallimax]);
        }

        /*
         * Nettoyage des fichiers
         */
        for (i=0; i<NBCOMPS; i++)
        {
                sprintf(ch, "rm %s.data", comp[i].fichier); system(ch);
                sprintf(ch, "rm %s.info", comp[i].fichier); system(ch);
        }

        return(TRUE);
}

/*
 * Tous les repertoires quotidiens du style "20050805" anterieurs a celui du dernier fichier
 * ferme sont deplaces sous DATADIR apres renommage en aaaa-jjj. Il y a donc normalement un
 * deplacement de ce genre chaque jour peu apres 00h00.
 */
void menage()
{
        int     resul;
        int     aa, mm, jj, jul;
        DIR     *dirp;
        struct dirent *dp;
        char    ch[MAXLEN];
        char    *pt;
        FILE    *liste;

        /* Le fichier temporaire contient tous les deplacements a effectuer */
        resul = chdir (TMPDIR);
        if (resul != 0)
        {
                sprintf(ch, "menage: Impossible d'acceder au repertoire %s.\n", TMPDIR); erreur(ch);
                exit(-1);
        }

        liste = fopen (TMPFILE, "w");
        if (liste == NULL)
        {
                sprintf(ch, "menage: Impossible de creer le fichier temporaire %s.\n", TMPFILE); erreur(ch);
                exit(-1);
        }

        resul = chdir (FTPDIR);
        if (resul != 0)
        {
                sprintf(ch, "menage: Impossible d'acceder au repertoire %s.\n", FTPDIR); erreur(ch);
                exit(-1);
        }

        resul = chdir (STATION);
        if (resul != 0)
        {
                sprintf(ch, "menage: Impossible d'acceder au repertoire %s.\n", STATION); erreur(ch);
                exit(-1);
        }

        dirp = opendir (".");
        if (dirp == NULL)
        {
                sprintf(ch, "menage: Impossible d'acceder au contenu du repertoire %s.\n", STATION); erreur(ch);
                exit(-1);
        }

        for (;;)
        {
                dp = readdir(dirp);
                if (dp == NULL)
                {
                        closedir(dirp);
```

```
                        fclose(liste);
                        break;
                }
                if (!strcmp(dp->d_name, "."))
                        continue;

                if (!strcmp(dp->d_name, ".."))
                        continue;

                if (strlen(dp->d_name) != 8)
                        continue;

                if (strcmp(veille, dp->d_name) <= 0)
                        continue;

                pt = dp-> d_name;
                strcpy(ch, pt); ch[4] = '\0'; aa = atoi(ch); pt += 4;
                strcpy(ch, pt); ch[2] = '\0'; mm = atoi(ch); pt += 2;
                strcpy(ch, pt); ch[2] = '\0'; jj = atoi(ch);
                jul = qt(aa, mm, jj);

                /*
                 * Deplacement des repertoires quotidiens
                 */
                fprintf(liste, "mv -f %s/%s/%s %s/%04d-%03d\n",
                        FTPDIR,
                        STATION,
                        dp->d_name,
                        DATADIR,
                        aa,
                        jul);
                /*
                 * Deplacement des fichiers log quotidiens (apres elimination des lignes en double)
                 */
                fprintf(liste, "cat %s/logs/%s.log | uniq > %s/%04d-%03d/%04d-%03d.log ; rm -f %s/logs/%s.log\n",
                        DATADIR, dp->d_name,
                        DATADIR, aa, jul, aa, jul,
                        DATADIR, dp->d_name);
        }
        /* Le fichier temporaire contient tous les deplacements a effectuer */
        resul = chdir (TMPDIR);
        if (resul != 0)
        {
                sprintf(ch, "menage: Impossible d'acceder au repertoire %s.\n", TMPDIR); erreur(ch);
                exit(-1);
        }

        liste = fopen (TMPFILE, "r");
        if (liste == NULL)
        {
                sprintf(ch, "menage: Impossible de relire le fichier temporaire %s.\n", TMPFILE); erreur(ch);
                exit(-1);
        }

        for (;;)
        {
                if (fgets(ch, MAXLEN, liste) == NULL)
                {
                        fclose(liste);
                        break;
                }
                system(ch);
        }
}

void exam()
{
        int     resul;
        DIR     *dirp;
        struct dirent *dp;
        char    prec[MAXLEN];
        char    cour[MAXLEN];
        char    ch[MAXLEN];
        char    ch2[MAXLEN];
        FILE    *listing;
        FILE    *listing2;
        char    *pt, *pt2;

        resul = chdir (FTPDIR);
        if (resul != 0)
        {
                sprintf(ch, "exam: ** 1 ** Impossible d'acceder au repertoire %s.\n", FTPDIR); erreur(ch);
                exit(-1);
        }
```

```
        resul = chdir (STATION);
        if (resul != 0)
        {
                sprintf(ch, "exam: ** 2 ** Impossible d'acceder au repertoire %s.\n", STATION); erreur(ch);
                exit(-1);
        }

        dirp = opendir (".");
        if (dirp == NULL)
        {
                sprintf(ch, "exam: ** 3 ** Impossible d'acceder au contenu du repertoire \".\".\n", STATION);
erreur(ch);
                exit(-1);
        }

        sprintf(ch, "%s/%s", TMPDIR, TMPFILE);
        listing = fopen (ch, "w");
        if (listing == NULL)
        {
                sprintf(ch, "exam: ** 8 ** Impossible de creer le listing des repertoires recents.\n");
erreur(ch);
                exit(-1);
        }

        for (;;)
        {
                dp = readdir(dirp);
                if (dp == NULL)
                {
                        closedir(dirp);
                        fclose(listing);
                        break;
                }
              . if (!strcmp(dp->d_name, "."))
                        continue;

                if (!strcmp(dp->d_name, ".."))
                        continue;

                fprintf(listing, "%s/%s/%s\n",
                        FTPDIR,
                        STATION,
                        dp->d_name);
        }

        /* Trier */
        sprintf(ch, "cat %s/%s | sort > %s/%s2 ; \\mv %s/%s2 %s/%s\n",
                TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE);
        resul = system(ch);
        if (resul == -1)
        {
                sprintf(ch, "exam: ** 8 bis ** Erreur dans le tri du fichier temporaire.\n"); erreur(ch);
                exit(-1);
        }

        sprintf(ch, "%s/%s", TMPDIR, TMPFILE);
        listing = fopen (ch, "r");
        if (listing == NULL)
        {
                sprintf(ch, "exam: ** 8 ter ** Impossible de lire le fichier temporaire.\n"); erreur(ch);
                exit(-1);
        }

        strcpy(ch, "");
        strcpy(prec, "");
        strcpy(cour, "");

        for (;;)
        {
                if (fgets(ch, MAXLEN, listing) == NULL)
                {
                        pt = prec; pt += strlen(FTPDIR); pt ++; pt += strlen(STATION); pt ++;
                        strcpy(veille, pt); veille[8] = '\0';
                        fclose(listing);
                        break;
                }

                /* Enlever le retour-chariot a la fin */
                pt = ch; pt +=strlen(ch); --pt; *pt = '\0';

                if (strcmp(ch, cour) > 0)
                {
                        strcpy(prec, cour);
```

```
                strcpy(cour, ch);
        }

}

/*
 *      Maintenant on a les deux repertoires les plus recents. On va les balayer
 *      pour determiner le fichier ferme (.dat) le plus recent et le fichier en cours
 *      d'acquisition (.dat.acq) le plus recent. Il peut en effet y avoir plusieurs
 *      fichiers .dat.acq (en cas de plantage ou de redemarrage du PC par exemple).
 */
resul = chdir (prec);
if (resul != 0)
{
        sprintf(ch, "exam: ** 6 ** Impossible d'acceder au repertoire %s.\n", prec); erreur(ch);
        exit(-1);
}

dirp = opendir (".");
if (dirp == NULL)
{
        sprintf(ch, "exam: ** 7 ** Impossible d'acceder au contenu du repertoire %s.\n", prec);
erreur(ch);
        exit(-1);
}

sprintf(ch, "%s/%s", TMPDIR, TMPFILE);
listing = fopen (ch, "w");
if (listing == NULL)
{
        sprintf(ch, "exam: ** 8 ** Impossible de creer le listing des repertoires recents.\n");
erreur(ch);
        exit(-1);
}

for (;;)
{
        dp = readdir(dirp);
        if (dp == NULL)
        {
                closedir(dirp);
                break;
        }
        if (!strcmp(dp->d_name, "."))
                continue;

        if (!strcmp(dp->d_name, ".."))
                continue;

        fprintf(listing, "%s/%s\n", prec, dp->d_name); fflush(listing);

        sprintf(ch, "%s/%s", prec, dp->d_name); fflush(listing);
}

resul = chdir (cour);
if (resul != 0)
{
        sprintf(ch, "exam: ** 11 ** Impossible d'acceder au repertoire %s.\n", cour); erreur(ch);
        exit(-1);
}

dirp = opendir (".");
if (dirp == NULL)
{
        sprintf(ch, "exam: ** 12 ** Impossible d'acceder au contenu du repertoire %s.\n", cour);
erreur(ch);
        exit(-1);
}
for (;;)
{
        dp = readdir(dirp);
        if (dp == NULL)
        {
                closedir(dirp);
                fclose(listing);
                break;
        }
        if (!strcmp(dp->d_name, "."))
                continue;

        if (!strcmp(dp->d_name, ".."))
                continue;

        fprintf(listing, "%s/%s\n", cour, dp->d_name); fflush(listing);
```

```
	)

	/*
	 *	Trier le listing des repertoires/sous-repertoires en commencant par les plus
	 *	recents. On s'arretera ensuite des qu'on y aura trouve un fichier .dat et un
	 *	fichier .dat.acq.
	 */
	sprintf(ch, "cat %s/%s | sort -r > %s/%s2 ; \\mv %s/%s2 %s/%s\n",
		TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE);
	resul = system(ch);
	if (resul == -1)
	{
		sprintf(ch, "exam: ** 13 ** Erreur dans le tri du fichier temporaire.\n"); erreur(ch);
		exit(-1);
	}

	/*
	 *	On boucle sur tous les repertoires listes
	 *
	 */
	sprintf(ch, "%s/%s", TMPDIR, TMPFILE);
	listing = fopen (ch, "r");
	if (listing == NULL)
	{
		sprintf(ch, "exam: ** 14 ** Impossible d'acceder au listing.\n"); erreur(ch);
		exit(-1);
	}

	sprintf(ch, "%s/%s3", TMPDIR, TMPFILE);
	listing2 = fopen (ch, "w");
	if (listing2 == NULL)
	{
		sprintf(ch, "exam: ** 15 ** Impossible de creer le listing des repertoires recents.\n");
erreur(ch);
		exit(-1);
	}

	for (;;)
	{
		if (fgets(ch, MAXLEN, listing) == NULL)
		{
			fclose(listing);
			fclose(listing2);
			break;
		}
		ch[strlen(ch)-1] = '\0';

		dirp = opendir (ch);
		if (dirp == NULL)
		{
			sprintf(ch2, "exam: ** 16 ** Impossible d'acceder au contenu du repertoire %s.\n",ch);
			erreur(ch2);
			exit(-1);
		}

		for (;;)
		{
			dp = readdir(dirp);
			if (dp == NULL)
			{
				closedir(dirp);
				break;
			}
			if (!strcmp(dp->d_name, "."))
				continue;

			if (!strcmp(dp->d_name, ".."))
				continue;

			if (strlen(dp->d_name) < 6)
				continue;

			fprintf(listing2, "%s/%s\n", ch, dp->d_name);
		}
	}

	/*
	 *	Trier le listing des repertoires/sous-repertoires en commencant par les plus
	 *	recents. On s'arretera ensuite des qu'on y aura trouve un fichier .dat et un
	 *	fichier .dat.acq.
	 */
	sprintf(ch, "cat %s/%s3 | sort -r > %s/%s4 ; \\mv %s/%s4 %s/%s3\n",
		TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE, TMPDIR, TMPFILE);
	resul = system(ch);
```

```
        if (resu1 == -1)
        {
                sprintf(ch, "exam: ** 17 ** Erreur dans le tri du fichier temporaire.\n"); erreur(ch);
                exit(-1);
        }

        /*
         *      On parcourt la liste triee repertoire/sous-repertoire/fichier
         *
         */
        sprintf(ch, "%s/%s3", TMPDIR, TMPFILE);
        listing = fopen (ch, "r");
        if (listing == NULL)
        {
                sprintf(ch, "exam: ** 18 ** Impossible d'acceder au listing.\n"); erreur(ch);
                exit(-1);
        }
        lastopenedOK = FALSE; strcpy(lastopened, "");
        lastclosedOK = FALSE; strcpy(lastclosed, "");

        for (;;)
        {
                if (fgets(ch, MAXLEN, listing) == NULL)
                {
                        fclose(listing);
                        break;
                }
                ch[strlen(ch)-1] = '\0';
                pt = ch; pt += strlen(ch); pt -= 4;
                pt2 = ch2;
                *pt2 = *pt; ++pt; ++pt2;
                *pt2 = *pt; ++pt; ++pt2;
                *pt2 = *pt; ++pt; ++pt2;
                *pt2 = *pt; ++pt; ++pt2;
                *pt = '\0'; *pt2 = '\0';

                if (lastclosedOK == FALSE)
                        if (!strcmp(ch2, ".dat"))
                        {
                                strcpy(lastclosed, ch);
                                lastclosedOK = TRUE;
                        }

                if (lastopenedOK == FALSE)
                        if (!strcmp(ch2, ".acq"))
                        {
                                strcpy(lastopened, ch);
                                lastopenedOK = TRUE;
                        }

                if ( (lastopenedOK == TRUE) && (lastclosedOK == TRUE) )
                        break;
        }
}

int main (int argc, char **argv, char **arge)
{
        char    ch[MAXLEN];
        char    ch1[MAXLEN];
        char    tmp1[MAXLEN];
        char    tmp2[MAXLEN];
        char    tmp3[MAXLEN];
        char    lastclosedandchecked[MAXLEN];
        char    *pt, *pt0;
        float   maxi;
        int     intensitemax, nouveau;
        booleen flag;
        int     aa, ms, jj, hh, mm, resul;
        FILE    *log;

        strcpy(lastclosedandchecked, "");

        initscr(); /* Initialisation de curses */
        for (;;)
        {
                /* On determine le dernier fichier ferme et celui en cours d'acquisition */
                exam();

                /*
                 * Deplacer les repertoires quotidiens, cad ceux anterieurs a celui auquel appartient le dernier
                 * fichier ferme. On deplace qussi le fichier log correspondant.
                 */
                menage();
```

```
/*
 * On depouille uniquement si ça n'a pas ete deja fait lors de l'iteration
 * precedente
 */
if (strcmp(lastclosedandchecked, lastclosed))
{
        /* Depouillement pour le dernier fichier ferme */
        flag = depouille(lastclosed);
        if (flag == TRUE)
        {
                /*
                 * On rajoute la ligne correspondante dans le fichier log
                 * quotidien. Pour cela, on teste l'existence du repertoire
                 * des fichiers log (/home/geosci/data/logs), et ensuite
                 * sous ce repertoire l'existence du fichier log quotidien.
                 */
                strcpy(ch1, DATADIR);
                strcat(ch1, "/logs");
                /*
                 * Creer le repertoire uniquement s'il n'existe pas
                 *
                 */
                if (access(ch1, F_OK) != 0)
                {
                        sprintf(ch, "Creation du repertoire %s.\n", ch1); erreur(ch);
                        mkdir(ch1,
                                S_IRUSR | S_IWUSR | S_IXUSR|
                                S_IRGRP | S_IXGRP |
                                S_IROTH | S_IXOTH);
                        sprintf(ch, "mkdir %s", ch1);system(ch);
                }

                resul = chdir (ch1);
                if (resul != 0)
                {
                        sprintf(ch,
                                "Impossible d'acceder au repertoire %s.\n", ch1);
                        erreur(ch);
                        exit(-1);
                }
                strcpy(ch1, lastclosed);
                pt = ch1;
                pt = strstr(ch1, STATION);
                pt += strlen(STATION); ++pt; pt0 = pt;
                *(pt+8) = '\0';
                strcat(ch1, ".log");
                if (access(pt0, F_OK) != 0)
                {
                        sprintf(ch, "Le fichier %s n'existe pas.\n", pt0);
                        erreur(ch);
                }
                /*
                 * Maintenant, pt contient un nom de fichier sous la forme
                 * aaaammjj.log
                 */
                log = fopen(pt0, "a");
                if (log == NULL)
                {
                        sprintf(ch, "Impossible d'ouvrir %s.\n", pt0); erreur(ch);
                        return(FALSE);
                }

                /*
                 * Affichage des fichiers en cours (mode plein ecran
                 * utilisant curses)
                 */
                pt = strstr(lastclosed, STATION); pt += strlen(STATION); ++pt;
                strcpy(ch, pt); ch[4] = '\0'; aa = atoi(ch); pt +=4;
                strcpy(ch, pt); ch[2] = '\0'; ms = atoi(ch); pt +=2;
                strcpy(ch, pt); ch[2] = '\0'; jj = atoi(ch); pt +=3;
                strcpy(ch, pt); ch[2] = '\0'; hh = atoi(ch); pt +=3;
                strcpy(ch, pt); ch[2] = '\0'; mm = atoi(ch); pt +=3;

                /* Impression de la ligne courante */
                maxi = 0.0;
                if (comp[0].maxi > maxi)
                {
                        maxi = comp[0].maxi;
                        strcpy(tmp1, comp[0].mercalli);
                        strcpy(tmp2, comp[0].mercalli2);
                        strcpy(tmp3, "Z");
                }
                if (comp[1].maxi > maxi)
                {
```

```
        maxi = comp[1].maxi;
        strcpy(tmp1, comp[1].mercalli);
        strcpy(tmp2, comp[1].mercalli2);
        strcpy(tmp3, "N");
}
if (comp[2].maxi > maxi)
{
        maxi = comp[2].maxi;
        strcpy(tmp1, comp[2].mercalli);
        strcpy(tmp2, comp[2].mercalli2);
        strcpy(tmp3, "E");
}

fprintf(log, "%02d:%02d\t%d\t%s\t%s\t%s\n",
        hh, mm,
        (int) maxi,
        tmp1,
        tmp2,
        tmp3);

intensitemax = atoi(tmp2);

fclose(log);

/*
 * Positionnement des valeurs des leds. On actionne en plus le buzzer
 * si le maxi depasse une valeur donnee (4)
 */
switch (intensitemax)
{
        case 8: mask = LED7;
                break;
        case 7: mask = LED6;
                break;
        case 6: mask = LED5;
                break;
        case 5: mask = LED4;
                break;
        case 4: mask = LED3;
                break;
        case 3: mask = LED2;
                break;
        case 2: mask = LED1;
                break;
        case 1:
        default:
                if (ancienmask == NOIR)
                {
                        mask = BLANC;
                        ancienmask = BLANC;
                }
                else
                {
                        mask = NOIR;
                        ancienmask = NOIR;
                }
                break;
}

if (intensitemax >= SEUILBUZZ)
        mask = mask | BUZZON;
else
        mask = mask & BUZZOFF;
mask = mask & 0xff;

if (argc == 1)
        AfficherLeds("/dev/console");
else
        AfficherLeds(argv[1]);

/* Initialisations */
memset(&insn, 0, sizeof(insn));
insn.subdev    = subdev;
insn.n         = 4;
insn.chanspec  = CR_PACK(chan, 0, 0);
insn.insn      = INSN_WRITE;

EcrireRelais(&insn, mask);

mvprintw(1, 23, "%-50s", "***** DERNIER FICHIER FERME *****");
mvprintw(3, 2, "%-78s", lastclosed);
mvprintw(4, 2, "%-6d points, %02d %s %04d %02d:%02d

        comp[0].nbpoints, jj, mois[ms-1], aa, hh, mm);
```

```
                        mvprintw(5, 2, "Z: %12.3f (%-4s), N: %12.3f (%-4s), E: %12.3f (%-4s)",
                                comp[0].maxi, comp[0].mercalli,
                                comp[1].maxi, comp[1].mercalli,
                                comp[2].maxi, comp[2].mercalli);
                        strcpy(lastclosedandchecked, lastclosed);
                }
                else
                {
                        mvprintw(1, 23, "%-50s", "***** DERNIER FICHIER FERME *****");
                        mvprintw(3, 2, "%-78s", lastclosed);
                        mvprintw(4, 2, "------------------------------------------------------------------------
----");
                        mvprintw(5, 2, "Z: ------------ (----), N: ------------ (----), E: ------------ (-
---)");
                }
        }

        /* Depouillement pour le fichier en cours d'acquisition */
        flag = depouille(lastopened);
        if (flag == TRUE)
        {
                pt = strstr(lastopened, STATION); pt += strlen(STATION); ++pt;
                strcpy(ch, pt); ch[4] = '\0'; aa = atoi(ch); pt +=4;
                strcpy(ch, pt); ch[2] = '\0'; ms = atoi(ch); pt +=2;
                strcpy(ch, pt); ch[2] = '\0'; jj = atoi(ch); pt +=3;
                strcpy(ch, pt); ch[2] = '\0'; hh = atoi(ch); pt +=3;
                strcpy(ch, pt); ch[2] = '\0'; mm = atoi(ch); pt +=3;

                maxi = 0.0;
                if (comp[0].maxi > maxi)
                {
                        maxi = comp[0].maxi;
                        strcpy(tmp2, comp[0].mercalli2);
                }
                if (comp[1].maxi > maxi)
                {
                        maxi = comp[1].maxi;
                        strcpy(tmp2, comp[1].mercalli2);
                }
                if (comp[2].maxi > maxi)
                {
                        maxi = comp[2].maxi;
                        strcpy(tmp2, comp[2].mercalli2);
                }

                nouveau = atoi(tmp2);

                if (nouveau > intensitemax)
                {
                        /*
                        * Positionnement des valeurs des leds. On actionne en plus le buzzer
                        * si le maxi depasse une valeur donnee (4)
                        */
                        switch (nouveau)
                        {
                                case 8: mask = LED7;
                                        break;
                                case 7: mask = LED6;
                                        break;
                                case 6: mask = LED5;
                                        break;
                                case 5: mask = LED4;
                                        break;
                                case 4: mask = LED3;
                                        break;
                                case 3: mask = LED2;
                                        break;
                                case 2: mask = LED1;
                                        break;
                                case 1:
                                default:
                                        break;
                        }

                        if (nouveau >= SEUILBUZZ)
                                mask = mask | BUZZON;
                        else
                                mask = mask & BUZZOFF;
                        mask = mask & 0xff;

                        if (argc == 1)
                                AfficherLeds("/dev/console");
                        else
                                AfficherLeds(argv[1]);
```

```
                        /* Initialisations */
                        memset(&insn, 0, sizeof(insn));
                        insn.subdev    = subdev;
                        insn.n         = 4;
                        insn.chanspec  = CR_PACK(chan, 0, 0);
                        insn.insn      = INSN_WRITE;

                        EcrireRelais(&insn, mask);
                }

                mvprintw(7, 20, "%-60s", "***** FICHIER EN COURS D'ACQUISITION *****");
                mvprintw(9, 2, "%-78s", lastopened);
               mvprintw(10, 2, "%-6d points, %02d %s %04d %02d:%02d                                                ",
                        comp[0].nbpoints, jj, mois[ms-1], aa, hh, mm);
                mvprintw(10, 2, "%-6d points, %02d %s %04d %02d:%02d",
                        comp[0].nbpoints, jj, mois[ms-1], aa, hh, mm);
                mvprintw(11, 2, "Z: %12.3f (%-4s), N: %12.3f (%-4s), E: %12.3f (%-4s)",
                        comp[0].maxi, comp[0].mercalli,
                        comp[1].maxi, comp[1].mercalli,
                        comp[2].maxi, comp[2].mercalli);

        }
        else
        {
                mvprintw(7, 20, "%-50s", "***** FICHIER EN COURS D'ACQUISITION VIDE !!! *****");
                mvprintw(9, 2, "%-78s", lastopened);
                mvprintw(10, 2, "------------------------------------------------------------------------");
                mvprintw(11, 2, "Z: ------------ (----), N: ------------ (----), E: ------------ (----)");
        }

        refresh();
        sleep(DELAY);
    }
}
```

Majsis : Programme de nettoyage lancé tous les jours (voir dans crontab) pour effacer tous les répertoires quotidiens plus vieux que 180 jours (six mois). Situé sous /home/geosci/bin/majsis. Exécute le programme ecart (cf. page suivante).

```
#!/bin/csh

set maxjours = 180

cd /home/geosci/data

foreach i (2???-???)
   set ecart = `/home/geosci/bin/ecart $i`
   if ($ecart >= $maxjours) then
      \rm -rf $i
   endif
end
```

Ecart.c : Exprime l'écart en jours entre un répertoire aaaa-jjj et le jour courant. Aaaa est l'année et jjj le jour julien dans cette année.

```
#include <stdio.h>
#include <string.h>
#include <time.h>

#define	MAXLEN		1024

int	nbj[12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

/*
 * Number of days in year
 */
int dysize(int year)
{
	if (year % 4 != 0)
		return (365);
	if (year % 400 == 0)
		return (366);
	if (year % 100 == 0)
		return (365);
	/*
	 * Otherwise leap year
	 */
	return (366);
}

main(int argc, char **argv)
{
	char	ch[MAXLEN], *pt;
	int	annee;
	int	i, jour, jours, moiss;
	struct	tm	*ptime;
	struct	tm	hier, hoy;
	time_t	yesterday, today, ecart;

	if (argc != 2)
		exit(1);

	if (strlen(argv[1]) != 8)
		exit(1);

	if (argv[1][4] != '-')
		exit(1);

	pt = argv[1];
	strncpy(ch, pt, 4); annee = atoi(ch);

	if (dysize(annee) == 366)
		nbj[1] = 29;

	pt = argv[1]; pt += 5;
	strncpy(ch, pt, 4); jour = atoi(ch); jours = jour;

	hier.tm_year	=	annee-1900;
	hier.tm_mon	=	0;
	hier.tm_mday	=	0;
	hier.tm_hour	=	0;
	hier.tm_min	=	0;
	hier.tm_sec	=	0;

	for (i=0; i<12; i++)
	{
		if (jours-nbj[i] < 0)
		{
			hier.tm_mday	=	jours;
			break;
		}
		++hier.tm_mon;
		jours -= nbj[i];
	}

	yesterday =	mktime(&hier);
	time(&today);

	ecart		=	today-yesterday;
	printf("%ld\n", ecart/86400);

}
```

Initialisation : Après avoir lancé la session root sous KDE, ouvrir au moins deux fenêtres. Dans la première, taper « initialisation », dans le seconde, taper « acquisition » pour lancer l'acquisition proprement dite.

```
#!/bin/tcsh

#
#       Initialisation de comedi. A faire s'il y a reboot.
#
/sbin/modprobe comedi
/sbin/modprobe pci1760
/usr/sbin/comedi_config /dev/comedi0 pci1760

#
#       Tty de la fenetre d'etat des LEDs
#
tty >! /tmp/terminal.txt
```

Acquisition : Après avoir lancé la session root sous KDE, ouvrir au moins deux fenêtres. Dans la première, taper « initialisation », dans le seconde, taper « acquisition » pour lancer l'acquisition proprement dite.

```
#!/bin/csh
\rm /tmp/logexam.txt
\rm /tmp/examine.err
set i = 1
while 1
    (date ; echo "Demarrage n° "$i) >> /tmp/logexam.txt
    examine `cat /tmp/terminal.txt`
    set i = `expr $i + 1`
end
```

The Comedi project develops open-source drivers, tools, and libraries for data acquisition.

Comedi is a collection of drivers for a variety of common data acquisition plug-in boards. The drivers are implemented as a core Linux kernel module providing common functionality and individual low-level driver modules.

Comedilib is a user-space library that provides a developer-friendly interface to Comedi devices. Included in the Comedilib distribution is documentation, configuration and calibration utilities, and demonstration programs.

Kcomedilib is a Linux kernel module (distributed with Comedi) that provides the same interface as Comedilib in kernel space, suitable for real-time tasks. It is effectively a "kernel library" for using Comedi from real-time tasks.

Features

- Integrated real-time support for most hardware
- High-level library (comedilib)
- Application-level device independence
- Works with Linux 2.0, 2.2, 2.4, 2.6 kernels

Une fois implanté comedi et le driver pci1760, le périphérique associé à la carte est /dev/comedi0. Il est nécessaire de procéder à une réinitialisation en cas de redémarrage du système (cf. annexe 2, initialisation) :

```
#
#       Initialisation de comedi. A faire s'il y a reboot.
#
/sbin/modprobe comedi
/sbin/modprobe pci1760
/usr/sbin/comedi_config /dev/comedi0 pci1760
```

3.4 – Les logiciels développés

Le logiciel de traitement des données, développé en C, est contenu dans le fichier examine.c (sous /home/geosci/src/examine/). S'y ajoutent quelques utilitaires d'initialisation (acquisition, initialisation) et de gestion des répertoires quotidiens pour la gravure des DVD (majsis, ecart).

Le logiciel examine l'arborescence des fichiers irae et pour la tranche courante de deux minutes, en examine les valeurs numériques échantillon par échantillon, pour en déterminer le maximum, et convertir cette valeur en intensité de Mercalli entre I et VIII. Les données de la Titan sont au format constructeur et pour les dépouiller, il est nécessaire d'avoir installé les utilitaires de dépouillement, en particulier cvtit, développés par J.-F. Fels. On trouve ces utilitaires à ftp://renass.u-strasbg.fr/pub/people/fels/. Par commodité, les exécutables ont été installées sous /home/geosci/bin. Dépouiller les valeurs numériques revient à créer un ficher temporaire dans lequel sont écrites les valeurs « en clair » (mais en binaire), fichier que l'on balaie ensuite pour en extraire la valeur maximum. L'appel de cvtit correspondant (cf. source de examine.c (annexe 2) est le suivant : ***cvtit <fichier origine> -bin -shortblk sta=yate | grep \"Output: \" > fichier (binaire) de valeurs numériques"***. Un fichier quotidien de log est mis à jour avec pour chaque ligne, la tranche de deux minutes, et l'intensité mesurée (cad date,

Annexe 3 :
Mise en œuvre des logiciels Irae sous Linux
(voir également http://www.agecodagis.fr)

==========
COMMENT FAIRE DEMARRER IRAE SUR UN PC LINUX SUSE ?

Christophe Maron - 26.06.2002
==========

Irae est constitue de 3 modules :

IraeTx recoit les donnees en flux continu
en provenance du numeriseur, soit directement
par liaison serie native, soit via un convertisseur
RS232 --> USB de type Keyspan. Dans ce cas, il est
possible de mettre en entree non pas 1 numeriseur,
mais 4 numeriseurs. Il y autant d'IraeTx a lancer
qu'il y a de numeriseurs en entree.
Chaque IraeTx porte le nom de la station d'acquisition
correspondante.

IraeRx prend les donnees transmises par IraeTx et
les met sous forme de fichiers Titan longs de 2
minutes. Il les stocke en local, d'une part, et les
transmet via ftp, soit dans une zone ftp d'un PC
distant, soit sur l'un de ses propres repertoires.
IraeRx doit porter un nom, le plus simple etant de le
nommer du lieu geographique ou se trouve le PC1.

IraeCx permet le controle a distance de l'acquisition
et du transfert des donnees.

Dans les explications qui suivent, on appelle PC1 le PC Linux
sur lequel tournent IraeTx et IraeRx, on appelle PC2 le PC
Linux sur lequel arrivent les donnees par ftp. PC1 et PC2
peuvent etre la machine.

On suppose que seront crees les comptes 'root' et 'irae'.

On suppose que sera branche un seul numeriseur en entree
d'IraeTx. Appelons cette station 'NOU'.
Appelons 'NOUMEA' l'IraeRx qui recoit les donnees d'IraeTx
'NOU'.

Dans toute la procedure qui suit, on est logge root.

A/ INSTALLATION STANDARD DE LINUX

On suppose Linux deja installe sur les PC1 et PC2.
En particulier :

. partitionnement,
. creation d'un compte root (Irae tourne sous root)
  et d'un autre compte (ex. : irae),
. choix d'une installation standard sans Office
  (sur un PC de bureau classique, les differents
  constituants sont automatiquement reconnus et
  les modules sont installes sans difficulte),
. configuration de la partie reseau (IP, NFS, NIS,
  route, demarrage automatique de inetd),
. installation supplementaire de :

```
package d :             g77,
                        kaffe
package xapplication :  xnetdiag
package net :           gkermit,
                        netdiag,
                        trafficvis,
                        xntp
```

[Pour cela : mettre le CDROM numero 1 dans le lecteur ;
lancer yast ;
selectionner la source d'installation (CDROM EIDE) ;
mettre a jour l'installation ;
definir l'installation ;
aller dans le package desire ;
cocher la case desiree grace a la barre d'espace ;
sortir du package avec F10 ;
une fois tous les packages desires parcourus :
demarrer l'installation ;
suivre les indications (changer le CD ...) ;
remonter les menus grace a Escape ;

```
                    sortir de yast.
                    eject cdrom ]

        . dans /etc/inetd.conf, decommenter la ligne comprenant
          in.ftpd" pour permettre les transferts ftp.
        . choix l'environnement login graphique xdm a la place de kdm.
        . rebooter sans CDROM pour voir si tout se passe bien.


B/ LA PARTIE USB (sur PCl uniquement, et uniquement en cas
                  d'utilisation du Keyspan.
                  Sinon, passer au C/)
-------------------------------------------------------

B.1 / Dans /proc/devices, on doit avoir :

==================
Character devices:
  1 mem
  2 pty
  3 ttyp
  4 ttyS
  5 cua
  6 lp
  7 vcs
 10 misc
 13 input
 29 fb
128 ptm
136 pts
162 raw
180 usb
188 usb/tts/%d

Block devices:
  1 ramdisk
  2 fd
  3 ide0
  7 loop
========

     Si la ligne "188 usb/tts/%d est absente et ne peut pas
     etre ajoutee, ne pas insister.

B.2/ Dans /dev,
     a la commande "ls -l ttyUSB*",
     on doit obtenir la reponse (eventuellement aux dates pres) :

=============================================================
crw-rw----    1 root     uucp     188,   0 nov  6 18:34 ttyUSB0
crw-rw----    1 root     uucp     188,   1 jan 19  2001 ttyUSB1
crw-rw----    1 root     uucp     188,   2 jan 19  2001 ttyUSB2
crw-rw----    1 root     uucp     188,   3 jan 19  2001 ttyUSB3
=============================================================

B.3/ Dans le fichier /etc/modules.conf,
     ajouter quelque part les 2 lignes :

================================
# RS232 --> USB Keyspan converter
alias char-major-188 keyspan
============================

B.4/ A la commande "echo > /dev/ttyUSB0",
     on ne doit pas avoir un message d'erreur tel que
     "no such device" apres la 2eme tentative.


C/ INSTALLATION DE IRAE
-----------------------

C.1/ Dechargement du CDROM irae :

        . cd /tmp
        . mkdir ageco-irae
        . cd ageco-irae
        . mount /dev/cdrom /cdrom
        . cp /cdrom/* .
        . umount /cdrom
        . eject cdrom

C.2/ Deploiement de Java :
```

```
        . chmod a+x j2sdk-1_3_1_01-linux-i386-rpm.bin
        . ./j2sdk-1_3_1_01-linux-i386-rpm.bin
          et taper Space jusqu'a la fin.
        . rpm -U ./j2sdk-1_3_1_01-linux-i386.rpm
        . dans /etc/profile,
          a la fin de la 1ere ligne contenant "PATH=",
          ajouter : ":/usr/java/jdk1.3.1_01/bin".

C.3/ Deploiement du driver serie :

        . cd /tmp/ageco-irae
        . rpm --nodeps -U ./AgecoJSerial-1.0-20011004.i386.rpm

C.4/ Deploiement de Irae :

        . cd /tmp/ageco-irae
        . rpm -U ./AgecoIrae-5.3-20011102.i386.rpm

        . cd /home/irae
        . mkdir IraeRx
        . cd IraeRx
        . cp /tmp/ageco-irae/ftprdv.ser .

        . cd /root
        . mkdir Agecodagis
        . cd Agecodagis
        . mkdir IraeCx
        . cd IraeCx
        . cp /tmp/ageco-irae/ftprdv.ser .

        . dans /opt/Agecodagis/Java/etc/java.sh,
          a la place de "which java",
          mettre : "type -path java".

        . cd /opt/Agecodagis/Java/lib
        . /bin/rm irae-*.jar
        . cp /tmp/ageco-irae/irae-*.jar .

        . uniquement si utilisation du Keyspan :

          dans /opt/Agecodagis/Java/run/IraeTx,
          apres la ligne "do DIR=$IRAETXHOME-$ntx" (ligne numero 22),
          ajouter la ligne : "echo > /dev/ttyUSB$ntx"
          pour atteindre une premiere fois le device USB.

C.5/ Parametrisation de l'acquisition Irae :

        . dans /opt/Agecodagis/Java/etc/irae.conf,
          sur la ligne contenant "IRAETXHOME=",
          mettre apres le "=" le nom du repertoire de travail,
          a savoir : "/home/irae/IraeTx".
        . dans /opt/Agecodagis/Java/etc/irae.conf,
          sur la ligne contenant "IRAERXHOME=",
          mettre apres le "=" le nom du repertoire de travail,
          a savoir : "/home/irae/IraeRx".
        . dans /opt/Agecodagis/Java/etc/irae.conf,
          sur la ligne contenant "NIRAETX=",
          mettre apres le "=" le nombre de numeriseurs en entree,
          a savoir : 1 sur PC1 (cf. hypotheses de depart).

C.6/ Premier lancement de Irae :

C.6.a/ Sur PC1 : cd /ftp
                 mkdir -p irae/NOUMEA
                 cd /
                 chown -R irae:users ftp

C.6.b/ Faire du menage dans les repertoires remplis par Irae :

       Sur PC1 :

       Si /home/irae/IraeTx-0 existe          : tout y effacer.
       Si /home/irae/IraeRx existe            : tout y effacer sauf ftprdv.ser.
       Si /home/irae/IraeRx n'existe pas      : cd /home/irae
                                                mkdir IraeRx
                                                cd IraeRx
                                                cp /tmp/ageco-irae/ftprdv.ser .
       Si /root/Agecodagis/IraeCx existe      : tout y effacer sauf ftprdv.ser.
       Si /root/Agecodagis/IraeCx n'existe pas : cd /root
                                                mkdir -p Agecodagis/IraeCx
                                                cd Agecodagis/IraeCx
```

```
                                                cp /tmp/ageco-irae/ftprdv.ser .

        Sur PC2 :

        /bin/rm -R /ftp/irae/NOUMEA/*

C.6.c/ IraeTx (sur PC1) :
        Autant de fenetres s'ouvrent qu'on a defini de Tx sur PC1.

        Pour chacune de ces fenetres :

                Remplir les champs :

                Name                    NOU
                Target Rx IP            IP de PC1
                Data IP Port            1000            <-- ne pas changer !
                Import Serial From IP                   <-- laisser vide
                COM Port                /dev/ttyS0

                Go
                Plot
                Iconiser la fenetre.
                La rouvrir : les signaux apparaissent.
                Close : la fenetre de tracer se ferme.

                Ne pas sortir de IraeTx.
                Iconiser la fenetre IraeTx.

C.6.d/ IraeRx (PC1) :

        OK
        Remplir les champs :

        Name                    NOUMEA
        Location                IRD Noumea      <-- infos au format libre
        Fixed IP Adress                 IP de PC1
        Phone Number
        FTP Site                IP de PC2
        FTP login               irae
        FTP Password            mot de passe d'irae
        FTP Path                /ftp/irae/NOUMEA
        Window (H1-H2)          0-23            <-- transfert 24 h/24
        E-Mail
        Smtp
        Max data Size (MB)      25000           <-- volume de donnees tjs dispo

        OK

        Le "Control Panel" apparait : 1 bouton par Tx porte le nom de sa station.

        Pour chaque bouton :
        Cliquer dessus.
        Iconiser la fenetre.
        La rouvrir : les signaux apparaissent.
        Close : la fenetre de tracer se ferme.

        Ne pas sortir du "Control Panel" de IraeRx.
        Iconiser le "Control Panel".

C.6.e/ IraeCx (PC1) :
        Mot de passe = 'noumea'
        OK
        New
        Remplir les champs de la meme facon exactement que pour Rx.
        Ftp Up...
        FtpXfers : New
                   Si il apparait "Always AllDone" : Delete, puis New, puis OK.
                   Si il n'apparait rien           :              New, puis OK.
                   OK
        OK
        Quit

C.6.f/ Sortir de IraeRx (PC1)

        . Dans le "Control Panel", cliquer sur "Quit".

C.6.g/ Sortir de IraeTx (PC1)

        . Cliquer sur "Exit".

C.7/ Lancement automatique de Irae (PC1)
```

. dans /opt/Agecodagis/Java/etc/irae.conf,
  sur la ligne contenant "STARTIRAERX=",
  mettre "TRUE" a la place de "FALSE".
. dans /opt/Agecodagis/Java/etc/irae.conf,
  sur la ligne contenant "STARTIRAETX=",
  mettre "TRUE" a la place de "FALSE".
. init 6 pour faire rebooter le PC
  et controler qu'Irae demarre correctement.

C.8/ Controle des donnees grace a Cx (PC2)

. IraeCx
  mot de passe = 'noumea'
  OK
  selectionner le Rx desire ou
  le creer en remplissant les champs de la meme facon
  que le Rx.
  Connect NOUMEA
  Cliquer le nom du Tx pour plotter les signaux correspondant.
  Sortir ou laisser defiler (mais pas a perpette !).

C'est fini !

D/ COMPLEMENTS :

D.1/ Irae se lance en mode automatique et on desire que cela cesse.

Dans une fenetre Terminal, editer le fichier /opt/Agecodagis/Java/etc/irae.conf.
Sur la ligne contenant "STARTIRAETX=", remplacer "TRUE" par "FALSE".
Sur la ligne contenant "STARTIRAERX=", remplacer "TRUE" par "FALSE".

Sortir le plus proprement possible du maximum de fenetres Irae.
Rebooter le PC grace a init 6.