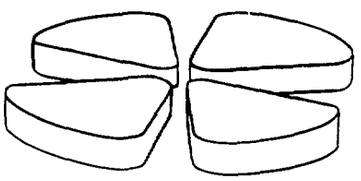
GANIL



Nouveau système de contrôle

ORGANISATION

des PROGRAMMES ADA

GANIL A 92 12

Secteur Exploitation
Informatique Machine
L. DAVID

GANIL 533/92
le 01/07/1992
version 2.0

Nouveau système de contrôle
ORGANISATION
des PROGRAMMES ADA

La seconde version de ce rapport inclut en plus la répartition des programmes Ada du Ganiciel entre les différents répertoires et librairies du Vax de développement.

Sur VAX/VMS, le compilateur ADA s'intègre dans un atelier de génie logiciel ACS qui permet un développement cohérent par contrôle des versions des sources et des exécutables, séparation des applications en différents niveaux de visibilité et gestion des liens existants entre les différents modules composants une même application.
Dans le cas du nouveau GANICIEL et d'IMAGIN, les applications existantes ou en cours de développement sont :

pour IMAGIN avec des automates SIEMENS :
- la source ECR et l'O.A.I
- les mesures de températures H.F
- les grilles de commutation

pour les processus avec un RTVAX :
- les profils de faisceau dans un chassis Vme
- L'extension en phase dans un chassis Camac
- Le cyclage des aimants des CSS dans un chassis Camac

pour le Ganiciel sur station Vax :
- La désignation des équipements (ex touch-panel)
- Les blocs de 8 dials (ex shafts)
- Les alarmes centrales au p.c.p

pour le Ganiciel sur chassis RTVAX :
- Les handlers d'accès aux équipements
- La base de données des équipements locaux à chaque chassis
- La communication et l'interprétation des ordres issus des stations

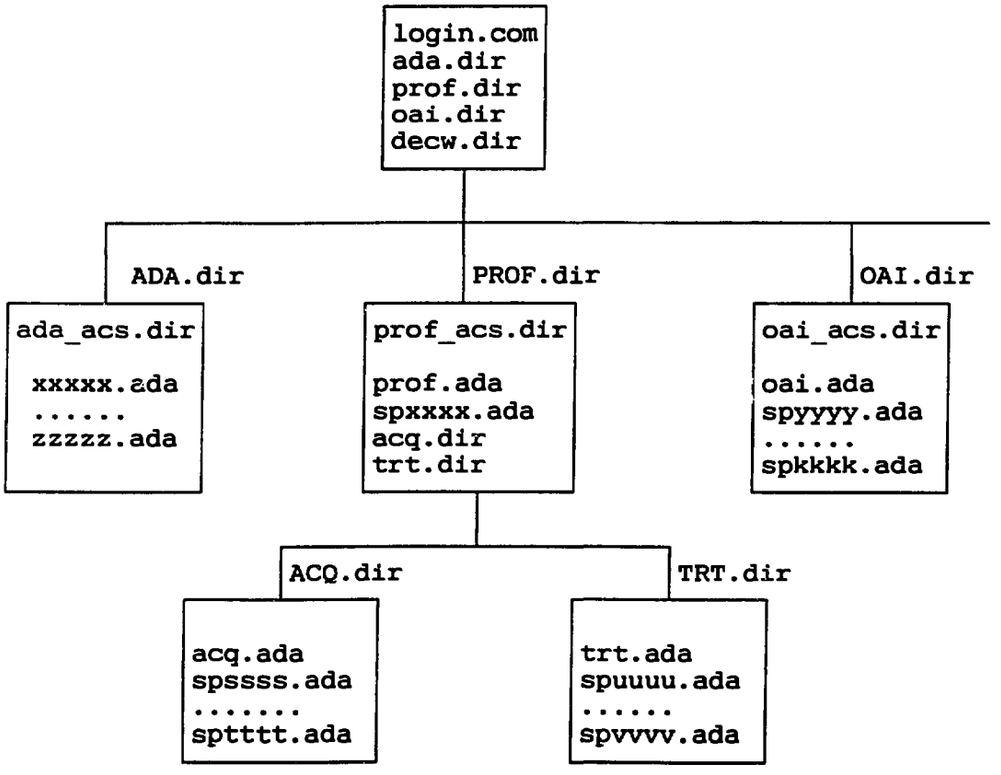
1. STRUCTURE D'UNE APPLICATION

1.1 Concepts

Les différents objets et sources relatifs à une même application et manipulés par ADA (ou par des procédures utilisant ACS de façon transparente pour l'utilisateur) doivent être regroupés dans un même répertoire, sous un même nom d'application.

Pour pouvoir passer d'une application à une autre, l'utilisateur devra utiliser impérativement cette notion, notamment grâce à la commande CA décrite dans le paragraphe 2. En effet, un simple changement de répertoire ne permet pas de prendre automatiquement en compte les changements de librairies et les reconstitutions de liens inter modules qui sont nécessaires au contrôle de cohérence imposé par ADA.

1.2 Structure d'un répertoire utilisateur



Le répertoire OAI correspond à une application IMAGIN s'exécutant sous VMS, alors que le répertoire PROF est relatif à l'application Profils qui s'exécute sous ELN sur un chassis VME ou CAMAC.

1.3 Les packages ADA

Les sources ADA sont regroupés dans des packages constitués d'un fichier pour la partie spécification et d'un fichier pour la partie exécutable (le corps du package).
Le fichier de spécifications contient ce qui est visible de l'extérieur :
- déclarations de constantes et de types communs à toutes les procédures du package.
- déclarations de procédures, de fonctions ou de tâches relatives au traitement de l'objet défini par le package, avec leurs paramètres.

Le fichier de corps contient ce qui n'est pas systématiquement visible de l'extérieur :
- définitions des constantes, variables ou types communs à toutes les procédures du package mais dont l'usage est local à celui-ci.
- corps de traitement des procédures, fonctions ou tâches correspondant aux spécifications ou en supplément par rapport à celles-ci pour réaliser un traitement local à l'intérieur du package.
- instructions d'initialisation du package qui s'exécutent avant le début du programme principal de l'application utilisant le package. Cette phase d'élaboration est déclenchée par la clause "with nom-package"

1.4 Application IMAGIN

L'application OAI est contenue dans le répertoire [OAI]. Celui-ci contient les sources du programme principal du même nom que l'application (ici OAI.ada) et de ses sous-programmes (procédures ada). Il contient aussi la librairie OAI_ACS qui est vue comme un répertoire mais ne peut être accédée comme tel car elle renferme les liens entre les modules et doit être gérée exclusivement par ACS. Les commandes du paragraphe 2 rendent cette manipulation transparente à l'utilisateur.
La création de l'application entraine la création du répertoire correspondant et des liens ACS nécessaires, ainsi que l'insertion d'un modèle de source applicatif composé de 4 fichiers:
- un fichier de spécifications du package P_APPLICATION
- un fichier de corps du package P_APPLICATION qui doit être modifié en fonction des messages IMAGIN à traiter
- un fichier d'exemple de procédure séparée DEMANDER_TABLES__.ada
- un fichier du nom de l'application qui constitue le programme principal traitant la boucle de dialogue avec l'animateur et qui ne devrait pas être modifié par l'utilisateur.

Dans le cas d'IMAGIN, l'application OAI est liée à une application INTERFACE_IMAGIN de niveau supérieur, située ailleurs, contenant les sous-programmes généraux d'IMAGIN, transparente pour l'utilisateur et hors de sa visibilité.

1.5 Application ELN

L'application PROF est contenue dans le répertoire [PROF], au même niveau que toutes les autres applications ADA de l'utilisateur. Ce répertoire contient les sources des sous-programmes et du programme principal, dont le nom est celui de l'application. Il contient aussi la librairie ACS de l'application: PROF_ACS
Dans le cas de cet exemple, l'application PROF contient deux sous-répertoires ACQ.dir et TRT.dir qui contiennent respectivement les sources des procédures correspondant par exemple au traitement et à l'acquisition. Il n'y a pas de librairie ACS à ce niveau car il ne peut y en avoir qu'une seule par application et elle se situe au niveau du programme principal.
La création de l'application entraine toujours la création du répertoire correspondant et des liens ACS nécessaires. La création des sous répertoires reste à la charge de l'utilisateur par la commande VMS : CREATE/DIR [.TOTO]
Dans le cas de VAX/ELN, l'application PROF est liée à l'application ACS_PROCESSUS de niveau supérieur et qui contient les sous-programmes généraux des processus ELN.
Si l'application nécessite aussi une partie sous VMS, cela correspond alors à une autre librairie ACS. Cette autre librairie est liée à l'application ACS_REGLAGE de niveau supérieur et qui contient les procédures générales d'interface utilisateurs.

1.6 Programmes de tests

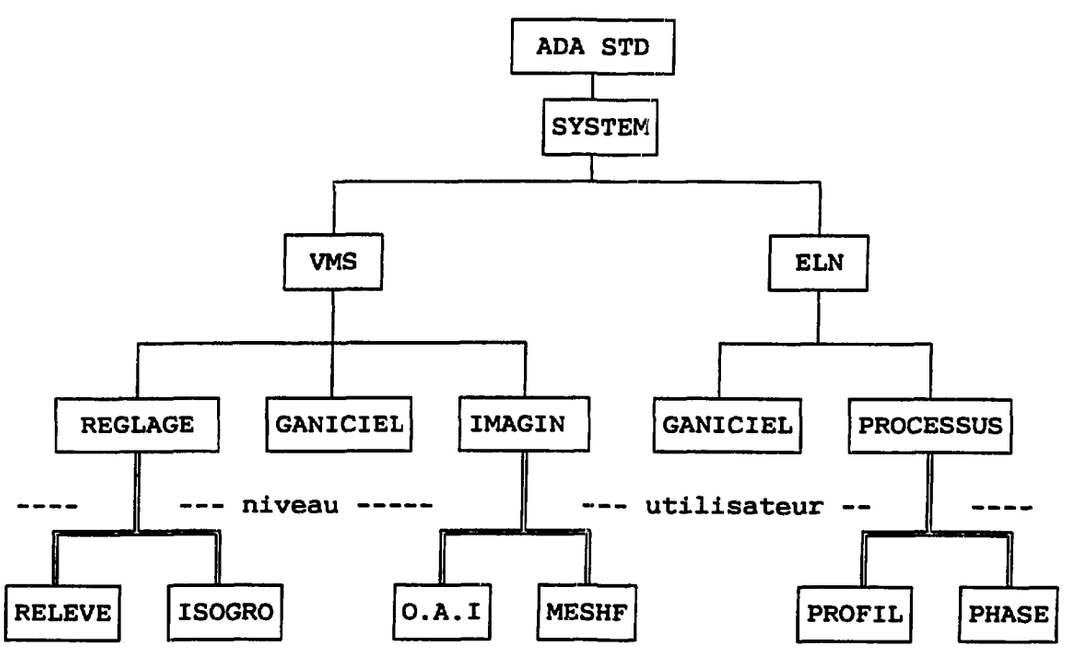
L'application ADA présente systématiquement chez tous les utilisateurs ADA doit être considérée comme un environnement pour petits programmes de tests. Elle contient les sources de ces programmes ainsi qu'une librairie ADA_ACS correspondant. Cette librairie n'est pas reliée aux librairies générales pour IMAGIN ou VAX/ELN mais possède uniquement un accès à la librairie générale ADA$PREDEFINED située à un niveau encore supérieur et qui contient les sous-programmes très généraux (packages pré-définis tels que text_io)

1.7 Liens inter applications

Les dépendances inter-applications, déclenchées par les clauses "with" du langage ADA, se font soit par filiation, soit par chemins d'accès, soit par entrées.
- Une librairie ACS peut contenir des entrées sur une autre librairie. Ce sont en fait des références à des packages sources situés ailleurs. S'il n'y a pas filiation, les entrées doivent être déclarées explicitement pour qu'un package défini dans une librairie ACS puisse être utilisé par une autre.
- Une librairie peut contenir une définition de chemin ("path") qui lui permet directement à l'ensemble des packages d'une autre librairie de niveau supérieur.
- Une librairie peut aussi être dépendante d'une librairie mère. Dans ce cas elle utilise les entrées ou chemins déjà définis dans cette librairie supérieure et peut ainsi accéder directement aux packages de celle-ci sans devoir déclarer les entrées ou chemins s'y rapportant.

Dans ce cas, si un utilisateur pense qu'une de ses procédures peut être d'usage plus général, il peut la remonter dans la librairie mère par une commande MERGE. Cette procédure devient alors accessible à toutes les librairies filles sans que les autres utilisateurs n'aient à déclarer de nouvelles entrées sur une librairie mère spécifique.

1.8 Hiérarchie des librairies ACS



Les liaisons en double trait représentent les filiations. Ainsi la librairie O.A.I est fille de la librairie ACS_IMAGIN, alors que celle-çi n'est liée à la librairie ACS_VMS que par une déclaration de chemin d'accès.
En principe les librairies de niveau utilisateur n'auront pas à déclarer ni entrées ni chemins car elles utiliseront ceux de leur librairie mère (ACS_IMAGIN dans ce cas).

2. BINAIRES EXECUTABLES

2.1 Sur VAX/VMS

Les fichiers exécutables sur Vax sont de 2 types :

- suffixe EXE s'ils sont issus de programmes ADA après compilation et link.

- suffixe UID s'ils sont issus de programmes UIL correspondant à des réalisation d'écrans par des widgets. Les programmes UIL peuvent eux-mêmes êtres issus de l'éditeur graphique VUIT.

Tous ces fichiers sont situés sur le disque du Vax 3800, que le programme soit destiné à s'exécuter sur ce Vax ou sur une station VS4000. Dans ce cas l'exécutable est chargé par le réseau depuis le disque du Vax vers la mémoire de la station, car les disques locaux des stations ne contiennent que des fichiers temporaires.

2.2 Sur RTVAX/ELN

Les exécutables sont aussi de 2 types :

- suffixe EXE s'ils sont issus de programmes ADA après compilation et link. Ils peuvent alors être téléchargés dans un chassis possédant déjà un système opérationnel par une commande LOAD exécutée depuis ECL (interpréteur de commandes de VAXELN).
Un fichier correspond alors typiquement à l'ensemble des tâches Ada formant un processus

- suffixe SYS s'ils sont issus du builder ELN qui regroupe :
  - le noyau de l'operating system Eln
  - le debugger et le process Edisplay
  - les drivers nécessaires (Camac ou Vme, réseau decnet et Lat)
  - le process d'initialisation
  - des process exécutables de type EXE

Ils sont alors téléchargés automatiquement depuis le Vax lors de la mise sous tension d'un chassis et peuvent être rechargés ensuite par une commande TRIG exécutée depuis DCL (interpréteur de commandes de VMS).
Il y a donc un fichier .sys commun à tous les chassis qui gèrent le Ganiciel et un fichier .sys spécifique à chaque processus nouveau, incluant aussi le ganiciel. Il est toujours possible de rajouter ensuite un nouvel exécutable par la commande LOAD précédente.
La correspondance entre un chassis et son fichier .sys est réalisée sur VAX/VMS par NCP lors de la description du load file de chaque noeud decnet.

3. LISTE DES PACKAGES

La liste et l'explication du contenu de chacun de tous les packages existants peut être obtenue en ligne sur le Vax par la commande "package". Les rubrique disponibles correspondent à chacun des répertoires décrits ci dessus:

- Commun pour les packages généraux
- Eln pour les packages camac et système coté Eln
- Vms pour les packages système coté Vms
- Ganeln pour les packages du ganiciel coté Eln
- Ganvms pour les packages du ganiciel coté Vms
- Processus pour les packages communs aux processus RTVAX
- Reglage pour les packages communs aux programmes de réglage

4. ARBORESCENCE DES REPERTOIRES

Les sources des applications sont situés dans les répertoires utilisateurs, ainsi que les sources UIL des présentations d'écrans.
Le disque logique Disk$control contient l'ensemble de l'arborescence rassemblant les composants logiciels suivants:
- les librairies ACS et les sources ADA communs
- les binaires exécutables (fichiers .exe)
- les fichiers de données (fichiers .dat)
- les widgets et présentations d'écrans communs (fichiers .uid)

4.1 Répertoire [SYSTEM]

Il contient l'ensemble des packages d'interface vers les différents environnements utilisés.

4.1.1 Répertoire [VMS] (nom logique acs_vms)

[Interface]: packages d'interface système Vms donnant l'accès aux mailbox, process, sections globales, lock manager...
[Motif] : fonctions ajoutées au logiciel Gobe pour manipuler les chaines et tracer des axes.
[Ingres] : package d'accès à des fonctions SQL.
[Binding] : corrections ou compléments aux bindings Ada erronés de DEC
[Acs] : librairie contenant les objets et références ada.

4.1.2 Répertoire [ELN] (nom logique acs_eln)

[Binding] : packages d'interface vers les librairies Camac et Vme écrites en "C".
[Handler] :
[Interface]: gestion des communications et de la base de données locale.
[Acs] : librairie contenant les objets et références ada.

4.1.3 Répertoire [COMMUN] (nom logique acs_commun)

[Bd_handler] : structures de données de la base des équipements
[Divers] : packages de traitement de la date, de chaines de caractères, de listes chainées.
[Msg_gan] : types de tous les messages du ganiciel
[Msg_proc] : types des messages spécifiques de chaque processus
[Acs] : librairie contenant les objets et références ada.

4.2 Répertoire [GANVMS]

Il contient d'une part le Ganiciel de base et d'autre part les packages permettant aux applications de réglage d'accéder au ganiciel.

4.2.1 Répertoire [GANICIEL] (nom logique acs_ganvms)

[Acs] : librairie contenant les objets et références ada.
[Alarmes] : traitement des alarmes
[Equipmt] : gestion de la base des équipements et traduction des noms operationnels
[Serveur] : packages permettant de générer des serveurs ganiciel
[Uil] : fichiers uid pour l'écran des alarmes.

4.2.2 Répertoire [REGLAGE] (nom logique acs_reglage)

[Acs] : librairie contenant les objets et références ada.
[Ecran] : affichage de texte et menus dans une fenêtre decterm
[Equipmt] : packages d'accès aux équipements
[Divers] : packages communs entre plusieurs applications de réglage.
[Processus]: packages d'accès aux différents processus

[Uil] : fichiers Motif réalisant l'interface graphique des programmes de réglage.

4.3 Répertoire [GANELN]

4.3.1 Répertoire [GANICIEL] (nom logique acs_ganeln)

Il contient l'ensembles des sources Ada des packages réalisant les fonctions du ganiciel, ainsi que l'accès à ces fonctions par les processus déportés dans les chassis Eln.

4.3.2 Répertoire [PROCESSUS] (nom logique acs_processus)

Ensemble des packages communs entre différents processus.

4.4 Le Répertoire [EXEC]

Ce répertoire est la copie de celui qui, sur le Vax 3800 contient l'ensemble des binaires exécutables sur station Vax ou chassis Eln. Il est lui aussi divisé en sous-répertoires.

- [ganeln] : nom logique disk$ganeln
fichiers .sys de l'exécutable du Ganiciel pour les chassis Camac avec ou sans processus.

- [processus] : nom logique disk$procexec
fichiers .exe des exécutables des différents processus (téléchargés par une commande de type LOAD)

- [ganvms] : nom logique disk$ganexec
  - [exe] : binaires des différents process du ganiciel vax
  - [data] : fichiers de données indépendants de la base Ingres
  - [uid] : fichiers écrans manipulés par les process
  - [com] : procédures DCL de lancement des process ganiciel

- [reglage] : nom logique disk$regexec
  - [exe] : binaires des différents process de réglage
  - [data] : fichiers de données indépendants de la base Ingres
  - [uid] : fichiers écrans manipulés par les process de réglage

- [ingres] : nom logique disk$ingrexec
  - [w4gl] : binaires des applications graphiques
  - [abf] : binaires des applications alphanumériques

5. COMMANDES DISPONIBLES

5.1 Syntaxe générale

Les commandes ont pour paramètre soit le nom de l'application, soit le nom du source. Une fois l'application créée, les commandes doivent être passées après s'être positionné dans l'environnement de l'application.

5.2 Création d'une application

CREE_IMAG crée le répertoire du nom de l'application Imagin et la librairie _ACS correspondante. Elle recopie les modèles des sources Imagin pour le programme et les procédures que l'utilisateur va ensuite modifier selon son besoin. Elle positionne enfin l'utilisateur dans ce contexte applicatif.

CREE_ACS crée le répertoire du nom de l'application ainsi que la librairie _ACS correspondante, puis elle positionne l'utilisateur dans le contexte de l'application.
La procédure demande le type d'application (eln, processus, réglage ou vms) et crée en fait une sous librairie fille d'une des librairies acs_ganeln, acs_processus, acs_réglage ou acs_ganvms.

5.3 Positionnement sur une application

CA permet de se positionner dans le contexte et le répertoire de l'application choisie en paramètre. Seule cette commande permet de passer d'une application à une autre car CD ne fait que changer de répertoire.

5.4 Compilation

ADA compile les programmes ou procédures modifiés dans le contexte de l'application. Le paramètre est le nom du module source.

WIDGET compile les fichiers de définition de widgets et d'écrans au format UIL pour en faire des fichiers au format UID, affichables par les fonctions "manage" de X Window.

5.5 Cohérence

RECOMPILE met à jour les liens entre les différents niveaux d'appel d'un sous-programme venant d'être compilé. Le paramètre est le nom du programme principal concerné.

5.6 Link

ACS LINK crée l'exécutable d'une application ADA standard en n'utilisant que la librairie ADA.

LKIMADA crée l'exécutable d'une application ADA en utilisant la librairie de l'application et celles nécessaires à IMAGIN. Si un second paramètre est spécifié en tant que "BD", elle ajoute les librairies utilisées par la base de données Ingres. Le binaire ne peut alors s'exécuter que si la base Ingres est active sur la station.
LKIMADA_BUG est identique à la précédente avec en plus l'accès au debugger.

LINKA crée l'exécutable d'une application Ada de type Réglage, Ganiciel ou Test suivant que le 2nd paramètre vaut "reg", "gan" ou "test" (défaut).
LINKELN crée l'exécutable d'une application ELN qui peut être téléchargée par une commande LOAD sous ECL, avec utilisation des librairies partageables d'ELN résidentes dans le fichier .SYS créé par le builder.

LINKELN crée l'exécutable d'une application ELN qui peut être téléchargée par une commande LOAD sous ECL, avec utilisation des librairies partageables d'ELN rédidentes dans le fichier .SYS créé par le builder.

5.7 Trig

TRIG réalise le téléchargement d'une application ELN prise par defaut dans disk$gansys: vers un chassis cible Camac ou Vme.
Le premier paramètre est le nom du chassis cible, le second est le nom de l'application et le dernier paramètre optionnel permet d'indiquer un nom de répertoire personnel.