N° d'ordre : 1196

THESE

présentée à

L'UNIVERSITÉ DES SCIENCES ET TECHNOLOGIES DE LILLE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ

en

PRODUCTIQUE, AUTOMATIQUE ET INFORMATIQUE INDUSTRIELLE

par

JEAN-LUC DEBOUCHÉ
INGÉNIEUR IDN

CONTRIBUTION À LA CONCEPTION DES SYSTÈMES DE CONTRÔLE-COMMANDE

Soutenue le 5 Novembre 1993 devant la commission d'Examen

M JP.	BOUREY	**Examinateur**
M JP.	FRACHET	**Rapporteur**
M JC.	GENTINA	**Examinateur, Directeur de Thèse**
M D.	JOUBERT	**Invité**
M JB.	KOECHLIN	**Examinateur, Directeur de travail**
M R.	MARTIN	**Invité**
M G.	MOREL	**Rapporteur**
M M.	STAROSWIESKI	**Président du jury**
M J.	SZYMANSKI	**Examinateur**

Si tu veux être plombier, mécano ou chercheur,
Si une vie de travail ne te fait pas peur,
Quel que soit ton métier, tu es motivé,
Ta vie professionnelle te fera progresser,

A une condition:
LA FORMATION.

Quand tu auras trouvé ta voie,
Engage tes forces et toute ta foi,
Le temps n'aura pas d'importance,
Une seule recette, persévérance.

Il n'y aura pas de sot métier,
Il n'y aura que des gens mal formés,
Pour obtenir la compétence,
Ne compte pas trop sur la chance.

Que ce soit "sur le tas" ou à l'université,
Acquérir le savoir est une nécessité,
L'expérience des autres est toujours un profit,
Si l'on sait retenir ce que l'on a appris.

Métier manuel ou intellectuel,
La chose également peut être belle,
Opposer l'un à l'autre est une ineptie,
Réunir les deux n'est pas une utopie.

Qu'est-ce qu'un métier manuel aujourd'hui,
Et demain ?
Un travail réfléchi ...
Qu'on fait avec les mains.

Dans un monde en évolution constante,
Ta formation devra être permanente,
Que tu sois mécano ou chercheur,
Il ne faut pas compter tes heures.

Mais si tu sais regarder écouter et comprendre,
Si ta volonté d'être meilleur est grande,
Si tu es sûr de toi mais jamais méprisant,
Alors tu seras un Pro, mon enfant.

Je dédie ces quelques mots à tous les *cifrés* qui comme moi, croient fermement en la formation par la recherche et ne réclament qu'une seule chose en contrepartie: trouver dans l'industrie sans cesse matière à appliquer ses propres idées.

REMERCIEMENTS

Je tiens tout d'abord à remercier Messieurs GENTINA et KOECHLIN pour le soutien qu'il m'ont apporté durant les trois années consacrées à l'étude retracée dans ce mémoire.

Ce travail enrichissant m'a permis de constater qu'il reste beaucoup à faire en matière d'organisation et de méthodologie pour maîtriser la conception de systèmes complexes.

Cette modeste contribution fournit un éclairage sur les méthodes en proposant, pour des systèmes appliqués de contrôle-commande industriels, de procéder à un phasage des développements guidé par des règles témoignant à la fois du bon sens, de la pratique et des principes fondamentaux permettant la réutilisation dans ce domaine précis.

Je tiens sur ce sujet, à remercier particulièrement messieurs MARTIN, SZYMANSKY, AMAR et BENOIT de l'école d'analyse de CEGELEC, pour m'avoir permis, d'une part, de recueillir les différentes techniques jusqu'à présent recensées dans cette activité et, d'autre part, de participer à l'extension de la méthodologie MODAL, dans sa version consacrée au développement des systèmes.

Je remercie également tous les membres du LAIL et plus particulièrement à messieurs BOUREY, CRAYE et MAIK pour leur soutien logistique et théorique dans cette tâche de longue haleine.

Une pensée toute particulière pour mes collègues de l'AGA qui, par leur expérience, ont su me convaincre de l'intérêt des méthodes pragmatiques, tirant profit des savoir faire cumulés des divisions d'applications du groupe CEGELEC, dans sa branche Controle Industriel.

J'adresse ma gratitude ainsi que mon profond respect à monsieur HAUET, directeur de la Branche Produits et Techniques de CEGELEC, qui a su m'accorder sa confiance malgré la conjoncture difficile que nous traversons actuellement.

Enfin, c'est un honneur enfin d'avoir pour rapporteurs Messieurs FRACHET et MOREL qui ne cessent d'œuvrer pour la promotion du Génie Automatique Français.

AVANT-PROPOS

L'étude et la mise au point d'un système automatisé de production industrielle est un exercice périlleux et de longue haleine. Il s'avère néanmoins très souvent nécessaire car il représente pour une entreprise du secteur secondaire un moyen efficace d'adaptation de sa capacité de production à l'évolution de la demande sur un marché de moyenne ou grande consommation.

Plus encore en période de récession économique, les qualités d'un bon SAP se révèlent lorsque, combinée à d'efficaces techniques de gestion, son exploitation assure une progression constante des résultats de l'entreprise alors que celle-ci ne dispose pourtant que d'une étroite marge de manœuvre.

L'industrie française a pris de ce fait peu à peu conscience de la nécessité de s'équiper de systèmes automatisés de production. Elle les réclame de plus, flexibles, évolutifs et rapidement rentables.

Ces quatre critères ne peuvent généralement pas tous être satisfaits dans les mêmes proportions.

Néanmoins, la performance et la flexibilité du système s'obtiennent au prix d'efforts importants d'optimisation du procédé à automatiser, son évolutivité dépend étroitement des choix de matériels et des techniques employées pour sa réalisation et sa rentabilité suit le ratio performance/coût.

Ainsi, pour un niveau de performance donné, seule la maîtrise des coûts de développement, d'exploitation et de maintenance d'un SAP permettent d'augmenter sa rentabilité et ces trois facteurs sont liés dans la mesure où un système bien conçu peut s'exploiter et se maintenir économiquement .

La priorité sera donc donnée à la méthodologie de développement du SAP et en premier lieu à sa conception dont découle la plupart des points précédents.

Nous exposerons dans cette optique une méthode permettant de ***concevoir*** *une partie importante de son* ***système de contrôle et de commande.***

Cette partie concerne l'automatisation du procédé.

La méthode présentée dans cet ouvrage contribue d'une part à:

- *identifier les équipements d'automatisme du système*
- *optimiser leur organisation*
- *matérialiser ces derniers par des assemblages de constituants standards;*

et d'autre part à:

- *déterminer les éléments de logiciels à implanter sur ces équipements,*
- *spécifier ces éléments de logiciel de façon à pouvoir ensuite les programmer dans différents langages métiers.*

Elle emprunte, pour ce faire, deux voies consécutives:

- *la description de l'architecture matérielle du système de contrôle-commande*
- *la conception préliminaire du logiciel de ses équipements.*

Sommaire

INTRODUCTION

1. LE SAP, SON SYSTEME DE CONTROLE COMMANDE ET SA PARTIE OPERATIVE

Un système automatisé de production (SAP) se divise généralement en deux parties: une partie "opérative", chargée de la mise en œuvre du procédé et une partie "commande", matérialisée par un système de contrôle-commande (SCC).

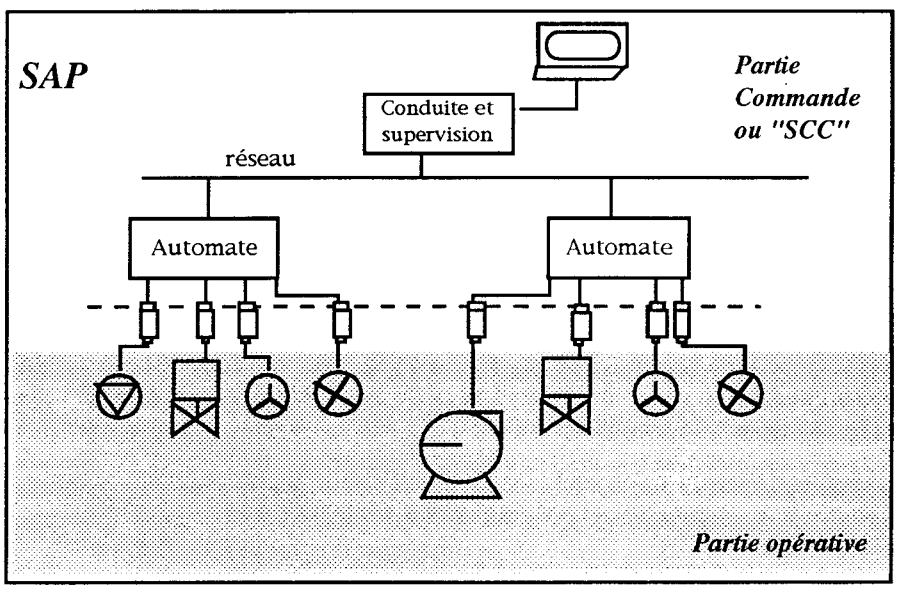


figure 1: Schéma de définition d'un SAP

Le SCC pilote, contrôle et commande la partie opérative constituée de capteurs et d'actionneurs dotés parfois d'une certaine complexité (capteurs intelligents par exemple).

Quel que soit le type de SAP envisagé, l'étude de sa partie opérative suit des règles imposées par le procédé industriel à automatiser. Elle tend à la mise en œuvre d'un appareillage conséquent de machines dont la conception exige généralement des compétences étroites dans des disciplines de base telles que la mécanique, la chimie, l'électronique ou encore l'électrotechnique.

L'étude du système de contrôle-commande n'exige pas, quant à elle, les mêmes compétences.
Elle s'adresse à des automaticiens, informaticiens ou plus généralement aux ingénieurs en systèmes complexes intégrant alors la connaissance de l'informatique à celle des réseaux de communication.

On note donc une séparation des métiers au sein des intervenants dans une affaire qui nous conduit à dissocier, concrètement, l'étude de la partie commande de celle de la partie opérative.

Dans ce qui suit, nous évoquerons essentiellement le développement des SCC. Leur partie opérative associée ne sera exposée qu'à travers ses interfaces avec le système de commande.

2. LA PROBLEMATIQUE DE CONCEPTION DU SCC

Sur le principe, tout le problème de la conception d'un système de contrôle-commande industriel réside dans le fait qu'il doit à tout instant maîtriser une quantité très importante d'informations en rapport avec **l'état** du procédé [RAY 91], ceci de façon à le faire évoluer dans des limites de charges régulièrement déduites des aspirations du marché.

La répartition constamment plus prononcée des machines de la partie opérative accentue encore la difficulté de mise en œuvre d'un SCC.

Sa complexité s'accroît aussi généralement par le mode de production occasionné par la **nature** du procédé, un mode de production continu qui impose bien souvent aux équipements d'automatisme le fait de mener des actions fortement coordonnées entre elles.

Elle évolue par ailleurs en fonction de la **précision** des opérations à effectuer pour automatiser le procédé, précision qui réclame un niveau de connaissance étroit du process et un savoir-faire important de la part des automaticiens.

3. LES MOYENS À METTRE EN ŒUVRE - CONTEXTE DE NOTRE ÉTUDE

L'évolution de la technologie permet peu à peu de répondre à la problématique précédente conception et accorde ainsi, plus souvent que par le passé, la possibilité aux SAP d'accomplir des tâches de plus en plus vastes, délicates et coordonnées.

La complexité de ces systèmes est telle qu'elle réclame nécessairement un travail en équipe et il n'est plus question, dans ce cas, de prétendre les développer sans **méthode** et sans **organisation**.

C'est pourquoi la plupart des projets actuels sont réalisés sous couvert d'une méthodologie préconisant l'emploi de techniques de gestion de projet et de méthodes spécifiques de développement.

Ces techniques et méthodes sont à l'origine d'outils informatiques d'assistances diverses et variées. Or, la disparité des matériels et des environnements proposés est telle que prétendre actuellement utiliser une chaîne d'outils communiquants correspondant au tracé couvert par la méthodologie relève d'un exercice très souvent vain.

Ainsi, seuls l'élaboration et le respect de standards en ce domaine peut aboutir à l'utilisation d'environnements de développements compatibles, respectant à la fois les mêmes types de **supports matériels**, les mêmes **ressources logicielles** et les mêmes **structures de données internes**.

De ces trois points de synergie est progressivement apparue la notion d'atelier de génie automatique dont les prémisses remontent au projet PTA.

3.1 LES INITIATIVES PTA ET BASEPTA OU L'IDÉE D'UNE HOMOGÉNÉISATION DES STRUCTURES DE DONNÉES D'APPLICATIONS

Le projet PTA (Poste de Travail de l'Automaticien) a permis dès 1984 [ALA 84] d'associer des offreurs et des utilisateurs, confrontés à des problèmes de mise en œuvre de systèmes de contrôle-commande de plus en plus complexes, à la rédaction d'un cahier des charges pour de futurs outils de génie automatique. Ce cahier des charges à mis en évidence la nécessité de former une chaîne d'outils communicants aux niveaux de laquelle chacun de ces outils respecterait un modèle de données unique, lié à son domaine d'activité.

La faisabilité d'une telle chaîne intégrée d'outils CAO a d'abord été démontrée dans [FRA 87] à l'issue d'un long travail de recherche, puis l'intérêt porté par les industriels à ce sujet a permis de poursuivre ces investigations par le projet Base-PTA, abordant alors une Base Application et Standard d'Echange pour le Poste de Travail de l'Automaticien.

Base-PTA a vocation d'être le méta-modèle conceptuel de données de toute structure d'information exploitée par un quelconque outil d'automatisme. Il représente ainsi potentiellement l'image conceptuelle des données d'application d'un atelier de génie automatique.

L'applicabilité de ses concepts n'a cessé d'être démontrée depuis que s'est fondé, pour la première fois en 1989 et à l'occasion de l'expérimentation ELEGA (Environnement Logiciel Expérimental pour le Génie Automatique), l'échange de données entre un outil d'ingénierie et une console de programmation d'automates sur un sous-ensemble de ce modèle [KOE 90].

Sa normalisation [Z68-92] a été assurée par le Centre Coopératif pour le Génie Automatique (CCGA) qui a comme objectif d'étendre Base-PTA à d'autres domaines d'intérêt et de promouvoir l'action du génie automatique notamment par des initiatives de formation à ces concepts.

☞ Le PTA puis Base-PTA ont montré à la fois par les idées et la pratique, l'intérêt économique que revêt l'homogénéisation des structures de données exploitées par des outils de génie automatique.

3.2. GENÈSE DE L'ATELIER OPÉRATIONNEL DE GÉNIE AUTOMATIQUE PAR LA PRISE EN COMPTE DES RÈGLES MÉTHODOLOGIQUES DE TRAVAIL

Disposer d'une structure de données d'application ne suffit pas à définir un atelier de génie automatique opérationnel. Il faut également établir une politique générale d'administration de ces données et fournir des éléments qui, à terme, permettent d'intégrer la **gestion des activités** de développement [GAG 93].

Ainsi, l'atelier de génie automatique doit nous amener à disposer, au sein d'un même système d'informations, de données d'applications et de paramètres de contrôle agissant directement sur l'ordre de fabrication de ces données.

Fort de cette constatation et riche de l'expérience acquise dans sa participation aux travaux du CCGA, le département RED de la Branche Produits et Techniques de CEGELEC a donné naissance au projet d'atelier AGA actuellement en cours d'étude et auquel nos travaux font référence.

3.3. LE PROJET AGA: UNE FINALISATION DE CES PRINCIPES

3.3.1. SES OBJECTIFS

L'ambition de l'AGA est de supporter les activités d'ingénierie que doivent mener les équipes de projet, c'est-à-dire:

- Spécifier le système à réaliser (le SCC),
- Concevoir son architecture,
- Développer ses sous-ensembles, matérialisés par des équipements, en réalisant une série d'activités: spécification, conception préliminaire et détaillée, programmation, test, intégration, mise en service,
- Gérer le processus de développement et les données produites à travers ce processus: gérer le projet, gérer les activités, gérer les configurations des éléments de projet,
- Assister la maintenance ultérieure du SCC,
- Produire la documentation adaptée aux différents intervenants (ingénieur d'affaire, concepteurs, réalisateurs et mainteneurs).

Dans le but:

- d'améliorer la productivité des équipes,
- d'augmenter la visibilité des applications,
- de diminuer les coûts de réalisation et les délais de livraison,
- d'améliorer la qualité des prestations, donc d'améliorer la qualité du produit développé,
- et enfin d'augmenter l'espérance de vie du système en le rendant plus facilement évolutif.

3.3.2 SA DÉMARCHE MÉTHODOLOGIQUE

L'AGA doit offrir les moyens de développer un SCC selon une méthodologie guidée par le cycle de vie du système. [MOD 93]

Il respecte pour ce faire les procédures d'application de cette méthodologie.

Il doit proposer pour la plupart des activités de développement le support d'un outil informatique adapté.

Il doit offrir la possibilité d'établir des documents à chaque fin de phase du cycle de vie système et selon des règles de rédactions prédéfinies.

Cf. figure 2 page suivante

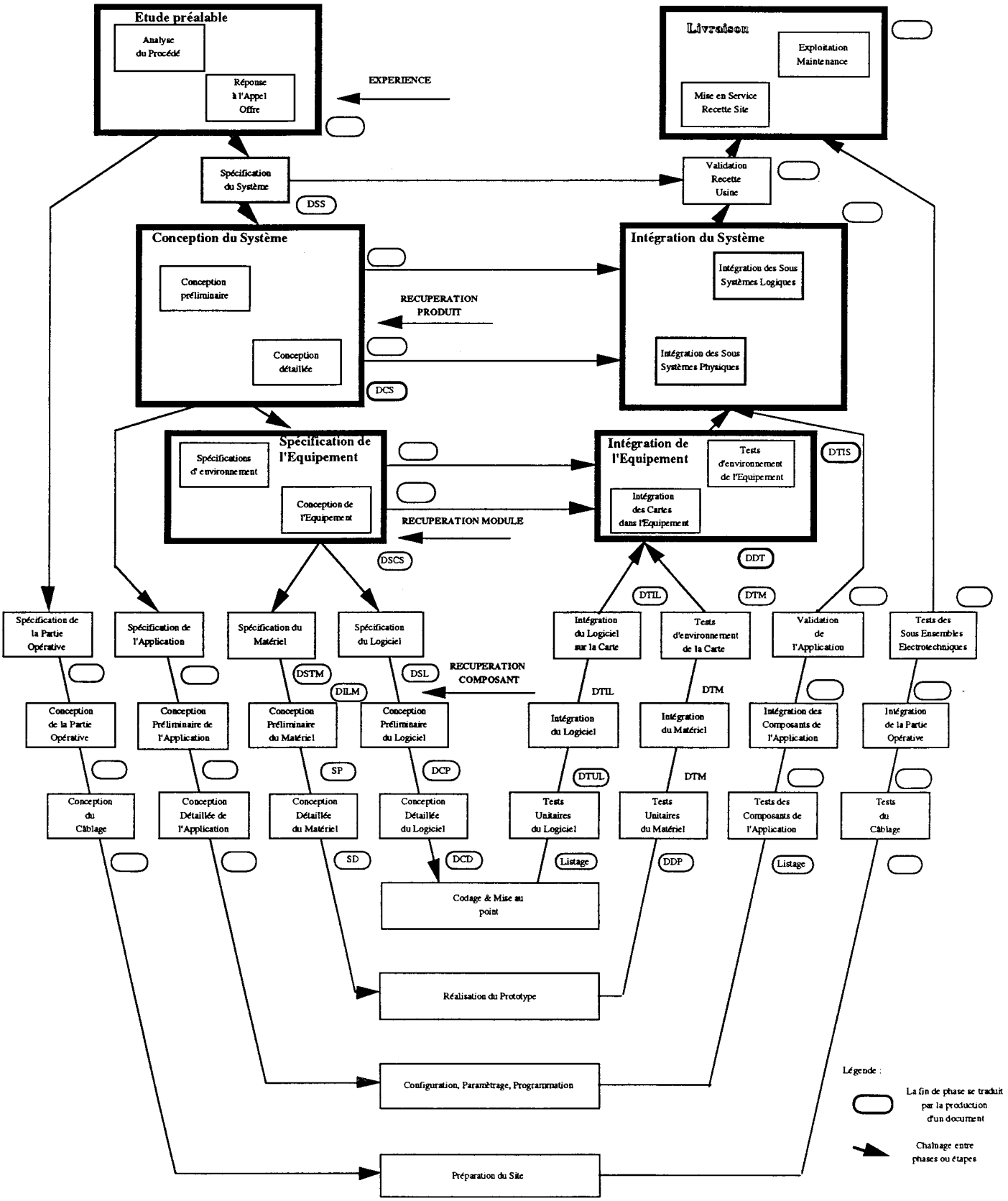


figure 2: carte générale des activités du développement d'un SAP

3.3.3 LE CONCEPT FÉDÉRATEUR DE L'AGA

De nombreux outils informatiques partiels existent déjà mais aucun ne facilite la gestion du développement des applications **réparties**. En effet, il est aujourd'hui très délicat d'organiser le travail des ingénieurs en parallèle, sans notamment entreprendre de pénibles phases de consolidation de données et ces outils sont incompatibles tant dans l'organisation des tâches, que dans les résultats qu'ils produisent.

De ce fait, même lorsque les consolidations entre résultats ont été obtenues, il n'est pas aisé de gérer les modifications continuelles et indispensables, qui affectent les systèmes de contrôle-commande et rendent nécessaire l'emploi d'un outil de "gestion des configurations".

L'intégration des outils réalisée par l'atelier de génie automatique est donc une nécessité que résume la figure suivante :

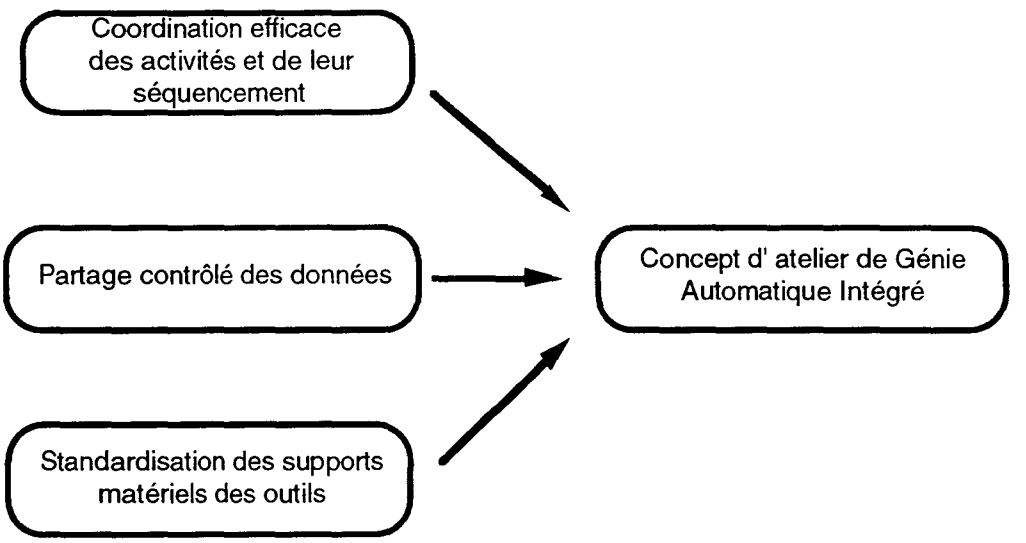


figure 3: Émanations du concept d'atelier de génie automatique

Selon sa configuration et le rôle qui lui est donné, l'AGA est sensé assister les équipes de développement sur toute ou partie du cycle de vie du système.

3.3.4 UNE PREMIÈRE CONFIGURATION-TYPE : SES FONCTIONS DÉTAILLÉES

Dans un premier temps, il a été choisit de faire porter nos études sur les activités liées au développement des systèmes **appliqués** de contrôle-commande, c'est-à-dire la partition suivante du cycle de vie d'un SAP:

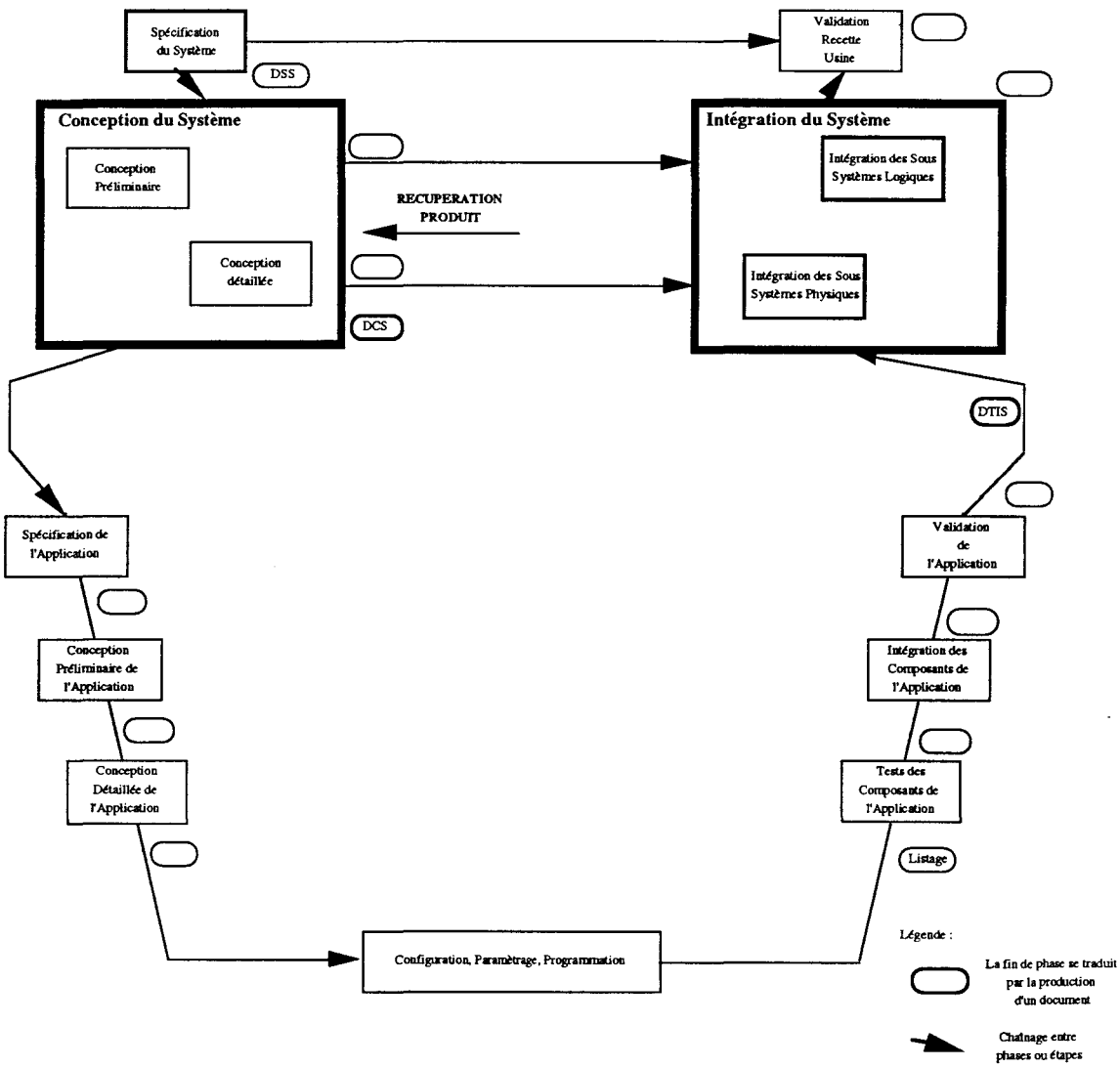


figure 4: carte des activités du développement d'un SCC appliqué

Cette partition doit donc nous permettre de traiter la liste des fonctions ci-dessous, classées par types de domaines d'activités:

Spécification et Conception Système

- Description de l'architecture matérielle
- Description des interfaces Partie Opérative
- Description des interfaces réseaux

Programmation des Automates / Contrôleurs de Traitement

- Conception des applications d'automatisme
- Edition des variables,
- Edition des programmes
  - .. en texte structuré,
  - .. en langage à relais,
  - .. en grafcet,
  - .. en langage de régulation,
- Fabrication de code exécutable.

Programmation des calculateurs spécifiques

- Conception des applications d'automatisme
- Programmation à l'aide de langages spécialisés
- Production de code exécutables.

Programmation des E/S

- Configuration des contrôleurs d'interfaces procédé et autres équipements d'E/S,
- Configuration des équipements de terrain.

Fabrication des tables de configuration des réseaux

Programmation des postes de conduite

Validation des systèmes

- Supervision de la mise en service,
- Gestion des configurations matérielles et logicielles,
- Téléchargement des cibles,
- Test à distance (mode télé-connecté).

Maintenance

- Aide à l'observation,
- Aide au diagnostic.

Services Communs de gestion

- Production de documents pour toutes les activités,
- Gestion de configuration des produits de l'AGA ,
- Gestion de bibliothèques de composants réutilisables,
- Couplage à des ateliers externes.

L'ensemble de ces fonctions doivent être réalisées autour d'une structure d'accueil qui garantit ainsi la cohérence et l'unicité des données acquises ou modifiées par un quelconque outil composant l'AGA. Ces principes sont largement évoqués dans le chapitre d'introduction de la norme AFNOR Z68901.

A terme seront à envisager d'autres configurations de l'atelier, concernant le développement des systèmes supports, l'étude et la mise en œuvre de la partie opérative des SAP.

3.3.5 LES SUJETS DOMINANTS DE RÉFLEXION LIÉS À CE PROJET

Ce projet comporte trois thèmes dominants de réflexion:

- la méthodologie de développement d'un système automatisé de production industrielle, élaborée par des groupes travail auxquels ont participé les représentants de la plupart des divisions d'application de CEGELEC,

- les fonctionnalités et l'architecture de la structure d'accueil, sujet traité par l'équipe chargée précisément d'étudier l'AGA,

- les méthodes de conception associées à la méthodologie, les outils supportant ces méthodes ainsi que leurs modèles de données internes, thème plus particulièrement lié à notre recherche que nous évoquerons à travers deux phases du cycle de vie du SCC;

  . la spécification et la conception du système (cf. 0.3.3.4.)
  . la conception de ses applications d'automatisme.

La méthode présentée ci-après propose de concevoir le système de contrôle-commande en réutilisant des produits standards d'automatisme.

Pour ce faire, elle tire parti d'une modélisation axée sur l'usage des constituants matériels utilisés pour composer une gamme d'équipements permettant de mettre en œuvre des systèmes de toutes tailles et de toutes complexités.

Elle tient également compte des caractéristiques des éléments de logiciels employés pour réaliser des systèmes **appliqués** de contrôle-commande.

PLAN DU MÉMOIRE

Notre mémoire débute donc par un premier chapitre s'attachant à définir ce qu'est un système appliqué de contrôle-commande et par quelles grandes phases doit passer son développement. Cette partie précise de plus nos hypothèses principales d'étude.

Il se poursuit, dans le contexte de l'élaboration d'un atelier de génie automatique, par l'exposé, au chapitre II, des principes permettant de modéliser l'architecture matérielle d'un SCC, puis, au chapitre III, par la présentation d'une série de concepts inhérants à l'application logicielle portée par les équipements de traitement d'un tel système.

Il rentre dans le vif du sujet aux chapitres IV et V en présentant notre proposition en terme de méthode de conception engageant la réutilisation de constituants matériels standards et de composants logiciels s'accordant à représenter l'aspect tranformationnel des programmes développés pour un système.

Une conclusion vient clore notre exposé en présentant les principaux apports de notre réflexion.

CHAPITRE I LES SYSTÈMES APPLIQUÉS DE CONTRÔLE-COMMANDE

I.1 LEURS CARACTÉRISTIQUES PROPRES

I.1.1 COUCHES DE STRUCTURATION

Nous avons donné en introduction de cet ouvrage un premier aperçu des SCC, en les caractérisant par rapport aux SAP.

Une vision plus détaillée de ces systèmes nous amènerait directement à considérer leurs composants caractéristiques, éléments matériels et logiciels qui, une fois assemblés, permettent au système de contrôle-commande d'assurer ses propres fonctions.

Mais parallèlement à cela, un SCC peut aussi être représenté en plusieurs couches, témoignant du niveau de service offert par ses composants. En effet, il peut, de cette façon, être fait état du rôle intrinsèque de chaque élément de la structure d'un système en respectant une classification proche de celle adoptée dans le modèle ISO des réseaux de communication (couches physique, couche liaison, couche transport, etc ...).

De cette classification émanent par exemple les 4 couches d'un système programmé, 4 couches aux niveaux desquelles se dégagent les ensembles réutilisés d'un projet à l'autre, les composants identiques à plusieurs types de systèmes, les éléments propres à un domaine particulier d'application .. et ainsi de suite, si l'on parcourt, de haut en bas, la pyramide ci-dessous:

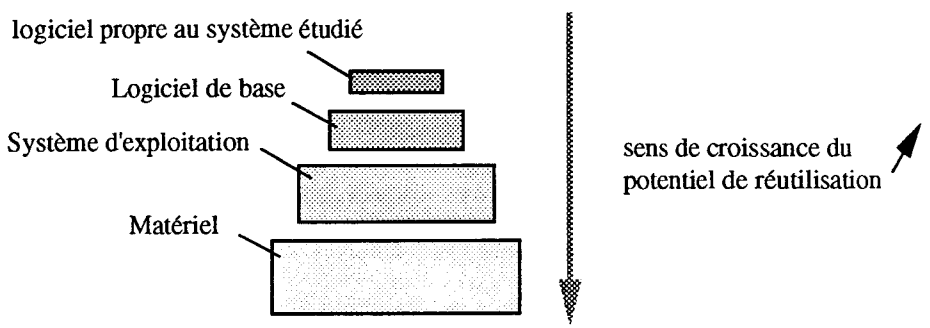


figure 5: les couches de structuration d'un système programmé

Aux couches de structuration du système sont donc associés des niveaux de réutilisabilité qui confèrent au SCC dans son ensemble et selon les circontances, un potentiel plus ou moins important de réutilisation.

Une classification par **niveaux de réutilisation** suit en particulier le modèle en couches présenté figure 5 et permet ainsi de faire la part entre les éléments réutilisés et les éléments spécifiques à l'application étudiée.

Certains systèmes laissent apparaître cette limite dès la couche "matériel", d'autres reprennent ou adaptent des constituants génériques jusqu'au niveau de leur logiciel de base (services de gestion de configurations redondantes ou, plus généralement, services de communication réseau).
Cette frontière n'est donc pas fixe. Elle définit le niveau de généricité de l'application et varie, en pratique, selon le type de process à automatiser (ou encore selon la nature de l'application). Certaines applications réclament en effet des développements spécifiques qui se sauraient être facilement réemployés dans d'autres projets.

I.1.2 SYSTÈMES SPÉCIFIQUES, SYSTÈMES APPLIQUÉS

La frontière précédente permet de distinguer les couches "application" d'un système de ses couches dites "support":

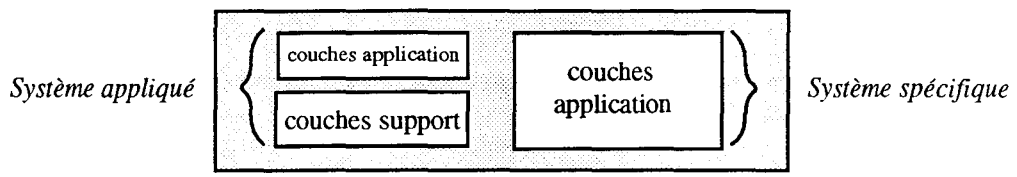


figure 6: Système appliqué / Système spécifique

Elle donne à la comparaison son extrême lorsque sont mis, face à face, un **système *spécifique*** (qui ne s'appuierait sur aucun savoir faire matérialisé) et un **produit** (système support ne réclamant au contraire aucune adaptation particulière).

Les couches supports sont faites d'éléments immuables offrant, chacune à leurs niveaux, des ressources pour les éléments des couches supérieures:

L'aspect matériel s'offre en effet en structure d'accueil au logiciel du système, constitué d'un système d'exploitation plus ou moins standard (VRTX, PSOS etc ...) et de couches de programmes adaptées aux types de traitements à mettre en œuvre (traitements d'acquisitions périodiques, traitements réflexes, traitements de contrôles séquentiels ... etc ...).

Le logiciel système gère des ressources exploitées par les niveaux plus élevés de structuration: files d'événements et files de messages entre tâches pour les couches basses, variables logicielles et passages de paramètres pour les couches les plus hautes.

Dans la pratique, tout élément nouveau, d'une couche application, peut être l'objet d'une réutilisation rapide, en particulier au sein de la même affaire. Il franchit dans ce cas la frontière qui le sépare des éléments des couches supports et prend place dans ce qu'il est coutume d'appeler le **système support** d'un SCC (cf. § suivant).

Par ailleurs, plus on descend dans la hiérarchie des niveaux structurels du système, plus la réutilisation à des chances de se faire jour. Le meilleur exemple en est celui du "matériel"; il est rare de devoir concevoir une carte de traitement en partant de zéro et de reprendre l'étude complète de son système d'exploitation.

La plupart des configurations actuelles, par exemple, utilisent plutôt des cartes aux formats standards VME, VMX ou compatible PC et vont même quelques fois jusqu'à leur préférer des stations de travail pour des traitements spécifiques de conduite et de supervision.

Beaucoup d'ambiguïtés proviennent d'un manque de caractérisation des systèmes et ne permettent pas de définir une politique très claire face au problème que pose leur conception. Les méthodes proposées à ce jour ([CAL 90], [PAN 91], [ROE 87] par exemple) se disent toutes pouvoir assister la conception mais font à priori peu état du type de système auquel elles sont plus spécifiquement dédiées et mieux encore, bien souvent de la manière dont est appréhendée la réutilisation d'éléments génériques dans cette étape.

Il s'en suit un décalage entre le niveau de besoin ressenti pour le développement et celui auquel satisfait la méthode effectivement utilisée. Le système est, de plus, souvent défini de A à Z sans qu'il soit possible d'y insérer une quelconque instance de constituant générique.

Notre proposition tente de pallier ces deux inconvénients en basant la méthode conception du système sur un modèle à plusieurs facettes sachant apprécier, sur chacune d'elles, les composants spécifiques à développer des constituants génériques à réemployer.

I.1.3 LES SYSTÈMES APPLIQUÉS DE CONTRÔLE-COMMANDE, UN MODÈLE EN TROIS COUCHES

Les systèmes "appliqués" de contrôle-commande industriels sont des systèmes programmés particuliers. Ils répondent de cette manière au mode de structuration précédemment exposé.

Trois couches les caractérisent plus spécifiquement:

- une couche "**matériel**",
- une couche dite "**domaine**",
- une couche "**application**" (la seule à jouer effectivement ce rôle dans ces systèmes).

Ces trois couches permettent de matérialiser la frontière entre le SCC appliqué et son système support, comme l'indique l'illustration suivante:

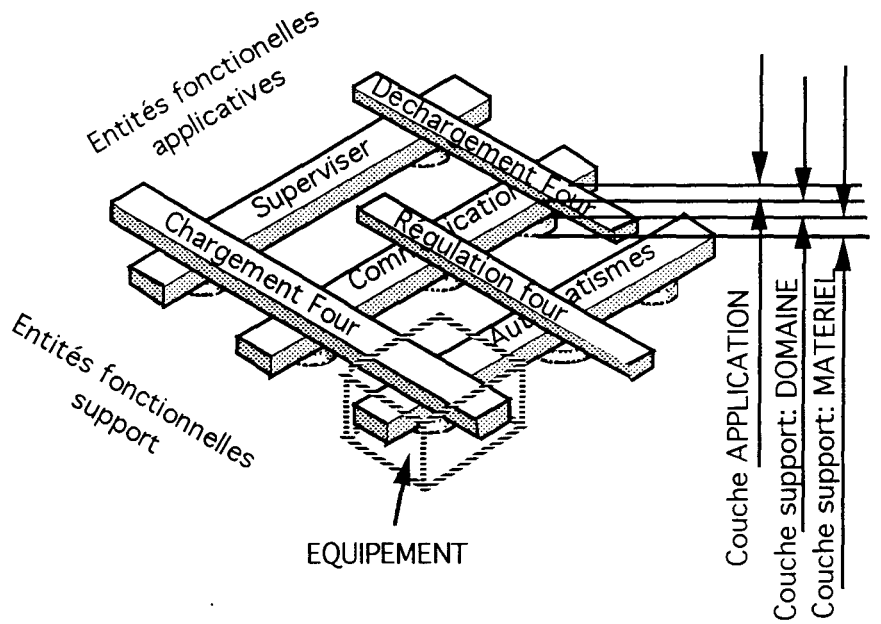


figure 7: Le SCC appliqué, un modèle en 3 couches

- La couche "matériel" constitue le socle du système. Elle est à la base de la structure du SCC et supporte une couche domaine assurant des services standards liés aux différents niveaux CIM de l'installation (cf. I.2.2).
  Exemples d'éléments de la couche domaine: services de communication réseau.
- La couche application est propre au système étudié. Elle dépend plus du process à automatiser et particulièrement des interfaces entre le SCC et sa partie opérative.

Un équipement regroupe, face à cette classification en trois strates successives, un sous-ensemble matériel distinct accueillant une partition de la couche domaine et de la couche application. Cette partition est liée à un sous-ensemble de la partie opérative mettant en œuvre un sous-procédé de production.

I.2 LA CONCEPTION DES SCC APPLIQUÉS

Concevoir un SCC appliqué nécessite de déterminer chacune des couches précédentes en accordant leurs composants aux spécificités du process à automatiser.

Le système est défini équipement par équipement et sont respectivement examinées:

- sa couche support matériel, en déterminant l'architecture matérielle du système,
- sa couche domaine, en choisissant des modèles d'équipements adaptés au traitements d'automatisme, de communication et de supervision,
- sa couche application, en concevant les logiciels implantés sur chacun des équipements de l'architecture.

Dans un ordre logique, est tout d'abord déterminée l'architecture de principe du SCC c'est-à-dire le squelette du système en liaison avec sa couche domaine. Ce squelette est obtenu à partir des éléments relevés en phase de spécification du projet.
Puis, sont ensuite listés les composants matériels de l'architecture (cf. 1.2.2), de façon à stabiliser la structure logique précédente.
En troisième lieu, sont déterminés les éléments de logiciel de la couche application (cf. 1.2.3).

I.2.1 PHASE 1: DÉTERMINATION DE L'ARCHITECTURE DE PRINCIPE DU SYSTÈME

L'architecture matérielle du SCC répond à une organisation de la commande guidée par son architecture de principe. Cette architecture de principe est décrite au début du chapitre II. Elle tient compte des contraintes liées à la topographie du site d'installation (organisation des locaux, répartition des interfaces selon ces locaux), des fonctions que doit réaliser le système ainsi que des types de matériels préposés au choix des équipements du SCC.

Elle réclame la détermination préalable d'entités fonctionnelles représentant de façon exhaustive chacune des fonctionnalités du système (les entités fonctionnelles sont obtenues en phase de spécification du projet). Cette étape permet de définir et caractériser les équipements logiques du SCC en les reliant entre eux par divers moyens de communication (segments de réseaux logiques, liaisons point à point etc ...)

Chaque équipement logique est, à cette occasion, choisi parmi les constituants d'une gamme de systèmes, développée par un constructeur de produits d'automatisme.
Ces constituants se présentent soit en familles de produits simplement compatibles entre eux, soit en ensembles plus spécialement étudiés pour répondre à des types de solutions particulières.
Le système P320 constitue par exemple, dans l'offre CEGELEC, le système support dédié au contrôle-commande de centrales thermiques et hydrauliques de production d'énergie. Il est spécifique par les traits suivants: traitements des signaux d'entrées-sorties, technique de concentration de l'information transmise au superviseur, etc ...
Ces traitements génériques sont figés dans le logiciel de base des cartes intelligentes et dans la bibliothèque de composants standards d'application liés à ce système support.

La réutilisation joue un rôle prépondérant dans cette première phase de conception. Elle réclame une bonne connaissance des produits du marché, connaissance que nous nous proposons de formaliser au chapitre II en dégageant les principaux concepts inhérents à une architecture logique et ses divers équipements.

I.2.2 PHASE 2: CONCEPTION DE L'ARCHITECTURE MATÉRIELLE DU SYSTÈME

Une fois la couche domaine précédente spécifiée intervient ensuite la matérialisation de chacun des équipements de l'architecture. Cette matérialisation est effectuée en regard des familles de constituants entrant dans la configuration des différents matériels connus jusqu'alors uniquement par des références à des modèles d'équipements.

Les équipements du SCC sont , dans cette phase, configurés en fonction à la fois du nombre et de la qualité de leurs interfaces avec la partie opérative. Leur organisation interne dépend par ailleurs des extensions que réclame l'application de certaines de leurs fonctions.

La réutilisation prend alors tout son sens puisqu'un système appliqué de contrôle-commande a, à ce niveau, la particularité d'être presque essentiellement conçu à partir d'équipements préexistants ne demandant qu'à être programmés, ou éventuellement configurés.
La méthode employée, pour déterminer l'architecture physique du système, doit donc permettre de sélectionner, en référence aux modèles d'équipements de l'architecture de principe, des composants entrant dans la constitution de produits à part entière.

La force de la méthode proposée à la fin du chapitre IV traitant de cette phase réside dans le fait qu'elle tire profit de la modélisation opérée tout au long du chapitre II pour faciliter le choix des constituants au moment de la configuration matérielle de chacun des équipements du SCC.

I.2.3 PHASE 3: CONCEPTION DU LOGICIEL D'APPLICATION

Le logiciel de la couche application s'étudie au moyen d'une technique empruntée à la conception des systèmes à fortes contraintes de temps. Cette technique sépare l'étude des comportements attendus du système, permettant de déterminer chacun des processus élémentaires caractérisant ce dernier, de la description de l'aspect transformationnel des éléments d'automatisme gérés par les unités de traitement des équipements de l'architecture.

L'aspect transformationnel d'un processus étant étudié indépendamment de son aspect réactif, la méthode vient à favoriser la réutilisation d'éléments génériques de commande utilisables dans des domaines relativement divers.

La conception du logiciel d'application jette ainsi les bases de la réutilisation d'objet de commandes, comme l'ont fait les types abstraits dans le domaine du logiciel avant qu'apparaissent les techniques objets actuellement utilisées aujourd'hui.

Notre proposition, présentée au chapitre V, s'applique à introduire un style de conception en accord avec les principes généraux de la réutilisation et les particularités d'un logiciel d'automatisme, où le temps revêt une signification particulière.

I.3 HYPOTHÈSES PRINCIPALES DE NOTRE ÉTUDE

I.3.1 SUR LA NATURE DES SYSTÈMES

On classe habituellement les systèmes de production industriels en trois catégories: les systèmes continus, les systèmes discrets et les systèmes mixtes [QUE 93], composés à la fois de sous-systèmes de production continus et de sous-systèmes de production discrets.

- Les systèmes **continus** permettent soit de produire de l'énergie en appliquant un processus de transformation de matière (cas par exemple d'une centrale thermique), soit de transformer de la matière en consommant de l'énergie (cas d'une fonderie). Dans les deux cas, on note un fort couplage entre la matière transformée ou produite et l'énergie obtenue ou consommée.

- Les systèmes à **événements discrets** n'ont pas cette particularité. Au contraire, on note toujours dans ces derniers la présence de "produits" en amont et en aval des différentes étapes du processus décrit par le SAP. Il y a, dans ce type de système, un réel découplage entre le produit subissant des transformations successives et l'énergie participant à ces transformations. Ceci est dû au fait même de l'atomicité relative de la structure du produit.

- Les systèmes de production **mixtes** prennent, quant à eux, partiellement les propriétés d'un système continu et celles d'un système discret [QUE 93]. Les flux de matière se transforment temporairement en flux de produit et inversement après chaque transition continue/discrète ou discrète/continue. Le procédé est alors sécable en sous-procédés disjoints par nature. Il peut ainsi être automatisé par tronçons.

La première partie de notre étude s'applique globalement à la conception du SCC des trois types de SAP précédents. Elle peut donc aussi bien être employée pour des systèmes manufacturiers (classe particulière de systèmes discrets) que des systèmes à production plus largement continue.

Le côté universel de la démarche tient au fait que quel que soit le SAP à mettre en œuvre, les équipements matériels qui composent son système de contrôle-commande sont physiquement similaires et l'on constate par ailleurs que leur organisation reflète, en grande partie, la structuration de la partie opérative (cf théorie du miroir [TIX 89] - tout se passe en effet comme si un miroir venait à être placé entre la partie opérative et la partie commande du système).

Ceci n'est pas le cas du logiciel d'application puisqu'il dépend pour beaucoup du type de procédé à automatiser. Nous ne prétendrons donc pas, dans la seconde étape de la démarche, savoir traiter, avec une égale efficacité, de toutes les catégories de systèmes avec la méthode présentée au chapitre IV.

Nous excluons à priori de cette partie de l'étude les systèmes manufacturiers (classe de systèmes discrets) pour lesquels des approches plus spécifiques de conception, notamment basées sur l'analyse des flux de produits [CRU 91], sont recommandées.

Etant donné que ces systèmes de production laissent constamment apparaître des produits dans le processus de transformation qui les caractérisent, l'étude de leur sous-système de commande revient à modéliser finement le "procédé" en décrivant chacun des états des produits subissant ces transformations.

I.3.2 SUR LE MODÈLE HIÉRARCHIQUE DES SYSTÈMES DE CONTRÔLE-COMMANDE

Nous considérerons des architectures proches du modèle CIM [CIM 88] pour lesquelles on distingue au minimum trois niveaux de commande:

- un niveau 0 d'interface avec la partie opérative: on y attache les équipements de terrain et les contrôleurs d'interface avec la partie opérative,
- un niveau 1 d'automatisme: on y trouve les équipements programmables, configurables ou de logique câblée chargés de l'automatisation du procédé,
- un niveau 2 de conduite qui rassemble les équipements de commande centralisée et de supervision du procédé.

Les niveaux supérieurs (3 et 4 du modèle CIM) sont à priori exclus de l'étude. Ils touchent à la gestion de la production et au respect du schéma directeur de l'entreprise, deux points assez éloignés de la mise en œuvre du procédé lui-même.

I.3.3 SUR LEUR CYCLE DE VIE, ÉQUIPEMENTS ET PHASES CONCERNÉES

Les phases présentées dans le chapitre d'introduction ont été établies par rapport à l'expérience des différentes divisions du groupe CEGELEC. Elle sont le reflet du savoir-faire de l'entreprise dans le métier du contrôle industriel. Nous pouvons nous fier à cet état de l'art.

Notre choix s'est porté en premier lieu sur l'étude des systèmes appliqués. Il représente plus de 80 % de l'effort habituellement consenti à la conception des systèmes de contrôle-commande. C'est pourquoi il est important de se doter d'une méthode efficace permettant de réutiliser, autant que faire se peut, le savoir-faire acquis sur des affaires antérieures, savoir-faire généralement matérialisé, un peu plus tard, par des produits d'automatisme.
La notion de produit (équipement générique que l'on commercialise) ne s'applique, à l'heure actuelle, qu'à des équipements matériels, et ceux-ci, pour des raisons évidentes de confidentialité. Des bibliothèques logicielles existent en effet déjà chez la plupart des offreurs ou sont dans le pire des cas, à l'étude. Les livrer à la concurrence serait commettre une erreur stratégique ne se justifiant pas par la faible masse de systèmes à réaliser.

La capitalisation du savoir-faire de l'entreprise est un objectif majeur en cette époque où une technologie devient désuète en quelques années. Bien résoudre la problématique de la conception passe donc, dans une certaine mesure, par un accompagnement de l'évolution technologique, en fixant, pour chaque composant développé, les clés de leur réutilisation ultérieure.

I.3.4 SUR LES MODÈLES REPRÉSENTATIFS DU SCC

Notre exposé repose sur un modèle standard d'architecture qui fait correspondre à chaque niveau hiérarchique du système de contrôle-commande un type de fonction réalisée par ce dernier.

Nous montrerons à travers la démarche exposée au chapitre IV que cette forme de modélisation satisfait tout à fait les objectifs recherchés en phase de conception "générale" du système.

Elle assure, de plus, une bonne continuité avec l'étape suivante du cycle de vie consacrée à la conception des logiciels de l'application, pour laquelle, nous exposerons une technique d'organisation des programmes, guidée par l'analyse du comportement attendu de la spécification des traitements que doit réaliser chaque équipement de l'architecture.

Nous constaterons alors qu'il est judicieux de concevoir les applications d'automatisme en ayant une approche mixte, c'est-à-dire à la fois descendante et ascendante qui nous fasse bénéficier, au moment le plus opportun, des avantages d'une décomposition par abstraction successive et des qualités que confère une méthode basée sur la réutilisation de composants génériques.

CHAPITRE II MODÉLISATION DE L'ARCHITECTURE MATÉRIELLE D'UN SCC

On constate, au vu d'un système opérationnel de contrôle-commande, que celui-ci est formé d'un ensemble d'équipements spécialisés, de calculateurs, d'automates, de réseaux de communication et de dispositifs d'interface avec la partie opérative.

Cet ensemble s'avère impossible à modéliser selon un seul point de vue.

En effet, l'architecture du système, traduisant l'organisation hiérarchisée de ses équipements, lui donne un sens logique particulier.

L'implantation et la composition de ces équipements matériels lui confère par ailleurs une structure physique propre.

Enfin, chacun de ces équipements assure une série de fonctions applicatives. Le système peut donc également être défini par une arborescence de services plus ou moins élémentaires.

Ainsi, nous choisirons de nous appuyer sur un modèle du SCC comportant trois volets principaux de description:

- un axe logique représentant l'architecture de principe du système,
- un axe physique lié à sa configuration matérielle,
- un axe fonctionnel décrivant les services rendus par chacun de ses équipements.

Les paragraphes suivants passent en revue les définitions des principaux concepts liés à ces trois axes de modélisation. Ils distinguent parmi ces concepts ceux relevant de l'usage des composants et ainsi montrent dans quelles circonstances les réutiliser.

II.1 INVENTAIRE DES COMPOSANTS DE L'AXE LOGIQUE

- L'architecture de principe s'attache à représenter la structure logique du système. Une brève illustration en est donnée figure 8.

Elle fixe ses principaux composants et sous-ensembles.

Les composants sont de deux types: équipements et segments de réseaux de communication.

- De plus, chaque équipement représente autant de "machines" d'exécution qu'il compte d'ensembles de modules assurant des traitements relatifs à l'application.
Ceci implique l'existence d'une entité plus fine de décomposition au sein de l'équipement que nous conviendrons d'appeler **unité de traitement**, par analogie aux produits proposés dans le commerce.

- Par ailleurs, chaque segment de réseau a pour fonction d'assurer une communication entre des équipements, en employant des protocoles et des services adéquats. Il regroupe donc, selon ce point de vue, une série **d'abonnés** logiques.

II.1.1 NOTION D'**EQUIPEMENT** (LOGIQUE)

Un équipement logique représente une unité de fonctionnement propre à un ensemble cohérent de ressources matérielles et de fonctions applicatives. C'est une unité de fonctionnement autonome de l'installation.

En réalité, cette distinction n'est pas aussi simple qu'elle paraît et peut porter à confusion si elle n'est pas suffisamment explicitée. Voici pour exemple deux configurations d'apparences semblables et pourtant bien différentes face à ce mode de décomposition .

Prenons l'exemple d'un automate et d'un contrôleur d'interface procédé, tous deux reliés par à un réseau de terrain. Cette configuration représente sans ambiguïté deux équipements distincts, chacun d'eux étant en effet capable de fonctionner de manière indépendante.

Si nous considérons maintenant le même automate relié à un bac d'entrées-sorties déporté par le biais d'un bus périphérique, cet ensemble ne fait cette fois plus apparaître qu'un seul équipement logique autonome.

La notion d'équipement est donc sujette à des interprétations ambiguës. Nous verrons dans la suite qu'elle est le tenant de la modélisation de l'architecture matérielle du SCC.

II.1.2 NOTION DE **SEGMENT DE RÉSEAU LOGIQUE**

Un segment de réseau est un support de communication connexe entre plusieurs équipements (2 et plus). Il a une portée limitée. Toutefois, celle-ci peut être étendue en lui adjoignant un nouveau segment de même type.
(Un réseau se définit comme un ensemble connexe de segments de réseau).

Un segment de réseau permet d'établir des points de connexion entre équipements logiques.
Il offre un certain nombre de ressources de communication, ressources qui prennent la forme de services élémentaires partageables par des unités de fonctionnement distinctes dans le système.
Il ne représente néanmoins qu'un médium.

II.1.3 NOTION D'**UNITÉ DE TRAITEMENT**

Une unité de traitement symbolise une ressource matérielle pour l'exécution des programmes applicatifs. Son architecture interne est masquée au reste du système et en particulier de ses unités de traitement voisines.

Nous faisons ici la distinction entre les éléments matériels assurant des fonctions de base du SCC, telles qu'acquérir des entrées par le biais d'un mécanisme systématique d'interface par exemple, et ceux assurant l'exécution des programmes de l'application.

II.1.4 NOTION **D'ABONNÉ** LOGIQUE

Un abonné représente une unité de traitement ayant accès aux services d'un segment de réseau.

La figure ci-après présente un exemple particulier d'architecture logique.
Cette architecture de principe identifie les équipements de contrôle-commande par rapport au lieu de leur installation, au moyen de repères sur le site.

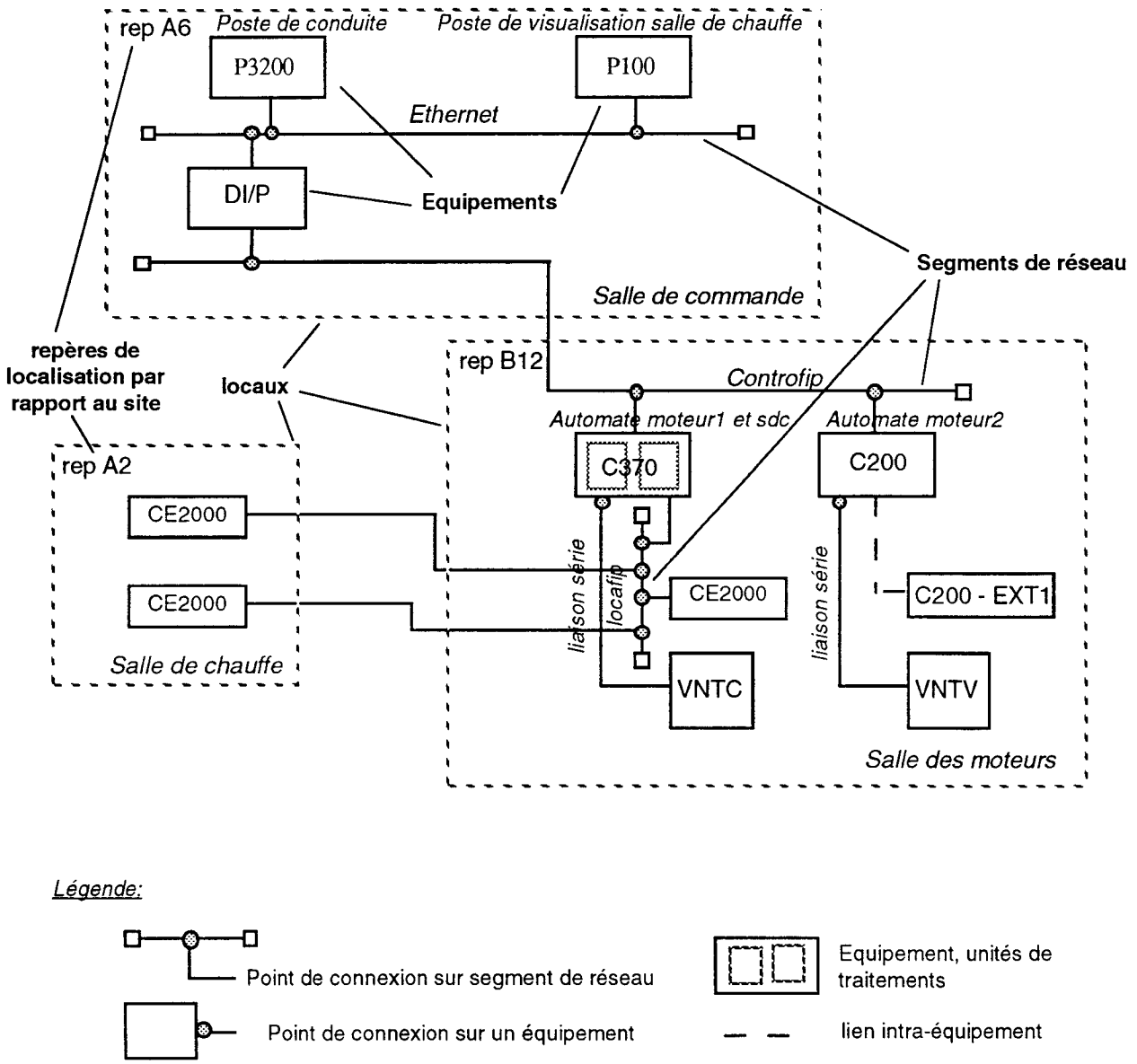
II.1.5 ELÉMENTS DE TOPOLOGIE: DISTINCTION ENTRE SEGMENT DE RÉSEAU ET LIAISON POINT À POINT

- Un segment de réseau représente une liaison multipoints décentralisée entre équipements. C'est un canal de communication sur lequel se connectent des abonnés.

Nous le représenterons donc par un segment de droite associé à deux délimiteurs logiques. (cf. figure ci-après)

- Les équipements de l'architecture peuvent également dialoguer deux à deux par le biais de liaisons séries ou parallèles ou plus généralement en mode multipoints centralisé.

Ces relations entre composants seront représentées par de simples points de connexion dans l'architecture:



Légende:

figure 8: Exemple d'architecture de principe

II.1.6 PORTÉE DE **LA COMMUNICATION** RÉSEAU

Un **segment** de réseau définit un espace de communication au sein du SCC.
Cet espace limite les échanges d'information entre équipements à des groupes d'abonnés.

Franchir la barrière d'un segment de réseau nécessite donc aux abonnés de faire appel aux services d'une passerelle ou d'un pont entre segments qui sont tous deux des équipements de types particuliers.

Une passerelle permet de relier deux segments de réseau utilisant un même protocole de communication alors qu'un pont établit un point de connexion en deux segments de utilisant des protocoles différents.
- L'équipement reliant les réseaux Ethernet et Controfip de la figure 8 constitue un exemple de Pont.

Le système de communication du SCC suit l'organisation hiérarchisée de ses équipements. Il est ainsi structuré en segments placés aux niveaux 0, 1 ou 2 de l'architecture.

II.2 AXE LOGIQUE: NOTIONS IMPORTANTES POUR LA RÉUTILISATION

"Bien réutiliser réclame avant tout de recenser l'existant et en particulier de s'en faire une idée aussi précise que possible".

Pour le cas des équipements de contrôle-commande, on adopte généralement une typologie à deux niveaux; un niveau dit de "couverture fonctionnelle" à partir duquel sont définis des équipement-types et un niveau proche des solutions matérielles s'appuyant sur des modèles d'équipements.

II.2.1 NOTION D'**ÉQUIPEMENT-TYPE**

Chaque équipement joue un rôle particulier dans le système. On note des types de rôle standards auxquels correspondent des types d'équipement ou "équipement-types". Ces genres sont liés au besoin et restent indépendants des solutions proposées.

Exemples d'équipement-types: automate programmable Industriel, équipement de terrain.

II.2.2 NOTION DE **MODÈLE D'ÉQUIPEMENT**

Un modèle d'équipement définit une solution répondant à un type de besoin. Il fait référence à une série de caractéristiques techniques définissant un produit dans une gamme de matériels d'automatisme.

Exemple: automate ALSPA C370

II.2.3 **TYPES** ET **MODÈLES** DE SEGMENTS DE RÉSEAU

De la même façon, *un segment de réseau type représente un genre de segment de réseau en lui assignant des utilisations types.*

Un modèle de segment de réseau définit en sus une série de caractéristiques techniques telles que le protocole supporté, la vitesse de transmission sur la ligne, le type de médium employé ou le mode d'accès réservé à ses abonnés.

Ces concepts sont clés pour la réutilisation car ils déterminent toute la sémantique du modèle logique de l'architecture.

- Comme le montre la figure 9, l'architecture de principe n'a de sens que si ses composants sont suffisamment caractérisés et notamment s'ils font référence à des modèles explicites d'équipements et de segments de réseaux de communication.

Nous définirons, en synthèse de tout ce qui a été énoncé jusqu'à présent et à l'aide du formalisme présenté en détail dans l'annexe 1 le sous-schéma conceptuel ci-dessous:

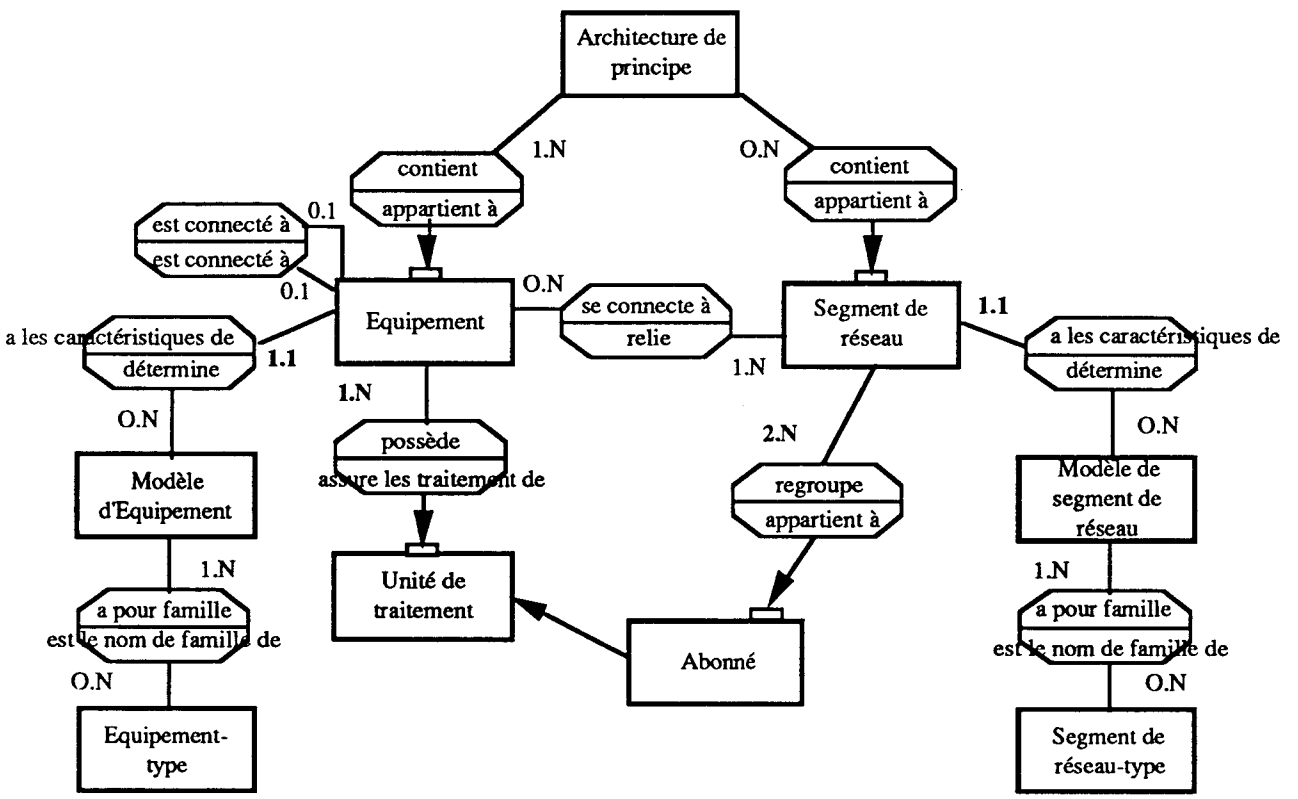


figure 9: modèle conceptuel binaire de l'axe logique

Remarques:

1) un "abonné" n'est qu'un rôle attribué à certaines des unités de traitement du SCC. Autrement dit, l'architecture logique du système n'est pas nécessairement constituée d'un ensemble de composants connexes.

2) les cardinalités qui apparaissent en lettres grasses sur la figure 9 expriment le minimum requis pour la recevabilité du modèle logique. Il va de soi que cette esquisse s'obtient par une estimation des ressources système nécessaires à la mise en œuvre des fonctions identifiées dans l'application.

Nous reviendrons sur ce sujet en présentant les concepts relatifs à l'axe fonctionnel et dans l'exposé de la démarche de conception de l'architecture de principe du SCC.

II.3 INVENTAIRE DES COMPOSANTS DE L'AXE PHYSIQUE

- Notre but n'est pas de modéliser la structure physique complète du système mais de fournir un cadre précis de description des équipements matériels contribuant à l'automatisation du procédé. Nous n'aborderons donc pas la décomposition des équipements de conduite et de supervision. L'architecture physique du système restera de ce fait, par endroits, incomplète.

- Un SCC se présente matériellement sous la forme d'un ensemble organisé de composants physiques. En tenant compte de la restriction de l'alinéa ci-dessus, nous pouvons admettre qu'ils sont globalement de 6 types:

- structures électriques et mécaniques d'accueil (armoire et coffrets)
- structures d'accueil de type fond de panier (bacs + fond de panier)
- modules électroniques
- cartes filles
- accessoires
- composants de segments de réseau,

sachant qu'une structure électrique d'accueil est le sous-système physique qui a pour but d'alimenter un équipement et qu'une structure mécanique d'accueil est le support mécanique de ses bacs de type fond de panier.

- Par analogie à la définition d'un équipement logique présenté au paragraphe **II.1**, un équipement matériel devient *l'ensemble des composants physiques matérialisant une unité de fonctionnement;* ce qui signifie que cette unité matérielle ne contient ni des structures électriques d'accueil (bacs de préalimentation par exemple), ni des structures mécaniques d'accueil.

Comme le montre la figure suivante, sa fourniture s'arrête à des bacs de types "fond de panier".

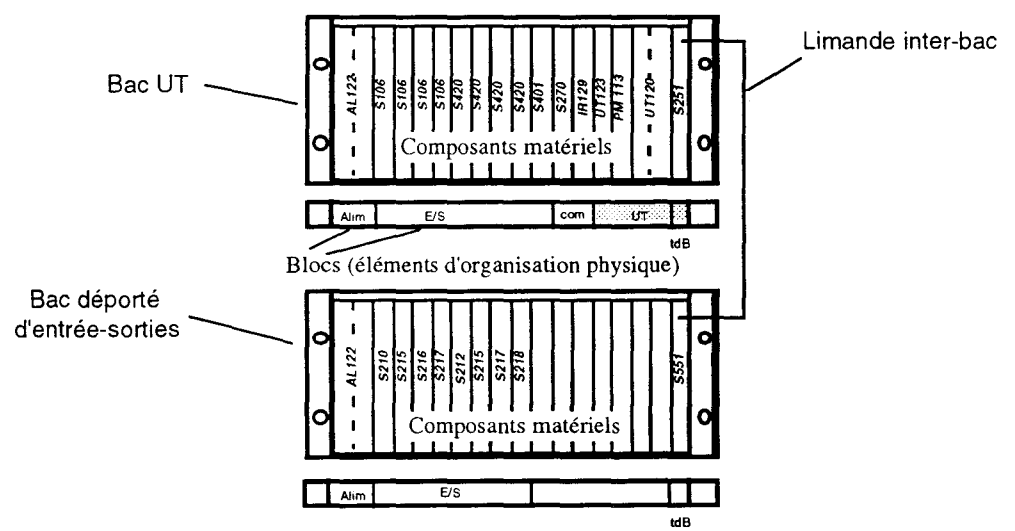


figure 10: Exemple de configuration matérielle d'un équipement

II.3.1 NOTION DE **BLOC**

Physiquement et comme l'illustre la figure précédente, un équipement regroupe un certain nombre de bacs dans lesquels sont enfichés des modules électroniques. Ces modules assurent des fonctions de plusieurs natures.

On regroupera les cartes de même nature dans des blocs afin de discerner leurs types de rôles respectifs.

Exemples de types de rôles: assurer des traitements applicatifs, réguler l'alimentation électrique du bac, assurer l'acquisition et la restitution de signaux d'interface avec la partie opérative ...

Exception faite des blocs matérialisant des unités de traitement, cette notion n'a pas trait à la décomposition logique de l'architecture.

En effet, prenons l'exemple des blocs "alim" précédents, chacun d'eux alimente un bac ne constituant qu'une partie de la configuration matérielle de l'équipement présenté figure 10.

➡ Ils n'ont de ce fait qu'un rôle partiel dans le fonctionnement de l'équipement.

II.3.2 LIENS ENTRE ÉQUIPEMENTS MATÉRIELS ET STRUCTURES D'ACCUEIL

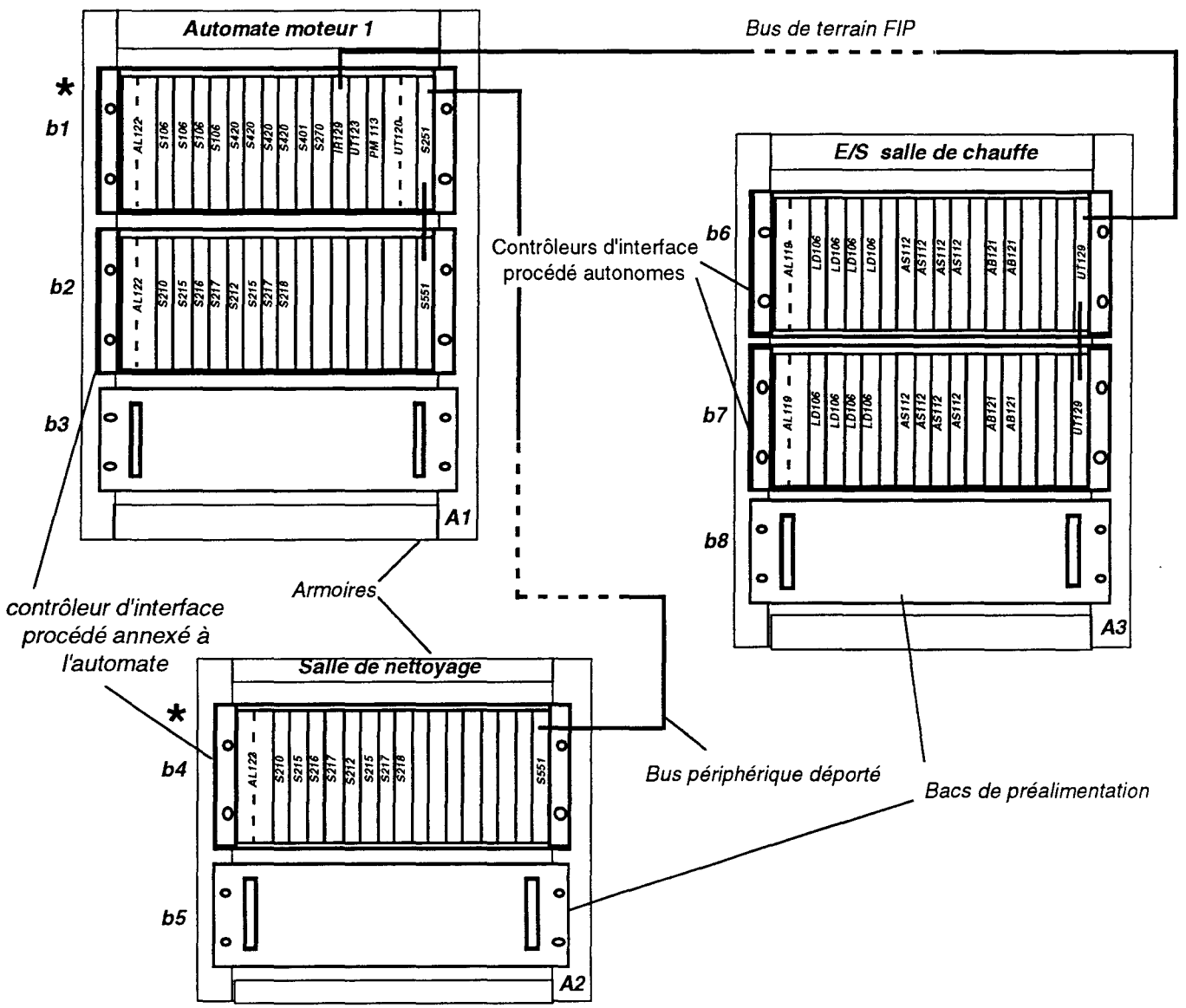
Nous constatons par ailleurs qu'une armoire peut accueillir des bacs de types fond de panier composants de plusieurs équipements logiques sans que la configuration matérielle de chacun de ces équipements ne soit intégralement portée par ces bacs.

L'organisation matérielle du système ne suit donc plus celle qui contribue à définir ses équipements et segments de réseaux . Elle répond à une modélisation en termes physiques de l'architecture du SCC.

Les équipements matériels sont en particulier structurés en fonction de leur aire d'influence sur la partie opérative, par composition des éléments d'interface avec le procédé.
Ils s'attachent de plus à respecter les contraintes générales d'environnement qu'impose l'utilisation de leurs composants respectifs à travers leurs caractéristiques de fonctionnement.

Ainsi, la description de l'architecture physique du système épouse la topographie du site d'installation en représentant l'arborescence des composants matériels du SCC.

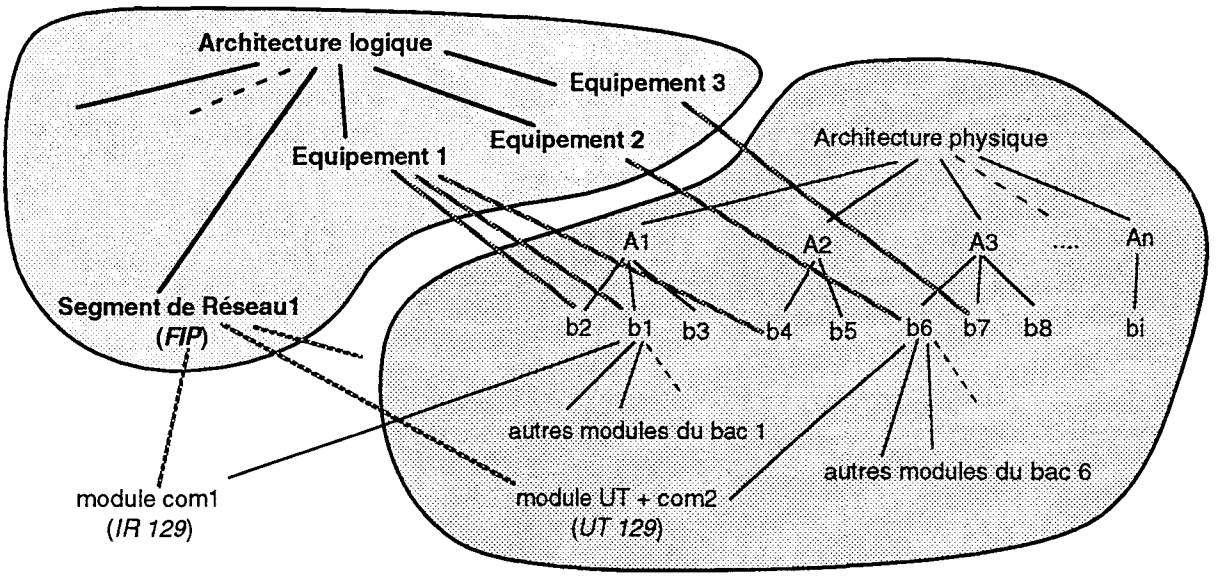
Elle ne traduit donc pas les liens sémantiques existant entre ces composants, que traduit précisément l'architecture logique du système. (le bac b1 ci-dessous fait en effet partie du même équipement logique que les bacs b2 et b4)



(L'exemple ci-dessus intègre la configuration matérielle de l'équipement développé en figure 10. Deux bacs correspondant à la configuration précédente sont marqués d'un astérisque)

figure 11: Partition d'une architecture physique

La partition de la figure 11 se traduit par les deux architectures suivantes:



A1..An	*liste des structures mécaniques d'accueil*
b1 ...bi	*Liste des bacs de type fond de panier*

figure 12: Structurations physique et logique des composants matériels de la partition de système présentée figure 11

Nous distinguons ici pleinement la complémentarité des deux modes de décomposition physique et logique.

L'architecture logique s'arrête d'une part à la description des équipements et segments de réseaux et lie ces composants à des bacs et des modules électroniques. Elle formalise d'autre part les unités de traitement de ces équipements pour les relier à des ensembles de constituants matériels répondant à la notion de bloc physique.

L'interface entre les axes logiques et physiques de représentation du SCC ne se limite néanmoins pas à ces deux types de relations. Elle s'inscrit également dans des liens du type segment de réseau - accessoires de continuité ou segment de réseau et constituants de types filaires.

Pour des raisons de place, ces derniers ne sont pas évoqués dans le schéma partiel de la figure 13. (Se reporter à l'annexe 2 pour plus de détails)

nota: le modèle de la page suivante exprime 4 relations d'interface entre les axes logique et physique de décomposition de l'architecture. Deux d'entre elles sont équivalentes et reliées par une contrainte procédurale. Ceci est dû au fait de la double approche équipement - segment de réseau de communication et signifie en pratique qu'il y a équivalence occurrentielle entre un module matérialisant un composant d'un segment de réseau logique et le même module associé au bloc représentant l'abonné à ce segment de réseau.

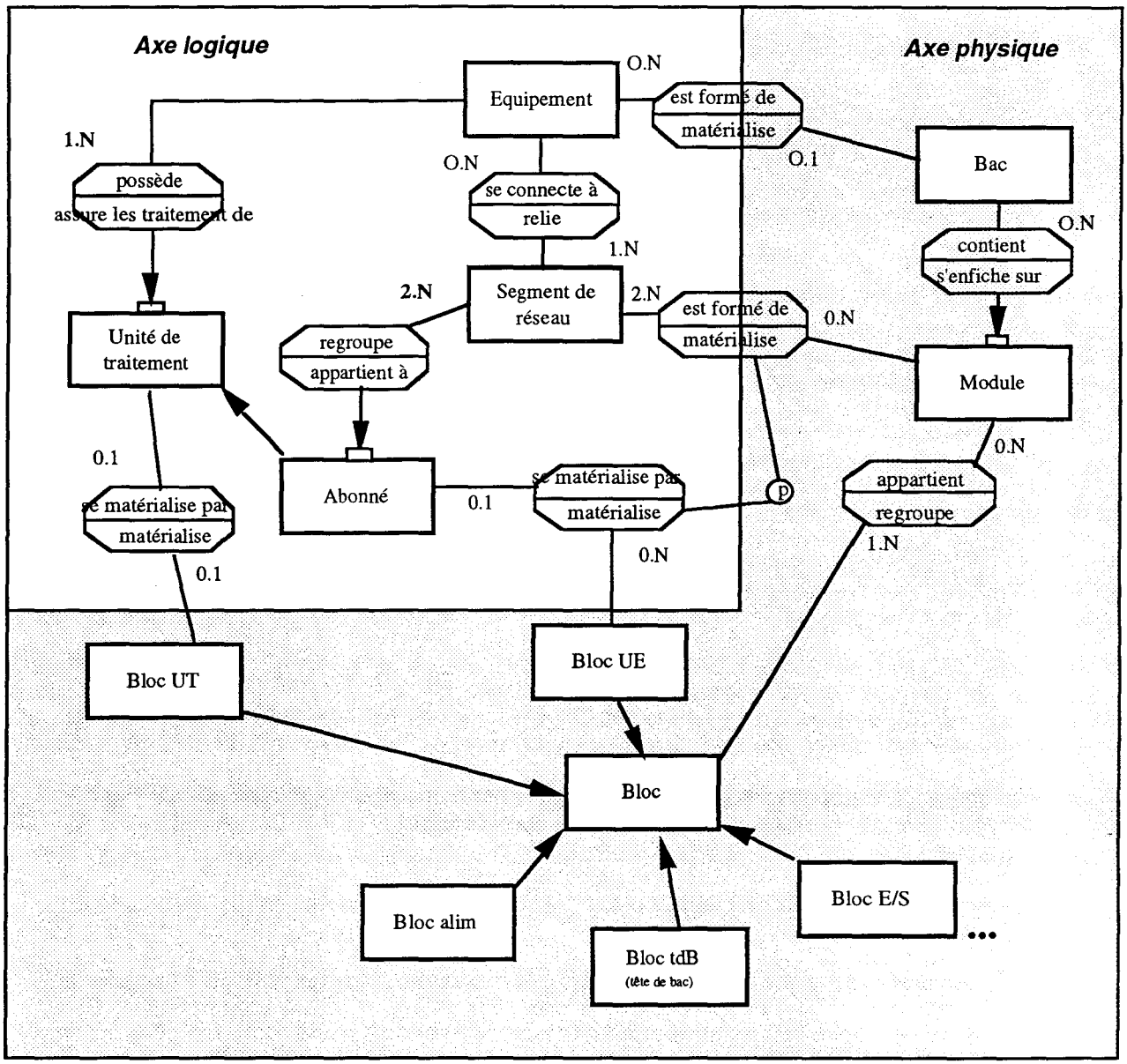


figure 13: schéma de définition de l'interface physico-logique.

II.4. AXE PHYSIQUE: NOTIONS DE RÉUTILISATION

II.4.1 NOTION DE FAMILLE DE CONSTITUANTS MATÉRIELS

-Les constituants entrant dans la composition des équipements et segments de réseaux de l'architecture respectent un certain nombre de standards faisant pour la plupart l'objet de normes nationales ou internationales. Qu'il s'agisse de caractéristiques dimensionnelles, électriques, électromagnétiques ou thermiques, ces constituants s'insèrent de fait dans des gammes de produits identifiées chez les fournisseurs.

Ainsi, CEGELEC commercialise sous la marque ALSPA une gamme de constituants d'automatisme respectant des caractéristiques dimensionnelles et électriques communes basées sur le format double Europe VME-VMX.

Nous conviendrons d'appeler une telle gamme de produits une famille de constituants matériels.

II.4.2 IDENTIFICATION DES PRINCIPAUX TYPES DE NŒUDS TECHNOLOGIQUES:

- Au sein d'une famille donnée et comme pour l'axe logique au niveau duquel ont été définis des types et modèles d'équipements et de segments de réseaux, l'axe physique comporte des composants répondant à des genres particuliers. Cette notion déjà présentée en II.1 ne fait toutefois pas directement référence à l'usage de ces constituants matériels. Nous conviendrons de faire référence à cet usage en introduisant deux concepts nouveaux correspondant à deux types de nœuds technologiques [Z99-1].

II.4.2.1 NOTION D'ENSEMBLE DE CONFIGURATION

Un ensemble de configuration désigne l'attachement d'un modèle de structure d'accueil à un modèle d'équipement et à un mode d'utilisation particulier. C'est en quelque sorte un morceau d'architecture-type auquel est associé une liste de constituants et des règles d'assemblages.

Toute structure d'accueil, électrique, mécanique ou de type fond de panier, se pliant à une utilisation particulière, c'est-à-dire devant assurer une liste déterminée de fonctions de base, laisse apparaître en plus des spécificités d'emploi.

Deux de ces spécificités sont remarquables dans le cas des bacs de type fond de panier: leurs capacités d'accueil en modules et les contraintes de placement des cartes permettant d'engendrer une telle configuration-type.

Ceci est principalement dû au fait qu'à un mode d'utilisation d'un bac correspond un sous-ensemble de modules susceptibles de rendre les services escomptés et que ces modules répondent, par ailleurs, à des règles d'association très précises. Ils occupent donc, par recoupement de possibilité, des emplacements compatibles avec leur fonction.

Nous attacherons ainsi des ensembles de configuration à chaque modèle de structures d'accueil relevé dans une famille de produits.

Exemple:

Le bac b1 de la figure 11 correspond à l'utilisation du modèle de bac ALSPA S802 en bac "mono unité de traitement".
Cet ensemble de configuration ne tolère qu'un nombre limité d'agencements intérieurs différents que définit la liste de réservation suivante:

emplacements -1..0: alimentation; emplacements 1..11: modules d'entrées-sorties ou de communication avec d'autres équipements; emplacement 12..15: modules de l'unité de traitement et emplacement 16 réservé au module tête de bac (tdB).

II.4.2.2 NOTION DE NŒUD TECHNO-FONCTIONNEL

Un nœud techno-fonctionnel associe, au sein d'une famille de constituants, des éléments liés à une même technologie et assurant la même fonction de base. C'est un nœud dans l'arbre représentant un catalogue de constituants. Il est le fruit d'une classification des constituants par le fournisseur de produits.

Un ensemble de configuration étant lui-même représentatif d'une collection de fonctions, nous pouvons en déduire qu'un nœud techno-fonctionnel regroupe les constituants rentrant dans l'élaboration d'un bloc physique tel qu'il a été défini en II.3.1.

Ainsi, des nœuds techno-fonctionnels seront attachés à l'exemple de structure d'accueil b1. Ils permettront de construire les blocs d'alimentation, d'entrée-sortie, de communication, d'unité de traitement et de tête de bac du bac S802 de l'exemple présenté figure 11.

☞ Toutes ces notions ne préjugent en rien du fait de disposer de relations ponctuelles de dépendance entre articles constituants.
Elles contribuent toutefois à limiter le nombre de ces relations et fixent à un niveau conceptuel, l'organisation de la base de "connaissance" à partir de laquelle nous pourrons envisager la réutilisation systématique de composants matériels d'automatisme, en exploitant directement les données d'un catalogue informatisé de constituants couplés à des outils d'assistance à la conception des systèmes.

La figure suivante donne une image synthétique de l'organisation préconisée pour un tel catalogue électronique:

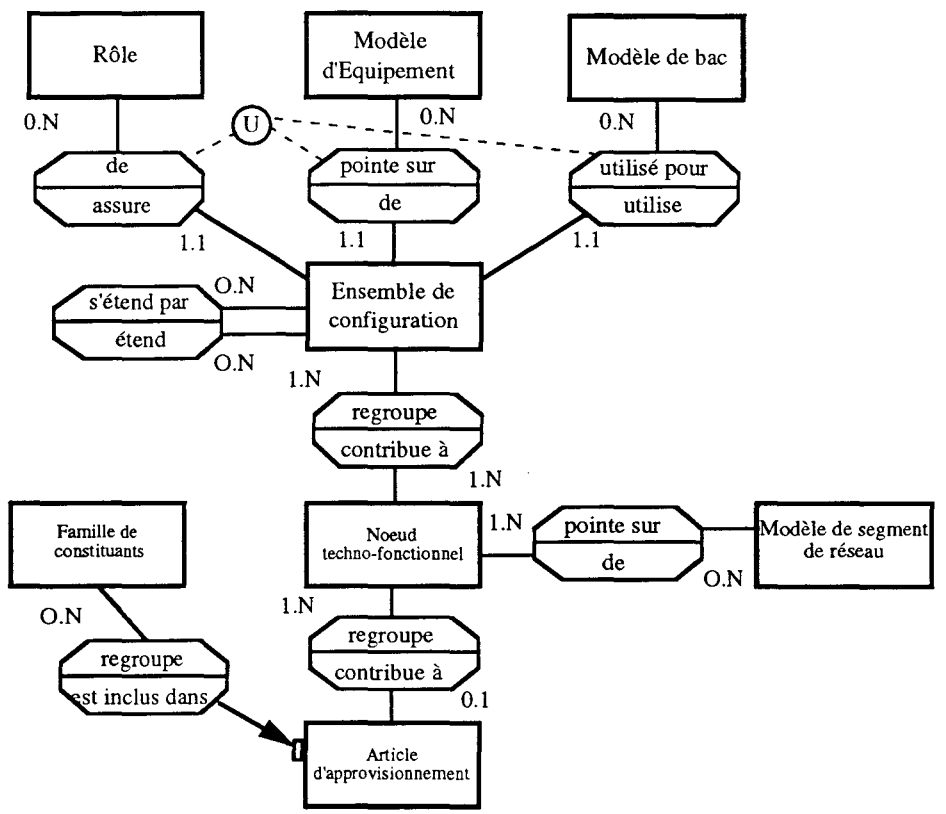


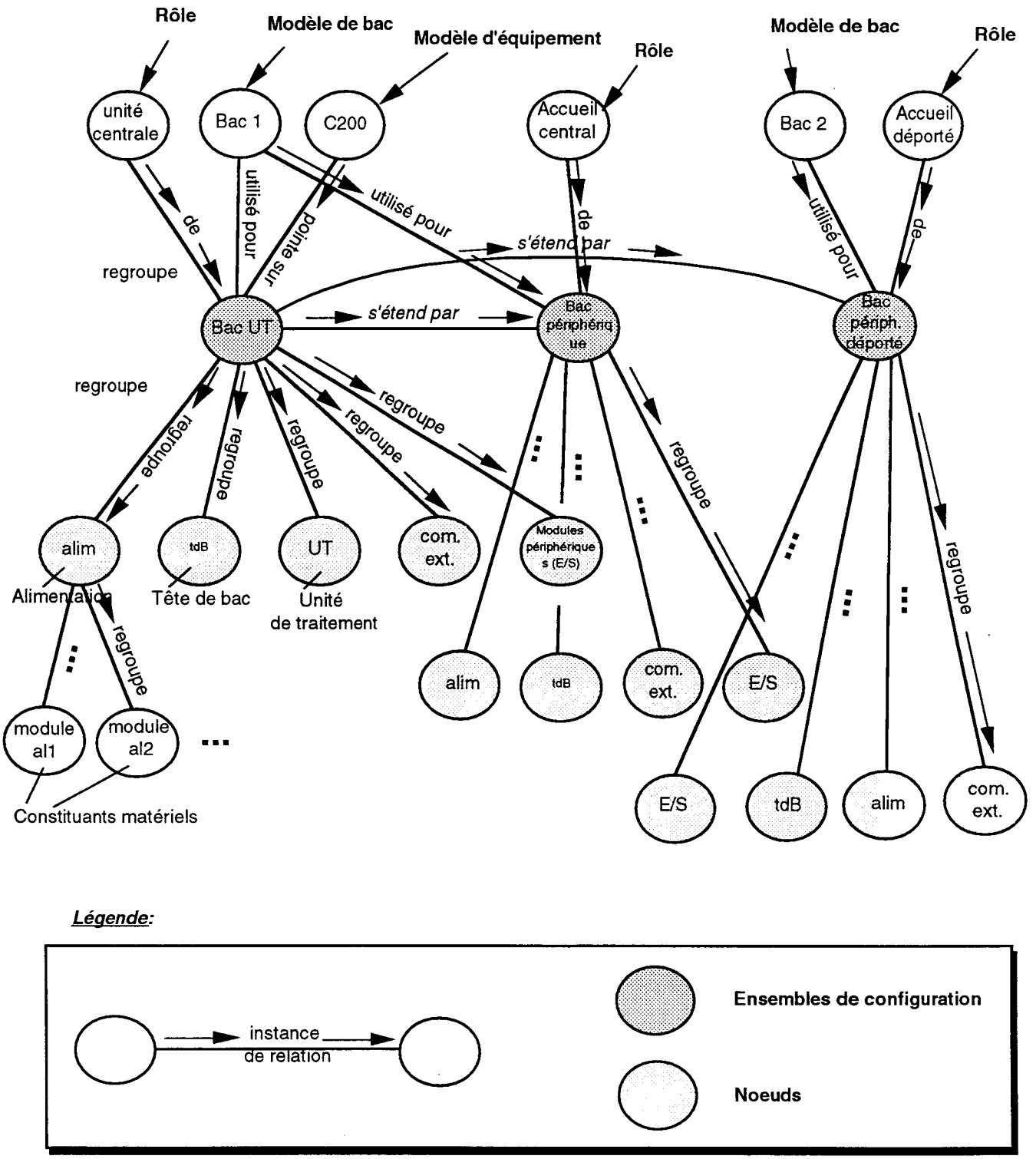
figure 14: schéma partiel de structure d'une bibliothèque de constituants matériels

Cette structure partielle de la base doit en particulier être complétée par des sous-schémas fixant un cadre d'accueil pour données et caractéristiques techniques. L'annexe 2 présente une organisation plus complète des informations caractérisant une bibliothèque informatisée de constituants.

Elle présente en particulier un premier niveau de décomposition attaché à des catalogues-produits matérialisant les familles de constituants compatibles identifiées dans une gamme donnée de produits d'automatisme.

La figure située page suivante tente d'éclater la structure générique du schéma 11 en instanciant quelques nœuds techno-fonctionnels et ensembles de configuration.

Schéma d'instanciation-type du modèle conceptuel précédent pour un automate particulier pourvu de bacs d'entrées-sorties périphériques centraux et/ou déportés:



Légende:

figure 15: Structure développée d'une bibliothèque de constituants

II.5 INVENTAIRE DES COMPOSANTS DE L'AXE FONCTIONNEL

Nous abordons maintenant la dernière facette de modélisation de l'architecture d'un SCC. Elle a pour but de représenter l'arborescence fonctionnelle de l'application formée par ce système.

Nous distinguerons à ce niveau 5 notions:

II.5.1 L'ENTITÉ FONCTIONNELLE

Une entité fonctionnelle est le résultat d'un partitionnement du système en sous-systèmes par grande fonction de contrôle-commande. C'est une unité d'œuvre issue du découpage du procédé dans un choix qui convient plus particulièrement à l'application.

II.5.2 L'ENSEMBLE TOPO-FONCTIONNEL (ETF):

Un ensemble topo-fonctionnel conduit au partitionnement du SCC selon la localisation de ses fonctions.

Il y a en effet similitude fonctionnelle entre l'organisation des équipements d'automatisme et l'organisation de la partie opérative qu'ils contrôlent.

- Un ETF regroupe par conséquent les fonctions assurées par les différentes unités de traitement des équipements situés dans un même lieu.

II.5.3 LES FONCTIONS

Une fonction est un partitionnement du SCC à concevoir selon un critère de service rendu.

Seules sont considérées à ce titre les fonctions applicatives du système, c'est-à-dire les services du logiciel de l'application.

II.5.4 LES INTERFACES EXTERNES

Une fonction applicative peut être liée à des interfaces externes au SCC.
Ces interfaces sont de trois types:

- interfaces avec la partie opérative,
- interfaces de conduite,
- interfaces avec d'autres systèmes.

II.5.5 LES INTERFACES INTERNES (OU FONCTIONNELLES)

Une fonction peut également être liée à une autre fonction par l'intermédiaire d'interfaces (internes au SCC).

II.6 AXE FONCTIONNEL: RÉUTILISATION

II.6.1 RÉUTILISATION SUR PLUSIEURS AFFAIRES

Bien que cet axe soit, par essence, propre à chaque système appliqué, il est toutefois possible, quelle que soit la méthode d'analyse fonctionnelle employée pour sa construction, de privilégier un point de vue qui permette finalement d'aboutir à des fonctionnalités "génériques", c'est-à-dire des fonctionnalités qui pointent vers des éléments de solutions réutilisables entre plusieurs affaires.

Ces fonctions apparaîtront au titre de fonctions techniques constructeurs dans le chapitre III. Elles sont liées au domaine de l'étude et constituent la réponse du fournisseur au besoin énoncé par le client. (cf. couche domaine du chapitre précédent)

II.6.2 RÉUTILISATION AU SEIN DE LA MÊME AFFAIRE

Le caractère répétitif de certains procédés ou de certains organes de la partie opérative, incite, par souci de standardisation influant de manière considérable sur la qualité et la maintenabilité du système, à décomposer ses fonctions applicatives en sous-fonctions élémentaires appliquées à résoudre une "tranche" réutilisable du procédé.

Cette technique sera employée chaque fois qu'il est possible. Elle nécessite, avant de spécifier le système de contrôle-commande, d'en factoriser les besoins pour dégager des sous-ensembles de services génériques à l'affaire.

II.7 CONCLUSION: LES TROIS FACETTES D'UN ÉQUIPEMENTS

☞ En conclusion de tout ce qui précède, un équipement se représente selon trois facettes distinctes:

- une facette logique dévoilant ses unités de traitement
- une facette physique représentant sa configuration matérielle détaillée,
- une facette fonctionnelle exprimant les services réalisés par chacune de ses UT.

Cette remarque nous permet de fixer de façon plus conséquente les liens existant entre les différentes entités représentatives des ces trois axes de modélisation.

Nous ne donnerons néanmoins en page suivante qu'un schéma partiel simplifié du modèle du SCC pour garder une certaine cohérence dans notre présentation.

Le schéma de la figure 16 est partiel dans le sens où il ne représente que les composants hiérarchiquement les plus élevés de l'architecture du SCC, exceptée l'architecture elle-même.

Un modèle plus détaillé est proposé en annexe 3.

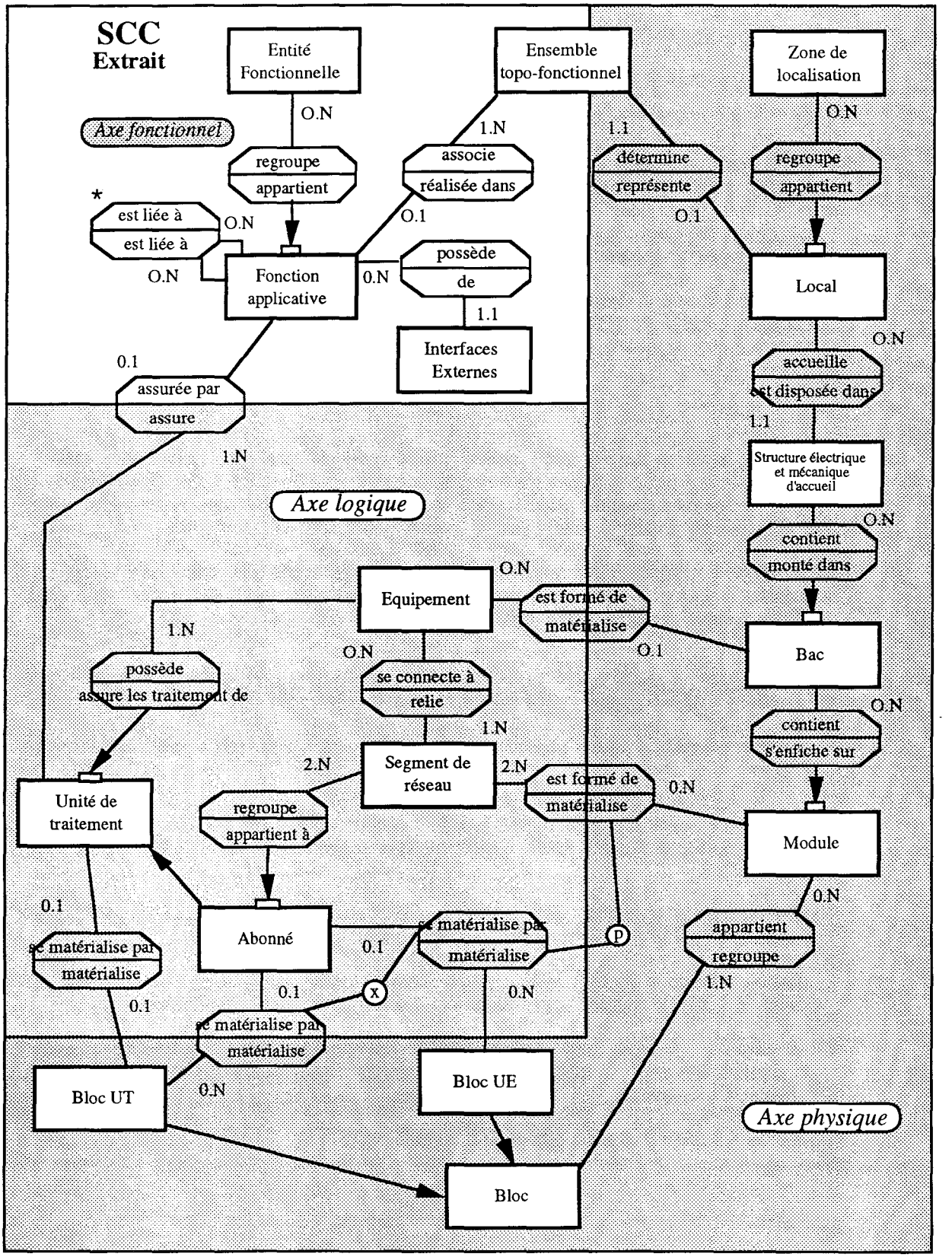


figure 16: jonction des trois axes de modélisation

Nature des liens entre une application matérielle et son catalogue de référence:

1) Les composants matériels d'un système appliqué de contrôle-commande sont liés par des liens de référence aux constituants matériels inscrits au catalogue du constructeur.

2) Les liens associant des ensembles de configurations et des nœuds techno-fonctionnels aux configurations particulières des équipements de l'architecture ne représentent que des traces de choix émanant du processus de conception du SCC.

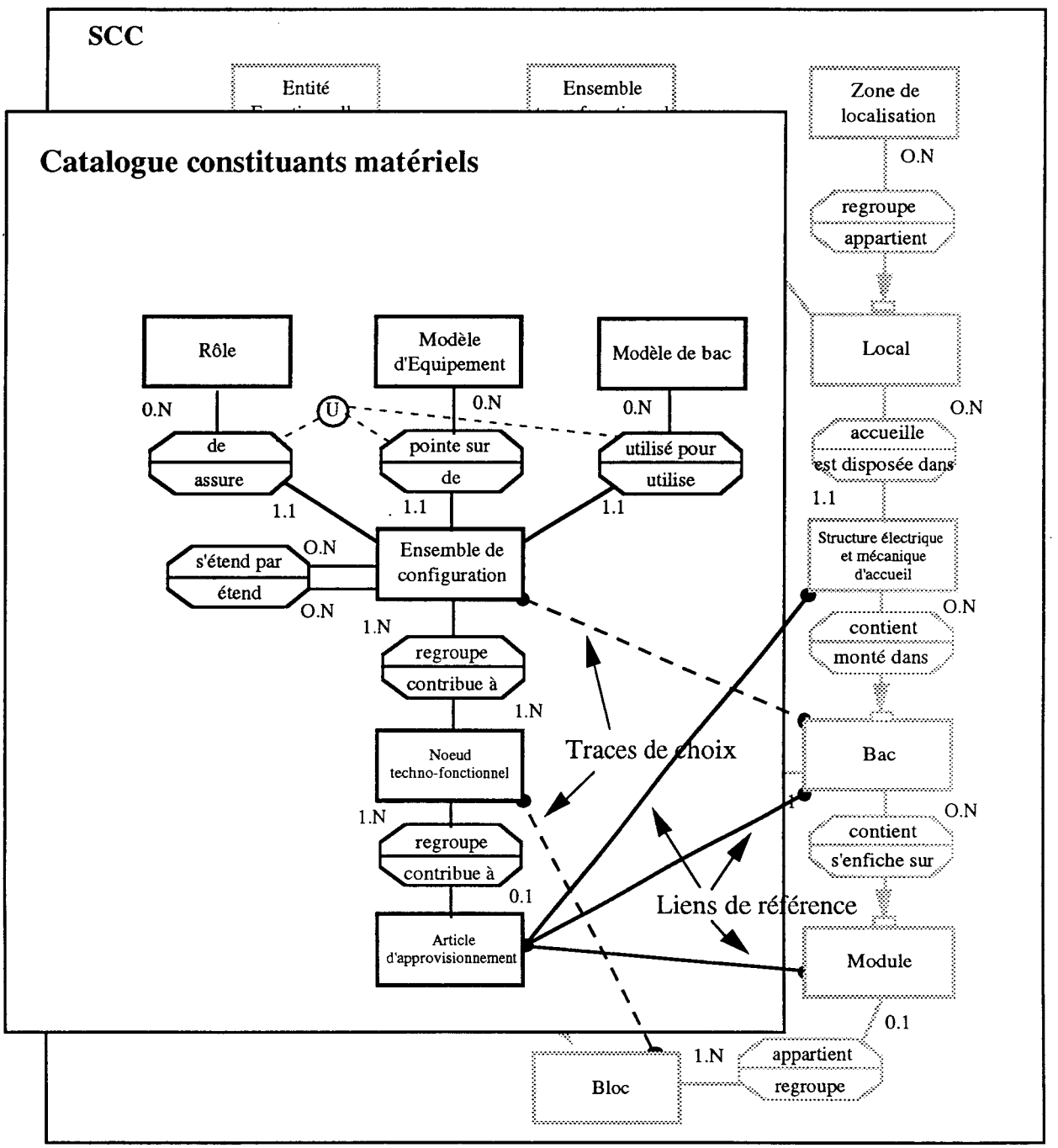


figure 17: Liens entre catalogue-produit et schéma d'application

De la même façon, chaque équipement et segment de réseau logique défini dans l'architecture est respectivement attaché à un type et un modèle particulier de produit au catalogue.

La méthode présentée dans le chapitre III décrit les principes d'une démarche permettant d'identifier les équipements d'automatisme d'un SCC et de construire progressivement chacune des trois facettes de leur représentation.

Elle passe successivement de l'analyse fonctionnelle à la décomposition organique du système et s'achève par une description détaillée de son architecture physique en exploitant des données de réutilisation inscrites en bibliothèque de constituants.

II.8 LIENS ENTRE NOTRE MODÈLE ET LA NORME BASEPTA

Les schémas conceptuels présentés aux paragraphes précédents partent de l'aspect matériel du méta-modèle Basepta. [PER 89] [Z68-92]
Ils sont conformes à la norme dans son domaine d'intérêt.

En particulier:

L'entité **RS** (Référentiel Site) de Basepta trouve son application dans le concept de Local de notre modèle.

L'entité **CR** (Constituant Réel) s'applique respectivement en Structure électrique et mécanique d'Accueil, Bac, Module et autres composants matériels du SCC.

Les entités Basepta exprimant l'approvisionnement des constituants réels du système sont appliquées et étendues à un modèle inscrivant en sus les modes d'utilisation de ses constituants, en définissant des ensembles de configuration et nœuds techno-fonctionnels pour une famille donnée de produits d'automatisme.

L'entité **CME** (Constituant Matériel Elémentaire) s'applique aux concepts d'équipement et d'unité logique de traitement.

Le concept de Bloc ne trouve, quant à lui, pas d'entité de référence dans Basepta.

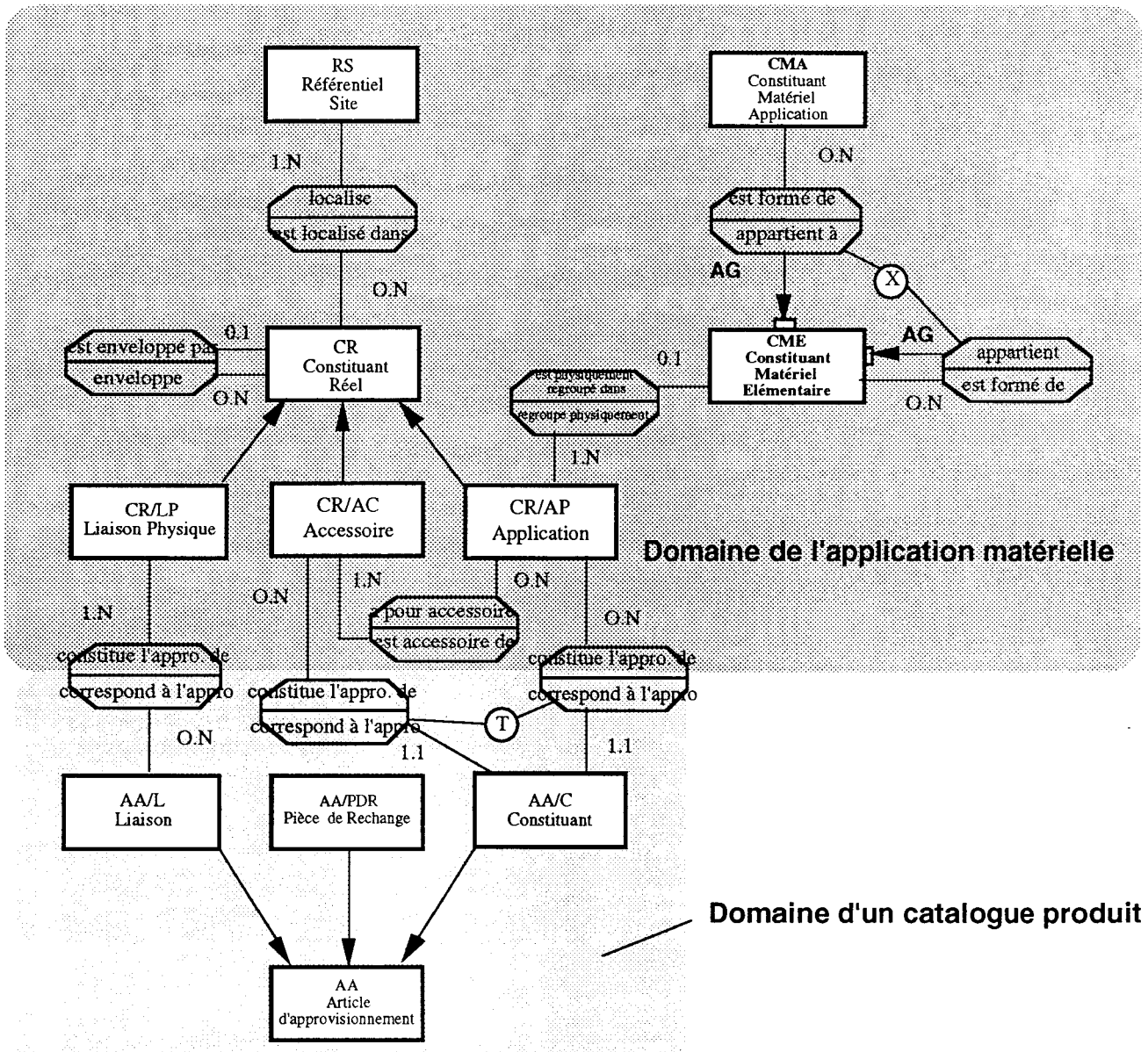
L'aspect fonctionnel développé au moyen d'entités fonctionnelles, de fonctions applicatives inter-reliées et d'interfaces externes au SCC n'est pas non plus abordé dans la norme Z68-901.

Les travaux entrepris à l'occasion du projet GAMA [GMA 93] ont confirmé la nécessité de modéliser ce dernier point. Ils proposent depuis lors d'intégrer cet aspect en spécialisant l'entité ERS (élément résultant de structuration) de BasePTA.

BasePTA à vocation de représenter les données d'un système opérationnel. N'y sont donc pas modélisées toutes les informations afférant au développement du système. C'est pourquoi certaines entités de structuration, bien utiles pour concevoir le SCC, ne trouvent pas place dans ce modèle.

☞ La modélisation proposée dans les paragraphes précédents constitue une extension de Basepta au domaine couvert par un atelier de génie automatique dédié à la conception des SCC. Elle est compatible avec la norme précédente et ne remet donc pas en cause les concepts de base ayant donné cette norme.

Extrait de BasePTA, aspect matériel:



Comme tout schéma appliqué de BasePTA, notre modèle de référence éclate la structure générique de la norme pour l'adapter à un domaine particulier d'application.

L'entité CME se transpose de cette façon en équipements logiques et unités de traitement, lesquelles représentent des abonnés logiques à un segment de réseau de communication.

La structure bouclée autour de l'entité CR permet de représenter de manière discrète l'arborescence des composants matériels d'un équipement, auxquels correspondent des articles d'approvisionnement particuliers, chez un fournisseur donné.

Nous avons distingué dans ce qui précède trois axes pour modéliser l'architecture matérielle du SCC.

Ces axes nous ont permis de décrire:

- *la composition **physique** du système en fixant l'organisation de ses composants matériels et en particulier celle de ses équipements.*
- *Les équipements représentant des unités de fonctionnement autonomes, nous les avons considérés à la base d'une architecture **logique** qui, selon le même point de vue, fait apparaître des unités de traitement d'informations liées au procédé.*
- *Chacune de ces unités de traitement réalise par ailleurs une part des **fonctions** applicatives du SCC.*

☞ *L'architecture logique se situe donc à un niveau de représentation intermédiaire entre les modèles fonctionnel et physique de représentation du SCC. Nous nous appuierons sur ce modèle intermédiaire pour concevoir l'architecture matérielle du système.*

Abordons maintenant l'aspect logiciel de l'application en définissant les concepts liés aux données de conception des programmes implantés sur les unités de traitement des équipements de niveau1 du modèle CIM.

CHAPITRE III CONCEPTS LIÉS À L'APPLICATION LOGICIELLE D'UN SCC

III.1 AXES PRINCIPAUX DE MODELISATION

Un équipement d'automatisme peut être décrit par les effets observables de son comportement vis à vis de sa partie opérative.
Ce comportement applique des lois de commande aux processus techniques gérés par l'équipement et se distingue, jusqu'à un certain niveau, de l'organisation des programmes qui, indirectement, permettent de le générer, par interprétation ou compilation et édition de liens interposées.

Ainsi, dans le domaine du logiciel comme dans le domaine du matériel, il faut distinguer les modèles qui concourent à la conception d'une application, des modèles sur lesquels s'appuie l'organisation effective des programmes implantés sur les machines de traitement.
C'est pourquoi nous optons ici pour un modèle de représentation du logiciel détaillant d'une part le **comportement** attendu d'un sous-système de commande et identifiant par ailleurs les **ressources** nécessaires à la mise en œuvre de ce comportement.

Le comportement sera détaillé sur un axe privilégiant l'aspect réactif du sous-système [DEL 93]. Les ressources liées à l'exécution de ce comportement contribueront pour leur part à définir l'aspect transformationnel des processus dont ce dernier est le siège.

Tout comme pour la modélisation de l'architecture matérielle du SCC où nous avons précédemment distingué les interfaces entre données de spécification (fonctionnelle), données de conception (organique) et données de réalisation (physique), la modélisation par comportements et ressources proposée dans ce qui suit n'est digne d'intérêt que dans la mesure où il est fait référence à ses liens avec les composants logiciels mis en œuvre ultérieurement dans le cycle de vie de l'application.

Nous déterminerons en particulier quel degré de proximité existe entre ces ressources et les éléments de réalisation émanant de la phase de conception détaillée des programmes d'automatisme (cf. figure 2).

Cette analyse nous permettra d'argumenter la technique présentée à la fin du chapitre IV.

Mais avant d'aborder tous ces sujets, cadrons tout d'abord le débat en présentant l'architecture interne standard d'un calculateur d'automatisme, ses unités d'exécution et comparons de cette manière les modes d'exécution des tâches de ce calculateur aux types de processus de commande à mettre en œuvre dans le SCC.

III.2 INTRODUCTION AUX TYPES DE PROCESSUS ET AUX MODÈLES D'EXÉCUTION ASSOCIÉS

III.2.1 ARCHITECTURE INTERNE D'UN CALCULATEUR D'AUTOMATISME

La majorité des calculateurs d'automatisme se caractérisent par une architecture interne qui respecte le modèle de la page suivante. Ces calculateurs représentent des équipements regroupant des unités de traitement (UT) matériellement distinctes. Les UT gérent elles-mêmes des tâches sur lesquelles s'appuie l'exécution des programmes de l'application.

Une tâche est, de ce fait, assimilée à une **unité d'exécution**.

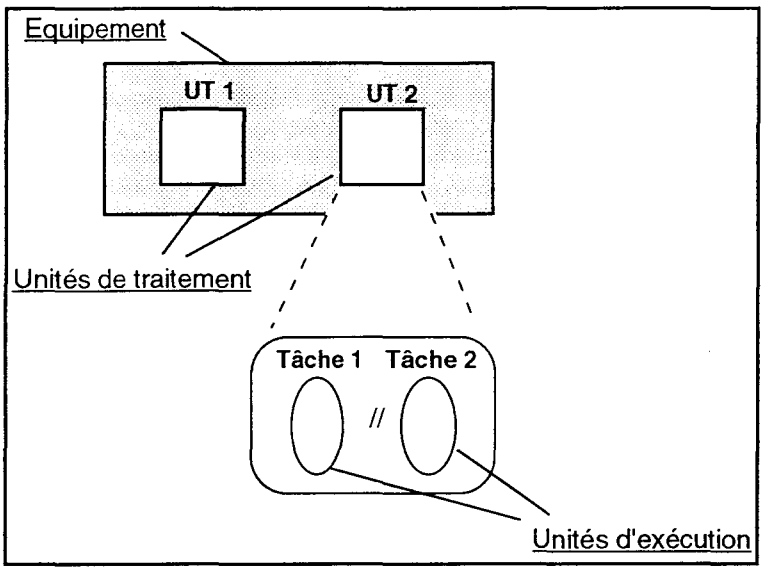


figure 18: "équipements et tâches"

III.2.2 TYPES DE PROCESSUS TECHNIQUES

Etant donné la taxonomie des systèmes présentée en II.1.1., on peut admettre ici l'existence de trois types de processus techniques élémentaires associés à la réalisation du procédé industriel. Ces processus sont soit à évolution continue, soit à évolution purement séquentielle, soit mixtes.

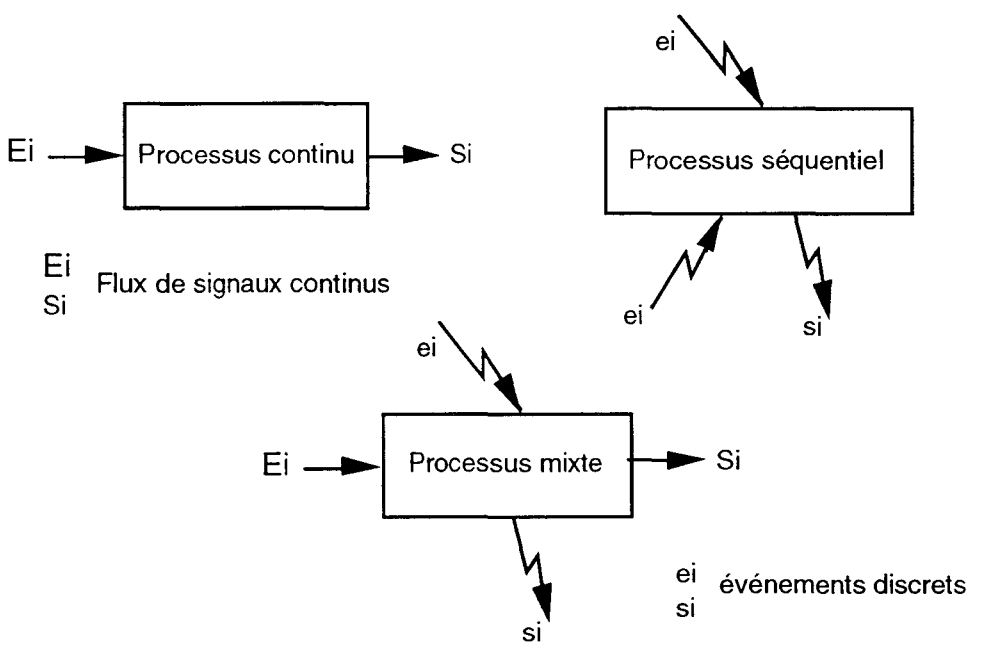


figure 19: types de processus techniques

- Les processus techniques continus se caractérisent par des flux continus de signaux d'entrées et de sorties.
- Les processus techniques purement séquentiels réagissent à des événements discrets dans le temps.
- Les processus techniques mixtes sont animés par les deux types de processus élémentaires précédents.

Exemple de processus technique continu: la régulation de vitesse d'un moteur; ce processus adapte de façon continue la tension à appliquer sur les bornes d'entrée du moteur à la consigne de vitesse de ce dernier.

Exemple de processus technique séquentiel: le processus que décrit un distributeur de monnaie; ce processus réagit à l'arrivée d'événements "pièces" et émet des événements analogues en sortie.

Exemple de processus technique mixte: un processus de déformation à chaud qui n'opère la déformation désirée que sur une plage donnée de température et fait varier l'effort de déformation en fonction de cette température.

III.2.3 MODÈLES D'EXÉCUTION DES TÂCHES

La nature des processus précédents implique un nombre limité de choix d'implantation qui, dans le cas des calculateurs numériques d'automatisme, sont caractéristiques du type d'exécution permis par leurs unités de traitement et plus précisément par ceux de leurs unités d'exécution.

En effet, contrairement à l'époque des calculateurs analogiques où l'on notait une équivalence point à point entre les modèles servant à la programmation des processus et leurs modèles d'implantation, l'utilisation exclusive de machines à traitements numériques nous conduit désormais à employer des structures d'exécution adaptées à la nature variée des sous-systèmes à commander et permet ainsi d'associer, sur un même équipement, des traitements répondant aux catégories précédentes.

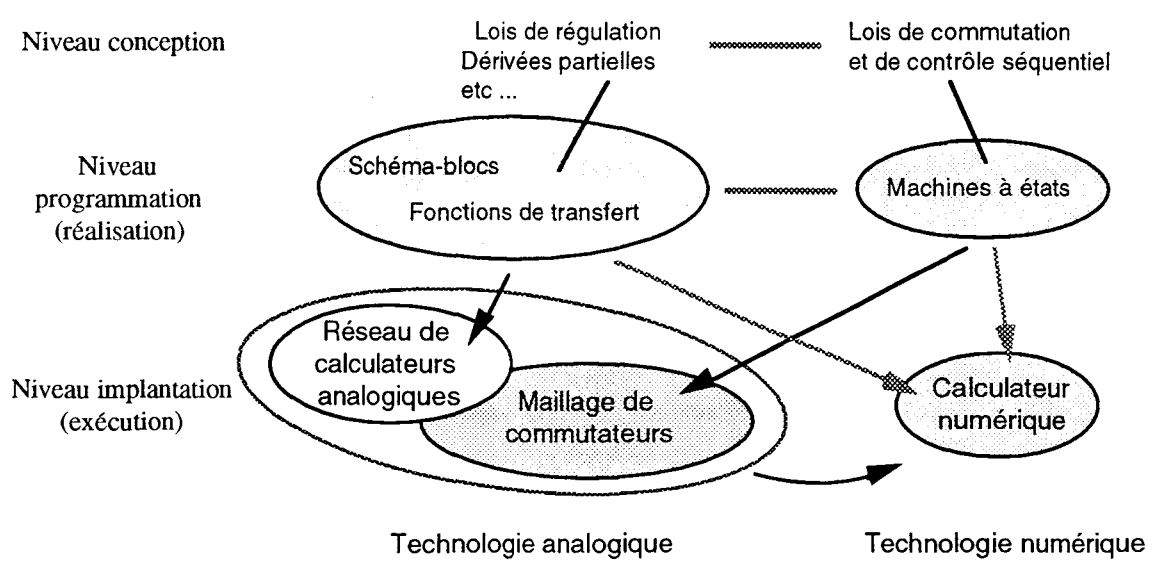


figure 20: Passage à plusieurs schémas d'implantation

Le passage de la technologie analogique à la technologie numérique a donc nécessité de considérer d'autres modèles d'exécution que ceux habituellement sous-entendus derrière les divers schémas de principe des composants électroniques alors employés.

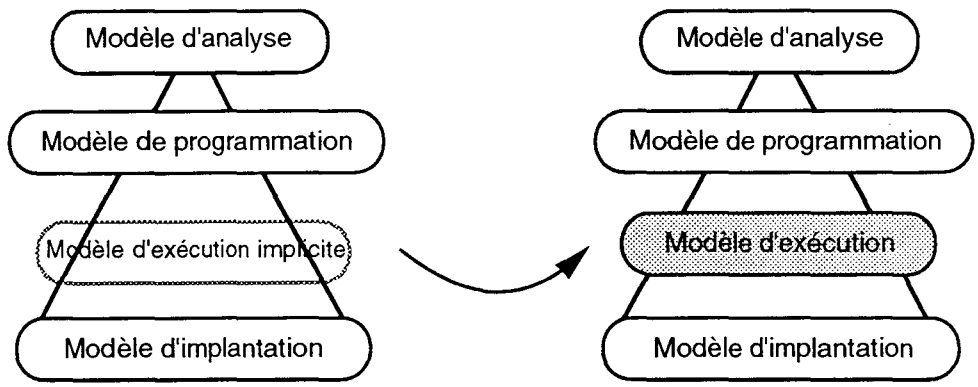


figure 21: Modèles de référence pour les différentes phases du développement

III.2.3.1 Mode 1: Processus Continus

Ainsi, par application de la théorie des systèmes échantillonnés, un **processus continu** nécessite de s'appuyer sur un modèle d'exécution **cyclique** des traitements qui, à chaque cycle, entreprend successivement trois groupes d'opérations:

- l'acquisition des entrées de ce sous-système
- le calcul des fonctions de transfert du modèle numérisé
- la restitution des sorties qui ont variées vers la partie opérative

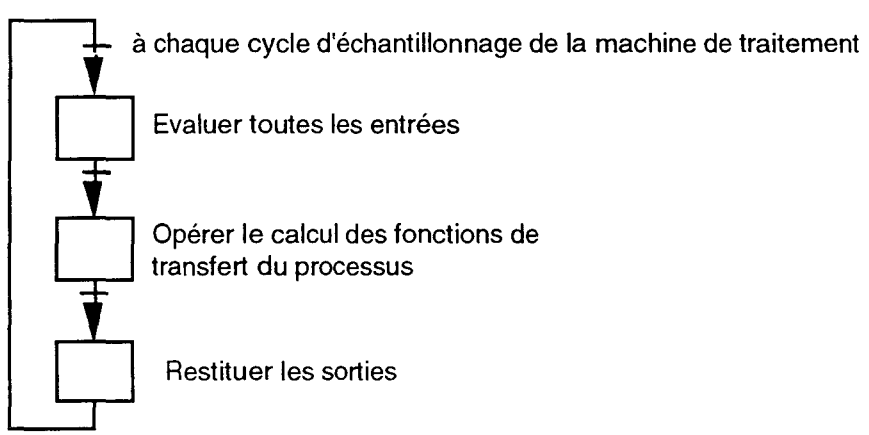


figure 22: modèle d'exécution des tâche d'un automate régulateur

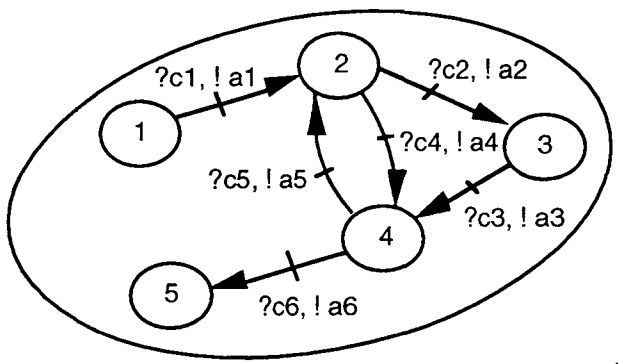
III.2.3.2 Mode 2: Processus Événementiels

- Un processus purement **séquentiel** ne nécessite pas, quant à lui, de porter régulièrement attention à tous les événements pouvant contribuer à son évolution, indépendamment de l'état dans lequel il se trouve.

Il suffit en effet d'évaluer, à chaque état stable, la liste des événements susceptibles de le faire évoluer vers un état voisin.

Un deuxième modèle d'exécution est donc proposé.
Comme le montre la figure suivante, il est associé au modèle programmé du processus.
(cf. modèles à états de représentation des modules ESTEREL une fois compilés)

Modèle programmé d'un automate

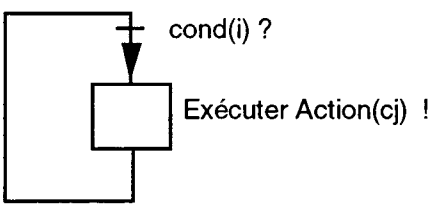


Conditions d'évaluation selon les places et actions associées

cond(1) = c1
cond(2) = c2 xor c4
cond(3) = c3
cond(4) = c5 xor c6
cond(5) = nil

et action(cj) = aj

Modèle d'exécution



(avec i, n° d'ordre des états de l'automate séquentiel et j, indice condition de transition)

figure 23: Deuxième mode d'exécution

Ce deuxième modèle d'exécution nous engage à une description exhaustive de chacun des états du processus séquentiel à réaliser et devient de ce fait contraignant dès que plusieurs de ces processus sont à intégrer sur une même unité d'exécution.

III.2.3.3 Mode 3: Processus événementiels combinés

Deux solutions permettent dans ce cas de s'affranchir de cette tâche manuelle et fastidieuse:

- soit employer un langage de programmation qui intègre le concept **synchrone** et supplée alors à cet exercice en réduisant les deux automates en un seul, (cf. § suivant)
- soit laisser les automates dans leur forme initiale et transformer le mode d'exécution précédent de sorte qu'il soit simulé un fonctionnement synchrone de deux machines à états sur une même unité d'exécution.

➡ Le mode d'exécution devient dans ce dernier cas pseudo-périodique. Il est illustré par la figure de la page suivante. C'est en particulier le schéma retenu pour définir la sémantique opérationnelle du Grafcet [G7 92] dans le récent ouvrage consacré à ce langage par l'AFCET. Les interpréteurs ou compilateurs G7 respectant ce modèle adopteront donc de facto le mode d'exécution présenté figure 24.

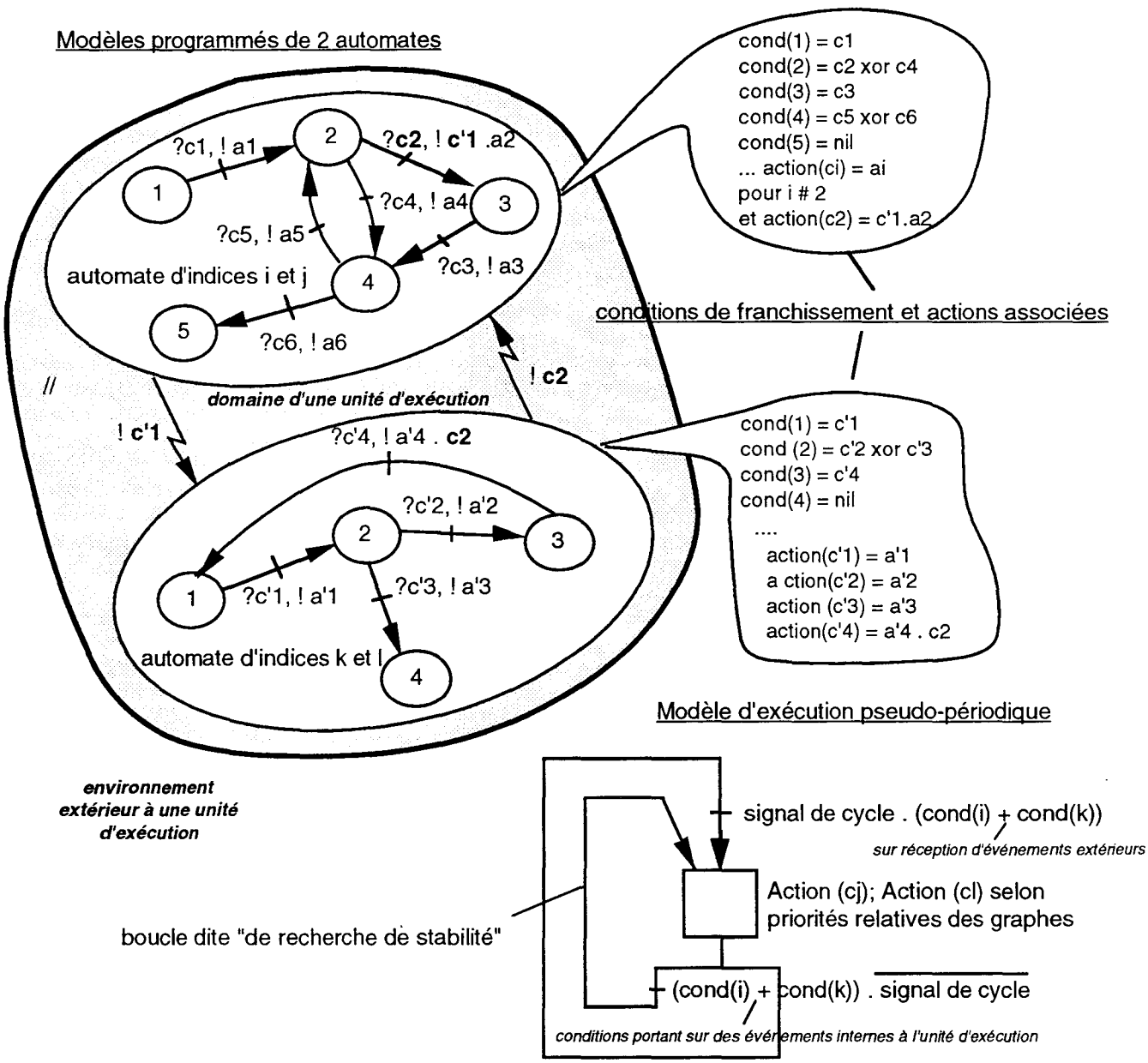


figure 24: Troisième mode d'exécution

Seule la première des deux solutions précédentes repose réellement sur un mode d'exécution synchrone des traitements et garantit ainsi un comportement entièrement déterministe de la machine programmée gérant dans l'absolu plusieurs **processus séquentiels communiquants**. [HOA 87]

Cette solution s'avère donc nécessaire à chaque fois qu'il est question de mettre en œuvre des automates en intéractions à la fois multiples et répétées.

La deuxième solution, qui consiste à employer le mode pseudo-périodique, n'est qu'une approximation "au premier ordre" d'une exécution idéalement synchrone. Ce mode est en effet cadencé au rythme de deux horloges des temps:

- une horloge externe, au rythme de laquelle sont pris en compte les événements externes (ce rythme est adapté au comportement du process à automatiser),
- une horloge interne, à la cadence de laquelle sont hiérarchiquement pris en compte les événements émis et reçus par les programmes associés à une unité d'exécution.

L'horloge interne évolue beaucoup plus rapidement que l'horloge externe. Elle permet normalement d'atteindre, sauf conception érronée et avant toute nouvelle manifestation de la partie opérative, un état stable sur chacun des automates finis mis en parallèles.

III.2.3.4 LES PROCESSUS MIXTES

Un processus **mixte** se caractérise à la fois par des réactions continues et séquentielles. Sa composante continue nous contraint à employer le mode 1 profilant le comportement de base d'un automate régulateur.

A chaque cycle d'évolution seront donc à la fois acquises, des entrées continues échantillonnées et évaluées, toutes les conditions de franchissement de l'automate séquentiel mêlés à la partie continue du processus.
De la même façon, la restitution des sorties discrètes d'un processus mixte sera groupée avec la restitution de ses signaux continus de sortie à chaque fin de cycle de tâche.

Rque: les processus mixtes étant élémentaires, leur partie séquentielle se décrit de façon **unitaire** et il n'est généralement pas nécessaire d'appliquer au modèle 1 la boucle de recherche de stabilité proposée dans le mode 3.

III.2.4 L'APPROCHE PAR LES LANGAGES SYNCHRONES

La programmation synchrone est issue d'une théorie qui considère que les traitements sont constamment réalisés en synchronisation parfaite avec le procédé.

Partant de cette hypothèse, le phénomène peut être abordé de deux manières distinctes selon que l'on considère le temps de transition de la machine d'un état à un autre quasiment nul ou que l'on assimile le système à modéliser à un circuit logique.

- la première approche permet de considérer le temps comme un élément discret et multiforme. Cette considération fournit intrinsèquement la possibilité d'effectuer des réductions sur les équations de logique temporelle traduisant l'automatisme à réaliser [BER 87] [MIL 83] [WOL 83] [CLA 83].

Elle donne lieu à un style de programmation impératif que synthétise le langage ESTEREL.

- la seconde approche est du type "flots de données" et tire directement parti des équations booléennes définissant un circuit logique [LGU 86] [CHPP 87]. Elle aboutit, après réduction des équations caractérisant le circuit, à l'expression de l'automate à états finis du processus.

Cette approche donne lieu à une programmation déclarative et est en particulier proposée dans les langages LUSTRE et SIGNAL.

Limites d'usage:

Les techniques sous-entendues derrière l'approche synchrone permettent de générer des automates à états en réduisant la complexité d'instructions temporelles lors d'une passe de compilation, ou initialement d'interprétation.

La puissance sémantique de ces langages est telle que l'expansion du code résultant de cette passe peut atteindre, si l'on y prend pas garde, des limites qui dépassent le cadre des hypothèses dans lesquelles s'inscrit l'exécution du code final. (exécution en un temps nul)

La difficulté est donc de savoir limiter la description à des automates préfigurant un nombre d'états et de transitions (entre états) acceptable par la machine cible.

Il faut donc impérativement se fixer des règles pour s'affranchir de ces problèmes.
L'une d'elle est très simple; elle consiste à limiter la taille des programmes par le nombre de leurs entrées-sorties.

III.2.5 LES MODÈLES D'EXÉCUTION EMPLOYÉS DANS LA PRATIQUE

Les équipements d'automatisme proposés en pratique appliquent directement ou indirectement les trois modèles théoriques d'exécution précédents pour les utiliser à des fins plus ou moins diverses.

- Ils les appliquent **directement** lorsque leur modèle d'exécution est inscrit dans le "firmware" de la machine de traitement.
- Ils les appliquent **indirectement** lorsque ce modèle est pris en compte par la console servant à générer les programmes de l'équipement.

Le mode 1, à la base de tout automate de régulation, s'applique aussi en pratique aux processus séquentiels.
La technique employée consiste à exécuter **pas à pas** et au rythme **périodique** du cycle présenté figure 18 le programme séquentiel décrivant un processus discret.
A la différence de l'exécution d'un processus continu échantillonné où la totalité de sa fonction de transfert est évaluée à chaque cycle d'évolution de la machine, l'exécution d'un processus séquentiel ne réclame dans ce cas que de "calculer" partiellement (en réalité entre deux états stables) la situation dans laquelle se trouve l'automate.

L'avantage de cette technique est de ne considérer qu'un seul mode d'exécution pour tout type de traitement.
Son inconvénient est de ne s'appliquer, en théorie, qu'à des modèles programmés de processus séquentiels **non communicants**.
Il serait par conséquent abusif de vouloir lui prêter d'autres intentions !

Le mode 2 est implanté sur des automates de moyenne à haute gamme ainsi que des microcalculateurs.
Ces équipements exécutent les traitements applicatifs sur des tâches **non périodiques** dont la durée peut de surcroît être limitée dans le temps.
Exemples d'équipements respectant ce modèle: l'automate C350 (ALSPA CEGELEC) et le microcalculateur S800.

Etant donné l'émergence nouvelle d'environnements complets de développements basés sur les langages synchrones, il n'est pas exclu de voir à termes s'intégrer, sur des machines respectant ce mode particulier d'exécution, des programmes développés avec de tels langages (SAGA de VERILOG, AGEL de ILOG ...). Nous verrons, à titre d'exemple dans le chapitre IV, comment certains éléments résultant de la conception préliminaire de l'application peuvent donner lieu à des modules de programmes décrits à l'aide du langage ESTEREL.

Le mode 3 n'est implanté que sur peu d'automates. Un seul est répertorié à ce titre dans les produits proposés par CEGELEC; le modèle C170.

III.3 MODÉLISATION DES DONNÉES DE CONCEPTION: AXE DES COMPORTEMENTS

III.3.1 COMPORTEMENTS

Le comportement d'un équipement d'automatisme, chargé par son système d'exploitation et son logiciel d'application, témoigne de sa **réaction** aux sollicitations de l'environnement extérieur. Il présente les **effets** que produisent l'arrivée, à des instants aléatoires, d'informations provenant soit de la partie opérative, soit d'autres systèmes, soit de l'interface de conduite du SAP.
Au niveau d'un automate, ce comportement est porté par des tâches et un ordonnanceur reproduisant les caractéristiques (idéales) de l'un des trois modes d'exécution décrits en III.2.3.

III.3.1.1 NOTION DE TÂCHE

Selon la norme automates CEI 1131-3 [CEI-92], une tâche est *"un élément de commande d'exécution périodique ou déclenché, d'un groupe de programmes associés".*

Une tâche anime donc des programmes assurant une part des fonctions d'automatisme que doit réaliser une unité de traitement dans le système.
Elle regroupe dans ce sens les conditions d'activation de ces programmes pour former une unité d'exécution cohérente vis à vis des processus de commande à mettre en œuvre dans l'application. [SYS 91]

III.3.1.2 NOTION DE PROCESSUS DE COMMANDE

Dans ce qui suit, nous associerons chaque processus technique élémentaire caractérisant le procédé au modèle de comportement du sous-système de contrôle-commande qui permet son déroulement, en d'autres termes au processus de commande qui le représente dans le SCC.

De cette manière, on note trois types de processus de commande:
- des processus à évolution continue
- des processus séquentiels
- des processus mixtes.

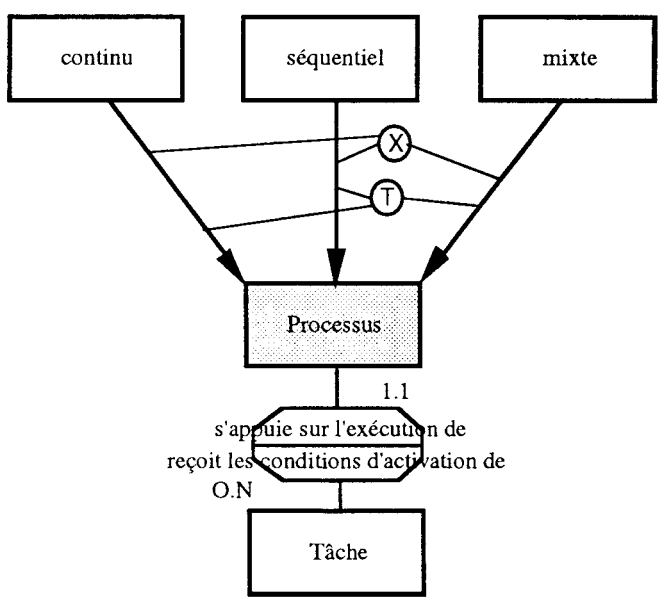


figure 25: Taxonomie des processus de commande

III.3.1.3 NOTION DE **COMPORTEMENT**

Face à la définition précédente, nous associerons à la notion de comportement la description formelle des réactions d'une unité de traitement ou toute partie d'une unité de traitement aux sollicitations de son environnement extérieur.

Avec cette hypothèse et partant des modèles détaillés des opérations à effectuer pour chaque processus de commande associé aux processus techniques élémentaires caractérisant le procédé, il nous est possible de retracer progressivement le comportement lié à une unité d'exécution (cf. III.2), à plusieurs d'entre elles ou bien même le comportement global d'une unité de traitement.

Un comportement sera donc défini *comme l'évolution caractérisée d'un sous-système dont la taille est limitée par les éléments de représentation logique du SCC.*

Compte tenu de la définition d'un processus (de commande), il lui correspond naturellement le concept de comportement élémentaire, comme l'illustre la figure ci-dessous:

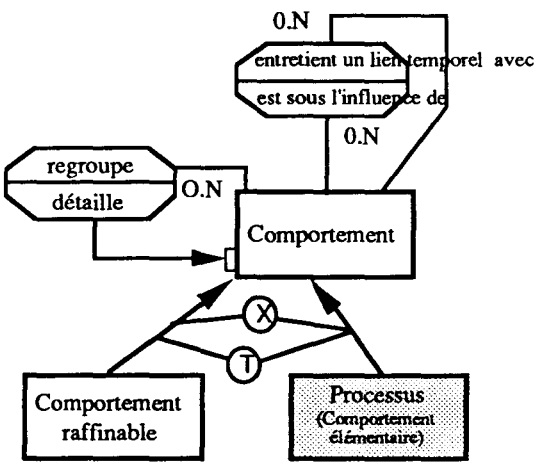


figure 26: Dévoloppement d'un comportement

La relation d'agrégation du sous-schéma précédent se développe alors comme suit:

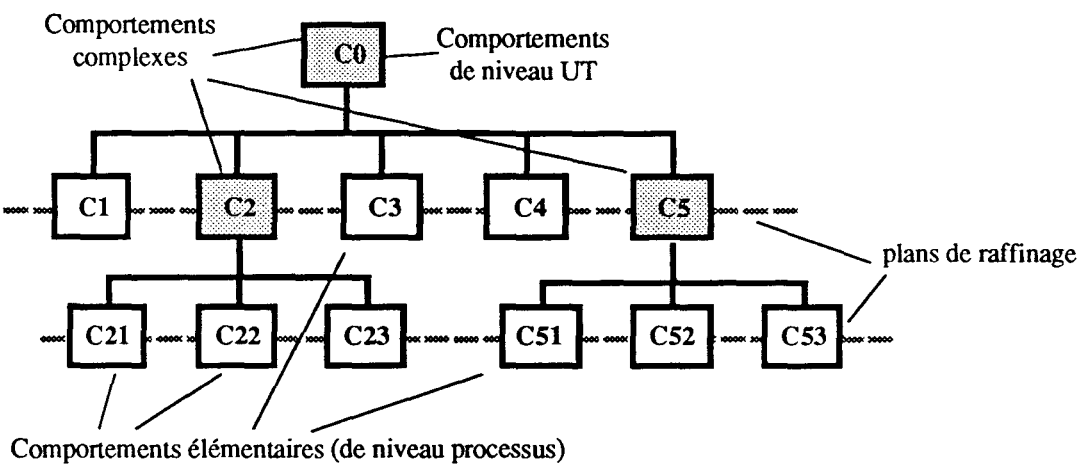


figure 27: Arborescence de comportements

Les comportements raffinables apparaissent comme des éléments complexes ne s'identifiant pas directement à un processus de commande.

☞ Ainsi, décrire un comportement du point de vue de la **conception** revient à raffiner ce dernier en sous-comportements moins complexes puis à réitérer la procédure autant fois que nécessaire pour obtenir finalement des comportements élémentaires correspondant à des processus de commande.

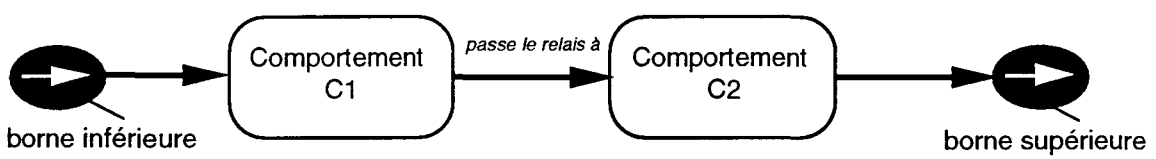
III.3.2 LIENS TEMPORELS

Les comportements de chaque niveau de l'arborescence (ou chaque plan de raffinage) sont par ailleurs liés entre eux par des règles spécifiques d'ordonnancement qui représentent des liens temporels particuliers entre comportements.

La méthode TOCCATA [TOC 92] , sur laquelle s'appuie notre exposé, associe ces règles d'ordonnancement à des opérateurs temporels ayant une sémantique stricte et respectant les principes de base de la programmation structurée.
Elle met en évidence 5 opérateurs. Chacun d'eux est associé à des connecteurs bornant les expressions élémentaires de comportement formées en les employant.

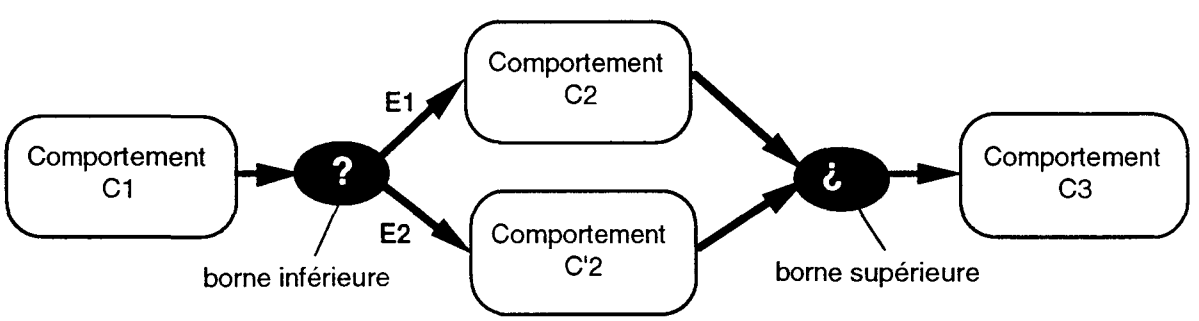
III.3.2.1 OPÉRATEUR DE SÉQUENCE:

Cet opérateur permet d'exprimer une séquence de comportements abstraits bornée par un connecteur "début" et un connecteur "fin".



III.3.2.2 OPÉRATEUR SÉQUENTIEL DE CHOIX OU "SÉLECTION DE COMPORTEMENT"

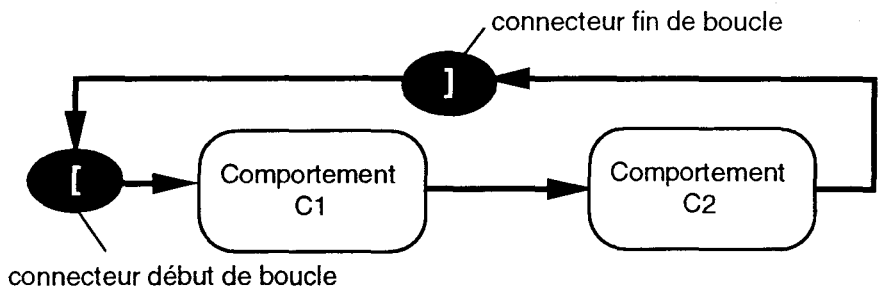
Cet opérateur est associé à l'évaluation de conditions portant sur des choix exclusifs.



(C1 peut, selon la valeur de la condition de Ei, basculer alternativement vers C2 ou C'2)

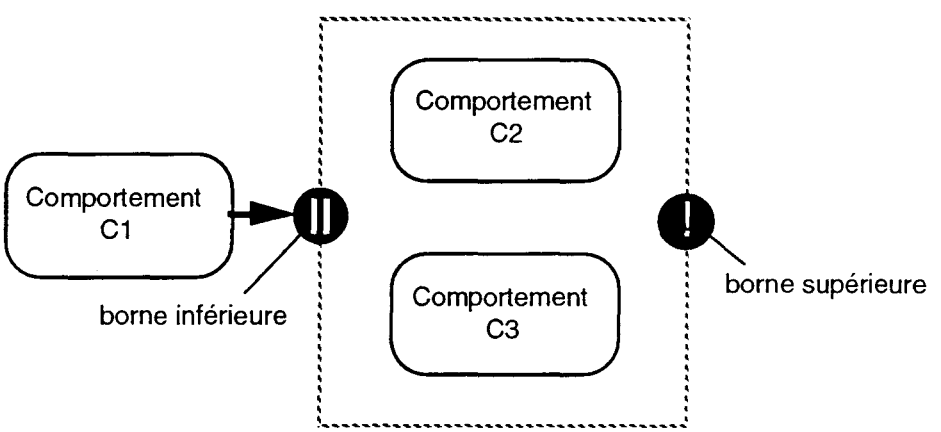
III.3.2.3 OPÉRATEUR DE BOUCLE OU ENCORE L'ITÉRATION DE COMPORTEMENT

Cet opérateur reprend les propriétés d'un opérateur de séquence et y adjoint le rebouclage de ses bornes extrêmales.



III.3.2.4 OPÉRATEUR DE PARALLÉLISME

Cet opérateur fait référence à des éléments de comportement à priori non séquentiels.

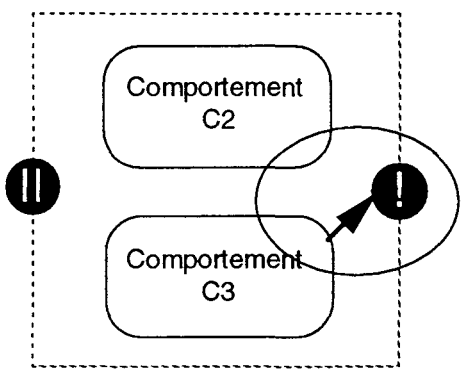


III.3.2.5 LA SORTIE DE BOUCLE OU L'ÉCHAPPEMENT DE BRANCHES PARALLÈLES

Cet opérateur revient à désactiver un ou plusieurs comportements. Il se mêle aux expressions de possédant pas de terminateur explicite, c'est-à-dire les expressions construites avec les deux types d'opérateurs précédents.

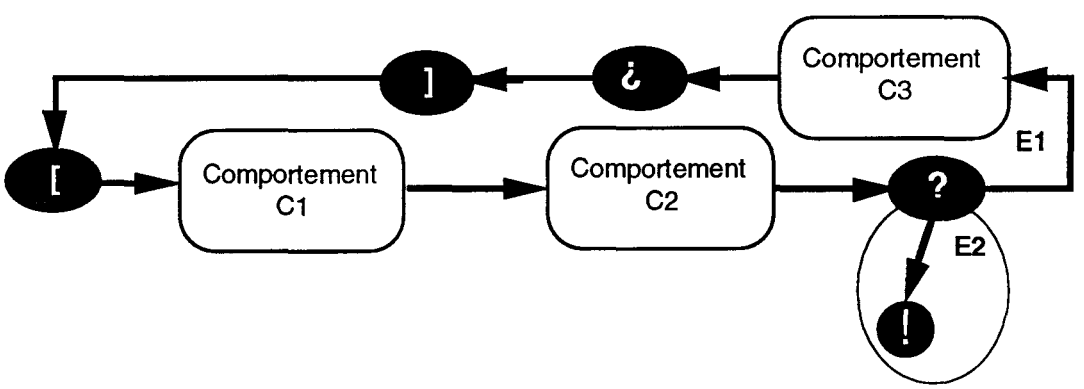
Exemples:

a)



(la fin de C3 provoque l'arrêt de C2)

b)



(la condition portant sur la vraissemblance de E2 conduit à sortir de la boucle)

nota: Il va de soi qu'en l'absence de terminateur spécifique, les comportements meurent de leur belle mort.

Remarque:
Conférer une sémantique précise à ces liens permet d'associer un graphe d'états à tout comportement de niveau élevé et d'en vérifier la cohérence sur un niveau de détail plus ou moins fin, en sélectionnant tout ou partie de son sous-arbre.
Cette vérification peut par exemple permettre d'inspecter les modes de fonctionnement d'un équipement programmé afin de s'assurer que ce dernier ne se dirige jamais vers un état puits.

☞ Nous composerons ces opérateurs temporels dans des **expressions de comportements** caractérisant chaque plan de raffinage des applications portées par les unités de traitement du SCC. [TOC 93]

III.3.3 LIENS OPÉRATIONNELS

III.3.3.1 NOTION DE PROTOCOLE

Les comportements s'échangent par ailleurs des informations par des moyens divers de communication. Nous distinguerons, pour la modélisation conceptuelle, les informations échangées des moyens observés pour les communiquer en introduisant les concepts respectifs d'information et de protocole.
Un protocole [TOC 93] sera donc défini comme *un moyen générique de communication entre processus de commande ou encore, en phase décomposition, un mode d'échange d'information particulier entre sous-comportements.*

Une information ne représentera quant à elle que la donnée intervenant dans chaque échange.

Il n'est fait ici aucun à priori sur la réalisation concrète de ces échanges puisque nous nous plaçons en phase de conception de l'application logicielle. Il va tout de même de soi que chaque unité d'exécution des équipements d'automatisme supporte une liste donnée de protocoles types.

Le schéma suivant fixe les notions d'information et de protocole et exprime de plus, à travers la contrainte de totalité marquée d'un astérisque, que la présence d'un comportement dans un graphe est justifiée dès lors que ce dernier produit ou consomme au moins une information.

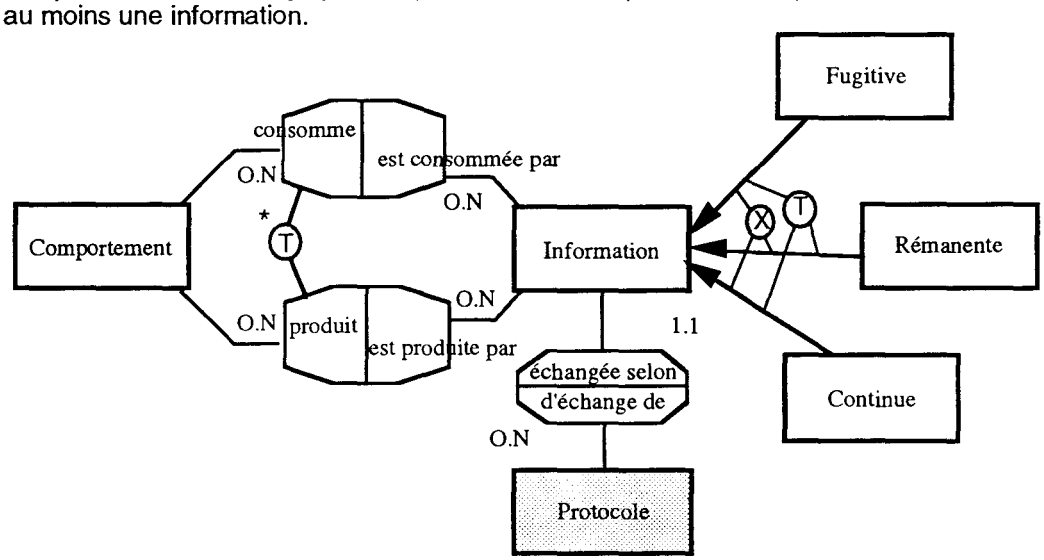


figure 28: Comportements - protocoles

Un comportement est le siège d'intéractions diverses qui se traduisent, dans un système tout à fait général, par trois types de liens opérationnels:

- des liens véhiculant des informations **fugitives** ou discrètes dans le temps (événements) ,
- des liens véhiculant des informations **rémanentes** ou mémorisées dans le temps,
- des liens véhiculant des informations prenant un caractère **continu**.

La méthode TOCCATA, initialement définie pour concevoir le logiciel des systèmes temps réels, donc à forte connotation réactive, ne fait apparaître que des protocoles véhiculant des informations fugitives ou rémanentes.
Notre proposition élargit le champ d'application de la méthode en prenant en compte des processus dits "continus", intéragissant de manière permanente avec leur environnement, en l'occurrence ici, la partie opérative.

exemples d'informations fugitives: top départ d'un cycle, événement d'alarme, coup de poing de sécurité ...

exemples d'informations rémanentes: message déposé dans une boîte aux lettres, file FIFO, variable en mémoire, commande temporisée d'ouverture d'une vanne ... etc

exemples d'informations continues: niveau de tension instantanée du secteur, vitesse instantanée de rotation d'un moteur ...

De plus, les informations véhiculées par ces protocoles ont une **portée** limitée dans l'espace caractérisant le système. Cette portée est directement liée aux éléments d'organisation logique de l'architecture du SCC. (cf. II.1)

Souhaitant pour notre part associer un modèle de comportement à chaque unité de traitement présente dans cette architecture, nous scinderons ces protocoles en deux nouvelles catégories représentant distinctement:

- des interactions entre sous-comportements caractérisant une UT
- des échanges d'informations entre une UT et son environnement extérieur.

1) Protocoles liés à des informations confinées à une Unité de Traitement

Une Unité de Traitement n'est, ni plus, ni moins qu'un calculateur temps réel. Les communications entre tâches de ce calculateur s'opèrent donc de la même façon que pour des systèmes temps réels. Nous sélectionnerons parmi les modes d'échange d'informations les plus connus de ce domaine un échantillon de protocoles représentatifs des types de ressources nécessaires au coordonnement de processus d'automatisme, et en particulier:

- 4 types de protocoles véhiculant des informations **fugitives**:
  - .. l'événement
  - .. le sémaphore (signal d'arrêt)
  - .. la technique du rendez-vous
  - .. la diffusion radio (cas particulier de l'approche synchrone ESTEREL)

- 2 types de protocoles véhiculant des informations **rémanentes**:
  - .. la variable en mémoire
  - .. la machine partagée

Chacun de ces protocoles autorise par ailleurs un nombre limité de **configurations d'échanges**.

En effet, un sémaphore exprime généralement la disponibilité d'une ressource unique; nous postulerons que celle-ci ne peut donc être "produite" que par un seul processus, qui, à force de compositions successives, ne peut être associé qu'à un seul comportement sur un plan donné de raffinage. (cf. figure 27)

L'information transportée par ce protocole peut, de plus, être diffusée à plusieurs processus à un instant donné, donc à plusieurs comportements issus d'un même plan de raffinage.

Les caractéristiques d'un sémaphore donnent donc: 1 producteur - N consommateurs

Tous les protocoles véhiculant des informations discrètes, continues ou rémanentes doivent ainsi être dotés de règles de cardinalités analogues:

- l'événement est produit par un comportement à un instant donné et peut être simultanément consommé par plusieurs, d'où la règle (1-N)
- la technique du rendez-vous se caractérise par les règles (N-1) ou (1-1), selon les langages de programmation utilisés pour réaliser les logiciels
- la radio-diffusion fait intervenir N producteurs et N consommateurs simultanément actifs

et

- la variable en mémoire, qui à priori est globale puisque l'on ne considère pas la structuration du programme qui génère le comportement, peut être produite et consommée par n processus, d'où la règle (N-N)
- la machine partagée (exemples: temporisation, compteur) peut être attaquée par N processus et servir des informations à N autres éléments de ce type .

(cf. annexe 4: Définition des protocoles)

Le modèle conceptuel de données de l'axe des comportements se complète donc de la manière suivante:

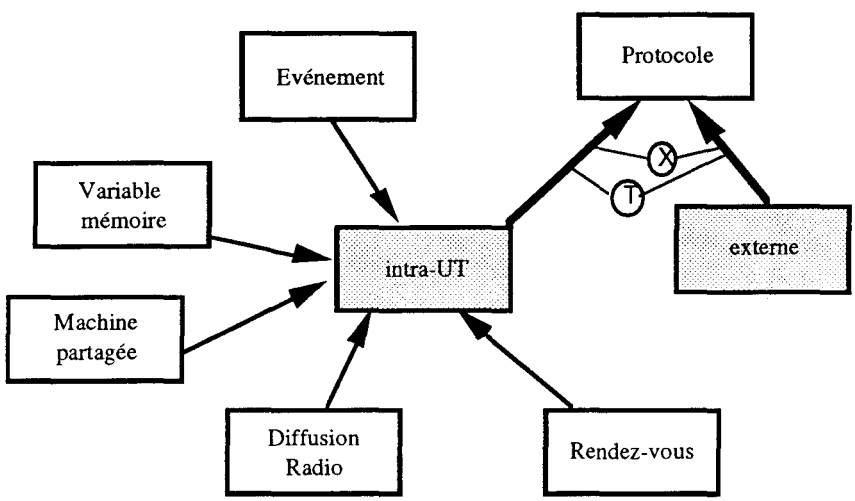


figure 29: types de protocoles relatifs à des communications intra-UT

Interprétation de la rémanence des informations:

La rémanence des informations échangées via les protocoles a des implications directes sur l'axe des temps de la modélisation des comportements.
En effet, un protocole véhiculant une information rémanente peut rester actif durant une séquence de comportement,

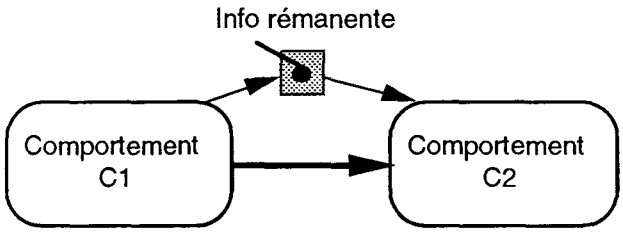


figure 30: Particularité d'une information rémanente

alors qu'un protocole véhiculant une information fugitive ne peut pas, selon cette modélisation, alimenter deux comportements abstraits participant à une même séquence d'opérations, s'agissant alors d'émettre et de recevoir un même événement à deux instants différents.

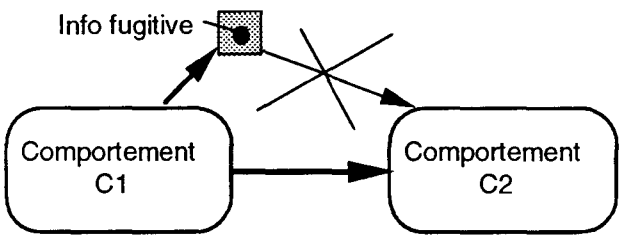


figure 31: Particularité d'une information fugitive

2) Protocoles liés aux interfaces externes à une Unité de Traitement:

On note, selon la hiérarchie des composants logiques d'un système de contrôle-commande, deux types de protocoles régissant l'échange d'informations entre une UT et son environnement:

- un premier type caractérisant des informations échangées entre les UT d'un même équipement, (dites plus loin "internes" à un équipement)
- un second type lié aux interfaces externes à ces unités de fonctionnement.

a) Interfaces internes à un équipement:

Les informations transmises entre les UT d'un équipement peuvent être de nature tout à fait diverses. Les moyens employés pour communiquer ces informations sont tout aussi divers. Néanmoins, on note tout de même trois types de protocoles génériques caractérisant ces interfaces:

- la communication par mémoire commune (au sens physique du terme): mémoire partagée entre plusieurs processeurs

et dans le cas de traitements asynchrones:

- la communication par file d'attente
- la communication par boîte aux lettres.

Chacun de ces protocoles véhicule des informations **rémanentes**.

De plus, le mode de communication par mémoire commune est doté des règles de cardinalités (N-N), les modes de communication par file d'attente et par boîte aux lettres sont tous deux associés à la règle (N producteurs-1 consommateur).

b) Interfaces externes à un équipement:

Les liens entretenus entre un calculateur d'automatisme et son environnement extérieur sont pour leur part de trois types:

- interfaces avec la partie opérative (les seules à pouvoir être continues puisqu'en en intéraction directe avec un système physique),
- interfaces avec un réseau ,
- interfaces ponctuelles entre équipements du SCC ou avec un équipement d'un système de contrôle extérieur (autre SCC ou système informatique).

Interfaces partie opérative

Les interfaces entre un équipement et le sous-ensemble de la partie opérative qu'il contrôle obéissent à trois modes de communications particuliers via:

- des entrées-sorties **continues**
- des événements et des actions **fugitives**
- des signaux ayant un caractère **rémanent**

En respect des règles de configuration d'échange qu'impliquent ces trois sortes d'interfaces, nous sommes amenés à considérer 6 types de protocoles leur correspondant (cf. figure 32):

- des protocoles délivrant des entrées continues dans le temps ou assimilées comme tels (entrées échantillonnées par exemple).
  Selon notre modélisation, ces entrées n'émanent d'aucun processus caractérisant une unité de traitement dans le système de contrôle-commande, puisqu'elles sont issues de capteurs donnant l'état observé d'une composante du procédé. Ces entrées peuvent, par contre, être transmises à N consommateurs potentiels (N processus consommateurs), d'où la double règle de cardinalité X-N, (X voulant dire aucun)
- des protocoles délivrant des sorties continues qui ne sont originaires que d'un seul processus et à l'origine d'aucun autre, dans un quelconque diagramme de comportement relatant une UT.
  Ces sorties sont donc caractérisées par la règle (1-X),
- des protocoles associés à des fronts d'entrées, véhiculant par conséquent des informations fugitives et liées à la règle de cardinalité (X-N)
- des commandes à accès immédiat en sortie (commandes impulsionnelles), liées à la règle (1-X),
- des mémoires d'entrées (un seuil par exemple), obéissant à la règle (X-N),
- des actions mémorisées en sortie (une commande temporisée), obéissant à la règle de cardinalité (1-N).

Par ailleurs, nous ne ferons, pour la modélisation comportementale, aucune distinction entre des entrées-sorties diffusées par le biais d'un bus périphérique de données et des entrées-sorties, procédé empruntant la voie d'un service périodique de communication sur un réseau de terrain (cf. exemple présenté figure 8 §II.3.2).

Interfaces réseaux

Une UT programmable peut dialoguer, de manière asynchrone avec d'autres unités de traitement de même type dans le système, en utilisant le canal d'un segment de réseau. Nous symboliserons ces échanges par des **messages réseaux**, véhiculant des informations rémanentes dans le SCC. Ces messages seront associés à la règle de cardinalité "1 producteur" - "N consommateurs".

Interfaces ponctuelles:

Les interfaces relatives à des communications point à point, entre équipements, seront modélisées par des files FIFO particulières, de règle de cardinalité (1-1). Ces protocoles manipulent également des informations rémanentes.

Les sous-types de la figure 28 représentent la taxonomie des protocoles retenue pour formaliser les échanges entre comportements issus d'unités logiques de traitement distinctes.

Cette taxonomie ne prétend pas être exhaustive.
Elle a simplement pour but de proposer un cadre de modélisation permettant de distinguer la nature de ces interactions.

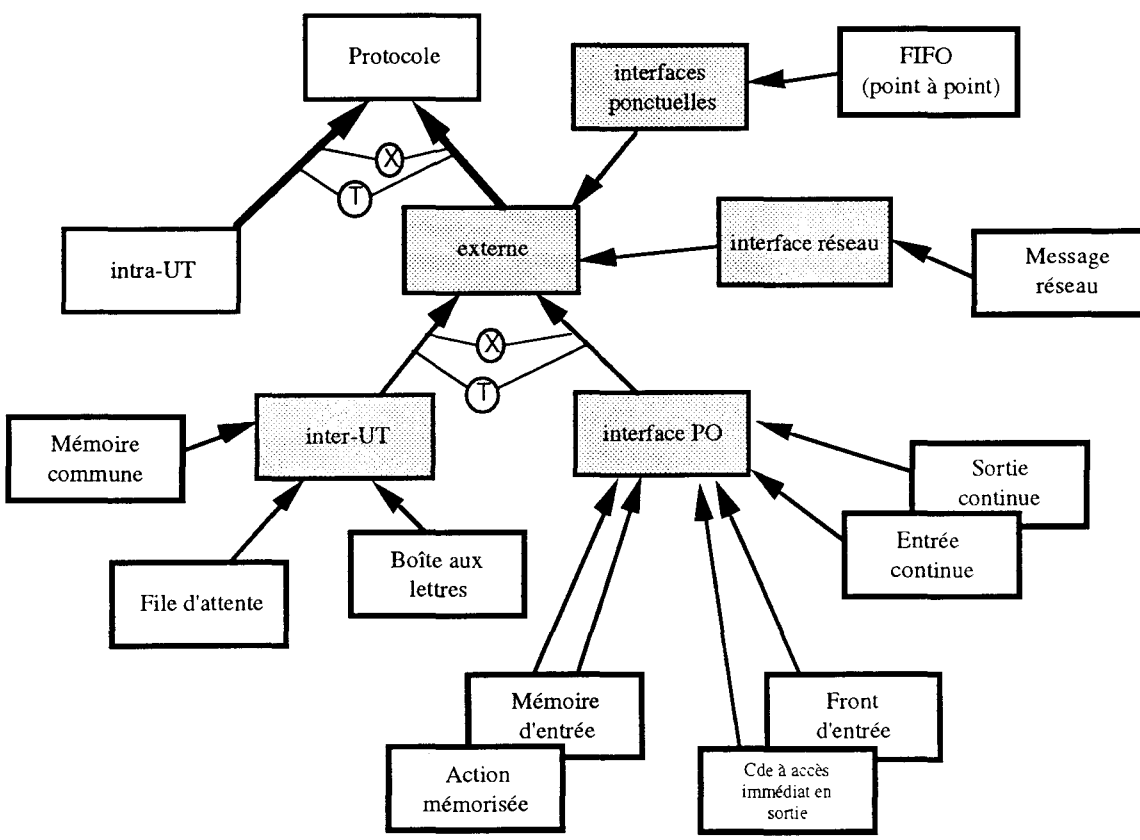


figure 32: Protocoles externes à une UT du sous-système d'automatisme

III.3.3.2 PROTOCOLES RAFFINABLES

Tous les protocoles listés ci-avant formalisent des modes de communication entre processus, c'est-à-dire des liens opérationnels entre comportements élémentaires du système programmé de contrôle-commande.

L' outil ORGUE, supportant la méthode TOCCATA, propose de décrire graphiquement ces liens et de représenter chaque type de protocole à l'aide d'un composant particulier d'interface.

- Il est, de plus, souhaitable d'accompagner la décomposition abstraite d'une application par des instances de protocoles dévoilant progressivement (sur plusieurs plans de raffinage) les informations échangées entre comportements. Nous utiliserons, pour ce faire, des liens opérationnels particuliers répondant aux caractéristiques de "protocoles raffinables".

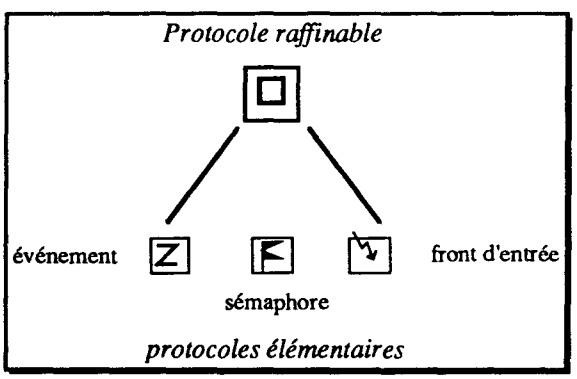


figure 33: exemple de protocole raffinable

Un protocole raffinable dévoilera ainsi à ses sous-comportements de niveaux inférieurs des instances de protocoles portant sur des informations réellement identifiées dans l'application.

... Les concepts évoqués dans ce paragraphe sont tous issus de la partie CA de la méthode TOCCATA.

Nous verrons dans le chapitre IV comment appliquer cette méthode à l'identification des processus gérés par les équipements d'automatisme du SCC.

Le paragraphe suivant est consacré à la définition des concepts liés à l'aspect transformationnel des processus et présente la hiérarchie des ressources identifiées pour chacun d'eux en phase de conception du logiciel. Il fait référence à la partie TA (Types Abstraits) de la méthode TOCCATA.

III.4 MODÉLISATION DES DONNÉES DE CONCEPTION : AXE TRANSFORMATIONNEL

Un processus opère des traitements sur des données.
Ces données sont plus ou moins complexes et plus ou moins bien isolées dans un sous-système d'automatisme.

Il convient donc d'une part de les considérer comme des ressources de contrôle-commande, encapsulées par des services et d'autre part, de les décomposer et les hiérarchiser de façon à réduire progressivement l'étendue de leur domaine d'intérêt [LIS 74] [LIS 2]

III.4.1 MACHINES ABSTRAITES DE CONTRÔLE ET DE COMMANDE

- Nous encapsulerons les données propres à un processus d'automatisme dans des **machines abstraites** de contrôle et de commande du procédé auxquels seront associés des **services** représentant les traitements accédant à ces données à tout instant d'exécution du processus [CHJ 86] [TOC 92].
Une machine abstraite sera représentée comme suit:

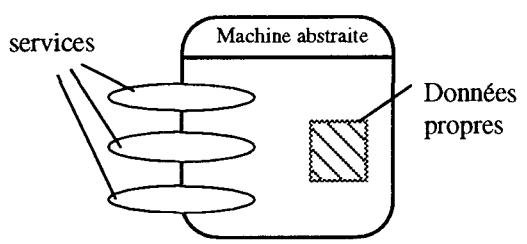


figure 34: Composants transformationnels de l'application

- Nous décrirons par ailleurs la hiérarchie des ces composants transformationnels en traçant les liens d'exploitation existant entre les machines abstraites caractériant un processus. Exemple de hiérarchie liée à un processus de transfert de gravats:

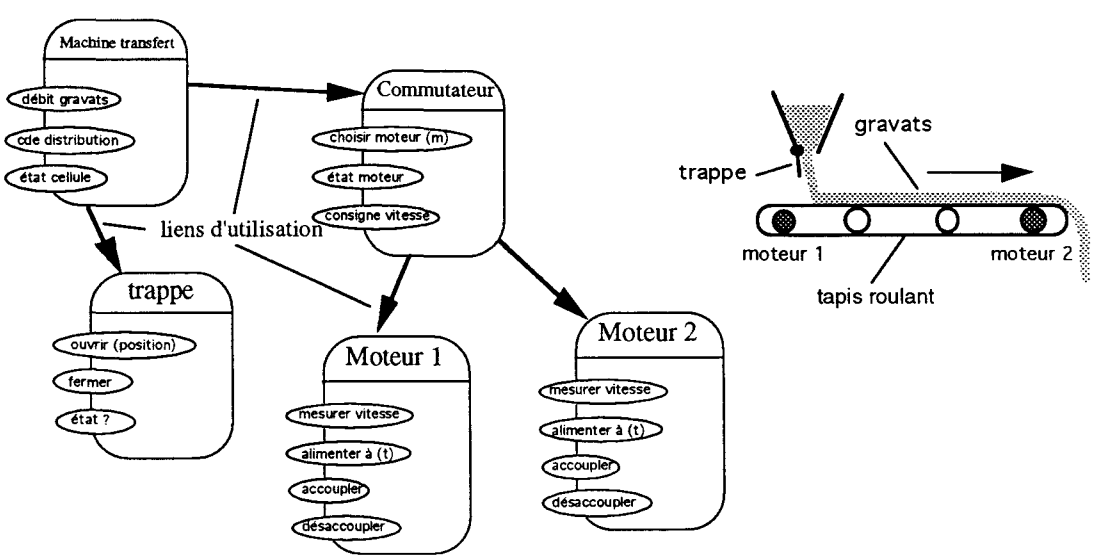


figure 35: hiérarchie des ressources de commande attachées à un processus

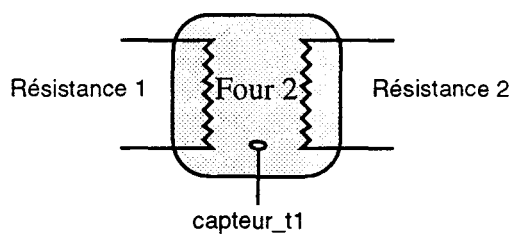
Dans la figure précédente, la machine abstraite "Machine transfert" fait appel aux services permettant de commander la "Trappe" déversant le gravats sur le tapis roulant et aux services de la machine "Commutateur" qui a, elle-même, recours aux services accédant aux ressources de contrôle et de commande des moteurs 1 et 2.

Une machine abstraite est donc d'une part définie à partir d'un ensemble de ressources propres caractérisant son état et d'autre part identifiée par rapport à une série d'opérations permettant de connaître ou faire évoluer cet état.

L'origine du terme "machine abstraite" vient des types abstraits qui sont à la base de bien des méthodes de conception objet en informatique ([BOO 86] ou [ROS 87] par exemple). Cette notion isole les propriétés significatives d'un objet en considérant un point de vue particulier d'analyse. Les propriétés ainsi obtenues remplacent alors l'objet dans tout contexte d'utilisation respectant le même point de vue [MEY 88].

Nous adopterons globalement ce principe pour identifier au chapitre IV les catégories de ressources liées à l'application du procédé, en faisant correspondre des machines abstraites, soit à des ressources liés à un sous-procédé distinct, soit aux ressources de contrôle-commande que symbolisent respectivement les organes, les capteurs et les effecteurs de la partie opérative. (cf. modèle de structuration du § **III.5**)

Exemple de ressources liées à l'organe "four" suivant:
Résistance 1, Résistance 2, capteur_t1



Liste de services visant à réguler la température de ce four en mode tout ou rien:

mesurer (température), alimenter, couper_alimentation, sélectionner (résistance de chauffage) ...etc...

Chacun de ces services répond soit à la commande d'un effecteur, soit à l'acquisition d'une information émanant d'un capteur spécifique.
Le service "alimenter" pilote par exemple l'alimentation de la résistance courante du four.

Nous encapsulerons les services d'accès aux ressources de ce four par une machine abstraite (**Four n°2**) ayant une structure de données propre affichant les variables d'état significatives de cet organe pour le procédé:

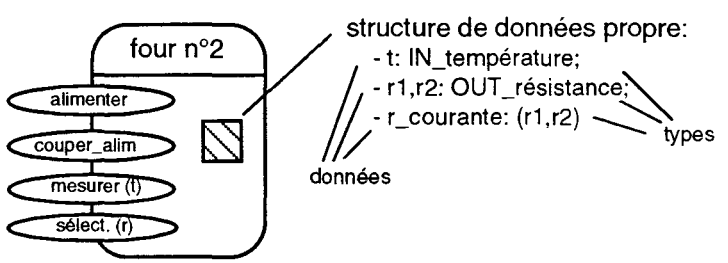


figure 36: Exemple de machine abstraite de commande liée à un organe PO

III.4.2 RAFFINAGE DES RESSOURCES APPLICATIVES

- La complexité de certaines données caractérisant un processus nous conduit à décrire ces dernières de façon progressive, c'est-à-dire par abstraction successive, en laissant apparaître des **machines abstraites raffinables** dans la hiérarchie des ressources identifiées lors de la conception d'une application.

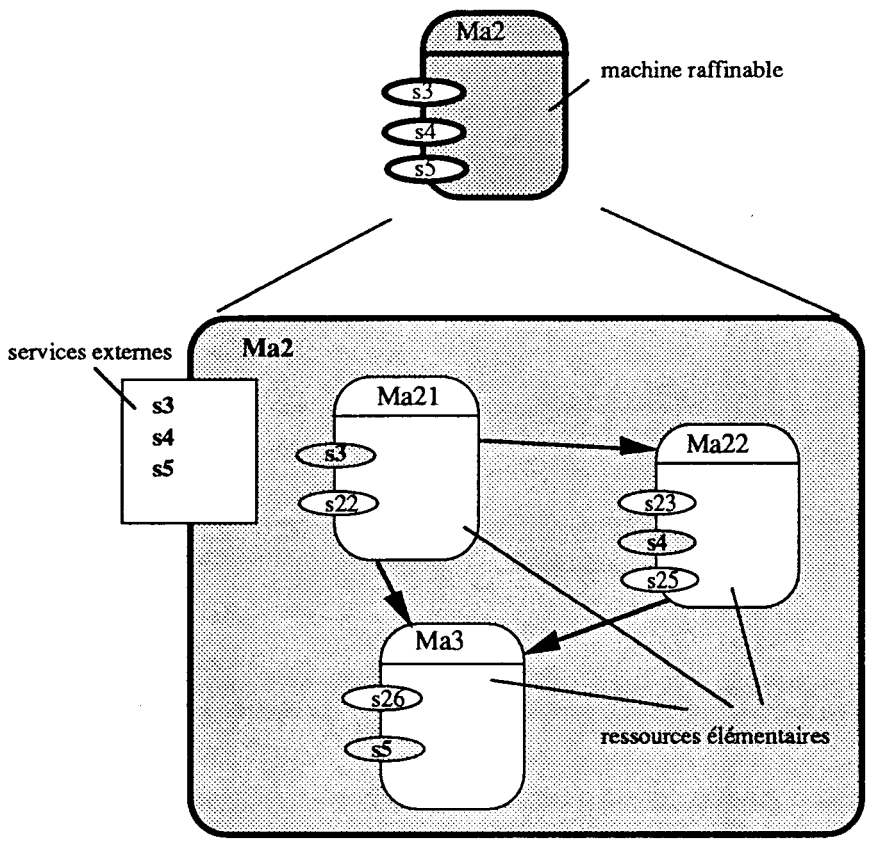


figure 37: Raffinage des ressources et des services

Ces machines sont raffinées à la manière d'activités statemate [HAR 90] (cf. figure 38). Elles ne s'attachent toutefois qu'à représenter l'aspect transformationnel des processus, contrairement à l'approche pré-citée ne faisant pas cette distinction. La démarche Statemate contribue en effet à faire apparaître, sur un même diagramme d'activités, les éléments issus d'une analyse événementielle et les composants représentant les transformations opérées dans un système.

N'émanent donc des représentations détaillées TOCCATA que des "flots de données" et aucun "flot de contrôle" (termes Statemate) hormis les liens d'utilisation statiques entre machine qui ont déjà étés évoqués au paragraphe précédent. Il correspond par conséquent aux machines abstraites de la décomposition proposée dans ce paragraphe, les activités dites "fonctionnelles" d'un diagramme Statemate.

Nous verrons un peu plus loin que les activités de contrôle, exprimées en statecharts [HAR 84] [HAR 87] dans l'approche de spécification à trois points de vue proposée par David HAREL (points de vue fonctionnel, comportemental et organique) prennent place, en TOCCATA, dans le réacteur de chaque processus.

Décomposition-type statemate:

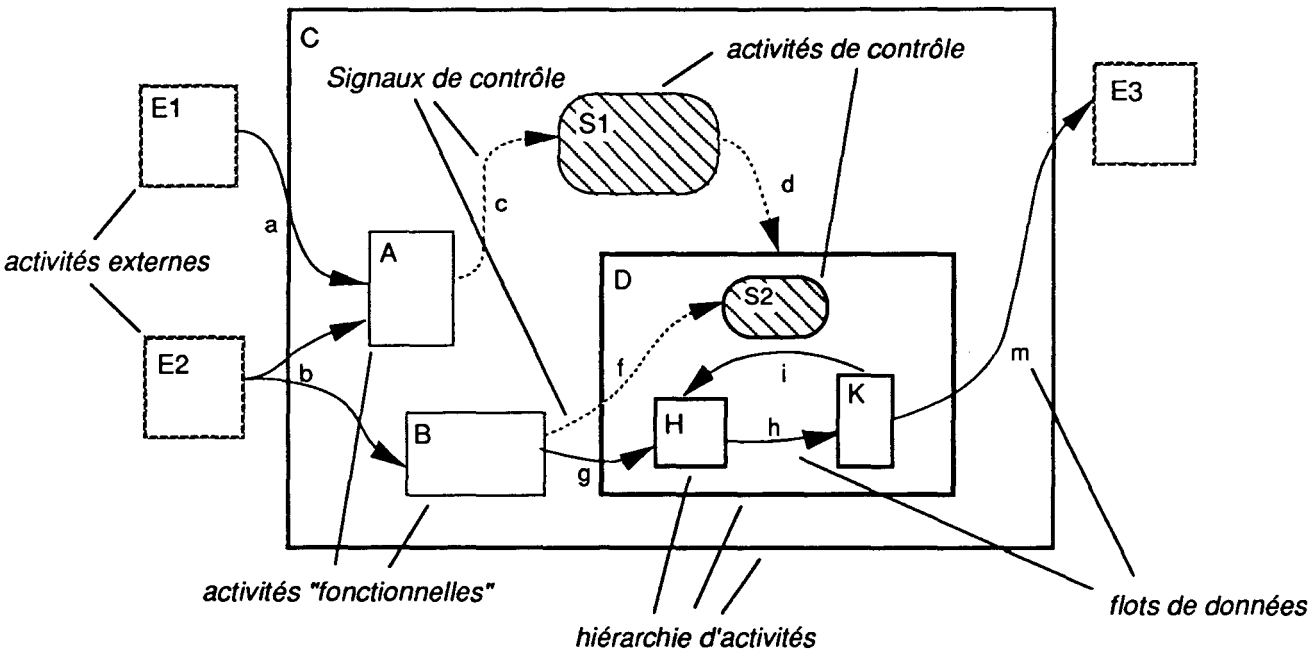


figure 38: Diagramme d'activité Statemate

Une machine abstraite raffinable masque, tout comme dans la représentation hiérarchique ci-dessus issue d'une autre approche, une ressource complexe de l'application.

Si les données qu'elle renferme se trouvent raffinées, les services opérant des traitements sur ces données sont ici bien réels et doivent trouver application sur une machine abstraite attachée à une ressource élémentaire du processus. C'est ce qui explique pourquoi s3, s4 et s5 de la figure 37 sont intégralement offerts (et non décomposés) par les machines Ma21, Ma22 et Ma3.

L'abstraction est donc toujours guidée par les données maîtrisées par un processus et non par ses traitements.
Cette optique est garante d'une bonne traçabilité entre les objets de conception et les éléments de réalisation manipulés ultérieurement dans le cycle de développement. A tout service déclaré au niveau d'une quelconque machine abstraite de l'axe transformationnel doit effectivement correspondre une part des traitements de l'application matérialisée par un élément de logiciel.

III.4.3 CLASSES DE PROTOCOLES: IMPLICATIONS SUR L'AXE TRANSFORMATIONNEL

Les liens opérationnels, associés à la modélisation des comportements caractérisant une unité de traitement (cf. III.3.3.1), ont été précédemment définis sur la base de protocoles véhiculant trois types d'informations:

- des informations fugitives,
- des informations rémanentes,
- des informations continues dans le temps.

Ces trois types d'informations sont en réalité soit partiellement, soit globalement, liés aux composantes réactives et transformationnelles des processus.

En effet, une information fugitive représente un signal "pur" auquel n'est associé aucun élément de donnée occasionnant un traitement. Ce signal est perçu ou produit à un instant t. Il disparait à l'instant suivant en ne laissant aucune trace de son apparition aux processus auxquels il était destiné. Seule la prise en compte de ce signal par un processus receptif à un tel simuli peut occasionner un traitement mettant alors en jeu des données.

☞ De telles informations fugitives sont donc uniquement liées à l'aspect réactif des processus.

A contrario, une information continue s'apparente à une donnée en perpétuelle évolution dans le système de production. Cette donnée fait constamment l'objet de traitements (opérés à des intervalles réguliers dans le temps) et ne revêt, à aucun instant, le caractère d'un signal.

☞ Elle concerne par conséquent uniquement l'aspect transformationnel du processus.

☞ Enfin, une information rémanante tient à la fois de la définition d'un signal et d'une donnée. Elle peut donc aussi bien apparaître sur un modèle réactif qu'un modèle transformationnel de processus.

Fort de ces trois remarques, il nous est maintenant permis de déterminer, quels types de protocoles dérivant, pour un processus donné, d'un modèle particulier de comportement, répondent aux caractéristiques de liens transformationnels et réactifs.

III.4.4 LIENS TRANFORMATIONNELS

On note, relativement à la taxonomie des liens opérationnels du § III.3.3.1, 5 types de protocoles délivrant des informations liées à la composante transformationnelle d'un processus:

- les entrées continues
- les sorties continues
- la variable en mémoire
- la mémoire commune entre plusieurs Unités de traitement
- la machine partagée

et 6 autres protocoles en leur qualité de mode de communication mixtes (véhiculant des informations rémanentes):

- la communication intra équipement par file d'attente
- la communication intra-équipements par boîte aux lettres
- les mémoires d'entrées
- les actions mémorisées en sortie
- les messages réseaux
- les files fifo "point à point" entre équipements

Ces liens transformationnels complètent la description du processus en attachant des ressources partagées et des interfaces spécifiques aux machines abstraites de ce dernier.

Ils témoignent des données échangées entre machines de traitement à un instant précis. Cette description est donc synchrone. Elle ne tient pas compte de l'appréciation du temps.

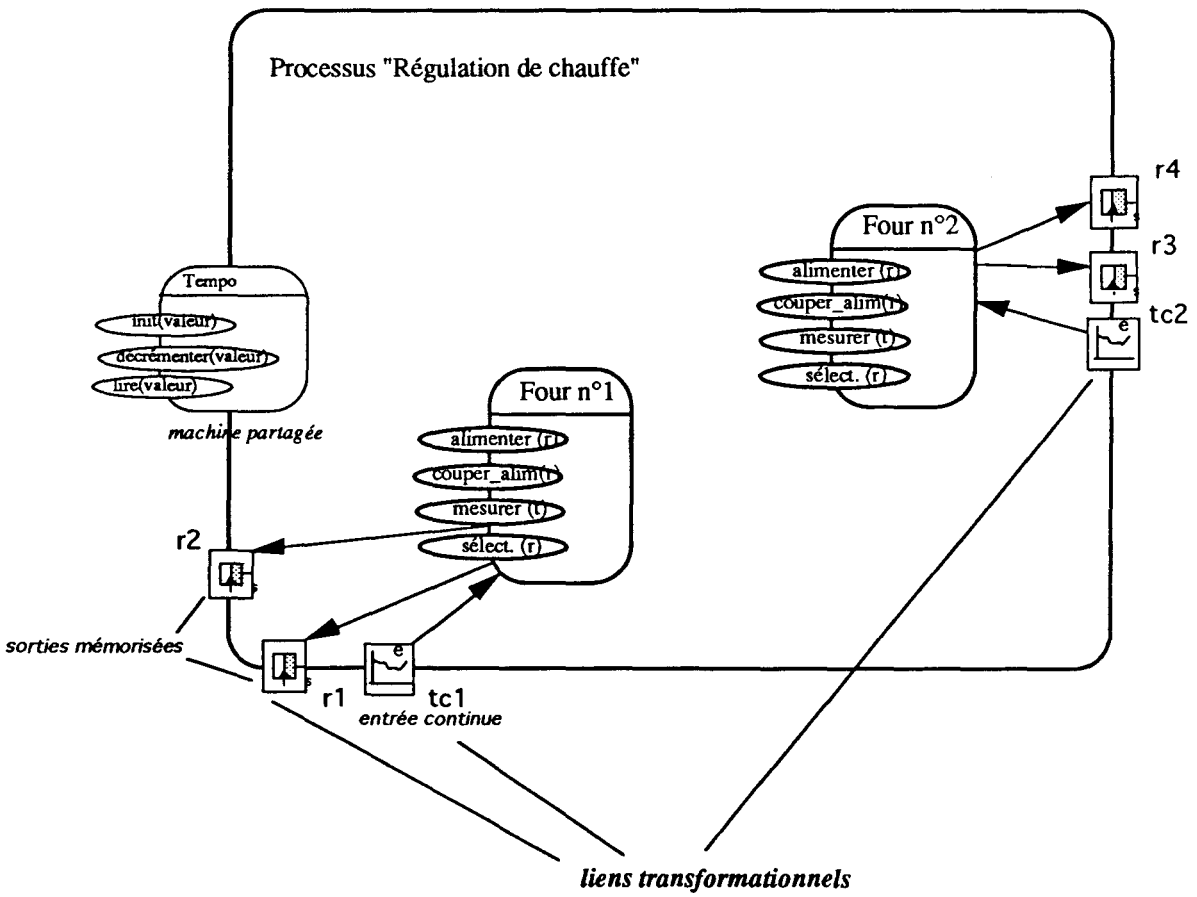


figure 39: liens transformationnels

Dans l'exemple ci-dessus, le processus "Régulation de chauffe" partage la machine "tempo" avec des processus tiers.

Par ailleurs, les machines abstraites "four n°1" et "four n°2" prennent en charge les données d'interface avec la partie opérative véhiculées par les instances de protocoles "r1", "r2", "r3", "r4" et "tc1", "tc2".

III.4.5 LIENS RÉACTIFS

6 types de protocoles répondent à la définition de liens réactifs. Il s'agit de:

- l'événement
- du sémaphore
- de la technique du rendez-vous
- de la diffusion radio
- des fronts d'entrées
- des commandes à accès immédiat en sortie ou commandes impulsionnelles.

A cette liste doivent être rajoutés tous les protocoles "rémanents", qui potentiellement peuvent permettre à un processus d'évoluer dans le temps.

Par analogie avec la méthode MACH (Méthode Appliquée de Conception Hiérarchisée) [MAC 85], TOCCATA centralise la prise en compte des liens réactifs d'un processus au niveau d'un élément unique appelé "réacteur".
Ce réacteur fédère les conditions d'activation du processus en regroupant tous les liens engageant une activité sur ce dernier. Le réacteur représente donc l'élément de logiciel qui est chargé de déclencher l'algorithme appelant les services des machines abstraites liées au processus.

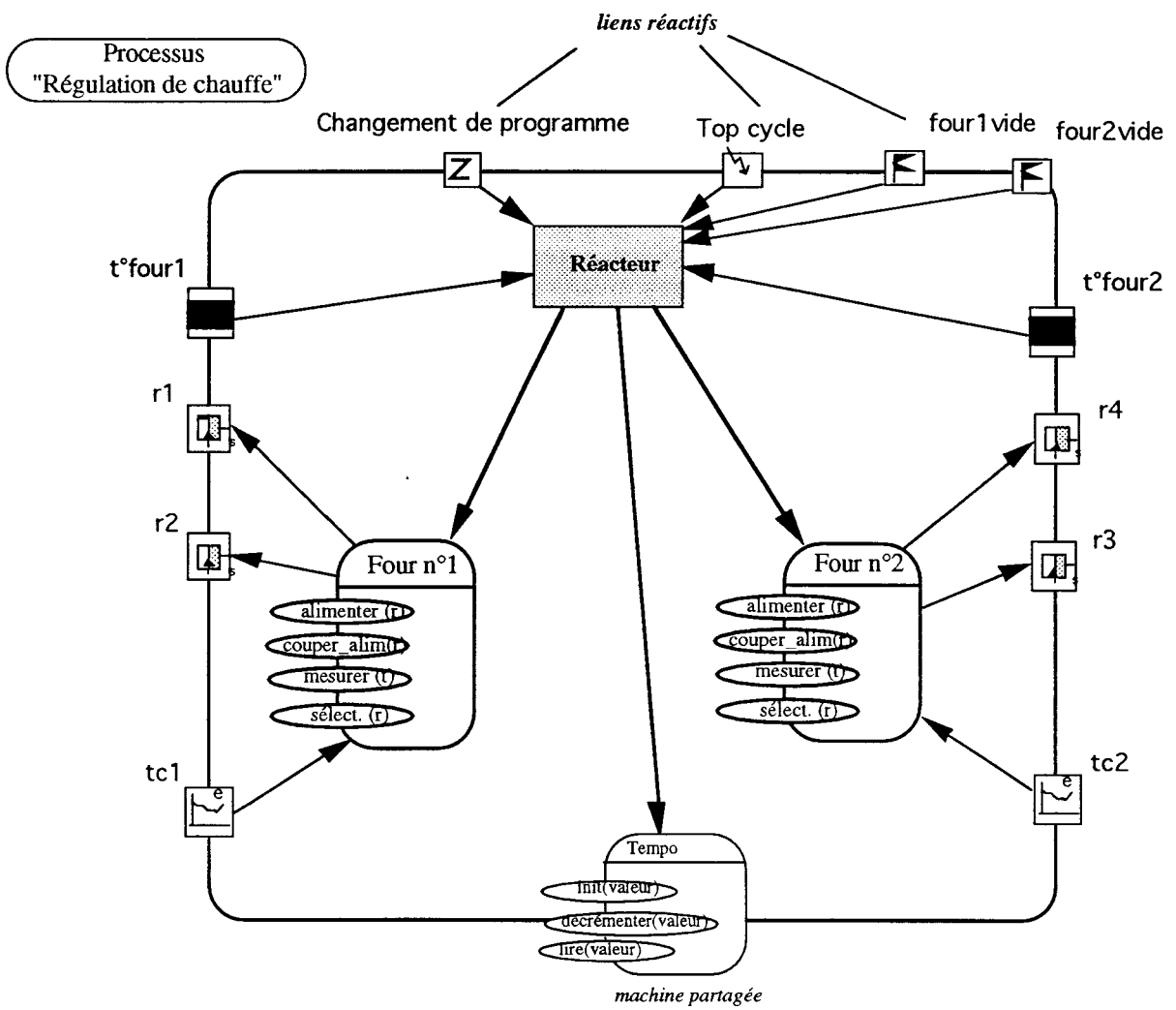


figure 40: Réacteur de processus

Comparé à l'approche Statemate [HAR 90], le réacteur regroupe, au sein d'un même composant, l'ensemble des éléments ayant la charge de produire le comportement, spécifié à l'aide d'un diagramme d'état (statechart), "d'activités" statemate liées à un processus. On note donc une différence sur ce point avec l'approche TOCCATA.

Ce composant représente néanmoins un **module** Statemate, élément de structuration du logiciel de l'application qui est associé à une représentation organique du système et définissant le "comment" alors que le "quoi" est formellement représenté à partir d'activity-chart et de statechart.

III.4.6 TYPE DE PROCESSUS / TYPES D'INTERFACES EXTERNES AUX PROCESSUS

En regard de la taxonomie adoptée en III.3.1.2, on reconnaît le caractère continu, discret ou mixte d'un processus à la nature de ses intéractions avec l'environnement extérieur:

et en particulier:

- un processus séquentiel, au fait qu'il est environné de liens opérationnels véhiculant des informations fugitives et/ou rémanentes;
- un processus continu, au fait qu'il gère des informations à la fois continues ou rémanentes;
- un processus mixte, au fait qu'il subit l'influence des trois types de protocoles précédents.

III.4.7 PARAMÈTRES D'ACTIVATION

Le réacteur d'un processus prend en outre le relais des liens temporels provenant de la description comportementale d'un sous-système. Ces liens temporels peuvent être le siège d'informations liées au déclenchement ou à la mort du processus. Nous les ferons apparaître sur l'axe transformationnel au titre de **paramètres d'activation**. Ces paramètres influeront sur le comportement du réacteur.

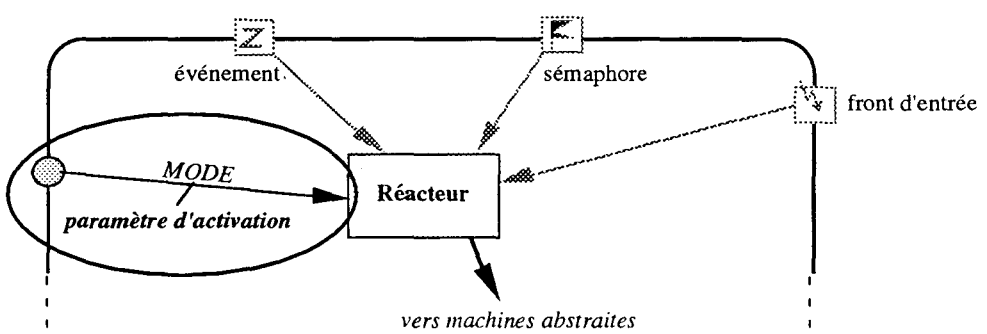


figure 41: paramètres d'activation

III.4.8 RÉACTEUR BOUCLÉ

Par ailleurs, dès lors qu'un processus connaît une interface continue, il se doit d'être cycliquement réactivé.
Nous associerons en conséquence un réacteur bouclé à tous les processus continus ou mixtes du SCC.

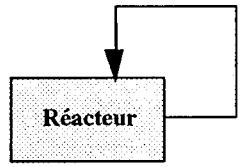


figure 42: Boucle de réactivation du processus

Cette condition n'est évidemment pas sans conséquence pour l'implémentation. Elle conditionne en particulier le choix du modèle d'exécution de la tâche sur laquelle doit être porté le processus. (cf. III.2.3)

III.4.9 MODÈLE CONCEPTUEL DE SYNTHÈSE

La plupart des concepts de l'axe transformationnel sont traduits dans le modèle suivant:

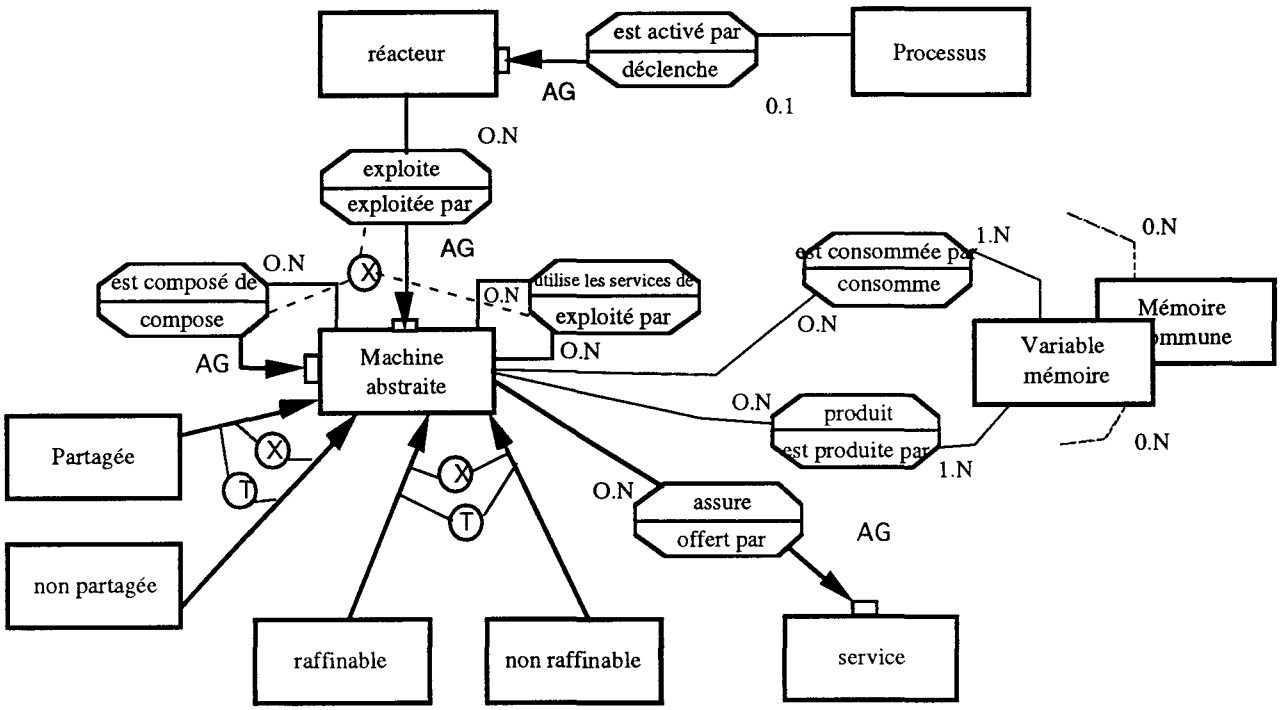


figure 43: schéma conceptuel de l'axe transformationnel

Seule la variable en mémoire justifie sa présence dès lors qu'elle est produite et consommée par une ou plusieurs machines abstraites du processus.
Par ailleurs, la contrainte d'exclusion portant sur les trois attributs de relation entourant la machine abstraite dans le modèle ci-dessus est garante du bon ordre hiérarchique des appels de services entre machines.

Le schéma précédent met en commun, avec le modèle de référence de l'axe des comportements, les concepts de processus et de réacteur, comme le montre la figure 44.

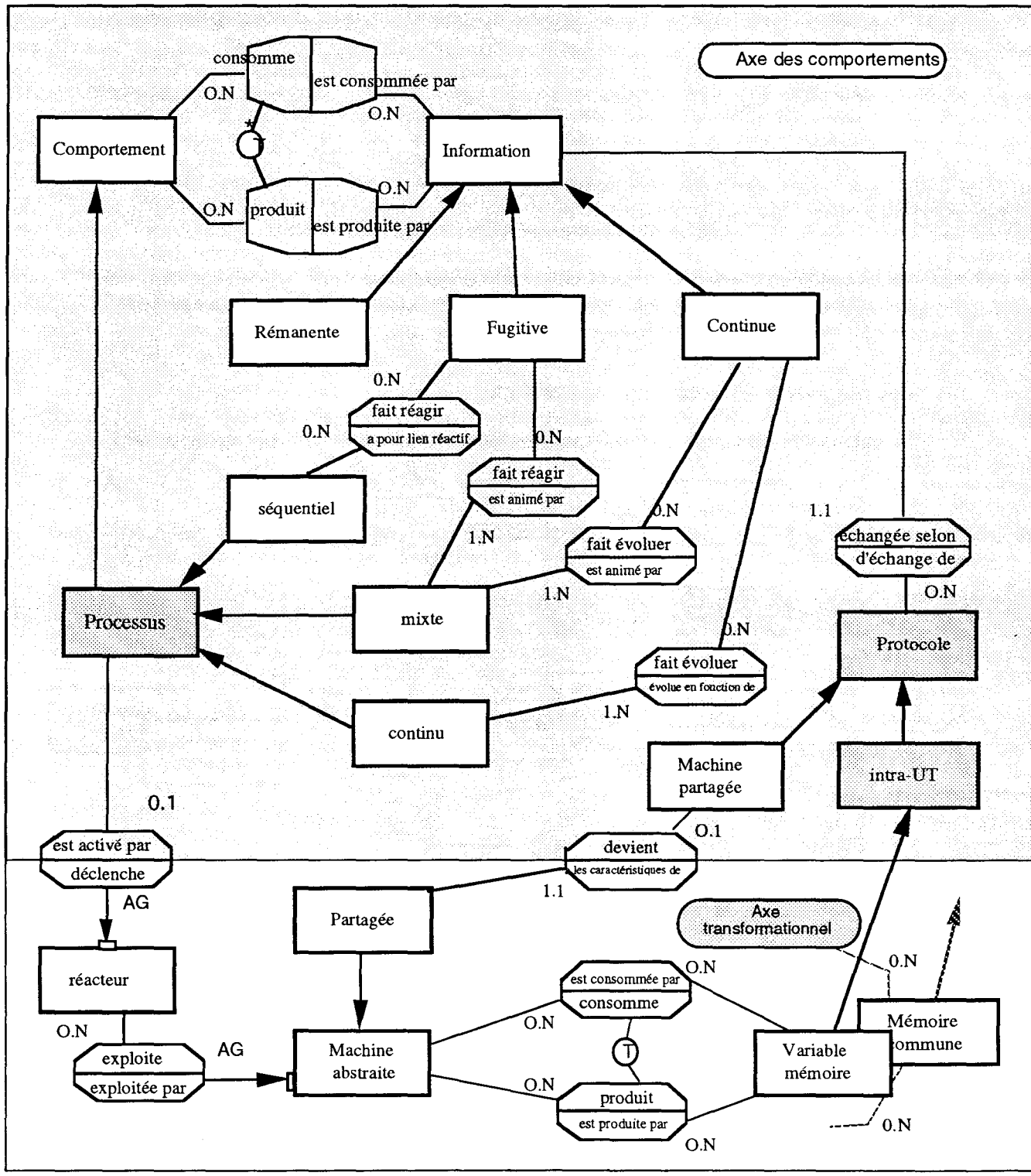


figure 44: Interfaces entre les axes comportemental et transformationnel

III.5 AXE TRANSFORMATIONNEL, NOTIONS DE RÉUTILISATION

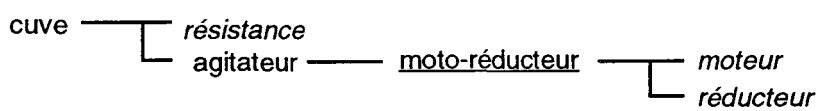
Nous avons précisé, dans le paragraphe II, que les machines abstraites apparaissant dans la description de l'axe transformationnel, encapsulent soit des ressources liées au procédé industriel, soit des ressources liées au contrôle direct d'organes de la partie opérative.

- Le procédé est généralement propre à l'application. Ses ressources et services associés sont, de ce fait, difficilement réutilisables à haut niveau. Elles ne conduisent donc pas systématiquement à des éléments génériques pour la conception.

- La partie opérative est, elle, constituée, quel que soit le type de système automatisé de production, **d'organes** et d'**objets physiques commandables** [CRU 91] qui, eux-mêmes, peuvent regrouper des **objets mono-fonctionnels**.

Exemple de sous-ensemble PO et d'objets physiques commandables associés:

Prenons la cuve d'un mélangeur comprenant une résistance de chauffage et un agiteur actionné par le biais d'un moteur électrique équipé d'un réducteur à deux vitessses:



Légende:

objet physique commandable

objet mono-fonctionnels

Comme le montre l'illustration ci-dessus, ce sous-ensemble de partie opérative compte deux organes principaux : la cuve et l'agitateur; et 5 objets physiques commandables dont 3 constituent des objets mono-fonctionnels.

Les objets mono-fonctionnels sont commandés et contrôlés via des signaux correspondant à des interfaces standards avec le SCC. Ils sont de plus souvent réutilisés d'une affaire à l'autre.
Il est donc souhaitable d'associer des instances de types abstraits à la commande de ces objets PO et de constituer par ce biais, au fil des applications, des bibliothèques d'éléments de conception réutilisables entre plusieurs affaires.

Prenons l'exemple d'un objet mono-fonctionnel "moteur" relié par un équipement de l'architecture matérielle de type "variateur de vitesse à contrôle vectoriel de flux":

A ce moteur piloté par un tel dispositif correspond, quelle que soit l'application visée, un élément de commande assurant l'interface entre les fonctions d'automatisme liées au procédé et le transfert des signaux envoyés au variateur. Il est souhaitable, de rendre cet élément générique en faisant, de la machine abstraite qui centralise les services d'accès aux commandes du variateur, un type abstrait particulier.
Dans le cas qui nous sert ici d'exemple, la machine abstraite encapsulant les ressources propres au couple moteur-variateur permettra de:

- *transmettre des ordres de marche et d'arrêt au moteur,*
- *préciser la rampe d'accélération et le couple nominal du moteur,*
- *donner sa vitesse en régime permanent,*
- *restituer des informations caractérisant son état (vitesse instantanée, vitessse nulle, vitesse atteinte ... etc ...)*

☞ Une machine abstraite intervenant dans la conception d'une application d'automatisme peut donc être l'instance d'un type abstrait, associé à une bibliothèque construite sur la base de capteurs et d'effecteurs standard (une vanne, un capteur de position, un capteur de température ... etc ...)

Des types abstraits peuvent également être associés à des capteurs et actionneurs intelligents. Ils donnent alors accès, lors de la conception, aux traitements attachés à ces interfaces spécifiques mais n'ont pas de représentant effectif dans le modèle de réalisation de l'application dans l'automate programmable (puisque sont implémentés sur les capteurs eux-mêmes).

On note en outre deux niveaux de standard:

- le standard pour une affaire (la redéfinition du "normal/secours" réutilisé une trentaine de fois dans une application donnée)
- le standard à plusieurs affaires (le type de commande préconisé par le fournisseur pour une vanne à retour de débit par exemple)

Certaines méthodes de conception d'automatismes proposent des approches orientées objets équivalentes, qu'elles soient ou non spécifiquement dédiées à une catégorie particulière de systèmes ([AMA 90] par exemple). Nous avons cité dans le chapitre d'introduction la méthode proposée par la société GEIODE qui, avec l'outil K-Sys, propose de concevoir une application avec une approche "bottom-up", c'est-à-dire allant de l'étude de la commande des éléments les plus fins de la partie opérative à l'élaboration des objets les plus sophistiqués d'automatisme.
Cette méthode n'offre toutefois pas de concepts et de mécanismes associés permettant de réutiliser les objets d'automatisme en les plaçant dans des bibliothèques de types abstraits.

La démarche présentée au chapitre IV a ceci d'original: qu'elle s'appuie, dès la phase de conception préliminaire de l'application, sur des concepts spécifiques permettant la réutilisation de tels composants.

Le sous-schéma conceptuel de la figure 43 se complète par les entités "Type Abstrait" (TA) et de "Bibliothèque de types abstraits" (BTA), entités liées à la machine abstraite par un mécanisme d'instanciation:

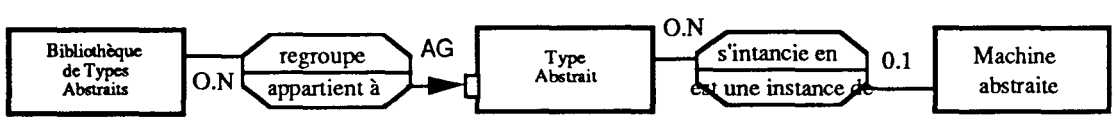


figure 45: machine abstraite - réutilisation

L'entité BTA tient la même place dans notre modèle des données de conception que l'entité EDL/B (Elément de Logiciel de type Bibliothèque) dans BasePTA pour les données de réalisation.

A l'identification des ressources et à la caractérisation des interfaces liées au réacteur des processus (cf. paragraphe précédent) s'achève la conception préliminaire du logiciel.
Intervient ensuite une phase de conception détaillée de ces différents éléments au niveau de laquelle le corps de chaque réacteur et celui de chaque ressource identifiés jusqu'alors, fait l'objet d'une description approfondie.
A cette description en pseudo-code se substitue en pratique très souvent la programmation des éléments de logiciel eux-mêmes, dans la mesure où ces derniers ne relèvent toutefois pas d'une grande difficulté de mise en œuvre.

Ne prétendant pas aborder dans ce mémoire ces deux phases en détail, nous ne donnerons, dans ce qui suit, que les éléments qui permettent de passer la description formelle d'un processus de commande à sa réalisation proprement dite.

III.6 PASSAGE DES DONNÉES DE CONCEPTION AUX DONNÉES DE RÉALISATION

III.6.1 NOTION D'ÉLÉMENT DE COMMANDE

En adoptant maintenant le point de vue de la réalisation, on note qu'un processus s'inscrit dans un programme implanté sur une unité de traitement. Ce programme décrit le déroulement du processus au moyen d'instructions spécifiques et fait appel à des **éléments de commande (EC)** qui, selon la norme CEI 1131 [CEI-91], correspondent:

- soit à des blocs fonctionnels
- soit à des fonctions

Rappel de la norme:

Un bloc fonctionnel regroupe un ensemble d'opérations programmées répondant à un besoin particulier d'automatisme. C'est une unité d'organisation qui peut renvoyer, après invocation de l'un de ces services, la valeur de plusieurs de ses données de sortie.

Une fonction ne renvoie, quant à elle, qu'un élément de donnée en retour. Son appel peut par conséquent servir d'opérande dans une expression littérale.

Une application d'automatisme est donc constituée de programmes structurés de la manière suivante:

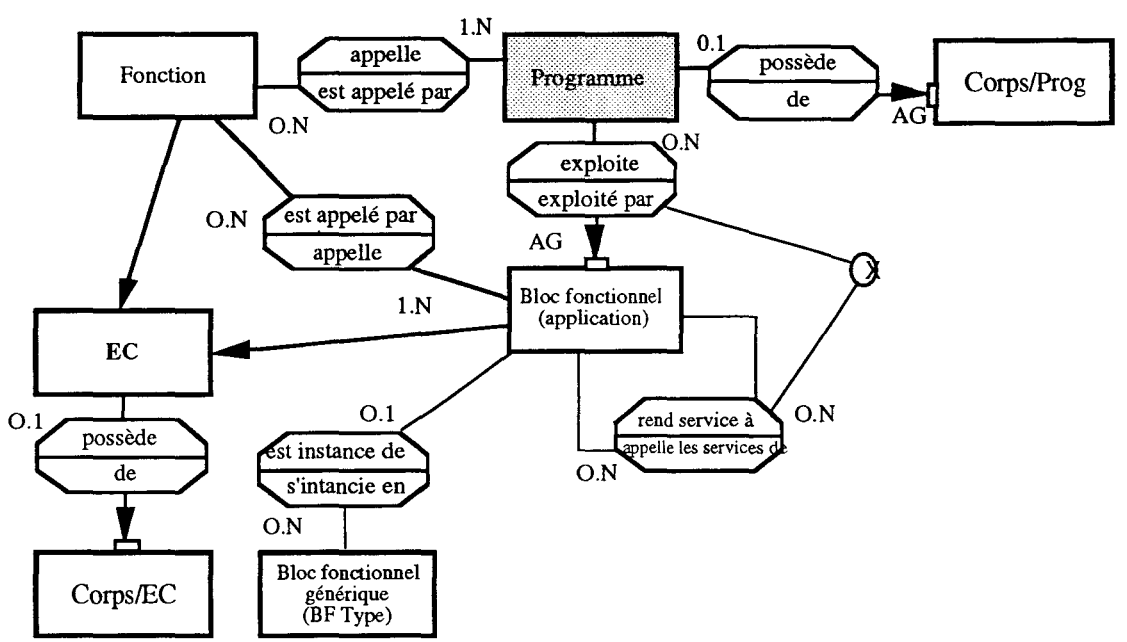


figure 46: Types d'éléments de commande

Les EC sont des composants procéduraux qui rendent les services attendus des machines abstraites identifiées en phase de conception de l'application.

Le corps du programme et de ses EC est de plus décrit à l'aide des langages de programmation adaptés aux types de processus à mettre en œuvre.

III.6.2 TRAÇABILITÉ ENTRE OBJETS DE CONCEPTION ET COMPOSANTS LOGICIELS D'UNE APPLICATION

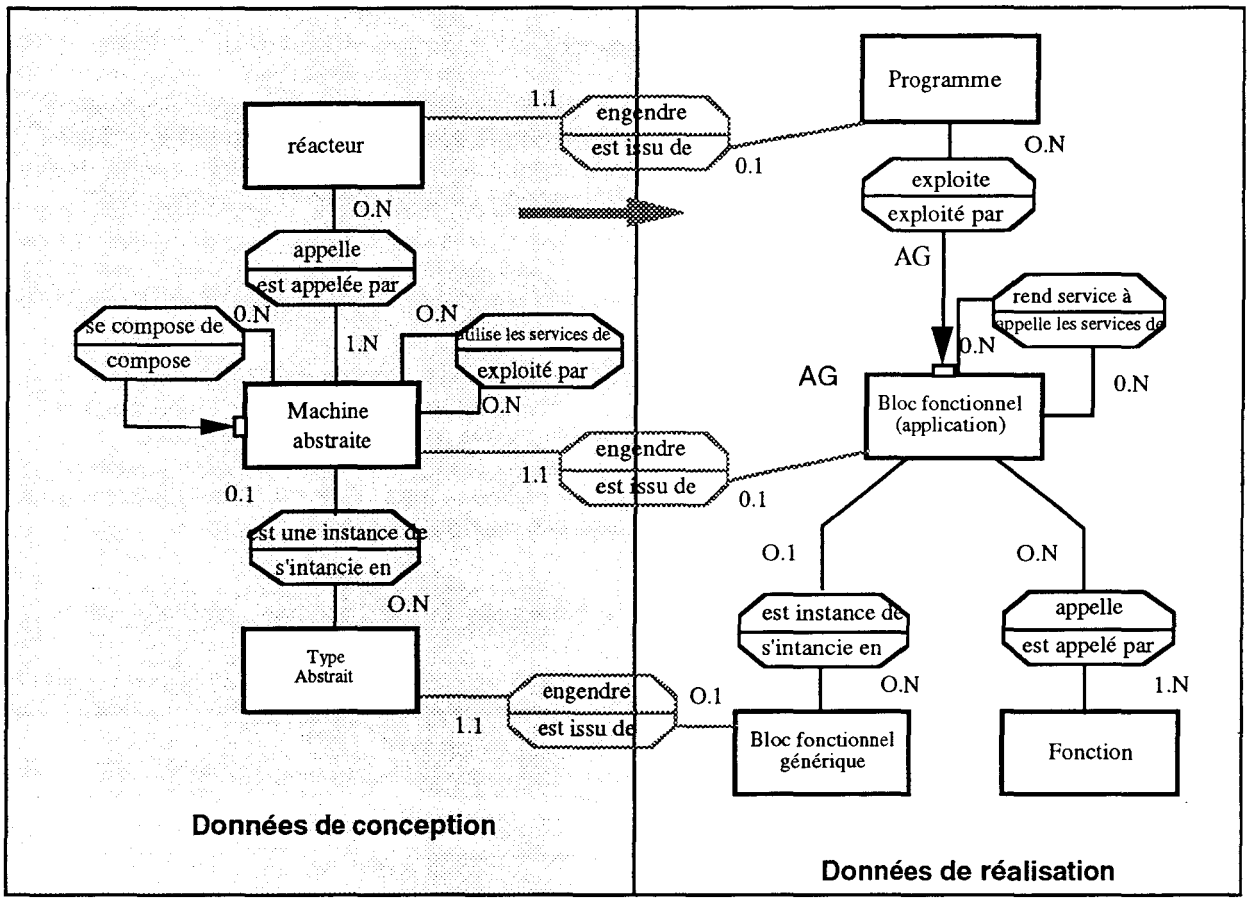


figure 47: Eléments de conception et composants logiciels de réalisation

La décomposition-type d'un processus de commande respectivement en un réacteur et des machines abstraites est en réalité très proche de l'organisation effective d'un programme d'automatisme.

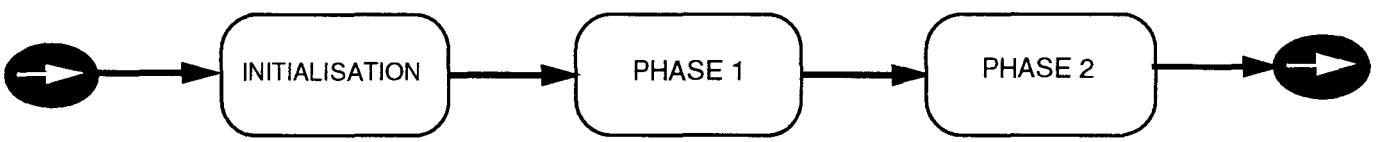
Son aspect réactif donne lieu au corps du programme générant le comportement attendu de l'exécution de ce dernier.
Son aspect transformationnel se traduit en une série de blocs fonctionnels et de fonctions qui, de la même manière que lors de la phase de conception préliminaire, sont généralement tirés de "Blocs Fonctionnels Génériques".

L'équivalence entre les notions de programme et de processus est importante. Elle est d'ailleurs exprimée dans les définitions de base de la norme automate 1131 part 3 [CEI-92], qui considère qu'un programme relève de *"l'ensemble logique de tous les éléments et constructions des langages de programmation nécessaires au traitement prévu des signaux requis pour la commande d'une machine ou d'un processus par un système d'automate programmable"*.

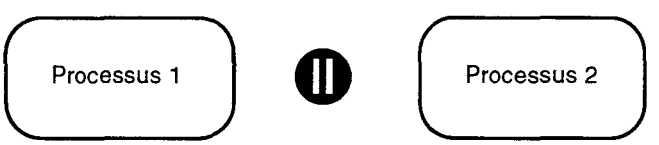
Un programme traduit donc la cohésion d'une partie des traitements de l'application.

Cette cohésion est à la fois structurelle (les traitements regroupés au sein d'un programme s'appliquent à des organes fonctionnellement dépendants) et temporelle (ces traitements sont par ailleurs régis par des règles spécificiques d'ordonnancement).

Le concept de programme n'a, en regard de sa définition, de réalité que par rapport aux traitements qu'il met en œuvre. Il peut, en pratique, regrouper l'exécution de plusieurs processus élémentaires (cf. III.3.1), à condition toutefois que ces derniers s'inscrivent bien dans une seule et même séquence d'opérations, comme par exemple la séquence suivante:



A contrario, la configuration qui suit ne pourra, en aucun cas, donner un schéma d'implantation unique:



puisque l'évolution de chacun de ces processus est considérée à priori indépendante.

- L'aspect transformationnel d'un processus (ses machines abstraites) se traduit par une arborescence de blocs fonctionnels et de fonctions. Le "graphe d'appel" de ces EC est inscrit dans le corps du programme traçant l'évolution du processus.

Ce corps peut par exemple être décrit de la manière suivante:

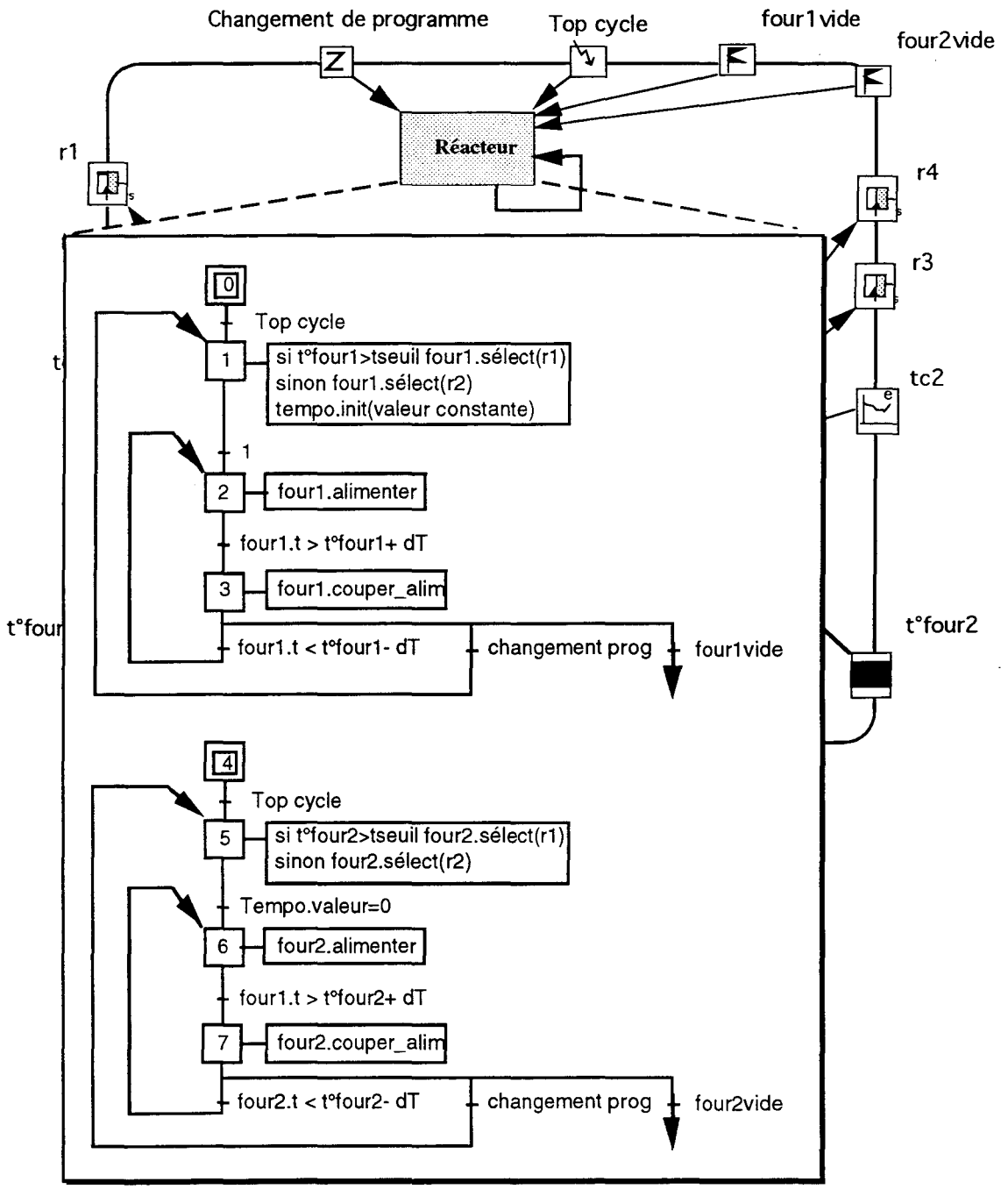


figure 48: conception détaillée de l'aspect réactif du processus présenté figure 40

Les instructions des 2 schémas grafcet ci-dessus font appel aux machines abstraites encapsulant les ressources four1, four2 et tempo de la figure 40. L'activation de ces machines est symbolisée par le "." suivant la désignation de chacune des ressources utilisées par le processus. (Comme par exemple: four1.t)

D'autres langages peuvent être utilisés pour décrire de façon détaillée le corps d'un réacteur. Tout dépend en fait de la qualité du process à représenter et surtout des langages de programmation offerts sur les consoles servant à configurer et programmer les divers équipements d'automatisme.

III.6.3 LIENS AVEC LES LANGAGES DE PROGRAMMATION

Si l'on se réfère à la classification soutenue depuis le début de ce chapitre, distinguant les processus continus des processus séquentiels et des processus mixtes, il nous est alors permis de préférer des types de langages à la description de chacun de ces types de sous-systèmes élémentaires.

En effet, la description d'un processus séquentiel se fait plus aisément à l'aide d'un formalisme basé sur une représentation à états-transitions, tels que le propose intrinsèquement le langage Grafcet [COL 91] ou les réseaux de Petri [PET 89]. Il s'avère donc préférable d'employer, autant que faire se peut, ces types de formalismes à chaque fois qu'il l'est permis (c'est-à-dire à chaque fois que l'outil servant à la programmation de l'équipement supportant l'exécution de ce processus propose ce type de langage)
Pourront être également utilisés, pour décrire en détail le réacteur d'un processus séquentiel, des langages à contacts, comme le langage à échelle par exemple. Ceux-ci permettent en effet d'exprimer les conditions de transition entre état du processus au moyens d'expressions combinatoires conditionnant le lancement d'actions élémentaires.

Les langages littéraux serviront plus spécialement à décrire des processus continus.

Les processus mixtes verront au choix leur réacteur être détaillé à l'aide de l'un ou l'autre de ces types de langages.

Certains processus, hautement réactifs, pourront être décrits à l'aide de langages synchrones, tel qu'ESTEREL [CHPP 87], qui par son approche événementielle, est tout à fait adapté à la représentation d'éléments de logiciel dont l'environnement est presque uniquement fait de signaux purs.

Nous exclurons néanmoins de toute cette approche les formalismes contribuant à définir des schémas "flots de données" et en particuliers les langages "à boîtes", qui constituent une alternative franche à une description séparée des aspects réactifs et transformationnels d'un processus.
Les liens entre boîtes sur ces types de langages véhiculent en effet indifféremment des signaux et des données (cf figure 28). Ils expriment par ailleurs des liens d'activation qui peuvent ne plus être associés à des opérateurs temporels ne permettant alors plus la description des traitements à un seul instant d'évolution du processus. La discrimination des protocoles en fonction des types d'informations échangées avec l'environnement extérieur, n'a également plus cours, puisque les ports de communication entre boîtes sont généralement dotés, dans de tels langages, d'une sémantique unique.

III.6.4 LIENS AVEC LES NORMES BASEPTA ET CEI 1131

De la même façon que dans le chapitre précédent, il nous est ici permis d'étudier les rapprochements entre notre modèle de données et l'aspect logiciel de la norme BasePTA.

Tous deux laissent apparaître des éléments de logiciels. Notre modèle en distingue deux sortes, BasePTA uniquement des EDL (Elément De Logiciel).

La structure récursive de la norme exprimentale Z68-901 trouve donc son pendant en conception au niveau des concepts de "réacteur" et de "machine abstraite".

BasePTA intègre d'autres notions, comme par exemple le SECI (Système d'Exploitation Configuré et Implanté) d'un sous-système automatisé de production, qui offre des ressources de trois types au logiciel de commande, RSU (Ressources Système Utilisé) de type Accueil, Logiciel ou Variable.

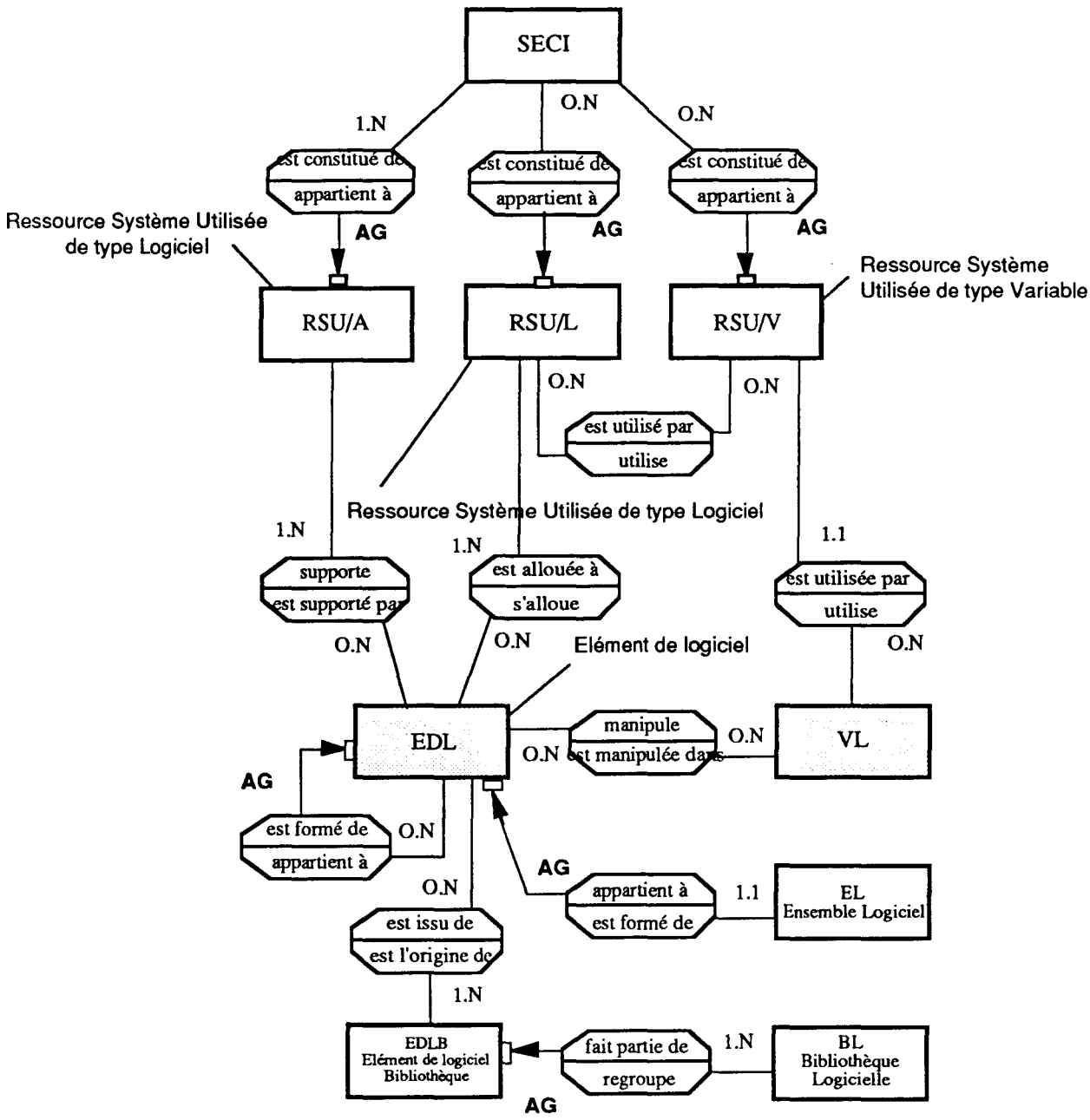


figure 49: Extrait de BasePTA, aspect logiciel

Ces ressources ne sont pas explicitement représentées dans notre modèle de base car elles ne sont pas, à proprement parler, nécessaire à l'étude du comportement attendu de l'application. L'allocation de ces dernières n'est qu'une conséquence de cette étude.

Notre modèle trouve donc, conformément aux dires du § III.6.2, son extension dans des sous-schémas "appliqués" de BasePTA (au sens précis donné à ce terme dans la norme).

BasePTA, tout comme notre schéma conceptuel, affiche par ailleurs des concepts se rapportant à la réutilisation de composants logiciels génériques. La traçabilité entre les modèles de données de conception et de réalisation n'en est, de ce fait, que plus simple à établir. A un élément en bibliothèque de types abstrait correspond en effet un bloc fonctionnel générique s'inscrivant dans le concept d'EDL/B précédent.

Notre modède se rattache par ailleurs aisément aux concepts de la norme automate CEI 1131 [CEI-92], qui recouvrent partiellement ceux de BasePTA, comme le montre le figure ci-dessous:

Concepts de la norme CEI 1131 - Partie 3

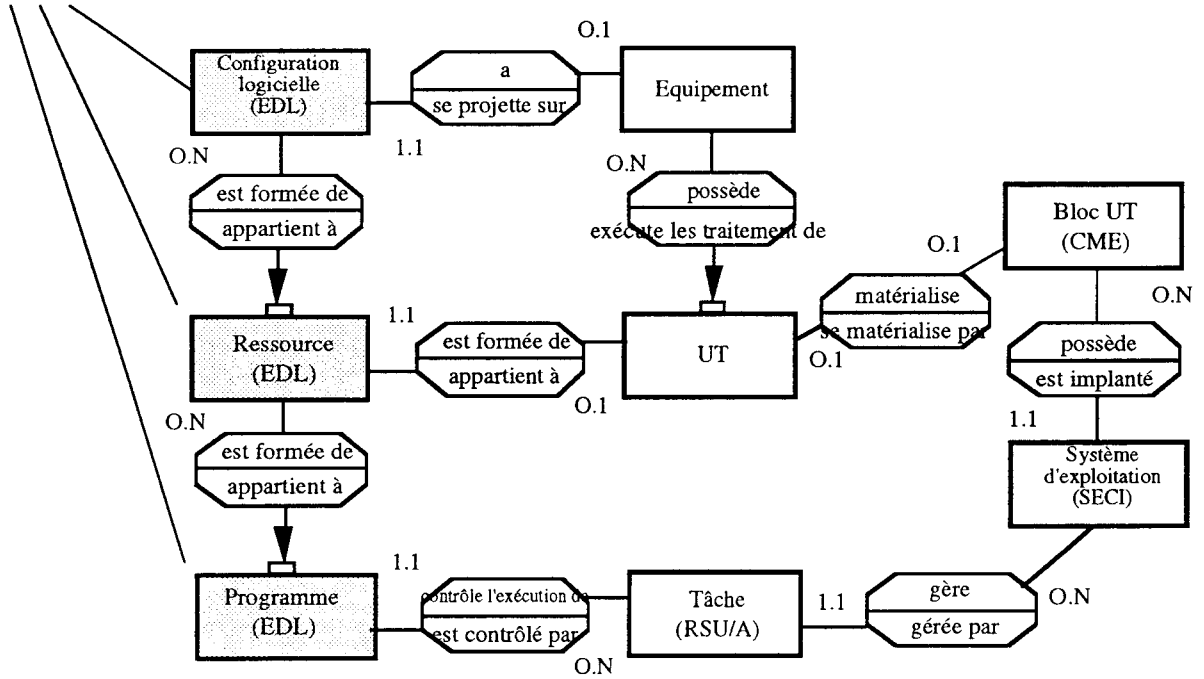


figure 50: Portion d'un schéma de consensus entre le modèle de la norme Automate CEI 1131, le modèle BasePTA et notre représentation spécifique du logiciel d'application

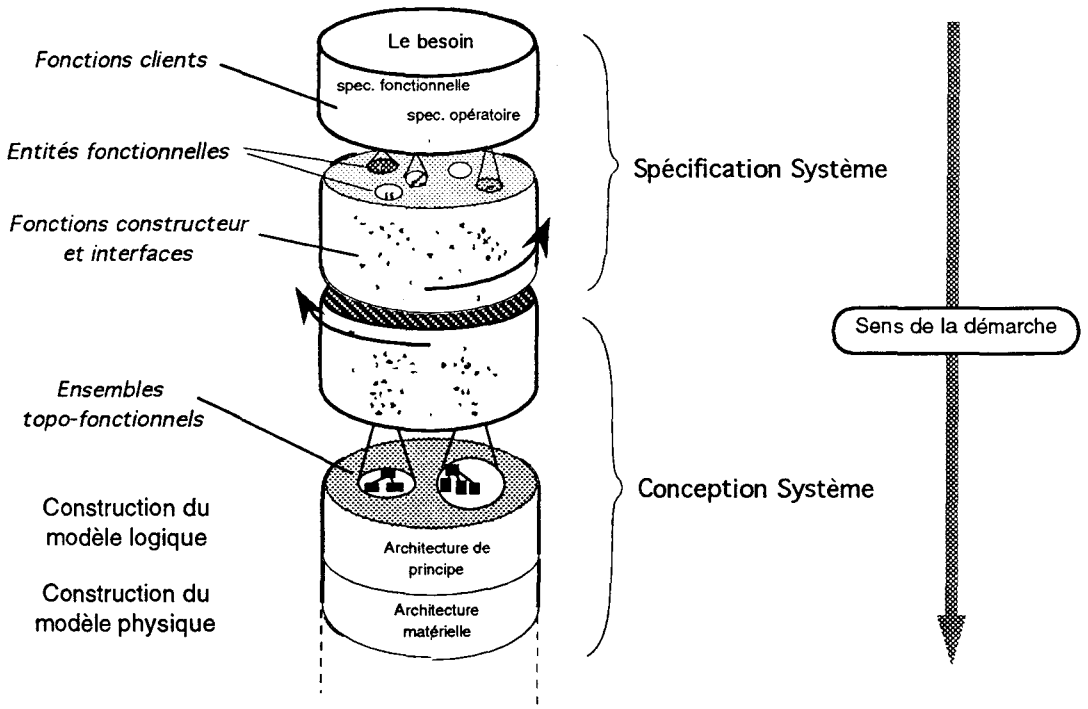
La configuration (logicielle) CEI regroupe l'ensemble des éléments de logiciel attachés à un équipement. Elle représente de plus l'EDL "chapeau" pointé par l'EL (l'Ensemble Logiciel BasePTA) associé à chaque équipement de l'architecture matérielle.

La ressource CEI est liée au logiciel attaché à une unité de traitement, matérialisé par le bloc UT représentatif de cette unité logique. Ce bloc UT n'est autre qu'un CME (Composant matériel élémentaire) de la norme BasePTA.

La notion de programme prend fait et cause dans les modèles de réalisation normalisés à la CEI et à l'AFNOR. Elle est liée au concept de tâche (RSU/A BasePTA), puisqu'un programme d'automatisme s'implante sur ce type de ressource allouée par une unité de traitement.

CHAPITRE IV DE LA DESCRIPTION DE L'ARCHITECTURE MATERIELLE DU SCC

Résumé du chapitre



"Vue intuitive du mapping fonction -> matériel"

Construire l'architecture matérielle du système de contrôle-commande réclame de connaître:

- les fonctions attendues du système,
- l'organisation de sa partie opérative,
- ses interfaces avec les opérateurs de conduites,
- ses interfaces avec d'éventuels autres SCC,
- les propriétés dynamiques estimées de ses traitements applicatifs,
- et enfin la variété de choix des équipements d'automatisme réutilisables pour l'affaire.

Ce chapitre montre comment sont à tour de rôle pris en compte ces éléments dans le processus qui mène à la définition de l'architecture matérielle.
Il se limite, après avoir exposé comment classer les différents produits issus de la spécification, à la description de la couche "automatisme" de modèle CIM et considère, pour ce faire trois étapes de conception successives:

- une première étape s'accordant à identifier les besoins fonctionnels relatifs à chaque zone sur site
- une seconde étape s'attachant à choisir et organiser les équipements répondant à ces besoins,
- une troisième étape se vouant à la configuration matérielle détaillée des équipements de l'architecture.

L'architecture matérielle d'un système de contrôle-commande prend forme dès la phase de réponse à l'appel d'offre (RAO). Elle puise sa source dans les documents rédigés à cette occasion ou établis lors de la spécification détaillée du système.

Elle arrive ensuite peu à peu à maturation en acquittant progressivement les contraintes et spécificités de fonctionnement qui incombent à l'application, en l'occurrence celles répondant directement au besoin initialement émis par le client ou celles qui, plus tardivement, sont induites par certains choix techniques retenus en phase de conception.

Nous tenterons dans ce chapitre de formaliser la démarche qui mène peu à peu à la définition de l'architecture matérielle d'un SCC en dégageant les constantes du raisonnement tenu en phases de RAO et de conception système.

Parcourons pour cela le cycle de vie dans le sens normal de son déroulement. (cf. carte des activité de développement figure 2 chapitre d'introduction)

IV.1. L'ÉTUDE PRÉALABLE DU SYSTÈME

IV.1.1 MODALITÉS CONTRACTUELLES

Le cycle de vie servant de référence à nos propos a l'originalité de regrouper, au sein d'une première phase, dite "d'étude préalable", les activités menées avant de conclure un contrat. Il adopte le point de vue d'un ensemblier sur le système de contrôle-commande en fixant comme préalable à la naissance d'une affaire, deux étapes distinctes respectivement consacrées à l'étude du procédé et la réponse à l'appel d'offre.

"On entend par étude du procédé la définition par le client ou par une société d'ingénierie du procédé à mettre en œuvre. Cette étude se traduit par la rédaction d'un cahier des charges"

La phase de réponse à appel d'offre consiste, à partir de ce cahier des charges, à déterminer les principales fonctionnalités du système et en déduire les équipements qui les supporteront. Elle permet d'estimer l'ensemble des risques encourus en termes de délais, financement et difficultés techniques et ainsi de chiffrer la solution qui sera proposée au client en constituant un dossier comprenant deux volets principaux:

une partie technique et une partie juridique et financière représentant "la proposition commerciale" du fournisseur.

Le volet technique comporte au moins:

- une explication sommaire des différentes fonctionnalités du SCC,
- une ébauche de l'architecture du système en terme d'équipements, à laquelle s'adjoint une liste de fournitures matérielles ou de services,
- les caractéristiques techniques de ce système et en particulier ses performances et facteurs de qualité;

Ce volet est rédigé en langage naturel.

La proposition commerciale contient pour sa part:

- un devis, c'est-à-dire une liste des fournitures et bordereaux de prix,
- un échéancier de livraison des documents, des équipements et de leur mise en service,
- les modalités de réception de l'ensemble industriel.

"L'échéancier peut comporter une étape de pré-étude aboutissant à la fourniture des dossiers de spécification et de conception du système de telle sorte qu'il soit obtenu un accord sur la solution technique proposée après signature du contrat"
Cet accord n'exclut pas l'engagement du fournisseur à chaque fois qu'il fait une offre.

IV.1.2 PROCÉDURE DE NÉGOCIATION

Les règles régissant cette étape "d'avant-vente" sont plus attachées au bon déroulement d'une procédure commerciale qu'à une approche scientifique rigoureuse nécessitant en particulier de prouver toute hypothèse à la base d'un raisonnement devant aboutir à des conclusions fermes et définitives.

La procédure de négociation observée à cette étape du cycle de vie s'apparente à une boucle au niveau de laquelle la solution technique est progressivement ajustée aux souhaits et aux ressources du client:

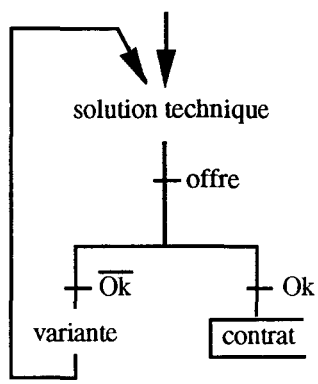


figure 51: cycles de négociation technique

Cette boucle s'achève dans le meilleur des cas par l'obtention du contrat qui clôt, pour dans chaque affaire, la phase de réponse à appel d'offre.

L'architecture matérielle du SCC est donc définie en deux étapes:

-une première ébauche en est fixée dès la RAO et permet d'étayer chaque offre; elle s'appuie sur une identification sommaire des fonctionnalités du système et de ses interfaces;

-une organisation plus détaillée du SCC est ensuite établie en regard d'un dossier complet de spécification dûment approuvé par le client; cette organisation décrit les équipements matériels du système, leur liens hiérarchiques ainsi que leurs rôles respectifs.

IV.1.3 INCIDENCES MÉTHODOLOGIQUES

La démarche méthodologique conduisant à l'architecture matérielle détaillée du système n'est donc, par suite des activités menées pendant et après l'étape de RAO, pas linéaire mais faite d'ajustements répétés qui permettent, pour l'essentiel, au client et au fournisseur de bien se comprendre.

On représente généralement cette démarche par une spirale qui, bien après l'étape conduisant à un accord signé entre les deux parties, se poursuit dans le but d'affiner la solution matérielle.

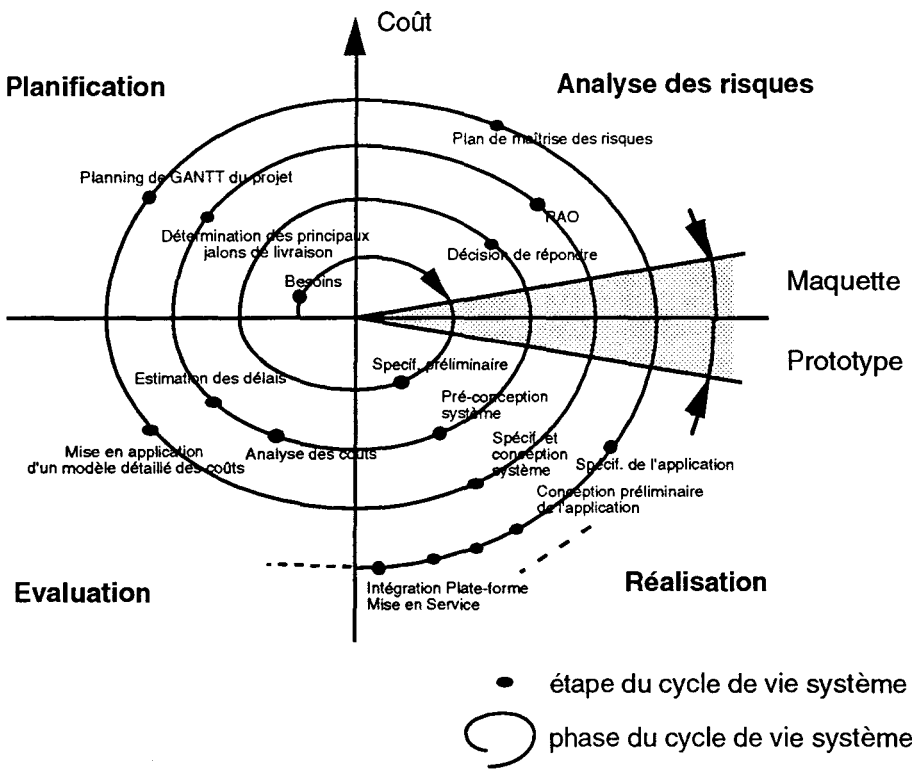


figure 52: Spirale de construction progressive d'une solution

Bien qu'elles n'y soient pas toutes représentées dans les cadrans ci-dessus, les étapes du cycle de vie système ont chacune un centre particulier d'intérêt qui est par ailleurs périodiquement examiné dans le cycle en spirale symbolisant l'évolution du projet et se focalise sur des données de plus en plus déterminées du système.

On note ci-avant respectivement comme centres d'intérêts les domaines de l'évaluation des coûts et des délais, de la planification du projet, de l'analyse des risques liée à chaque début de phase (pré-étude, spécification, conception codage et intégration), et enfin de la réalisation qui s'attache aux produits issus de ces phases.

Chacun des ces centres d'intérêt fait l'objet de recommandations et de procédures dans la méthodologie MODAL-v2 [MOD 93]

Aux étapes sont associées des méthodes spécifiques de développement qui servent généralement à plusieurs reprises dans le cycle de vie et au niveau d'un même cadran.

Ainsi sont par exemple employées des méthodes de spécification fonctionnelles pour d'une part effectuer la spécification du système puis, pour d'autre part, déterminer quelles doivent être les fonctions attribuées aux différents équipements de l'architecture, en phase de spécification de l'application:

Rappel de carte des activités:

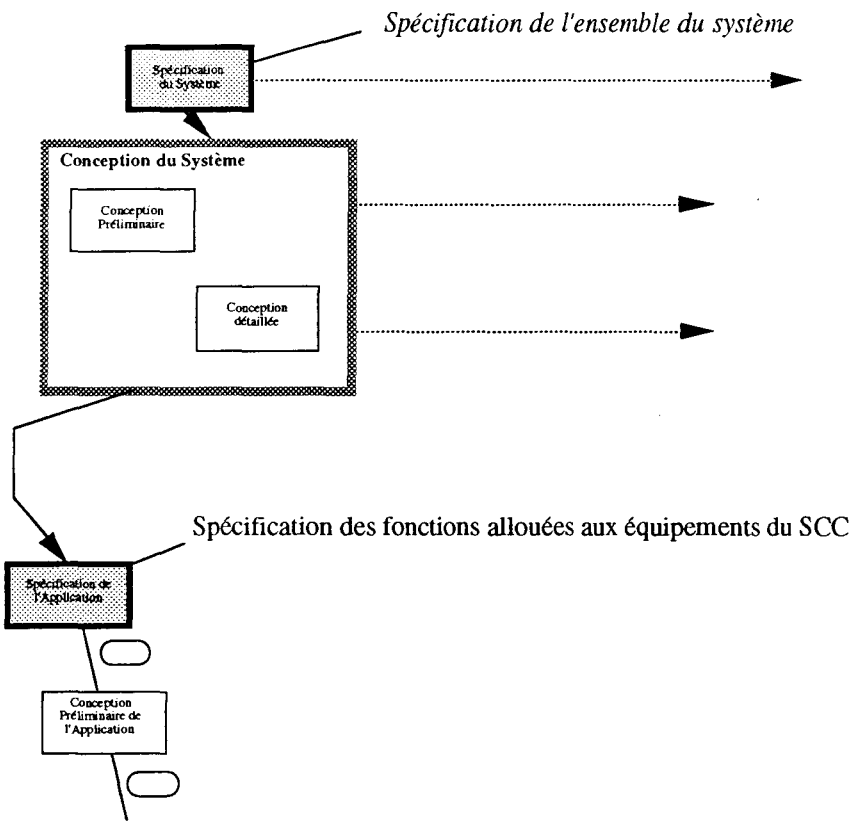


figure 53: les activités de spécification dans le cycle en "V" de l'application

L'établissement d'un devis constitue par ailleurs un risque pour le fournisseur qui, en se déterminant dès la phase de réponse appel d'offre, n'est pas certain de pouvoir appliquer l'architecture initialement présentée au client lorsqu'il aura connaissance des fonctions développées du système.

Il peut choisir, afin de limiter les risques, d'allonger le cycle de vie au niveau de la pré-étude lui permettant de répondre à l'appel d'offre, ce qui le conduit alors à une définition moins imparfaite de la solution mais il risque dans ce cas de voir, au fil des négociations, le besoin évoluer au gré des souhaits changeants du client.

Nous sommes donc ici confrontés au paradoxe de la prise de décision qu'illustre le tracé des courbes ci-dessous:

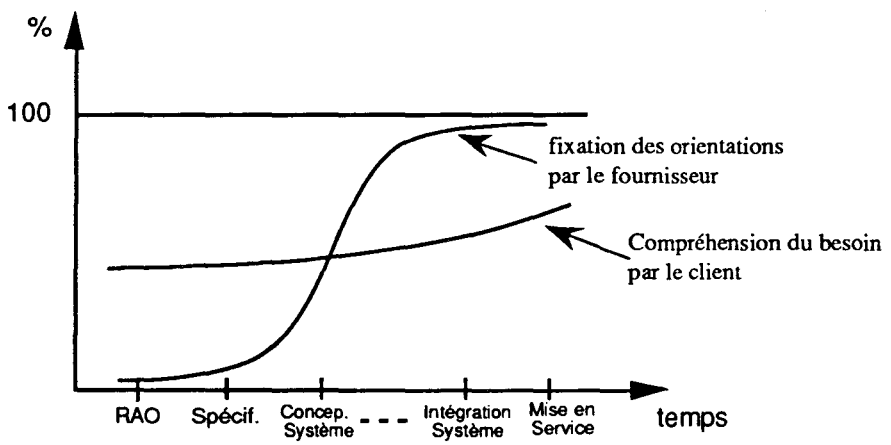


figure 54: Paradoxe de la prise de décision

Dans l'industrie, allonger le cycle de vie conduit généralement à une impasse qu'il faut à tout prix éviter. Ce n'est par contre pas le cas des projets militaires et spatiaux (Hermes par exemple) qui se caractérisent par des cahiers des charges très stables.

On voit, par rapport à ce qui précède, que le seul moyen de limiter les risques pour des projets à caractère industriel est de raccourcir la durée du cycle de vie du SCC en innovant peu et en réutilisant des solutions types capitalisant ainsi le savoir-faire de la société d'ingénierie impliquée dans la conception du système, ceci pour chaque domaine d'application donnée.

Conclusions:

La méthodologie est en conséquence fortement guidée par les principes de la réutilisation.

Les projets évoluent selon une technique d'approximation (projets en spirale) qui rend le modèle des phases du développement difficile à établir et réglementer.

IV.1.4. L'ÉTAPE DE RAO: PREMIÈRE ÉBAUCHE D'ARCHITECTURE DU SYSTÈME

La première ébauche d'architecture du SCC est donc établie à l'occasion de la réponse à appel d'offre.
Elle est issue de l'examen des grandes fonctionnalités du système.
Elle forme l'**architecture de principe** du SCC en plaquant les fonctions de ce dernier sur des équipements-types interconnectés.

L'ébauche construite à cette étape applique, aux niveaux de commande consacrés à l'automatisation du procédé, la théorie dite "du miroir" [TIX 89] selon laquelle l'organisation des machines de traitement répondant à ce rôle reflète, pour une large part, l'organisation de la partie opérative. Elle permet par ailleurs d'effectuer une première estimation des ressources nécessaires à la mise en œuvre des niveaux plus élevés de conduite et de supervision du procédé en tenant compte des prérogatives du client, notamment celles concernant l'imagerie du SCC.

IV.1.4.1 LES ÉQUIPEMENTS DU SCC

Cette esquisse du système voit le jour, en faisant correspondre, aux organes de la partie opérative, des équipements d'automatisme dimensionnés pour traiter avec les performances attendues, les informations du procédé. Elle tente de réutiliser des architecture types d'équipements répondant à des **modèles de produits** proposés par les différents fournisseurs d'automatismes.
La structuration de ces équipements doit permettre de répondre favorablement aux exigences de fiabilité, de sécurité, de disponibilité et de maintenabilité du matériel énoncées au cahier des charges.

"Certains clients imposent par ailleurs des marques de matériels, des types de réseaux spécifiques, des modèles d'équipements compatibles avec certains standards de communication ou avec des normes d'environnement plus ou moins sévères"

Toutes ces exigences, justifiées par les contraintes du client, doivent être prises en compte par la société d'ingénierie qui répond à l'appel d'offre.
Le savoir-faire de cette dernière doit donc être assez large pour ne pas dépendre des spécificités d'une seule gamme de produits d'automatisme.

Une architecture-type doit en particulier pouvoir dériver sur plusieurs types de matériels, d'où l'intérêt de suivre de très près les produits répondant aux normes récentes de communication comme FIP par exemple.

IV.1.4.2 LES SEGMENTS DE RÉSEAU DE COMMUNICATION DU SCC

Au fur et à mesure qu'est construite l'architecture de principe du SCC sont tissés des liens de communication entre les équipements de cette organisation. Ces liens font apparaître des segments de réseaux dont chaque modèle est choisi en fonction du flot estimé d'informations sur ces canaux d'échanges.

L'ébauche d'architecture obtenue permet d'établir un devis matériel de l'installation en considérant des configurations approchées d'équipements, c'est-à-dire des configurations permettant à priori de prendre en charge les sous-ensembles d'interfaces identifiés pour chaque sous-système formé entre le SCC et sa partie opérative.

Ce devis sert de référence à la proposition émise à l'occasion de la réponse à l'appel d'offre. Son chiffrage tient à priori compte des aléas de la conception.

IV.2. LA SPÉCIFICATION DU SYSTÈME

Notre objet n'est pas de traiter de manière détaillée des procédures de spécification système. Il nous incombe surtout de connaître ici le résultat de leur application pour concevoir ensuite le SCC. Nous exposerons dans ce sens les éléments contenus dans les documents relevés à l'issue de cette phase, en l'occurrence les fonctions et entités fonctionnelles caractérisant le système.

IV.2.1 FONCTIONS CLIENT / FONCTIONS CONSTRUCTEUR

Le rôle de la spécification système est de décomposer le problème soulevé par le client en une somme de sous-problèmes que le fournisseur sache résoudre par son expérience dans le domaine.
Le besoin apparaît donc en terme de "**fonctions client**".
Il doit être traduit en une série de "**fonctions techniques constructeur**".

IV.2.1.1 LES FONCTIONS CLIENT

Les "fonctions client" listent exclusivement les services rendus par le système à ses utilisateurs (exemples: réguler température four 1, produire historique des pannes cellules 3, contrôler vitesse d'avance de la machine transfert etc ..).
Elles ne font pas état des services "internes" au SCC permettant d'assurer ces fonctions (exemples: communiquer entrées ANA à la supervision, transmettre données de recette aux automatismes etc ...).
Elles ne décrivent pas non plus comment sont liées ces fonctions.
Elles ne tiennent enfin à priori pas compte des critères permettant d'éclairer le besoin dans le sens de la réutilisation.

Le système est donc de ce fait généralement décrit par rapport aux sous-procédés à mettre en œuvre dans l'application ou en fonction des opérations à effectuer pour conduire et superviser le procédé. De ce point de vue essentiellement "utilisateur" ne ressort qu'une description imprécise du système, ne permettant pas de fixer le besoin avec un niveau de détail suffisant pour envisager sa conception.

Il est requis une autre description du SCC, que s'emploie à réaliser le fournisseur ou la société d'ingénierie responsable de la maîtrise d'œuvre du projet.
Cette description est composée d'entités fonctionnelles, faisant une synthèse des fonctions client, et de fonctions plus détaillées, attachées à des niveaux hiérarchiques précis de contrôle et de commande du procédé.
Ces dernières tendent à être, pour un petit nombre d'entre elles, plusieurs fois représentées dans la même affaire (fonctions alors qualifiées de génériques) ou bien issues de développements antérieurs.
Nous les distinguerons des "fonctions client" par l'appellation "fonctions techniques constructeur".

IV.2.1.2 DÉFINITION D'UNE FONCTION CONSTRUCTEUR:

Une fonction technique constructeur constitue un élément de proposition du fournisseur allant dans le sens d'une décomposition méthodique de toutes les activités du système de contrôle-commande.

Cette décomposition est axée sur la spécification détaillée des services rendus par l'application en attachant chacun de ses services à des organes précis de la partie opérative, à des fonctions de niveau 1, 2 ou 3 du modèle CIM, à des services externes ou internes du SCC.

IV.2.2 BREF DESCRIPTIF DE LA DÉMARCHE

Pour déterminer les fonctions "techniques constructeur" du SCC, le fournisseur adopte une démarche qui consiste tout d'abord à regrouper les fonctions client dans des entités fonctionnelles puis à traduire ces entités fonctionnelles par des fonctions plus précises, progressivement décomposées et jugées plus pertinentes pour la réalisation.

Les entités fonctionnelles, regroupant des ensembles cohérents de fonctions, suscitent quelques fois, à leur niveau, la réutilisation de composants matériels ou logiciels génériques. C'est en pratique surtout le cas d'affaires similaires ou de projets mettant en jeu plusieurs tranches identiques à spécifier.

La procédure ci-dessus revêt un double intérêt:

- celui de fournir, d'une part, un exposé clair et succinct du besoin
- celui d'être, d'autre part, le support de toute interprétation, par le client, des fonctions proposées par le fournisseur.

Les entités fonctionnelles sont généralement structurées par domaine de compétence ou par types d'équipements .
Nous pouvons citer pour exemple les domaines de l'imagerie, de la supervision, des traitements séquentiels, de la régulation ou de la sécurité du système; ou comme types d'équipements liés à ces domaines: des postes opérateurs, des postes de conduite et de supervision, des automates programmables industriels, des automates régulateurs, etc ..

IV.2.2.1 ORGANISATION DE LA DOCUMENTATION

Chaque entité fonctionnelle donne lieu à un document de spécification comportant au moins un volet fonctionnel et un volet opérationnel.

Ce document est appelé DSEF (Document de Spécification d'une Entité Fonctionnelle) dans la méthodologie MODAL v2.

Chaque interface entre entité fonctionnelle fait également l'objet d'un document divisé en 3 parties, distinguant selon les cas:

- les interfaces homme/machine alors mises en jeu
- les interfaces avec le procédé
- d'éventuels éléments d'interface avec d'autres systèmes

Un document chapeau a, par ailleurs, la charge de décrire l'organisation générale du SCC et adopte le point de vue utilisateur requis pour permettre un bon dialogue avec le client.

C'est le DDS (Document de Définition Système) de MODAL v2. Son organisation est plus complète.
Elle présente les grandes fonctionnalités du système, son environnement direct, ses comportements, ses principes de mise en œuvre, ses caractéristiques et performances, la manière dont celui-ci doit être maintenu et enfin l'organisation des documents de spécification lui faisant référence.

IV.2.2.2 CRITÈRE D'ARRÊT DE LA DÉCOMPOSITION FONCTIONNELLE

La décomposition des entités fonctionnelles du système en fonctions de plus bas niveaux s'effectue jusqu'à obtention de fonctions applicatives terminales, entretenant des liens **exclusifs** avec des interfaces externes au SCC. (cf. contrainte d'exclusion marquée d'un astérisque sur le schéma ci-dessous)

Ces interfaces sont de surcroît d'un seul type, c'est-à-dire:

- soit des interfaces entre le SCC et les opérateurs de conduite (assimilées à des éléments d'interface homme-machine),
- soit des interfaces entre le système et sa partie opérative (mélange de signaux liés à des capteurs, actionneurs ou machines spécifiques),
- soit des interfaces avec d'autres systèmes.

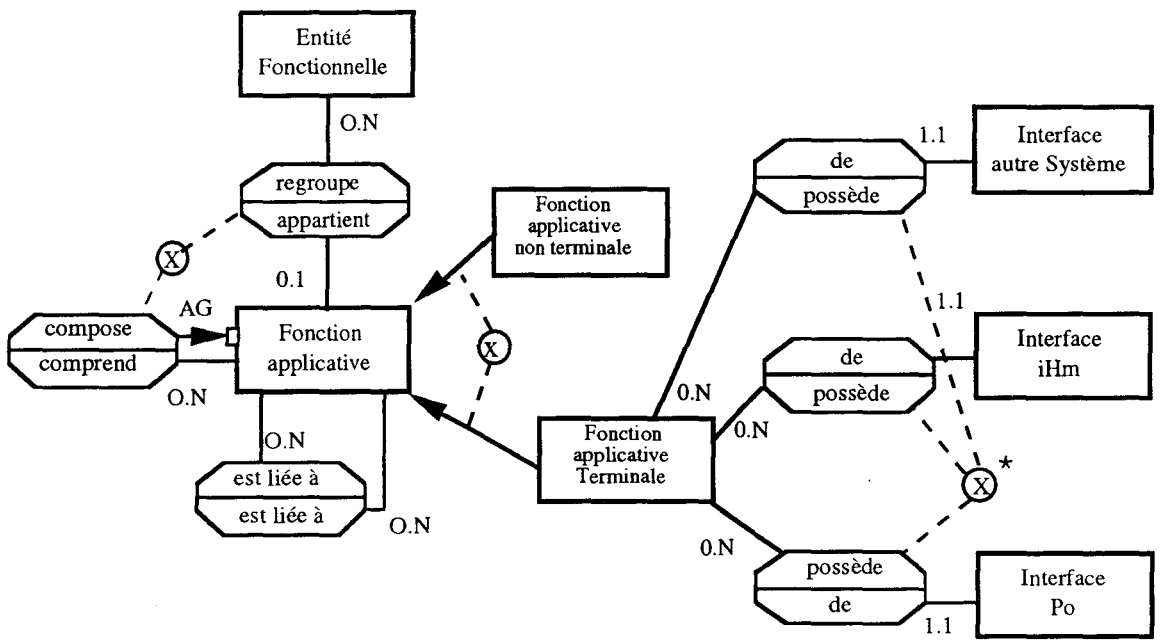


figure 55: Schéma conceptuel du modèle fonctionnel

On reconnaît une fonction terminale au fait que son rôle s'attache à un organe particulier de la PO ou à une série d'objets physiques commandables servant la même fonctionnalité de base.

Exemples de fonctions terminales : "vidanger cuves 1, 2 et 3", "charger four"

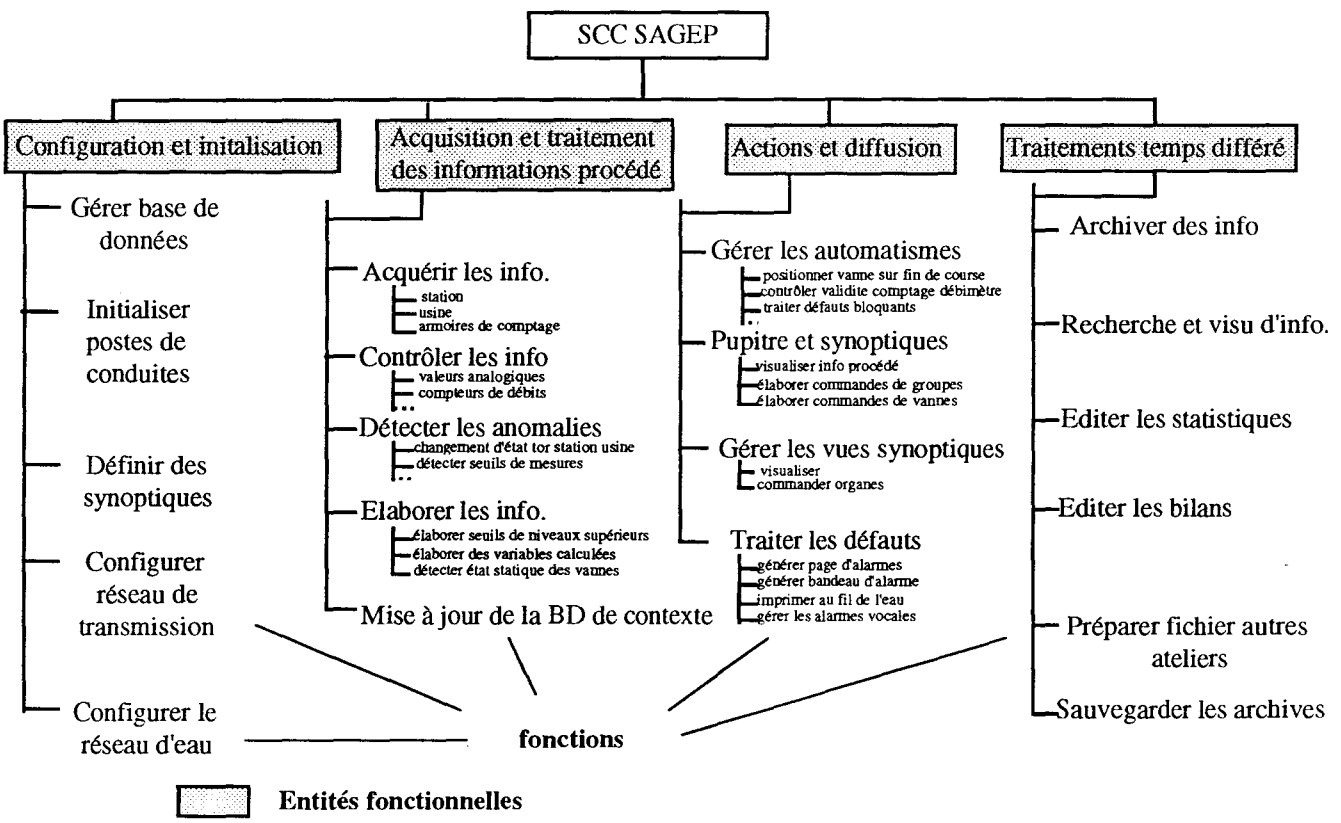
- "vidanger cuves 1, 2 et 3" s'adresse uniquement aux vannes de vidange des 3 cuves citées;
- "charger four" s'adresse à tous les objets physiques commandables d'un organe partie opérative de télé-manipulation.

IV.2.3 RÉSULTATS ATTENDUS DE LA SPÉCIFICATION

IV.2.3.1 ARBRE DE FONCTIONS

Les services rendus par le SCC peuvent être, une fois le critère d'arrêt atteint, synthétisés sous la forme d'un arbre de fonctions traduit de l'application de méthodes de spécification telles que SADT [IGL 89] [SAD 89], SA-RT [SAR 92] ou Gane et Sarson [GSA 82].

Voici par exemple l'arbre caractérisant les fonctions du système central de contrôle de production et de distribution des eaux de la ville de Paris, projet réalisé en 1990 par l'agence CEGELEC de l'île de France:



La figure ci-dessus ne représente que les deux premiers niveaux de décomposition du système pris en exemple. Elle ne fait donc pas apparaître la liste complète des fonctions terminales de ce système, qui se situe à un niveau de décomposition supérieur.

figure 56: Décomposition fonctionnelle partielle d'un système réel de contrôle-commande

IV.2.3.2 LISTES D'INTERFACES

Cette formalisation précise du besoin garde, par son regroupement en entités fonctionnelles, trace du point de vue initial du client.
Elle lie les fonctions applicatives terminales du système à ses interfaces externes qui représentent:

- soit des signaux composites relatifs à la partie opérative,
- soit des sous-listes organisées d'interfaces avec la conduite du procédé,
- soit des informations à échanger avec d'autres systèmes.

Exemple de signal composite d'interface avec la partie opérative:

Une interface de type "débit" est, dans le système précédent, exploitée par les fonctions "Acquérir les info. procédé", "Contrôler les info." et "Gérer les automatismes".
Cette interface permet d'apprécier le débit d'eau observé dans chaque station locale d'approvisionnement d'un quartier de la ville.

Une instance de "débit" comprend les informations élémentaires suivantes:

- la mesure instantanée d'un débit (information analogique)
- le comptage de ce débit (en m^3, pulsé par une information tout ou rien)
- le sens de circulation de l'eau dans les canalisations (+ ou -)
- le signalement d'un défaut de fonctionnement (info tout ou rien)

L'organisation des interfaces de conduite suit généralement la structuration des vues présentées aux opérateurs en salle de commande.
Si ce mode de structuration n'est pas assez fin, les vues sont elles-mêmes décomposées en sous-vues associées au contrôle de fonctionnalités plus réduites du procédé.

Exemple de vue opérateur:

Dans le système illustrant nos propos, une vue générale permet de contrôler le débit de chaque site de production en eau potable et de connaître le potentiel à produire de ces sites en cas de lestage momentané d'une partie du réseau.

Cette vue comprend:

- le signalement de la production totale instantanée d'un site (somme des débits instantanés des stations de pompages en fonctionnement sur ce site),
- le signalement de la production totale mobilisable du site (débits potentiels des sous-installations en état de marche)

Exemple d'interface avec un autre système:

Une interface "Minitel" permet, dans le même système, à tout opérateur éloigné de la salle de commande, d'obtenir par voie téléphonique et via un modem, des informations utiles à la maintenance du SCC:

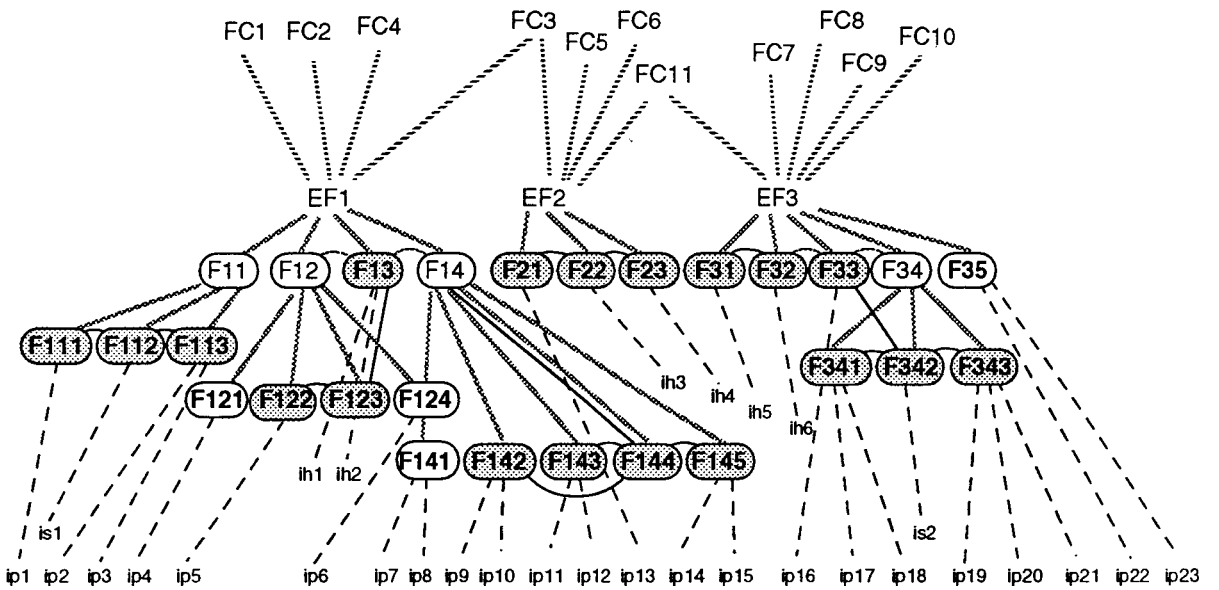
- l'historique des pannes matérielles observées en cours d'exploitation,
- l'état respectif des organes de chaque site de production ou de distribution d'eau.

Un signal composite ne regroupe pas, à priori, de signaux identifiés dans un lieu unique sur le site d'installation.

La liste détaillée des signaux élémentaires d'interface procédé est, quant à elle, accordée à la liste d'instruments retenue pour le système dès l'achèvement de la phase de conception de la partie opérative. (cf. figure 2)

Le critère d'arrêt qui consiste à considérer une fonction terminale dès lors qu'elle n'est caractérisée que par un seul type d'interface est applicable aux trois méthodes de spécification citées en référence (SADT, SA/RT, Gane et Sarson). Il témoigne de lui-même du bien fondé de la démarche puis qu'il oblige au raffinement des fonctions tant que les dépendances entre fonctions et interfaces restent floues à l'esprit du spécifieur, donc s'avèrent mal identifiées.

La figure 57 fournit un éclaté des liens -fonctions clients- / -fonctions constructeur- / -Interfaces externes tracés pour une application-type, qui, afin de simplifier la démonstration, ne considère qu'une arborescence à deux niveaux de fonctions et une liste réduite d'interfaces externes:



légende:

FCi: fonctions clients

Fi: fonctions constructeur terminales

Fj: fonctions constructeur de niveaux intermédiaires

ih: listes d'interfaces homme-machine

ip: listes d'interfaces partie opérative ou procédé

is: listes d'interfaces avec autres systèmes

EF: entités fonctionnelles

liens de composition — liens d'exploitation ——— interfaces fonctionnelles

figure 57: Arborescence de fonctions et d'interfaces d'une application

On distingue dans l'application ci-dessus des fonctions client pointant vers des entités fonctionnelles identifiés par le fournisseur.

Ces entités fonctionnelles regroupent des fonctions et sous-fonctions à priori réalisables par le constructeur et traduisant l'univers des FCi.

Par ailleurs, chaque liste d'interfaces ih, ip ou is de cette application-type est attachée à une et une seule fonction terminale de l'arborescence. Cette fonction est notée en gras sur la figure.

Les fonctions sujettes à interfaces fonctionnelles sont, pour leur part, présentées sur fond gris.

Une application industrielle moyenne fait généralement apparaître 4 à 5 niveaux de décomposition des fonctions et un nombre très important de listes d'interfaces.
Pour donner un ordre d'idée, le système présenté figure 56 compte pas moins de 75 listes d'interfaces procédé par site de production, le SAP complet devant par ailleurs gérer 9 sites de ce type.

Ne pouvant objectivement pas prétendre à l'exposé de la démarche proposée en IV.3 et IV.4 sur des exemples de cette taille, nous leur préférerons le support du projet-type de la figure 57.

IV.2.4 ADÉQUATION DES FORMALISMES AU BESOIN DE LA SPÉCIFICATION

Le choix d'un formalisme de spécification est une affaire de goûts et d'habitudes. Tous se valent à nos yeux dès l'instant qu'ils permettent de décrire la décomposition fonctionnelle de l'application et adoptent une représentation de type "flots de données" du système.

En SADT, les interfaces entre fonctions sont de plusieurs types.

Elles marquent:

- soit le contrôle d'une fonction sur une autre,
- soit des dépendances de ressources entre fonctions,
- soit des flots de données d'entrées ou de sorties.

Cette typologie n'est pas proposée en SA/RT ou dans des modèles équivalents reprenant le formalisme propre à "Gane et Sarson". Nous verrons en particulier au chapitre V que cette option n'est en définitive pas retenue dans la méthode proposée pour concevoir l'aspect logiciel de l'application. Elle n'est donc pas primordiale, tout comme celle qui considère des entités de stockage de données dans le système (représentées par exemple en SA/RT par deux traits horizontaux). Il paraît en effet prématuré de préciser ces entités alors que l'organisation du SCC n'est pas fixée et que ses fonctions sont encore très abstraites.

IV.2.5 CONCLUSION PARTIELLE

L'étape de spécification système entre dans la suite logique des activités de pré-étude d'un SCC. Elle est essentielle dans le sens où elle permet au fournisseur de s'accorder le temps de réflexion nécessaire à la mise en forme et l'expression détaillée des fonctions répondant au besoin.

Elle permet, durant son déroulement, de provoquer un dialogue fructueux entre le client et le fournisseur, dialogue qui aboutit normalement à la rédaction d'un dossier détaillé de spécification qui est visé, tel un contrat.

Ce dossier répond par ailleurs aux normes qualités en vigueur pour cette phase [ISO-1] [ISO-2] . Chacun s'accorde donc à y insérer ses expectatives propres aux caractéristiques techniques du SCC, en respectant la procédure, les recommandations, les guides d'utilisation (des méthodes SA/RT et SADT) et les règles de rédaction de la méthodologie MODAL v1 et v2.

Est obtenue en final un modèle formel du besoin, gage de réussite de la phase de conception qui va suivre.

Après avoir très succinctement présenté la forme à donner au résultat de l'étape de spécification système, nous allons maintenant montrer comment affiner la couche automatisme de l'architecture de principe en faisant correspondre ces éléments aux contraintes liées à la topographie du site d'installation.

Nous nous appuierons, pour ce faire, sur les concepts définis au chapitre II, notamment les notions d'ensembles topo-fonctionnels, d'équipement, de segments de réseaux et unités logiques de traitements applicatifs pour construire le modèle logique de l'architecture du SCC.

IV.3. CONSTRUCTION DU MODÈLE LOGIQUE

IV.3.1. TYPAGE DES FONCTIONS ÉLÉMENTAIRES DU SYSTÈME

Le schéma de décomposition proposé au paragraphe précédent exclut de relier une fonction terminale à des interfaces externes de natures différentes. Il révèle de ce fait quatre types de fonctions élémentaires assurées par le SCC:

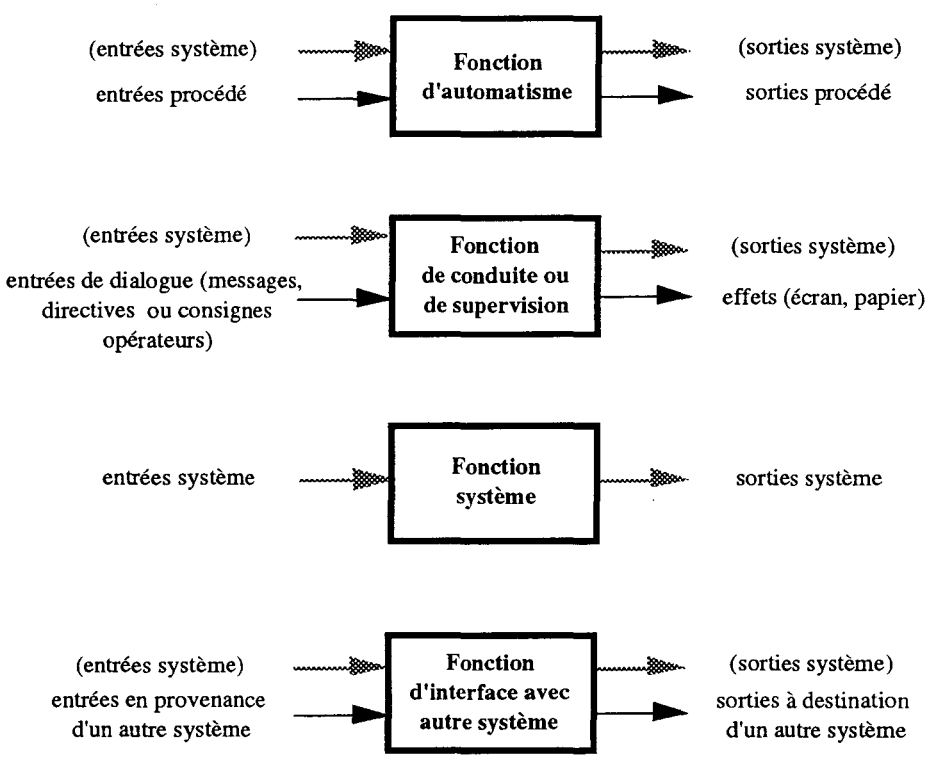


figure 58: types de fonctions applicatives terminales et types d'interfaces associées

- Les fonctions d'automatisme assurent la régulation du procédé industriel en élaborant les signaux de commande transmis aux organes de la partie opérative.
- Les fonctions de conduite et de supervision assurent l'interface entre le système de production et les opérateurs de conduite de l'installation.
- Les fonctions d'interface prennent en charge les interactions du SCC avec d'autres systèmes extérieurs.
- Les fonctions système assurent la cohésion du SCC en mettant à disposition des trois types de fonctions précédentes des ressources propres au procédé.

- Les entrées/sorties procédé sont déduites des listes d'interface ip du modèle de spécification,

- Les entrées de dialogues et effets visibles au niveau des postes opérateurs sont issues des listes ih,

- Les entrée/sorties avec d'autres systèmes traduisent les interfaces is de la figure 57,

- Les entrée/sorties dites "systèmes" représentent, quant à elles, des signaux matérialisant les interfaces fonctionnelles du SCC.

Notre propos étant d'identifier et d'organiser les équipements des niveaux 0 et 1 du modèle CIM, nous ne retiendrons dans la suite que les fonctions d'automatisme, d'interface et leurs fonctions système directement associées.

IV.3.2 MODÉLISATION DE L'ARCHITECTURE SITE

Nous supposerons par ailleurs la topographie du site d'installation du SAP traduite en une succession de lieux distincts permettant d'accueillir les équipements matériels de contrôle et de commande du procédé.

Ces lieux seront, dans le cas de systèmes très étendus, regroupés dans des zones de localisation.

Ils seront à l'origine des ensembles topo-fonctionnels du système (cf. figure 13 chapitre II)

La démarche proposée ci-après va tout d'abord consister à rapprocher les trois types de fonctions applicatives précitées des lieux désignés sur le site pour former des ensembles topo-fonctionnels cohérents en exploitant les liens qui existent entre ces fonctions et leurs interfaces.

Elle se poursuivra par l'identification des machines virtuelles de traitements applicatifs de niveau 1 du SCC.

Ces machines seront structurées en cellules appliquées à satisfaire les besoins de l'automatisation d'un ensemble cohérent d'organes de la partie opérative situés dans ou à proximité d'un ETF.

IV.3.3 MODE DE STRUCTURATION DES ÉQUIPEMENTS D'AUTOMATISME

IV.3.3.1 POURQUOI DES CELLULES D'AUTOMATISME ?

La mise en œuvre d'un procédé industriel peut faire intervenir un ensemble physiquement très étendu de machines. Son automatisation, ne peut, dans ce cas, se résoudre à la réalisation d'un seul "automate" centralisant tous les traitements nécessaires et possédant en plus la capacité de supporter quelques dizaines de milliers d'entrées-sorties.

La solution passe donc par la mise en œuvre d'un système réparti de contrôle-commande dont chaque sous-ensemble, qualifié de **cellule d'automatisme**, à la charge de gérer une partition distincte du procédé.

Comme nous le verrons un peu plus loin, les partitions retenus à cet effet relèvent d'organes de la partie opérative à priori situés à proximité les uns des autres.

Les cellules d'automatisme s'appliquent en outre à ne pas dépasser les limites imposées par la technologie, en particulier évitent certaines incompatibilités liés aux caractéristiques dynamiques des traitements de l'application.

Elle tiennent également compte des capacités nécessairement limitées des machines ou réseaux de communication proposés dans une famille donnée de constituants matériels.

Exemple d'incompatibilité liée à des caractéristiques dynamiques de traitements au sein d'une cellule:

Le fait de prétendre implanter trois tâches cycliques de régulation ayant respectivement les périodes 3, 10 et 25 ms sur une machine de traitement mono-UT oblige cette dernière à opérer à une cadence élémentaire de 1000 cycles à la seconde, ce que peu d'automates du marché sont capables de réaliser à l'heure actuelle sur un nombre important d'instructions.

La solution ne semble pas être dans cette voie.

Il paraît en effet plus raisonnable d'implanter ces tâches soit sur deux machines distinctes, soit sur une machine bi-unité de traitement doublant ainsi la capacité dynamique d'exécution de l'automate. Le choix de la configuration la plus appropriée serait, dans ce cas, conditionné par la masse d'informations à échanger entre les tâches rendues matériellement distinctes.

⇨ Ainsi, on ne peut concevoir l'exécution des différentes fonctions d'automatisme du SCC sans décentraliser leur traitements au sein de cellules distinctes employant des machines dont les besoins en communication puissent alors être satisfaits par l'emploi de réseaux et de protocoles standards de dialogue.

Nous nous fixerons donc pour but de déterminer ces cellules en réservant l'emploi de segments de réseaux à des échanges d'informations distants et les bus de données internes des automates au convoi d'interfaces propres à des sous-ensembles d'exécutifs cohérents.

IV.3.3.2 Deux Types de Cellules

Fort de cette remarque, il nous est néanmoins toujours permis de construire deux types de cellules:

- des cellules dites "à traitements centralisés" pour lesquelles les équipements assurant l'interface avec la PO ne sont reliés qu'à une machine de traitement applicatif,

- des cellules dites "à traitements répartis" aux niveaux desquelles l'exécution des traitements applicatifs s'effectue sur plusieurs machines distinctes.

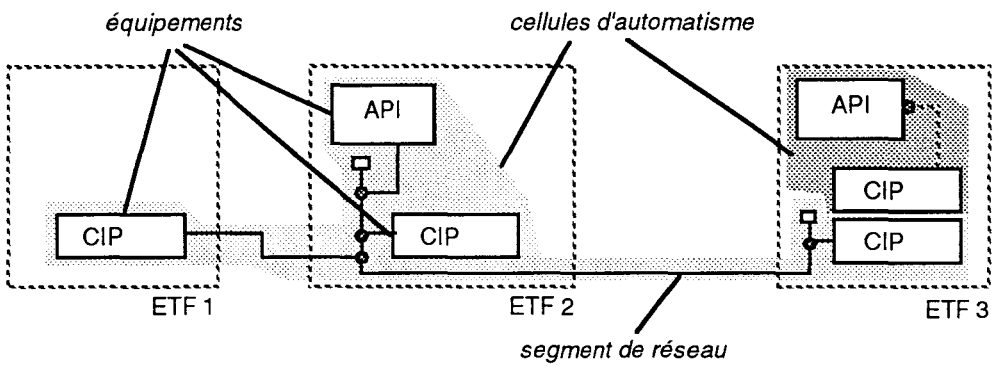


figure 59: exemple de cellules à traitements centralisés.

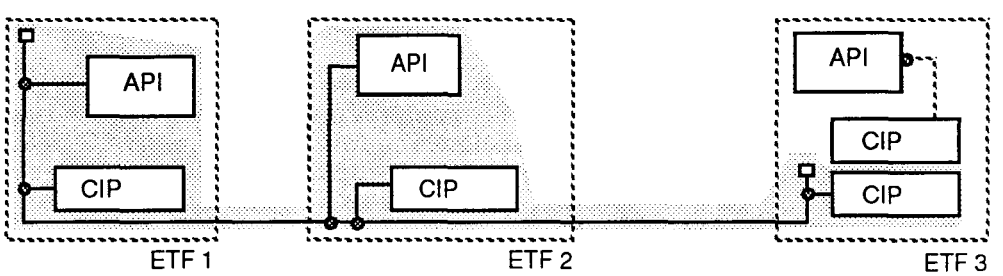


figure 60: exemple de cellule à traitements répartis

Choisir une structure à traitements répartis implique de solliciter plus fortement le segment de réseaux de communication qui cimente la cellule.

Ce choix, s'il est employé, doit être motivé par d'autres nécessités que celle de prétendre augmenter la disponibilité d'une telle structure, puisque contrairement aux idées reçues, il se démontre aisément que le fait de décentraliser les traitements d'une cellule à pour effet de la fragiliser dans son ensemble [BAZ 69].
Ce choix est par contre souvent jugé nécessaire pour assurer des traitements ponctuels de repli ou de sécurité dans le système.

En règle générale, la répartition des traitements sera guidée par la recherche d'une cohésion maximale et d'un couplage minimal entre les fonctions assurées par chacun des équipements d'automatisme. [MOD 89]

De ce fait, on préférera dans tous les cas commencer par évaluer la viabilité des solutions construites sur la base de cellules à traitements centralisés en regroupant au sein de celles-ci des fonctions en fortes dépendances fonctionnelle et/ou opérationnelle.

L'organisation fonctionnelle de l'application va, dans notre démarche, servir de référence à la formation des différentes cellules d'automatisme du SCC.

Nous déterminerons leur organisation en trois étapes:

- tout d'abord en identifiant les ensembles topo-fonctionnels du système,
- puis en tirant de ces ensembles topo-fonctionnel des machines virtuelles de traitement par intégration des contraintes portant sur les fonctions du modèle de spécification,
- enfin en matérialisant les machines virtuelles précédentes par des équipements logiques de traitements applicatifs auxquels seront associés des équipements d'interface avec le procédé.

IV.3.3. ETAPE 1: IDENTIFICATION DES ENSEMBLES TOPO-FONCTIONNELS DE L'APPLICATION

- Les interfaces procédé font référence à des capteurs et à des actionneurs localisés sur le site final d'installation du système
- Ces interfaces sont de plus liées aux fonctions applicatives terminales du SCC par des liens d'exploitation (cf. modèle présenté en IV.2.2.2).

Il nous est donc permis de délimiter la "zone d''influence" de chaque fonction applicative sur le site en relevant les unités de lieu dans lesquelles sont repérées ses interfaces.

Ainsi, en admettant par exemple que les interfaces entre la fonction F341 et la partie opérative du système décrit figure 57 sont localisées dans les locaux 2 et 3 de l'installation, F341 a alors comme zone d'influence ces 2 locaux particuliers:

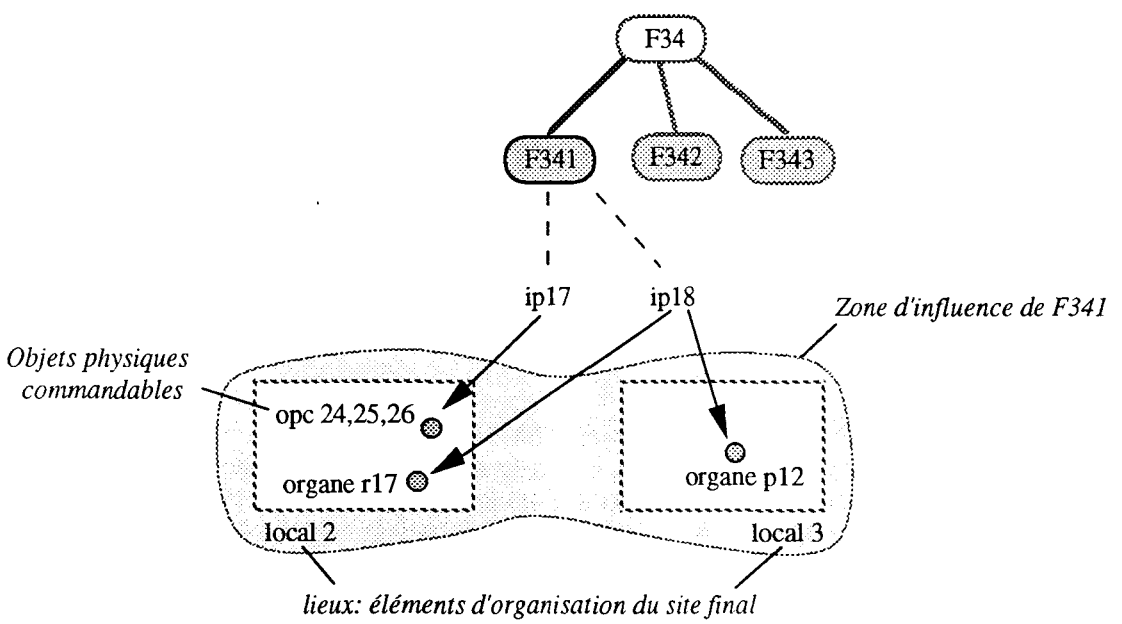


figure 61: Zone d'influence d'une fonction terminale

C'est en connaissant les zones d'influence des fonctions de la couche automatisme que le concepteur peut juger de la répartition des traitements de niveau 1 de l'application.

Il dispose, pour ce faire, du concept d'ensemble topo-fonctionnel, introduit en II.5, permettant d'associer, à toute unité de lieu de l'architecture site, les fonctions réalisées par les équipements implantés en ces lieux.

La détermination des différents ensembles topo-fonctionnels (ETF) constituera la première étape du raisonnement qui mène à la construction des cellules d'automatisme. Elle suppose bien évidemment de disposer de listes stables de fonctions et d'interfaces, sans quoi, le raisonnement risquerait d'être très vite invalidé.

Cette procédure, appliquée à l'application-type présenté figure 56, donne, en admettant les localisations suivantes des listes d'interfaces du système, les 3 ETF ci-dessous:

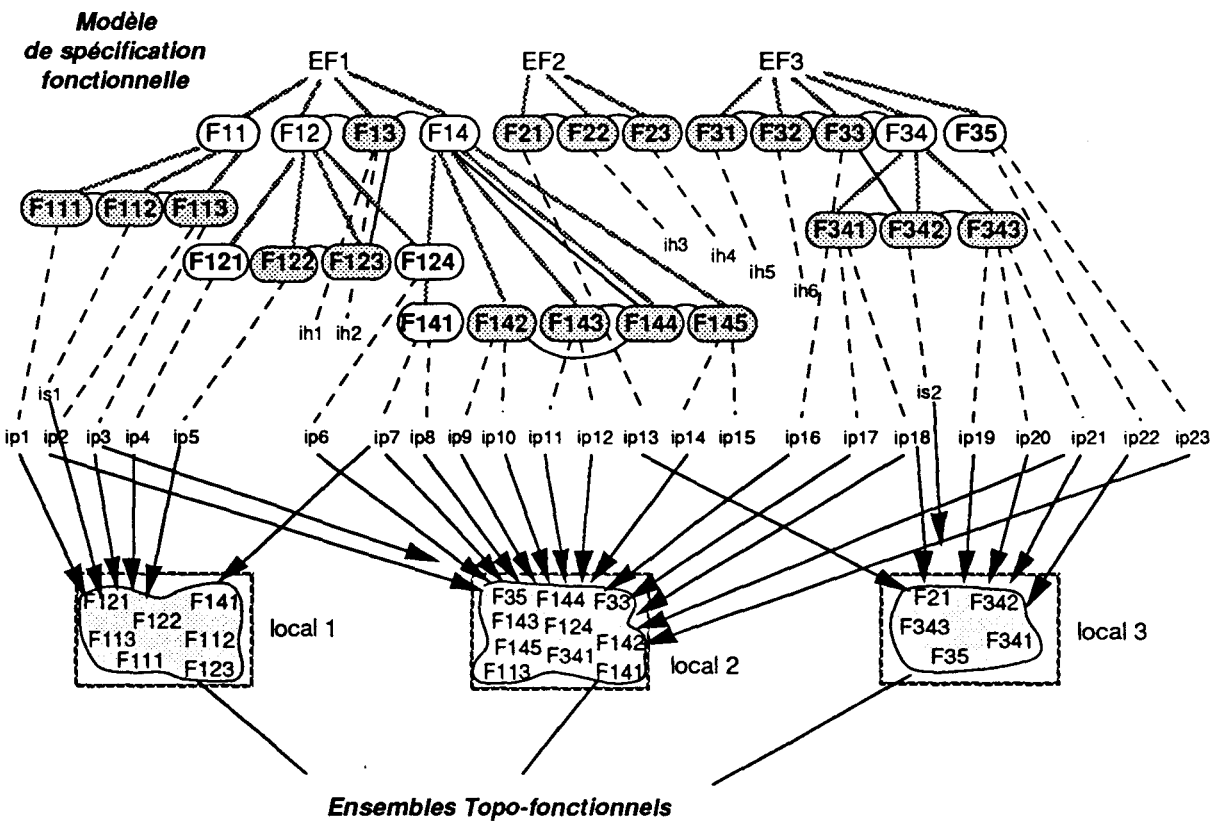


figure 62: Ensembles topo-fonctionnels déduits de la localisation des interfaces externes ip_i et is_j.

Ces trois ETF sont, comme dans la plupart des cas, connexes;

En effet, comme précisé figure 61, la fonction F341 est partagée par la nécessité de devoir être assurée à proximité de son organe de rattachement, situé au local 2, et de l'organe p12 situé, lui, dans le local 3. Ce phénomène est également observé sur les fonctions F113, F141 et F35.

Etant donné qu'il se dégage un point de vue fonctionnel dans le modèle de spécification système, il y aura de ce fait un certain nombre de fonctions pour lesquelles le concepteur aura à trancher, c'est-à-dire à préférer une implantation dans un local plutôt qu'un autre. Il ne faudra oublier que l'interface forcément mise à l'écart par ce choix doit user d'un autre moyen pour être représentée; utiliser un mode de communication distant semble être alors la voie la plus appropriée pour répondre, dans ce cas, à cette attente.

IV.3.4 ETAPE 2: DÉTERMINATION DES MACHINES VIRTUELLES DE TRAITEMENT

Les regroupements effectués dans l'étape précédente font potentiellement apparaître des cellules d'automatisme (une cellule par local de l'architecture site).
En vertu des propos tenus aux paragraphes IV.3.3.1 et IV.3.3.2, nous examinerons tout d'abord la possibilité de construire de ces cellules à traitements centralisés.

Pour ce faire, nous associerons, dans un premier temps, une machine virtuelle de traitement à chaque ensemble topo-fonctionnel caractérisant le sous-système d'automatisme. Cette machine aura la charge d'assurer les fonctions à opérer dans chacun des lieux ayant donnés ces ETF.

Les fonctions de niveau 1 seront, dans un deuxième temps, allouées aux machines virtuelles en respectant les contraintes émanant de la localisation de leurs interfaces et en faisant en sorte que soit respecté le principe rappelé en IV.3.3.2 de la plus forte cohésion et du plus faible couplage entre ensembles se dégageant d'une telle projection. Seront considérées, en cas de multiples affectations potentielles, par ordre de priorité, les interfaces internes au SCC, puis les informations échangées entre le système et sa partie opérative et celles évoquant des dialogues avec d'autres systèmes spécifiques.

La procédure, appliquée au système de la page précédente, se déroule comme suit:

- dans une premier temps, sont allouées aux machines virtuelles de traitement représentatives des trois ETF de niveau 1 du système, les fonctions terminales parfaitement circonscrites dans la zone couverte par chacune de ces machines:

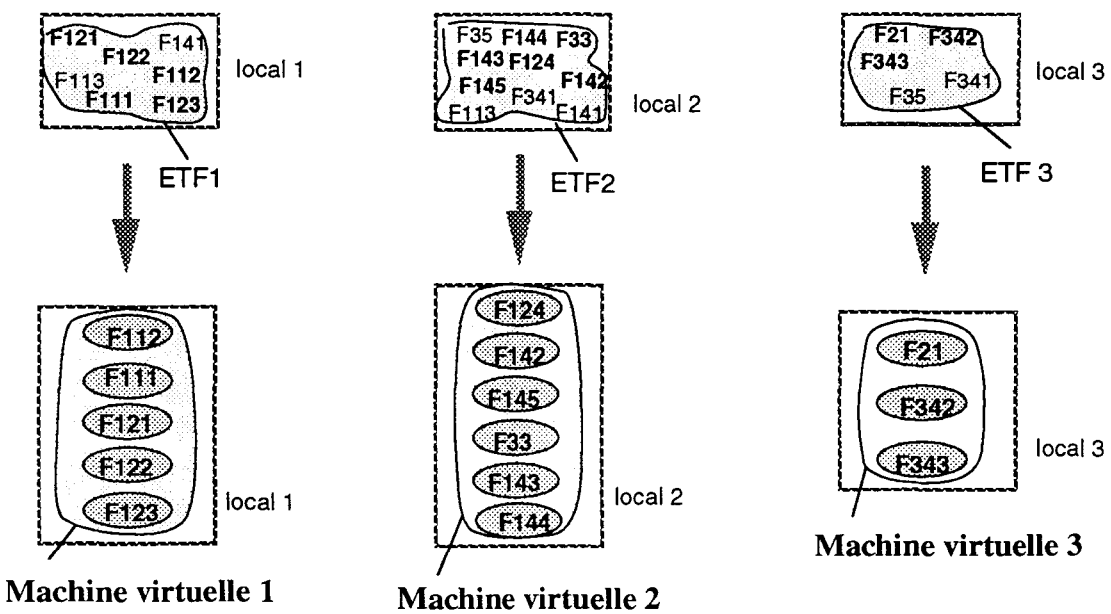


figure 63: Premier affinement des machines virtuelles de traitement

- dans un deuxième temps, sont évalués les liens entres les fonctions restant à attribuer et les différentes interfaces du système, en appliquant la règle de priorité proposée plus haut.

  . Ainsi, F113 étant liée par une entrée ou une sortie dite "système" à la fonction F112, se doit d'intégrer la machine virtuelle du local 1;

. F341 relevant d'un cas semblable, rejoint les fonctions de la machine MV3;

. F141 et F35 n'étant le siège d'aucune interface interne au système de contrôle-commande, elles sont allouées aux machines virtuelles assurant les traitements les plus fortement liés au sous-procédé primant dans l'une ou l'autre des unités de lieu sur lesquelles peut porter le choix du concepteur (cf. flèches d'allocation présentées en grisé sur la figure ci-dessous, témoignant de choix à priori arbitraires).

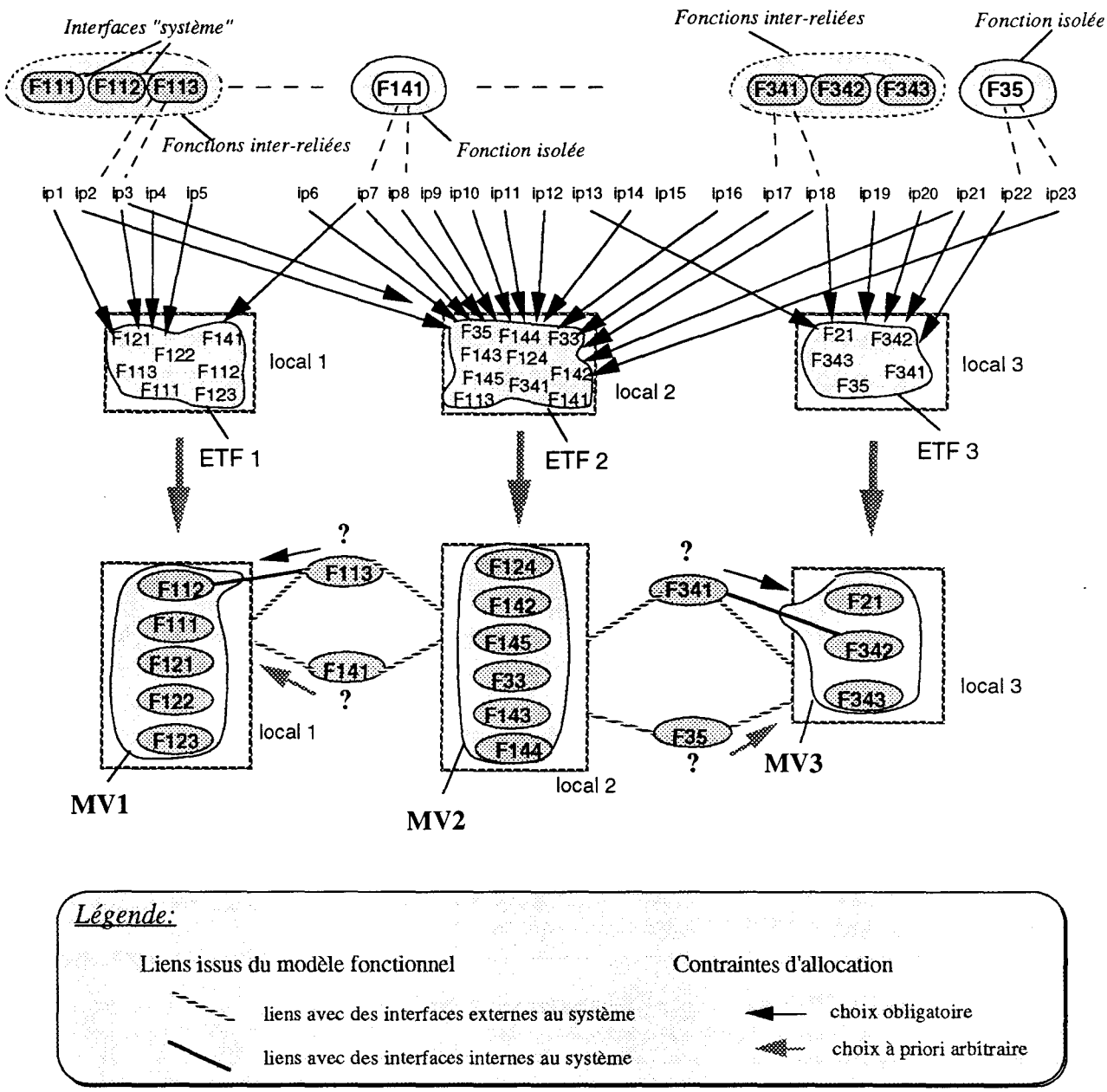


figure 64: Machines virtuelles de traitement

Trois facteurs rentrent donc en ligne de compte pour affecter les fonctions applicatives à des machines virtuelles:

- cohésion maximale d'un ensemble projeté,
- couplage minimal entre de tels ensembles,
- et si couplage il y a, optimisation des moyens de communication à mettre en œuvre.

Le plus important de ces facteurs est sans nul doute celui qui garantit la plus forte cohésion des fonctions affectées aux différentes machines de traitement de l'application. Cet aspect doit donc toujours être examiné avant les deux autres.

La procédure d'affectation ne présente en pratique que peu de fonctions pour lesquelles peuvent être opérés des choix de façon indifférente puisqu'une fonction terminale est, en règle générale, attachée à un organe particulier de la partie opérative, situé lui-même dans un lieu précis de l'installation.
Le modèle formel de spécification n'est, sur ce point, pas aussi abstrait qu'il n'y paraît. Il traduit l'organisation des fonctions assurant l'automatisation d'une partition distincte et bien localisée du procédé.

IV.3.5 ETAPE 3: CHOIX DES ÉQUIPEMENTS LOGIQUES DE TRAITEMENT ET D'INTERFACE DU SCC

Une fois les machines virtuelles de traitement déterminées, peuvent alors être définis les équipements logiques de traitement et d'interface des cellules d'automatisme:

Ce choix tient compte:

- des types de traitements à effectuer dans chaque cellule (traitements de régulation, traitements séquentiels, traitements spécifiques de calculs, traitements combinatoires etc ...)
  ➡ A ces types de traitements correspondent des équipements-types particuliers

- de la puissance de ces traitements, généralement en rapport avec le nombre d'interfaces partie opératives de la cellule
  ➡ A ce niveau pointent des modèles d'équipements d'automatisme

- de leurs caractéristiques dynamiques (période de rafraîchissement des informations procédé, performances attendues de certaines fonctions réflexes ..)
  ➡ Cet aspect conditionne en particulier le fait de devoir disposer de structures à traitements centralisés ou répartis (cf. IV.3.3.2)

- de leur niveau de disponibilité (si la commande tolère ou non un arrêt momentané de la cellule de production pilotée par la cellule d'automatisme)
  ➡ De ce point dépend l'architecture interne des équipements

- des types d'interfaces de la cellule (interfaces banalisées ou bien spécifiques)
  ➡ A des interfaces banalisées correspondent des contrôleurs "généraux" d'interface procédé, à des interfaces spécifiques correspondent des équipements spécialisés de terrain (variateurs de vitesse par exemple)

- enfin, de la distance séparant un équipement de traitement de l'endroit où sont effectivement localisées ses interfaces procédé (au sein d'un même lieu sur le site peuvent être observées des distances importantes entre les équipements, à cause par exemple des nuisance dues à l'environnement de production)
  ➡ Cet aspect conditionne également la structuration logique d'une cellule.

Nous retiendrons, des six points précédents, trois critères principaux permettant progressivement de définir l'architecture logique d'une cellule d'automatisme, en déterminant tout d'abord ses équipements-types de traitement et d'interface, puis, en définissant plus précisément les modèles des équipements de cette cellule et enfin, en fixant l'architecture interne de ces équipements.

Nous considérerons, pour ce faire, tour à tour:

- la nature des traitements opérés dans la cellule,
- son étendue physique,
- l'importance accordée ou non aux contraintes d'environnement et de disponibilité lors de la spécification du système.

IV.3.5.1 NATURE DES TRAITEMENTS OPÉRÉS AU SEIN D'UNE CELLULE

De la nature des traitements suscités par les fonctions allouées à chacune des machines virtuelles de la cellule dépend le choix d'équipements-types particuliers (cf. définition proposée en II2.1).

On choisira par exemple un "automate régulateur" pour assurer des fonctions avancées d'asservissement, un "automate programmable industriel" pour opérer des traitements de contrôles séquentiels, un "microcalculateur" lorsque sera exigé, à ce niveau de commande, l'intervention d'une machine de calcul performante ou l'exécution de logiciels dits "temps réels" etc ...

Les compétences "affichées" de ces équipements-types sont donc, de cette manière, accordées aux éléments du besoin issus de la spécification système.

Les machines résultant de l'étape 2 sont, de leur côté, forts d'une cohésion "par grappes" (grappes de fonctions inter-reliées (cf. figure 64)) qui, au delà de ces sous-ensembles ne font que rassembler les fonctions attachées à des organes situés à proximité les uns des autres.
Il ne faut donc pas s'attendre à ce qu'émane de l'analyse de ces fonctions un seul et même équipement-type.
Dans certains cas, la diversité des traitements d'une machine virtuelle nécessitera, en regard de la gamme d'équipements pressentie à ce choix, l'emploi de plusieurs machines de traitement.
Dans d'autres cas, pourra être retenue une solution intermédiaire, intégrant dans un même équipement, plusieurs unités de traitement distinctes.

Résulte donc, d'une première phase d'analyse, des fonctions attribuées à une machine virtuelle, des équipements-types de traitement. Voici par exemple ce que pourrait donner cette phase pour MV1, présentée figure 64:

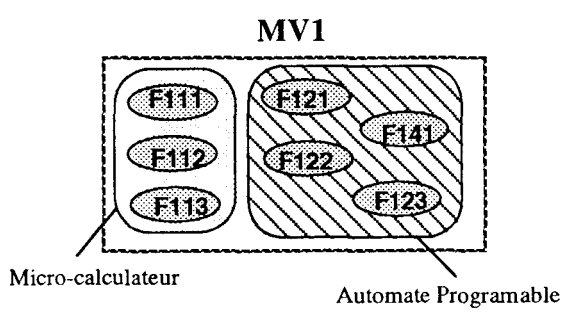


figure 65: Partitionnement des fonctions d'une machine virtuelle

IV.3.5.2 ETENDUE PHYSIQUE DE LA CELLULE

De l'étendue physique d'une cellule dépend ensuite sa structuration logique, faite d'équipements logiques de traitement et d'interface.

La structure d'une cellule varie en effet en fonction de la répartition des organes de la partie opérative gérés par cette dernière. Cette répartition contribue:

- soit à confondre les équipements de traitement avec leurs équipements d'interface,
- soit au contraire, en cas de répartition plus prononcée, à séparer les machines de traitement des machines chargées des échanges avec le procédé.

Dans le deuxième cas, apparaît un certain nombre d'équipements nouveaux au sein de la cellule, dans lesquels sont regroupées les interfaces partie opérative situées à un endroit précis d'un local de l'architecture site. (endroit entaché d'un repère particulier)

Certaines fonctions, projetées sur les machines virtuelles au moment de l'étape 2, détiennent encore des liens avec des objets physiques commandables placés dans d'autres locaux.
Ces liens emprunteront nécessairement la voie de segments de réseaux pour être effectifs et révéleront de nouveaux équipements logiques d'interface, comme le "CE2000", placé au local 2 dans la figure ci-dessous.

Structuration logique de la cellule liée à la machine "MV1":

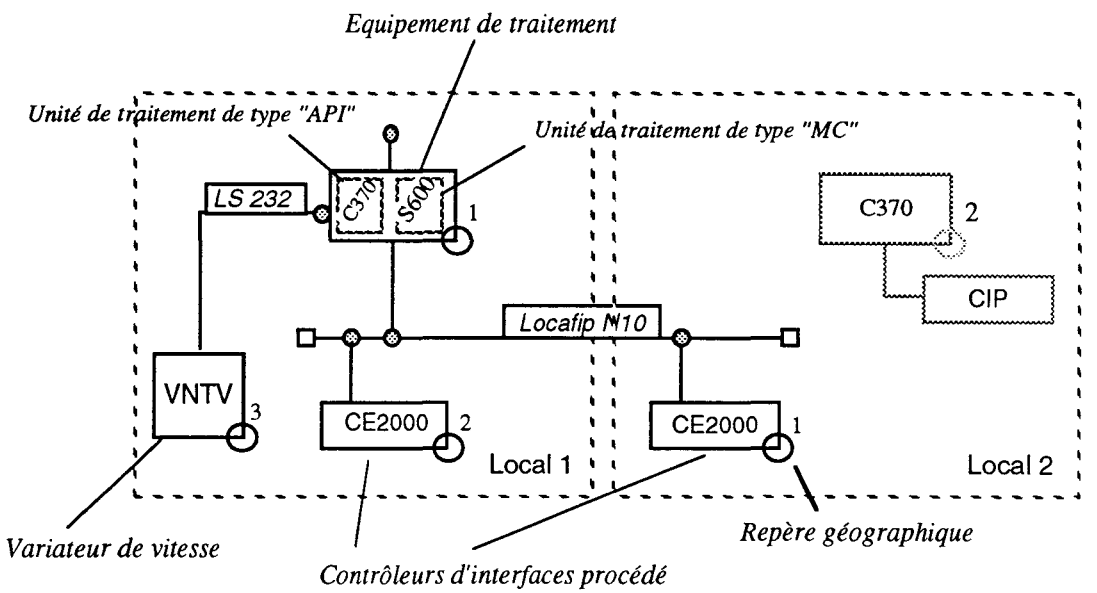


figure 66: Organisation logique d'une cellule

L'exemple ci-dessus accorde à chaque équipement de la cellule liée à MV1 un repère site précis.

Il regroupe les deux équipements-types initiaux (Microcalculateur et Automate Programmable) en un seul équipement logique, dans sa configuration bi-unités de traitement.

Le **modèle de segments de réseau** "locafip N10" vient enfin cimenter la cellule formée des équipements précédents.

On peut espérer dans l'avenir, ne plus avoir à regrouper de façon si systématique les interfaces avec la partie opérative dans des contrôleurs d'interface procédé puisqu'apparaissent depuis peu sur le marché des capteurs et actionneurs intelligents directement connectés aux réseaux locaux.

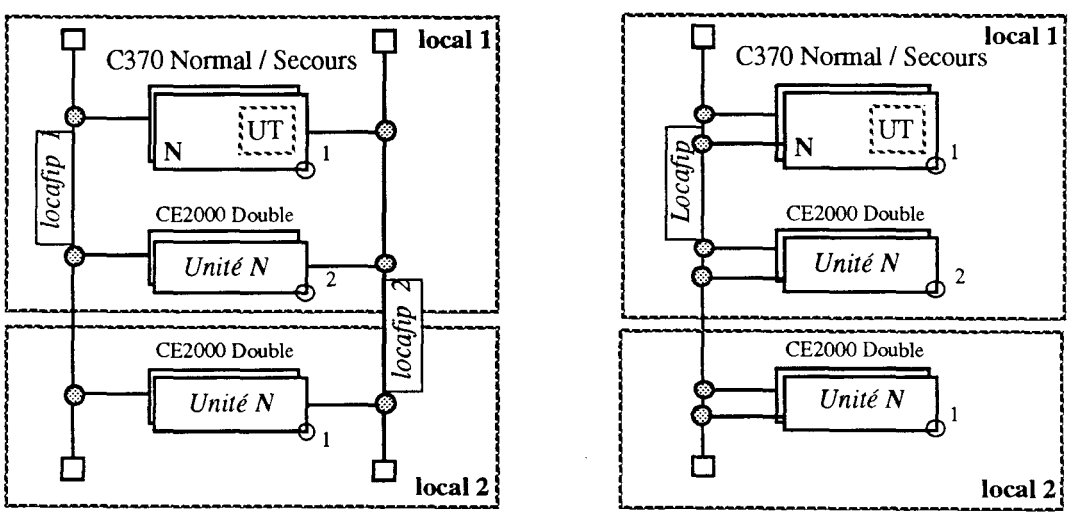
Le prix de revient d'une connexion à un signal composite est actuellement encore beaucoup trop élevé pour être appliqué à l'échelle d'un vaste système de contrôle-commande. Il saurait être plus abordable dans les années qui viennent. Ce procédé aura l'avantage d'être nettement plus souple que le précédent et de faire l'économie d'une bonne partie du câblage servant à raccorder le SCC à sa partie opérative. Il aura l'inconvénient de disperser les ressources de contrôle-commande un peu partout sur le site et de compliquer la vie aux mainteneurs et aux metteurs en service.

IV.3.5.3 EXIGENCES DE DISPONIBILITÉ ET CONTRAINTES D'ENVIRONNEMENT

Du respect des exigences de disponibilité et des contraintes dues à un environnement plus ou moins sévère dépend, là encore, la structuration de la cellule.

Ces exigences ne viennent généralement pas compromettre la définition des équipements logiques de cette cellule. Elles ne contribuent, tout au moins, qu'à déterminer l'architecture interne des ces équipements, en leur assignant un niveau de redondance donné et un mode de commutation en cas d'indisponibilité passagère.
Ainsi, certains équipements accepteront de voir leurs unités de traitement doublées, d'autres sauront même être triplés (c'est le cas des automates régulateur C500). Les segments de réseaux pourront également être disposés en double, de façon à pallier à une panne due au sectionnement malencontreux d'un câble ou à une perturbation prolongée d'une ligne.

Un certain nombre de configurations redondantes sont proposées par gamme de produits. Les plus utilisées sont celles qui, pour répondre à un besoin de disponibilité maximales, consistent à doubler les unités de traitement des équipements de traitement et d'interface et à doubler les réseaux en multipliant les voies d'accès à ces derniers.
Nous verrons un peu plus loin comment s'opère précisément ce dédoublement au moment de la configuration matérielle détaillée des équipements d'une cellule.
Dans sa représentation logique, une cellule en structure redondante prend, en fonction du mode de redondance retenu, l'une ou l'autre des 2 formes types suivantes:



Redondance double, "Réseaux et Equipements"

Redondance double, "Equipements"

figure 67: Représentation logique d'une cellule redondante

IV.4. CONSTRUCTION DU MODÈLE PHYSIQUE

L'étape précédente ayant permis de déterminer la structure logique des cellules d'automatisme, il nous faut maintenant construire le modèle physique correspondant à cette couche du SCC.

Nous considérerons pour ce faire à nouveau 3 étapes:

- une première étape, consistant tout d'abord à définir la configuration matérielle détaillée des équipements de ces cellules,
- une seconde étape, s'accordant ensuite à alimenter les composants issus de cette configuration,
- une troisième étape, permettant enfin de placer les éléments matériels obtenus dans des structures mécaniques d'accueil.

IV.4.1 ETAPE 1: CONFIGURATION MATÉRIELLE DES ÉQUIPEMENTS

Configurer chacun des équipements de l'architecture logique nécessite de se polariser sur plusieurs aspects distincts pouvant eux-mêmes être traités en ordre séquentiel;

En effet, nous pouvons dans un premier temps matérialiser les interfaces procédé de ces équipements en faisant état d'une liste de modules d'entrées-sorties attachés aux repères-site affectant chaque élément du modèle logique.

Il nous est ensuite permis de regrouper ces modules dans des bacs afin de juger de l'organisation matérielle de chaque équipement et déterminer ainsi les emplacements licites pour leur(s) unité(s) de traitement, leurs cartes d'extension et leurs modules "tête de bac".

Dans un troisième temps, peuvent être enfin choisis et ranger les modules chargés des communications internes au SCC (échanges réseaux par exemple) et du dialogue avec d'autres systèmes spécifiques.

Sont obtenus en final une liste de bacs configurés nécessitant d'être alimentés et placés dans des armoires ou des coffrets de protection.

IV.4.1.1 CHOIX DES MODULES PÉRIPHÉRIQUES D'INTERFACE PROCÉDÉ

Ce choix s'opère sur inventaire des signaux élémentaires contenus dans chacune des listes d'interfaces attachées aux équipements de l'architecture.
Il tient compte des types de signaux échangés avec la partie opérative, de leur plage de variation, de leur dynamique .. de façon générale, de leurs caractéristiques propres, pour choisir des modules électroniques sur lesquels affecter ces interfaces.

Les signaux sont affectés par lots aux voies des modules d'interface (lots de 8 pour des interfaces de type analogique par exemple, lots de 16 pour des entrées-sorties Tout ou Rien, lots de 4 pour des voie de communication multiplexée etc ...)

Les modules sélectionnés s'accordent bien évidemment aux modèles d'équipements retenus dans l'étape précédente. Ils sont pour cela choisis parmi les constituants matériels d'une même famille de produits *, attachés aux ensembles de configuration liés à chacun de ces modèles (cf. II.4.2).

(*) DEUX FAMILLES DE CONSTITUANTS SONT BRIÈVEMENT LISTÉES EN ANNEXE 2 AU POINT A2.3

IV.4.1.2 REGROUPEMENTS DES MODULES DANS DES STRUCTURES D'ACCUEIL ET POURSUITE DU PROCESSUS DE CONFIGURATION

Sont indirectement proposés, derrière chaque ensemble de configuration précédent, des modèles de bacs qui, pour chaque mode d'agencement retenu (chaque utilisation-type du bac) limite le nombre d'emplacements disponibles pour des modules d'interface.

Ainsi, un automate "C370" dans sa configuration "bi-unité de traitement" ne dispose, dans son bac UT, au maximum que de 8 emplacements pour modules "périphériques". Etendre cette configuration "de base" réclame ensuite d'introduire une nouvelle structure d'accueil de type "fond de panier".
Du nombre de modules ayant le même repère-site que cette machine de traitement dépendra donc le nombre de bacs matérialisant effectivement l'équipement.

Ce type de problème ne se pose pas dans le cas des contrôleurs d'interfaces autonomes puisque ces équipements particuliers n'ont pas la possibilité d'être étendus de la sorte.
Ainsi, si la capacité d'accueil d'un tel contrôleur ne suffit pas à matérialiser toutes les interfaces affectées un équipement placé à un repère site donné, il peut s'en suivre une modification sensible de l'architecture de principe du SCC contribuant à ajouter un nouvel équipement de ce type dans l'organisation initiale.

... Les modules d'interface retenus en premier lieu sont donc disposés dans des bacs aux emplacements requis par leur fonction, en respectant les contraintes signalées plus hauts. Ces bacs héritent par ailleurs du repère-site affecté à leurs cartes d'entrées-sorties.
Apparaît alors la structure physique de chaque équipement qu'il convient de compléter en y associant des modules de traitement, des modules d'extension mémoire et de "tête de bac" (module de traitement particulier se chargeant de l'acquisition, de la restitution et de la diffusion, au sein d'un équipement, des données échangées avec des périphériques).
Sont donc tour à tour déterminés chacun des **blocs** de modules (cf. II.3) repérés dans un bac: bloc E/S, puis bloc UT, puis bloc tdB. Exemples de configurations obtenues:

- dans le cas d'un automate,

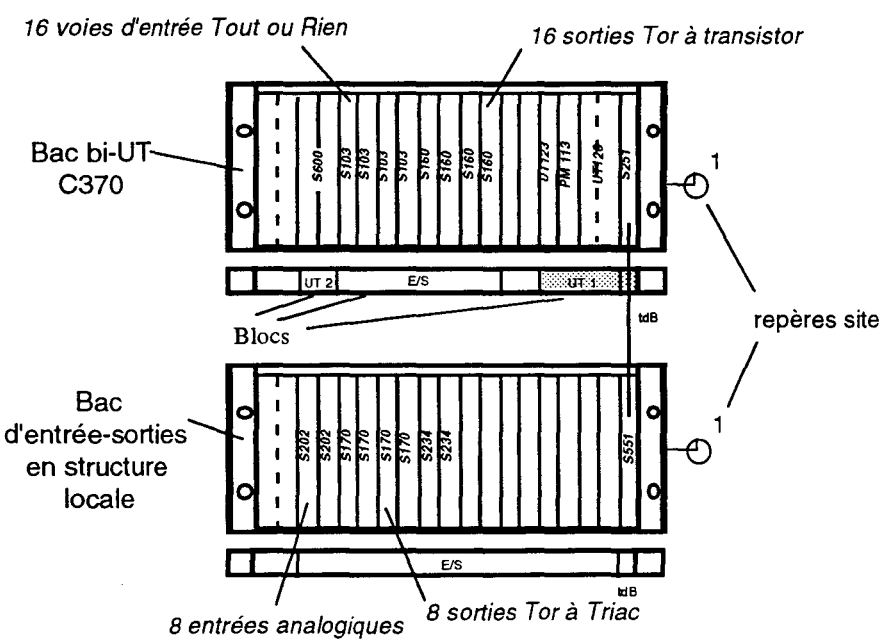


figure 68: Configuration matérielle d'un automate bi-UT doté d'entrées-sorties disposées dans un bac périphérique local

- dans le cas d'un contrôleur d'interfaces procédé,

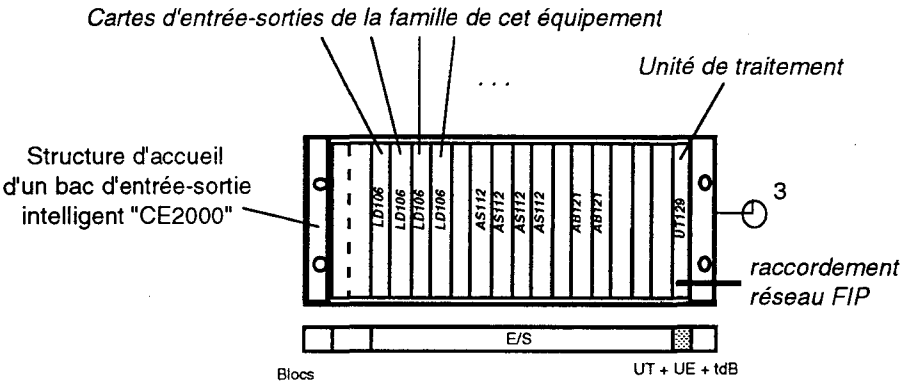


figure 69: Configuration matérielle d'un contrôleur autonome d'interface procédé

Les composants de ces blocs sont le fruit d'une sélection précise de constituants parmi ceux référencés par des **nœuds techno-fonctionnels** en bibliothèque.
Leur choix dépend des caractéristiques de ces matériels, caractéristiques fixées par des données techniques de différentes natures; pour un module de sortie Tout ou Rien par exemple: sa tension de service (12, 48 ou 220 Volts), ou pour une carte d'entrées analogiques: sa plage de variation de tension et la constante de temps de filtrage de chacune de ces entrées etc ...

Selon le schéma de données présenté en II.4.2.2, les nœuds techno-fonctionnels sont annexés aux ensembles de configurations caractérisant les modèles d'équipements d'une gamme de produits d'automatisme. Ils s'offrent donc en clé d'accès direct à la configuration d'un équipement de l'architecture logique.

IV.4.1.3 CHOIX DES MODULES DE COMMUNICATION EXTERNES

Reste, pour être complet, à placer dans les bacs précédents, des modules de connexion au divers segments de réseaux du système et des cartes assurant des échanges "point à point" avec des équipements de terrains ou d'autres systèmes extérieurs.

Selon les équipements concernés par ces échanges, sont sélectionnés des modules d'interface spécifiques, assurant des dialogues à différentes vitesses de transmission, sous différents protocoles, véhiculant des informations plus ou moins synchrones.

- Un équipement de terrain de type "variateur de vitesse" peut par exemple être connecté à une machine de traitement via une liaison série en respectant le standard RS 432 et le protocole XON/XOFF.

- La matérialisation d'une connexion à un segment de réseau s'effectue elle-même par le choix "d'unités d'échanges" respectant les couches 1 et 2 du modèle ISO (couches physique et liaison)

Ces cartes viennent se ranger aux emplacements prévus à cet effet, généralement à proximité des modules de traitement des différents équipements, quand ce n'est pas directement ces modules qui assurent eux-mêmes de telles fonctions (cas du "CE2000" de la figure ci-dessus)

Voici ce que donnerait par exemple la configuration complète du bac bi-UT de la page précédente:

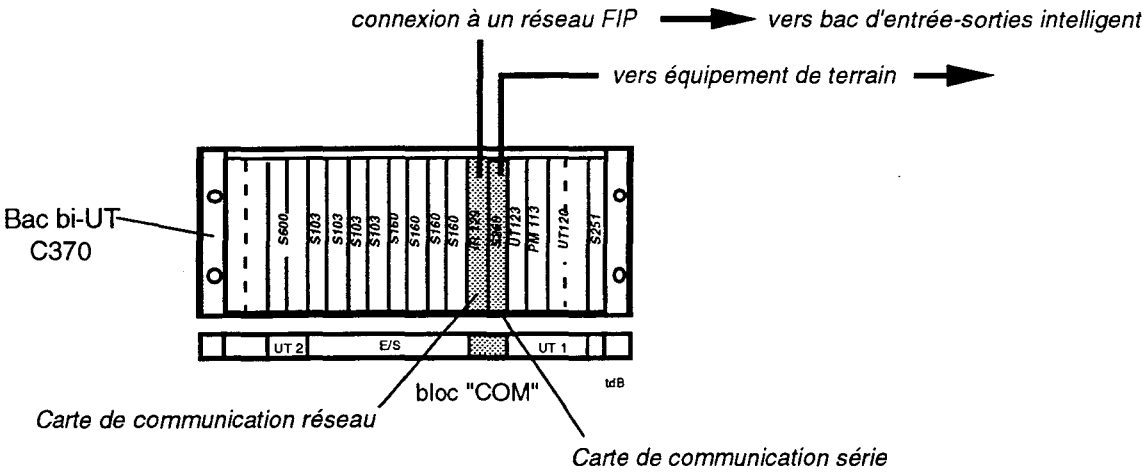


figure 70: Choix des modules de communication

- A ce stade, est également listé l'ensemble des accessoires venant compléter la fourniture d'un équipement: accessoires de continuité, limandes inter-bac, charges externes, etc ...

  Ces accessoires sont classés en 3 catégories:
  - accessoires de modules,
  - accessoires de bacs,
  - accessoires de liaison réseau ou de connexion "point à point" entre équipements.

  Ils sont sélectionnés de la même façon que le sont les modules électroniques d'une configuration matérielle, parmi les constituants regroupés dans les nœuds techno-fonctionnels d'une bibliothèque.

  Ainsi, un accessoire de continuité est pointé par le même nœud que les unités d'échanges servant à la connexion d'un modèle équipement à un modèle de segment de réseaux donné.
  Un accessoire de bac figure, pour sa part, dans le nœud techno-fonctionnel pointé par l'ensemble de configuration auquel ce bac fait référence.

IV.4.2 ETAPE 2: CHOIX DES ALIMENTATIONS

On note une multitude de choix possibles pour alimenter les composants d'une cellule d'automatisme. Ces choix varient en fonction de l'alimentation secteur utilisée et en fonction de la configuration retenue pour alimenter ces matériels.

Deux modes d'alimentation peuvent en effet être considérés:

- un mode "centralisé"
- ou une alimentation "en direct" des bacs de l'architecture physique

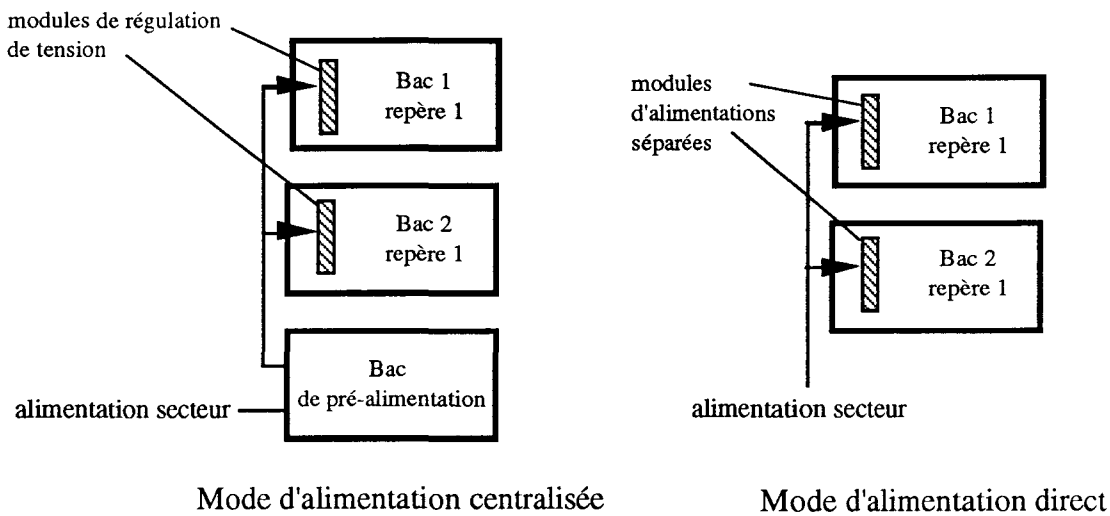


figure 71: Configurations d'alimentation

- Le premier mode s'emploie à alimenter les structures d'accueil entachées d'un même repère-site. Il requiert l'utilisation d'un bac de préalimentation et de régulateurs de tension.
- Le second mode permet d'alimenter directement les modules regroupés dans un bac dans la mesure où ce dernier est seul à être affecté à un repère-site précis. Il requiert l'utilisation de modules d'alimentation spécifiques se chargeant de convertir l'alimentation secteur en différentes sources de tension directement exploitables par les modules s'insérant dans ce bac.

IV.4.2.1 BILANS ÉLECTRIQUES

Quelle que soit la configuration d'alimentation retenue, le choix des alimentations du système s'opère sur un bilan, bac par bac, des consommations des différents modules contenus dans ces structures d'accueil.

- Ce bilan s'effectue, pour les modules des blocs physiques d'entrées-sorties, généralement en regard d'abaques permettant de calculer l'intensité du courant consommé par ces composants.

Exemple d'abaque de calcul relative à des modules de sorties Tout ou Rien:

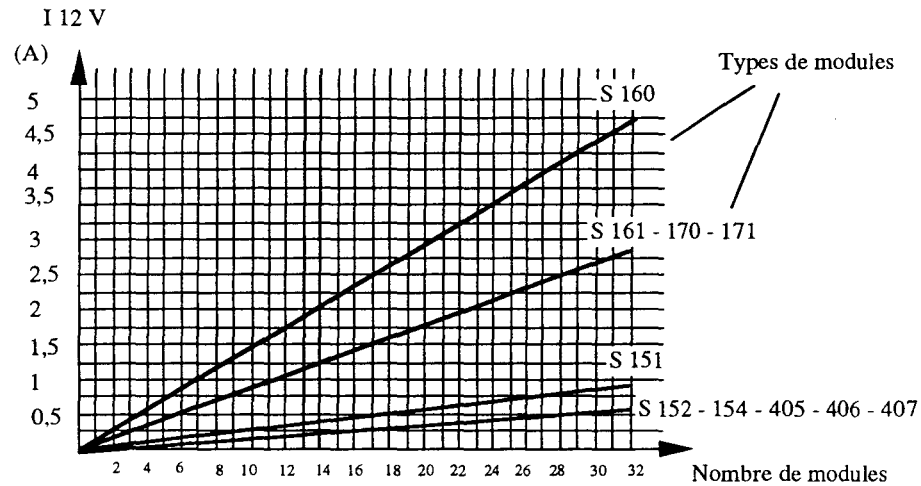


figure 72: Puissance consommée par n modules de même type

- Il se poursuit en adjoignant, au résultat du calcul précédent, les consommations des modules des autres blocs physiques d'un bac. (Bloc UT, bloc tdB, bloc COM, etc ...) Ces consommations font partie des données techniques associées aux constituants d'une bibliothèque de produits.

Un bilan par repère-site est par ailleurs effectué à chaque fois que plusieurs bacs sont localisés à un même endroit sur le site.

IV.4.2.2 CHOIX DES MODULES D'ALIMENTATION ET DES BACS DE PRÉALIMENTATION

Les bilans précédents permettent de déterminer les modules d'alimentation ou de régulation de tension de chacun des bacs de l'architecture physique.
Sont en effet choisis des modules délivrant les tensions souhaitées sur ces bacs et les intensités cumulées requises pour chaque niveau de tension exploité par les composants de ces structures d'accueil.

Les bacs de préalimentation s'accordent de la même façon au niveau de tension et au niveau d'intensité réclamés par les structures d'accueil entachées du même repère-site. Il s'adapte à l'alimentation secteur disponible à cet endroit précis de l'installation.

Ainsi sont par exemple proposés dans les gammes de produits commercialisés par CEGELEC des bacs de préalimentation acceptant des tensions d'entrée continues de 24 ou 48 Volts, des tensions alternatives de 110 ou 220 Volts.

IV.4.3 ETAPE 3: CHOIX DES STRUCTURES MÉCANIQUES D'ACCUEIL

Chaque bac d'une configuration matérielle doit figurer dans une structure mécanique d'accueil (une armoire ou un coffret de protection).

Le choix de cette structure d'accueil dépend d'une part de l'encombrement des bacs à y insérer et d'autre part des conditions générales d'environnement offertes aux équipements de l'architecture physique.

Dans certains cas, serons choisis des structures d'accueil fermées, obligeant à une ventilation forcée des composants matériels du système.
Dans d'autres cas, seront au contraire retenus des armoires ou coffrets ouverts et auto-ventilés.

Nous ne détaillerons pas ici comment est opéré le dimensionnement des éléments de ventilation. Sachons simplement qu'il s'effectue à partir d'un bilan thermique détaillé de chaque ensemble matériel contenu dans une structure mécanique d'accueil fermée.

Le processus se clôt donc par la détermination des éléments mécaniques assurant la prise en charge des composants matériels de l'architecture.
Il s'achève précisément par l'inventaire des accessoires de ces éléments mécaniques.

IV.5 CONCLUSION

Les trois sections précédentes nous ont permis de lister les différentes étapes de conception de l'architecture physique d'un SCC.

Ces étapes font suite au processus plus formel de détermination de la structure logique du système, caractérisant la couche "domaine" d'un système appliqué et réparti de contrôle-commande.

Elles se poursuivent par la conception des différents éléments de logiciel portés par les machines de traitement de l'architecture en respectant les affectations précises données à chacune de ses interfaces avec la partie opérative.

La construction de l'architecture physique du SCC vient donc entériner le choix des équipements effectué par projection d'un modèle formel du besoin sur des éléments distincts d'un système support.
Elle permet d'entamer l'étude de l'application logicielle en toute quiétude, connaissant le caractère définitif de la structure déterminée dans cette ultime phase de conception matérielle.

Nous retiendrons de ce chapitre, l'importance accordée à l'architecture site pour organiser, puis implanter les équipements de contrôle-commande d'un SAP; cette architecture site où sont localisées les interfaces entre le SCC et la partie opérative.
L'originalité de la démarche présentée dans ce chapitre vient de l'exploitation de cette architecture site pour représenter un premier modèle d'organisation de la commande qui, en respectant les développements possibles des constituants d'un système support, permettent de procéder par étape à la construction de l'architecture matérielle du SCC.

CHAPITRE V .. A LA CONCEPTION PRÉLIMINAIRE DU LOGICIEL DE SES EQUIPEMENTS D'AUTOMATISME

Nous disposons, à ce niveau de la démarche, d'une architecture logique composée d'équipements organisés en cellules d'automatisme. Ces équipements possèdent des unités de traitement dont il convient maintenant de concevoir les logiciels respectifs.

La méthode proposée dans ce chapitre consiste d'une part, à analyser le comportement attendu du sous-système de contrôle-commande attaché à chaque unité de traitement pour en déduire ses processus élémentaires et, d'autre part, à déterminer la hiérarchie des ressources permettant l'exécution de ces processus.

Cette méthode s'appuie sur les concepts définis au paragraphe II.3. Elle adapte la double démarche "événementielle et objet" de TOCCATA à la conception de sous-systèmes d'automatisme en considérant une typologie de protocoles permettant de représenter tous les types d'interfaces internes ou externes à un SCC.

En distinguant la composante "signal" de la composante "donnée" de ces interfaces, la typologie proposée permet d'étudier séparément les aspects réactifs et transformationnels des processus et fournit ainsi une base solide pour la réutilisation de composants logiciels d'une application à une autre.

V.1 LES PRINCIPES D'UNE BONNE MÉTHODE DE CONCEPTION

Quatre principes généraux s'appliquent à tout exercice de conception, qu'il s'agisse de concevoir un produit ou d'élaborer un prototype, que la solution recherchée soit d'ordre matériel ou logiciel:

Le premier principe découle du fait qu'il ne faut, en règle générale, jamais confondre **analyse du besoin** et **recherche de solutions.** Il tend à marquer cette distinction fondamentale en proposant une formalisation systématique du besoin, dans le cas présent fixé par la spécification du sous-système de contrôle-commande à implanter sur une unité de traitement, avant d'entamer la conception de ce sous-système.

Le second principe va dans le sens de la qualité du produit à réaliser. Il est basé sur le respect d'une **démarche progressive de conception** d'un système qui, à chaque étape du processus ainsi décrit, distingue des composants répondant à des ensembles cohérents de fonctions faiblement couplés entre eux.

Le troisième principe rappelle la nécessité de **valider une à une les étapes** de conception du système.

Le quatrième et dernier principe part du constat que l'**on innove mieux en copiant plus**. Il préconise donc de mettre tout en œuvre pour d'une part, identifier les éléments d'application relevant d'une certaine généricité et d'autre part, susciter, au moment adéquat, la réutilisation de ces composants génériques de commande.

V.1.1 PRINCIPE 1: FORMALISER LE BESOIN

Formaliser le besoin n'est pas un exercice facile. Il sous-entend une analyse détaillée du problème à résoudre et le choix d'un modèle de représentation adapté aux types d'interrogations ayant cours lors de la spécification.
Dans le domaine des systèmes de contrôle-commande industriels, comme pour bien des logiciels en général, MODAL [MOD 89] [MOD 93] propose d'affiner le besoin en considérant trois points de vue complémentaires:

- un point de vue fonctionnel,
- un point de vue lié aux données manipulées par le système (interfaces internes et externes),
- un point de vue opérationnel plus spécifiquement axé sur ses modes de fonctionnements propres.

Dans la pratique actuelle, bien que ces trois axes méritent tous une attention égale, n'est généralement bien traité que l'aspect fonctionnel du problème auquel est associé une vision globale des données gérées par le système. C'est ainsi qu'il est possible, à partir de la spécification détaillée de l'ensemble d'un SCC, dont les principaux points sont évoqués au paragraphe IV.2.2, de dégager un sous-ensemble de fonctions clairement délimité qui sera réalisé par les éléments associés aux tâches d'une unité de traitement.

Les modes de fonctionnement du SCC sont également un des aspects les mieux traités dans un document de spécification générale. Ils sont issus de l'analyse des "modes de marche" du système. Leur projection sur des unités de traitement particulières de l'architecture, n'est malgré tout, pas simple à obtenir. Elle est, selon les cas, plus ou moins guidée par la décomposition fonctionnelle de l'application (cf. V.3.4).

Certaines approches proposent d'aller plus loin dans la formalisation du problème, notamment en affinant le point de vue temporel sous-entendu derrière l'étude des modes de fonctionnement du système.
C'est le cas de SPEX [PAN 91] qui propose d'exprimer le caractère dynamique des fonctions les plus détaillées d'un diagramme SADT. C'est également le cas de l'environnement STATEMATE [HAR 90] qui, par le biais *d'activity-charts* et de *statecharts*, mêle récursivement dans une même démarche d'analyse, l'étude des fonctions du système et celle de ses états.

Ces types de démarches sont bien utiles lorsque le problème soulevé relève d'une problématique "temps réel", où l'appréciation du temps revêt alors autant d'importance, sinon plus, que l'exposé clair et précis des principales missions du système.
Elles permettent de valider très tôt dans le cycle de développement du produit, la cohérence des objectifs visés et c'est en cela qu'on les qualifie, à juste titre, de méthodes de spécifications exécutables.

Que ces approches se prêtent ou non au jeu de la spécification, leur exploitation ne nous évite toutefois pas l'effort important de création que réclame la conception de la solution. Les récentes évolutions de l'environnement SPEX en témoignent, notamment [GAL 93], qui propose de prendre en compte, dans une phase de conception, les contraintes matérielles induitent à la fois par l'architecture site et la spécificité du procédé, pour aider à la mise en forme d'un SCC. La démarche de conception s'écarte de ce fait naturellement du point de vue fonctionnel initial pour tendre vers une solution respectant les contraintes émanant de la structuration de la partie opérative.

Il y a donc bien deux étapes distinctes dans le processus de développement d'un système: une étape de spécification et une étape de conception, qui adoptent chacune un point de vue particulier sur ce dernier.
En d'autres termes, il faut bien avoir à l'esprit que formaliser le besoin ne sous-entend pas forcément qu'il soit fait, par la suite, de la conception dite "fonctionnelle".

V.1.2 PRINCIPE 2: PROCÉDER PAR ÉTAPES

On distingue, globalement deux types de démarches pour concevoir un système:

- des démarches par abstraction de type "top to down", considérant à priori le système comme une boîte noire et incitant à identifier progressivement ses composants en analysant ses interfaces, ses données propres et ses fonctions;
- des démarches de type "bottom-up" préconisant au contraire de procéder par composition (tel un légo) d'éléments répondant à des sous-ensembles de plus en plus complets du besoin.

Ainsi, que l'on choisisse d'adopter, à l'issue de la phase de spécification système, un point de vue de pure analyse (descendante) ou, bien au contraire, un point de vue de synthèse (montante), il apparaît toujours nécessaire de distinguer des jalons dans la démarche conduisant à l'élaboration de la solution.

Ces jalons permettent de fréquents retours en arrière (rephasages) faisant suite à la validation des étapes successives de conception.

V.1.3 PRINCIPE 3: VALIDER CHAQUE ÉTAPE DE CONCEPTION

Comme tente de l'illustrer la figure ci-dessous, le raisonnement qui mène peu à peu à la structuration du système est comparable à une boucle de régulation numérique dont la consigne, quasiment constante, est atteinte par sa sortie après plusieurs cycles de calcul.

La procédure de conception remplace dans la boucle le sous-système physique à piloter.
Le "Feed-Back" s'opère sur l'état de la solution à chaque cycle de scrutation.
La procédure de validation joue le rôle du comparateur ci-dessous et déclenche à intervalles réguliers l'exécution d'une nouvelle procédure de conception.

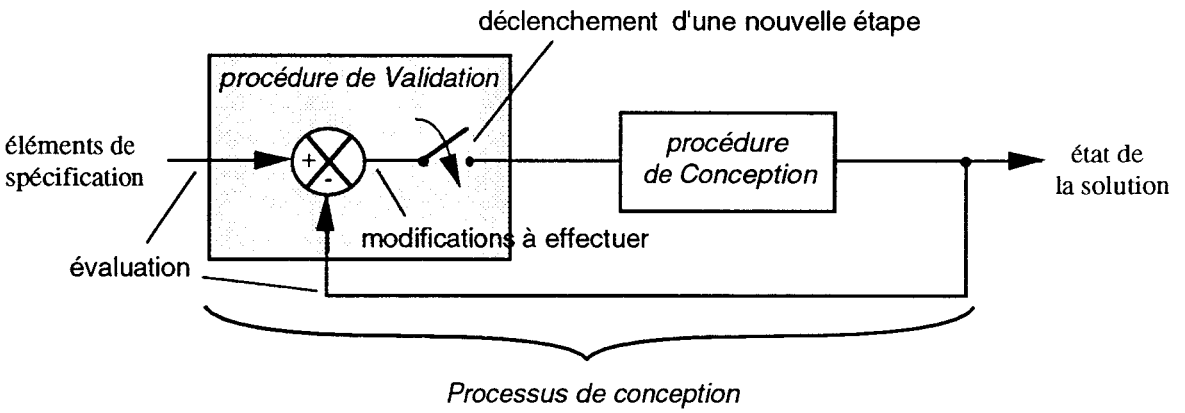


figure 73: Analogie entre processus de conception et système numérique de régulation

De la même façon qu'est soustrait de la consigne transmise au système la valeur de la variable d'état de l'objet physique à asservir, est appliquée à l'issue de chaque étape de conception une procédure de validation ayant pour objectif de comparer les composants issus de la conception aux éléments de spécification jusqu'alors pris en compte pour répondre au besoin.

Il va sans dire que cette étape nécessite au préalable de comptabiliser les éléments de spécification sur lesquels doit porter la comparaison.

Par ailleurs, tout comme le système numérique n'est efficace que si l'on observe une diminution régulière de l'écart mesuré entre sa consigne d'entrée et sa sortie, le processus de conception mérite aussi d'évaluer périodiquement le chemin parcouru entre deux jalons d'analyse ou de synthèse.
Ce chemin témoigne à la fois de l'évolution de la couverture fonctionnelle de la solution ainsi que de la précision de son organisation.

☞ La validation d'une étape de conception est donc tout aussi décisive que l'application de la procédure elle-même.

V.1.4 PRINCIPE 4: RÉUTILISER DES MORCEAUX DE SOLUTIONS EXISTANTES

Nous verrons, dans le paragraphe suivant, que la méthode TOCCATA supporte, l'une après l'autre, les deux types de démarches présentées en V.1.2 et en particulier propose de réutiliser des types abstraits de commande (cf. II.3.9) pour concevoir l'aspect transformationnel d'un sous-système logiciel.

Ces types abstraits sont les images de portions génériques d'application développées lors d'affaires antérieures. Ils sont répertoriés dans des bibliothèques spécifiques (BTA), classés par domaines d'intérêts et associés au contrôle-commande d'organes, de capteurs et d'actionneurs standards de la partie opérative.

V.2 RÉSUMÉ DE LA DÉMARCHE PROPOSÉE

V.2.1 SES GRANDES LIGNES

Par application des 4 principes précédents , la démarche présentée dans ce chapitre consiste tout d'abord, partant d'un modèle formel détaillé du besoin, à déterminer le diagramme de contexte de l'application relevant d'une unité de traitement.

Elle se poursuit par la description, en terme de comportements et sur un premier plan de raffinage TOCCATA (cf II.3.7.1), des modes de fonctionnement du sous-système de commande associé à cette UT.

Intervient ensuite le raffinement de chacun de ces modes en employant une analyse événementielle récursive, qui aboutit à force d'itérations, à l'obtention de processus élémentaires.

La démarche se clôt enfin par le recensement des machines abstraites (cf. II.3.8.1) liées à l'aspect transformationnel (aux traitements) de chacun de ces processus, machines abstraites qui, pour une bonne part, constituent des instances de types abstraits suggérés par des affaires similaires et antérieures.

V.2.2 SES PRÉSUPPOSÉS MÉTHODOLOGIQUES

V.2.2.1 CHOIX D'UN FORMALISME ADAPTÉ

Parmi les différents formalismes à notre disposition pour spécifier le besoin, nous choisirons, comme dans le cas de la conception de l'architecture de principe du SCC et pour les raisons évoquées au paragraphe IV.1.1, de partir d'un modèle fonctionnel du système (par exemple SADT) , se limitant au critère d'arrêt prescrit en III.2.2.

A l'issue de la phase présentée au chapitre III contribuant à déterminer les différentes unités de traitement du SCC programmé, est alloué à chacune de ces UT une série de fonctions élémentaires s'inscrivant dans ce modèle formel du besoin.

L'extraction du sous-arbre répondant à cette série de fonctions permet alors de recomposer facilement un nouveau diagramme fonctionnel de plus petite taille contribuant d'une part:

- à disposer d'une expression condensée et cohérente des services rendus par une UT, expression qui facilite la détermination de ses processus caractéristiques,

et d'autre part:

- à garder constamment trace d'un modèle de référence du besoin permettant ainsi de valider chacune des étapes de conception. (la validation intervient après chaque itération de procédure présentée figure 73)

V.2.2.2 RÈGLES DE VÉRIFICATION ASSOCIÉES

De cette façon, si l'application de la procédure qui consiste à déterminer le diagramme SADT d'une UT révèle la décomposition suivante:

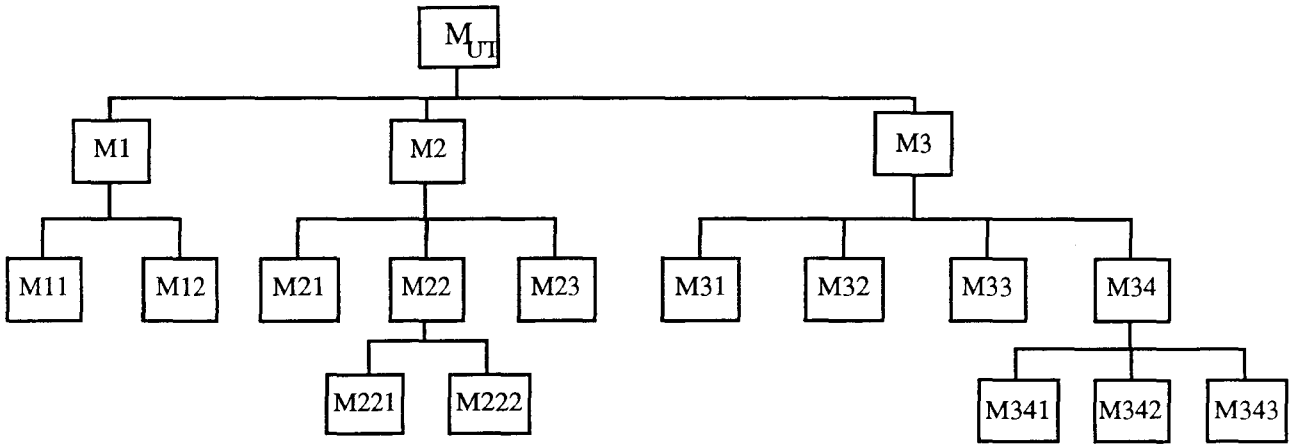


figure 74: Actigramme type

et que, par ailleurs, l'analyse événementielle préconisée par TOCCATA aboutit à la configuration ci-dessous:

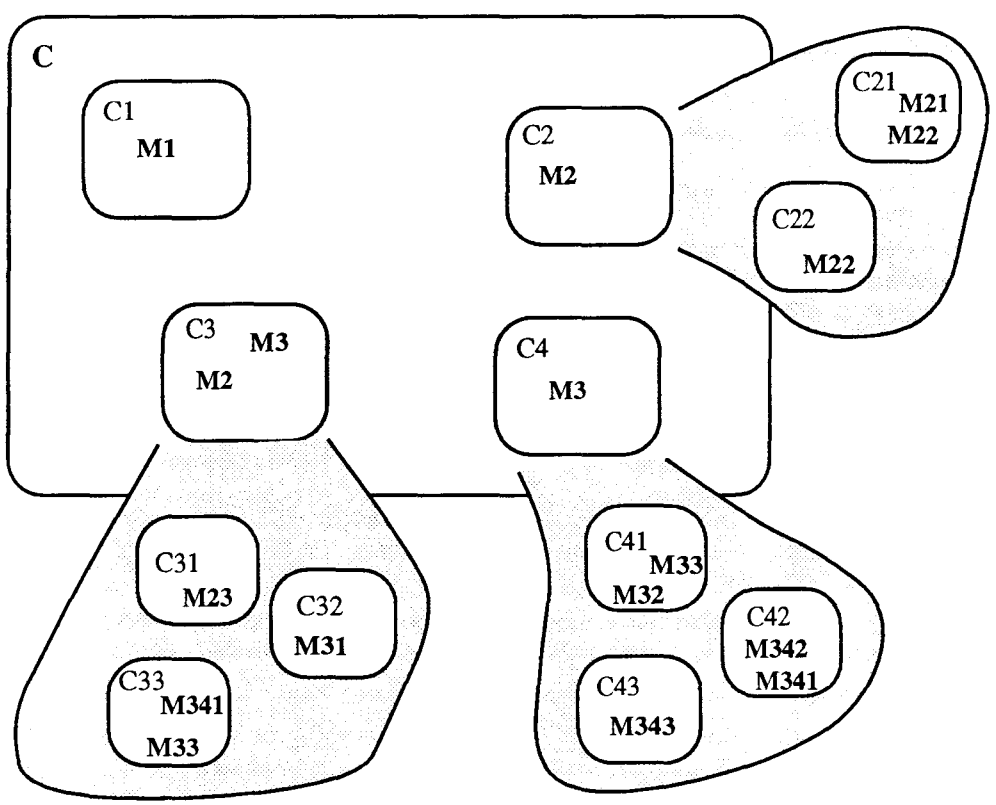


figure 75: Exemple type d'une décomposition comportementale

où, dans une première étape, le comportement C1 couvre l'ensemble des fonctionnalités initialement regroupées dans M1, le comportement C2 une partie des fonctions de M2, etc...
et dans une deuxième étape, C31 représente le processus assurant la fonction terminale M23, C32 le processus assurant la fonction M31 et C33 les fonctions M33 et M341, etc ...

Alors, on devra vérifier que les tous les sous-comportements TOCCATA couvrent bien, à chaque niveau de raffinage, l'ensemble du besoin relaté par le diagramme fonctionnel. (toutes les fonctions du diagramme doivent être prises en compte par des comportements et aucun comportement ne doit se trouvé isolé)

C'est ce que l'on constate en examinant les tableaux de couverture fonctionnelle des niveaux 1 et 2 de l'exemple présenté figure tt:

C	M1	M2	M3
C1	▨		
C2		▨	
C3		▨	▨
C4			▨

figure 76: Tableau de couverture fonctionnelle des comportements de niveau 1

	M1	M21	M22	M23	M31	M32	M33	M341	M342	M343
C1	▨									
C21		▨	▨							
C22			▨							
C31				▨						
C32					▨					
C33							▨	▨		
C41						▨	▨	▨		
C42									▨	
C43										▨

figure 77: Tableau de couverture fonctionnelle des processus de niveau 2.

- De la façon similaire, toutes les interfaces relatives à une fonction du modèle SADT doivent être prise en compte par les différents comportements assurant cette fonction:

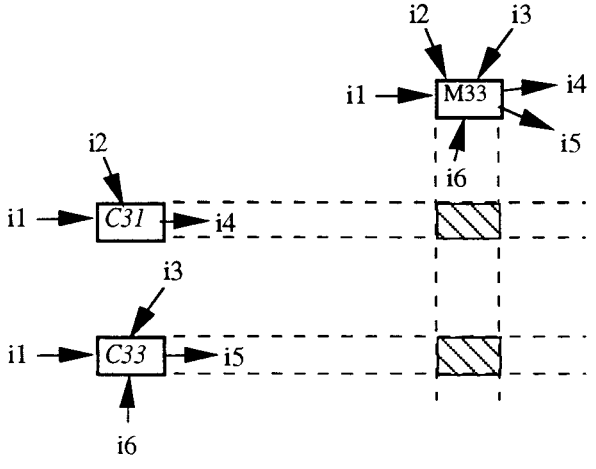


figure 78: Suivi des interfaces

V.3 EXPOSÉ DÉTAILLÉ DE LA DÉMARCHE

Après avoir rapidement brossé les grandes étapes de la démarche et choisi un formalisme permettant de fixer le besoin auquel doit répondre le sous-système d'automatisme, exposons maintenant en détail et sur un exemple simple d'application, la méthode proposée.

V.3.1 EXEMPLE D'APPLICATION

Considérons l'ensemble matériel suivant et estimons que l'application de la méthode présentée en IV aboutit à la projection des fonctions de ce sous-système sur une seule et même unité de traitement.

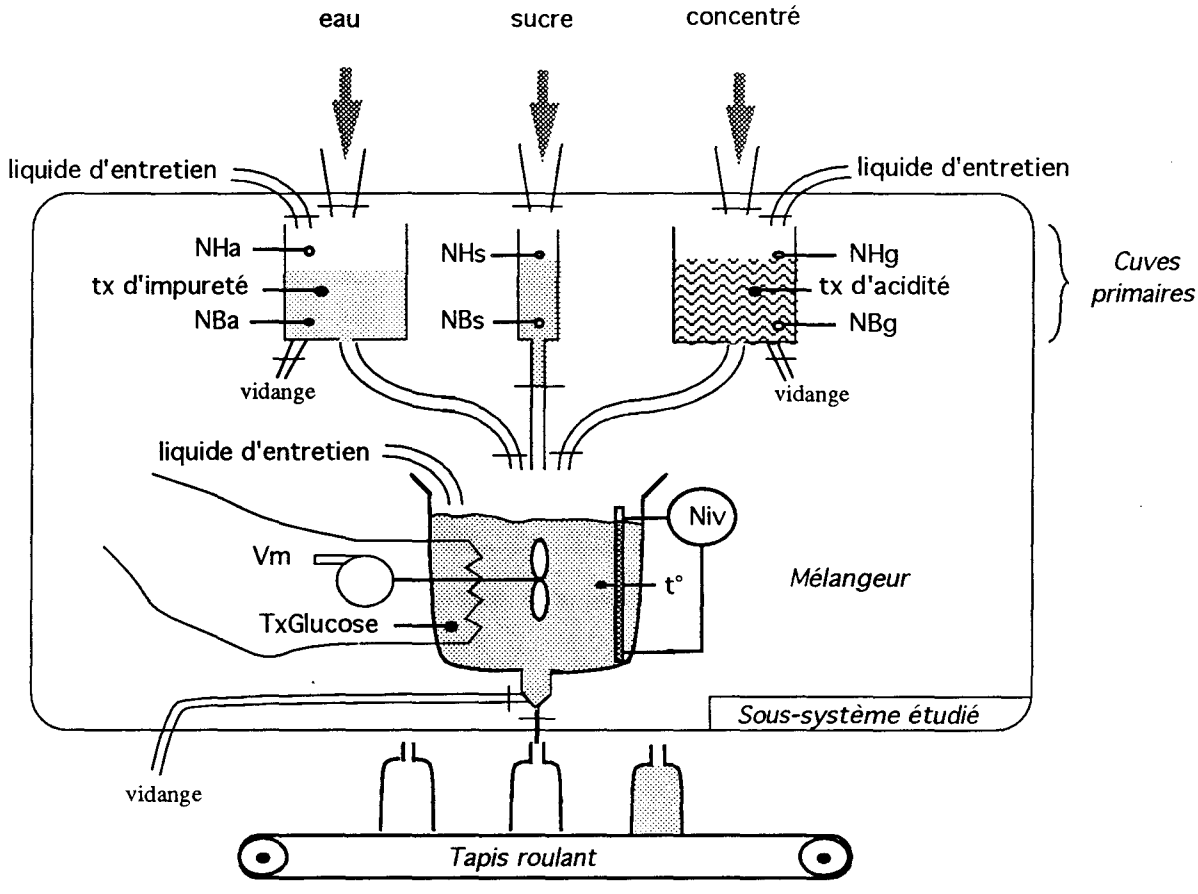


figure 79: Image de la cellule matérielle SDDS

Cherchons à automatiser le procédé spécifié en annexe 5, ayant pour missions:

- d'approvisionner de façon continue le mélangeur en ingrédients de bases (concentré de sirop, eau et sucre),
- de mélanger ces ingrédients à une température variant en fonction de la cadence d'arrivée des bouteilles sur le tapis roulant et du type de concentré de base,
- d'assurer le remplissage des bouteilles se présentant sous l'orifice de sortie du mélangeur.

V.3.2 ETAPE PRÉLIMINAIRE: SPÉCIFICATION FORMELLE DE L'APPLICATION

V.3.2.1 FORMALISATION DE L'EXEMPLE

L'analyse fonctionnelle détaillée de ce sous-système de contrôle-commande permet de tracer le premier diagramme SADT suivant:

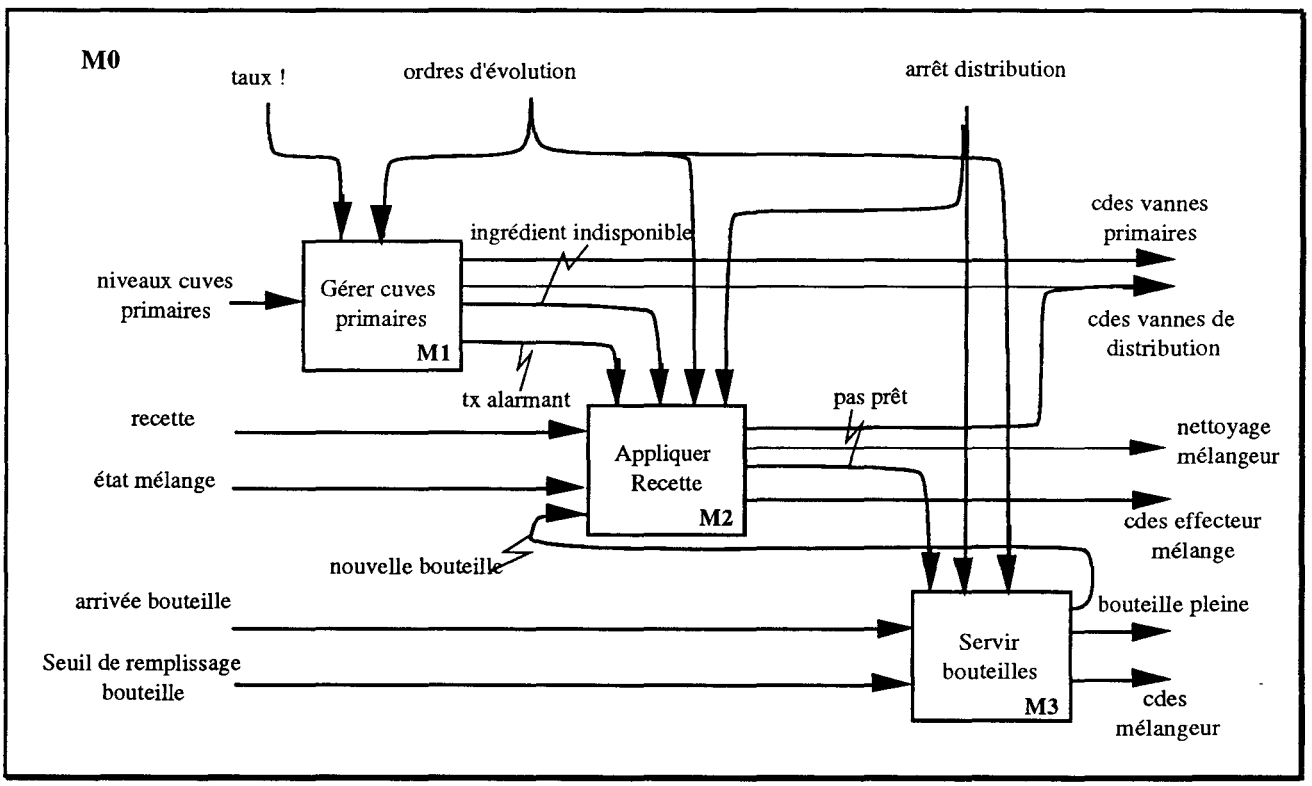


figure 80: Diagramme M0 du SDDS

Le SDDS comporte à ce niveau trois fonctions principales auxquelles sont associées plusieurs types de données d'interfaces:

- des données d'entrées, connectées aux faces gauches des fonctions;
- des données de sorties, reliées aux faces droites des fonctions;
- des données de contrôle, reliées aux parties supérieures de M1, M2 et M3.

Ces trois fonctions principales sont elles-mêmes décomposées en fonctions de plus bas niveaux, en l'occurrence pour M1 des fonctions "rincer", "vidanger cuves", "contrôler taux" et "gérer niveau", pour M2 des fonctions "rincer cuve mélangeur" à "mélanger" et pour M3 de "positionner verseur", "verser" et "signaler bouteille pleine", comme le montre l'arbre présenté page suivante:

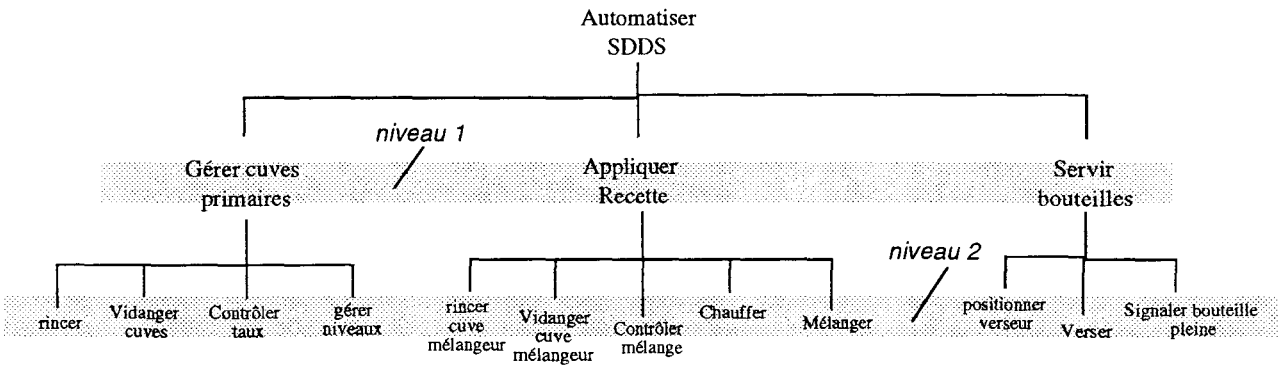


figure 81: Arbre de décomposition fonctionnelle du SDDS

Ainsi est formalisé le problème soumis à l'exercice de la conception. (cf. annexe 5 pour obtenir le détail de M1, M2 et M3)

Sa structuration s'étale sur deux niveaux. Elle peut, selon la complexité du système à modéliser, atteindre une profondeur de 3 à 4 niveaux.

Une unité de traitement peut en effet être en liaison avec l'ensemble des interfaces partie opérative d'un équipement. Elle est donc, en théorie, capable de supporter plusieurs centaines d'entrées-sorties et si l'on applique la règle de 5 (c'est-à-dire 1 fonction donne 5 fonctions maxi à un niveau inférieur, et une interface donne 5 nouvelles interfaces à un niveau n+1) préconisée par SADT, on obtient pour une profondeur de 4 niveaux un potentiel de 1250 entrées-sorties (10 E/S en M0 * 5^3), ce qui est très largement suffisant.

V.3.2.2 CHOIX DU POINT DE VUE D'ANALYSE

Toute décomposition fonctionnelle réclame l'adoption d'un point de vue constant d'analyse. C'est en respectant ce même point de vue que l'on obtient rapidement une modélisation complète et cohérente du système qui est à la base de toute bonne étude de conception.

Dans le cas présent, nous choisirons de nous appuyer sur la **marche normale** de la cellule. Ce choix tient au fait que l'entretien du SDDS s'effectue en grande partie avec l'appareillage utilisé en production. Il n'est donc pas pertinent de considérer les différents modes de fonctionnement du sous-système comme un critère majeur de décomposition de ses fonctions. Nous nous contenterons donc de reproduire les trois missions principales du SDDS sur le premier niveau de raffinage du diagramme SADT.

De façon générale, le choix du point de vue de l'analyse fonctionnelle est propre à l'application étudiée.
Il n'est donc aucun conseil à donner sur ce point si ce n'est à signaler quelques critères permettant d'évaluer la viabilité d'un diagramme déjà établi.

- On peut par exemple vérifier que l'arbre des fonctions de l'application est équilibré, c'est-à-dire qu'il comporte sur chacune de ses branches respectives des sous-arborescences équivalentes (en nombre de feuilles).

- On peut également vérifier que toutes les fonctions du diagramme placées à un niveau de profondeur donné témoignent de la même granularité de description.

- On peut enfin être amené à douter de la décomposition fonctionnelle obtenue si le taux de raffinage moyen des fonctions sur l'arbre est trop important (>5) ou n'est pas homogène entre les niveaux.

Par rapport à ce critère d'évaluation, l'arbre de la figure 81 présente un taux de raffinage moyen de 4 ((4+5+3)/3) au niveau 2 et un taux de 3 au niveau 1. La progression de la décomposition est donc quasiment constante. L'arbre obtenu est donc relativement homogène.

Sans trahir la méthode de spécification proposée par MODAL [MOD 89] [MOD 93], nous remarquerons tout de même que deux types d'approches sont, dans la pratique, préférées pour formaliser un sous-système d'automatisme:

- des approches s'accordant à regrouper sur une même branche du diagramme fonctionnel les fonctions liées un organe (ou un groupement d'organes) particulier de la partie opérative,
- des approches privilégiant au contraire les modes de fonctionnement du sous-système étudié, indépendamment de l'organisation de la partie opérative liée à ce sous-système.

Le choix de l'une ou l'autre de ces deux alternatives dépend surtout des habitudes du concepteur.

La décomposition événementielle TOCCATA va consister progressivement à prendre en compte les caractéristiques précédentes (fonctions et données) pour leur faire correspondre des éléments de comportements produisant les effets décrits dans la spécification.

Cette décomposition s'emploie à rester cohérente par rapport au besoin. Elle garde pour références inchangées les données identifiées au sein du sous-système ainsi qu'au niveau de son environnement extérieur.

La première étape se consacre, dans ce but, au report sur un diagramme "de contexte" des interfaces externes du modèle fonctionnel et à leur interprétation en terme d'informations fugitives, rémanentes ou continues.

V.3.3 ETAPE 1: ETABLISSEMENT DU DIAGRAMME DE CONTEXTE DE L'APPLICATION

V.3.3.1 INVENTAIRE DES INTERFACES

Nous proposons dans une première étape de faire l'inventaire des interfaces du sous-système porté par une UT en listant les informations identifiées à la périphérie du diagramme M0 de cette UT. (cf. V.2.2.1)

Ces données sont généralement peu nombreuses (pas plus d'une vingtaine).
Elles représentent, à ce niveau, rarement des informations élémentaires.

V.3.3.2 CHOIX D'INSTANCES DE PROTOCOLES TYPES

Une fois que sont listées les informations précédentes, peut alors s'opérer, pour chacune d'elle, le choix d'une instance particulière de protocole TOCCATA .

Cette instance traduit d'une part la nature de la communication supposée derrière chaque échange.
Elle fixe par ailleurs le type d'information véhiculée par cet échange.

Quatre cas de figures sont alors à prendre en considération, s'agissant soit de représenter:

- une interface composite, de sémantique par nature indéfinie,
- une interface révélant une information fugitive,
- une interface mettant en jeu une information rémanente,
- ou bien enfin une interface liée à une donnée continue du système de production.

- Aux interfaces composites correspondent des instances de protocoles raffinables qui se déclinent, au fur et à mesure de l'analyse événementielle, en protocoles élémentaires. (cf. III.3.3.2)

- Aux interfaces révélant des informations fugitives correspondent des instances de protocoles événementiels.
Exemples: Fronts d'entrée ou commandes à accès immédiat en sortie.

- Aux interfaces mettant en jeu des informations rémanentes correspondent des instances de protocoles ayant une certaine pérennité dans le temps.
Exemples: mémoires d'entrées, messages réseau, file d'attente, ... etc

- Enfin, aux interfaces liées à des données continues correspond des instances de protocoles actives à chaque instant d'évolution du système.
Exemples: entrées ou sorties continues échantillonnées (au rythme de l'évolution du système).

V.3.3.3 CATÉGORIES DE LIENS OPÉRATIONNELS

Les différents protocoles listés ci-avant ont une signification particulière vis à vis du diagramme de contexte de l'application.

Certaines des informations véhiculées par ces derniers répondent en effet à la définition **d'entrées** pour le comportement du sous-système de contrôle-commande. C'est en particulier le cas de l'information "recette" qui est "consommée" par le sous-système SDDS. (*1)

D'autres sont au contraire assimilables à des **effets** directs de sortie de ce comportement. C'est le cas de la commande des vannes liées au mélangeur ou bien du signalement "bouteille pleine" opéré de façon fugitive par le système. (*2)

On note également des protocoles véhiculant le flux des données de **contrôle** du sous-système de contrôle-commande. Protocole "taux !" par exemple. (*3)

Enfin, une dernière catégorie d'interfaces se prête à la représentation des **ressources** de contrôle du procédé.
Ces ressources constituent une "mémoire" sur le sous-système formé de la partie opérative (*4) puisque sont constituées d'informations rémanentes.

Un comportement entretient dont 4 catégories de liens avec son environnement extérieur, comme le montre la figure ci-dessous:

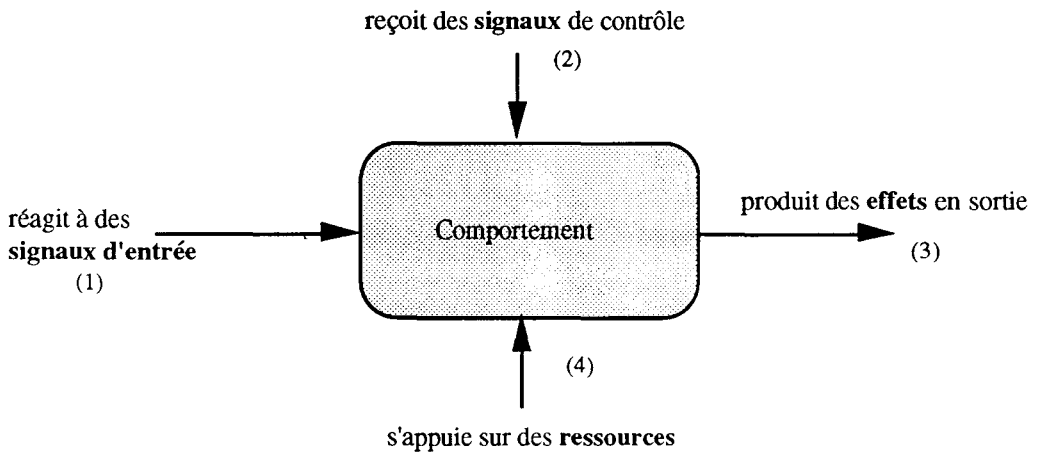


figure 82: Catégories de liens opérationnels

*Les (*i) font référence aux interfaces de la figure 83.*

Cette classification, appliquée à l'exemple SDDS, donne le diagramme de contexte suivant:

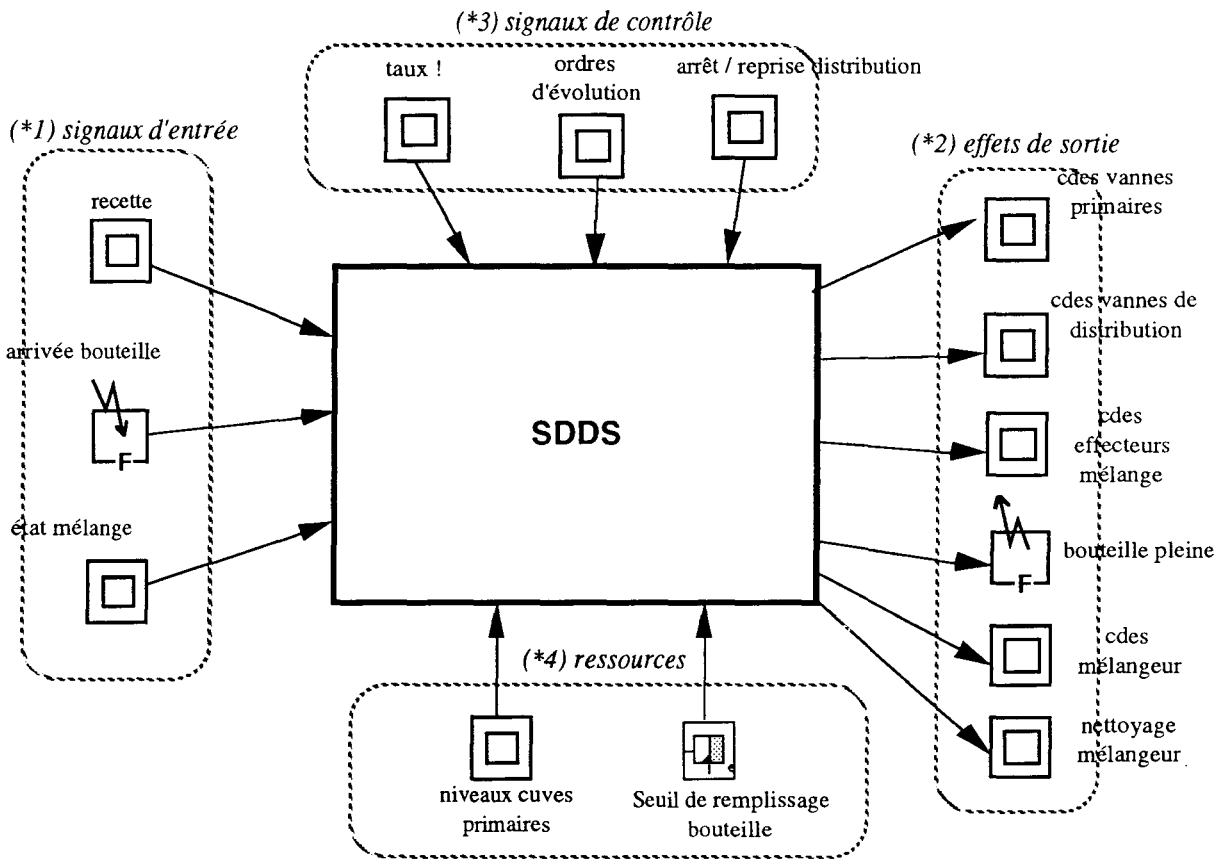


figure 83: Diagramme de contexte du SDDS

Nous constatons de ce fait qu'un comportement subit l'influence de 3 des 4 catégories de liens précitées dont 2 d'entre elles, nous le verrons un peu plus loin, contribuent directement à caractériser ses modes de fonctionnement propres.

Contrairement à ce que pourrait laisser penser l'illustration fournie figure 83, nous n'accorderons, dans les propos qui vont suivre, pas de sens précis aux facettes des éléments d'un schéma TOCCATA, de façon à ne pas contraindre la décomposition par abstractions successives que réclame l'application de cette méthode.

Nous abandonnons donc ici définitivement la sémantique SADT au profit de celle proposée par TOCCATA.

V.3.4 ETAPE 2: DESCRIPTION DES MODES DE FONCTIONNEMENT DU SOUS-SYSTÈME

L'étape 2 de la démarche s'appuie sur la spécification opérationnelle et sur les catégories d'interfaces externes (*1) et (*3) du sous-système pour déterminer ses modes de fonctionnement. L'analyse des signaux de contrôle et celle des signaux d'entrée permet en effet de marquer les différentes phases d'évolution du comportement de ce sous-système en employant des opérateurs temporels définis au § III.3.2.

V.3.4.1 DÉVELOPPEMENT DES INSTANCES DE PROTOCOLES RAFFINABLES

Le passage à tout nouveau plan de raffinage nécessite préalablement de détailler les protocoles formés à partir d'interfaces composites.
Tout comme il l'est déjà fait entre les niveaux de décomposition d'un diagramme SADT, où apparaissent alors successivement plusieurs couches d'interfaces entres les fonctions d'un arbre de ce type, cette opération dévoile des protocoles "développés" allant de pair avec la description de sous-comportements de niveaux inférieurs.

A ce stade de la démarche, sont plus particulièrement à détailler les protocoles appartenant à la catégorie des signaux de contrôle du processus automatisé, dont certains correspondent à des événements de transition entre modes de fonctionnement distincts de l'installation.
Ces événements doivent être étroitement mêlés à la recherche de ces modes, de telle sorte qu'ils soient tous pris en compte dans le modèle séquentiel de processus que décrit, sous cet angle particulier, l'application.

Certains signaux d'entrées peuvent également être liés aux modes de fonctionnement du système. Ce pourrait être par exemple le cas du protocole "recette" qui, s'il contenait dans le SDDS un événement caractérisant l'ordre de mise en application d'un nouveau dosage, aurait alors partiellement les propriétés d'un événement de contrôle.

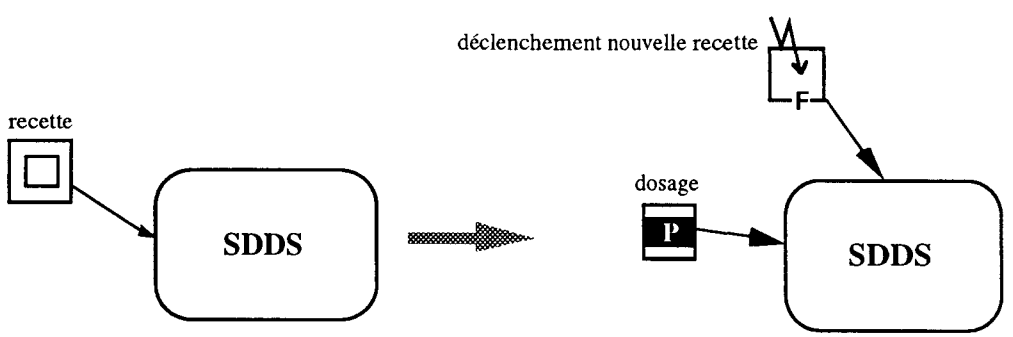


figure 84: Glissement de propriétés d'un protocole entre deux plans de raffinage

Nous comprenons maintenant mieux à la vue de cet exemple, pourquoi, contrairement à ce qui est proposé dans des modèles de types SADT, il n'est pas souhaitable d'assimiler, dans cette décomposition, les différentes facettes d'un comportement TOCCATA à des ports d'interfaces dédiés aux catégories de liens opérationnels présentées en V.3.3.3.

Il n'est pas non plus nécessaire, dans cette étape, de raffiner tous les liens tracés au niveau du diagramme de contexte. Seuls doivent être détaillés les protocoles associés, par composition, à plusieurs modes de fonctionnements.

V.3.4.2 IDENTIFICATION DES MODES

La spécification opérationnelle de l'application (chapitre 5 du Document de Spécification Système [MOD 93]) permet facilement en pratique de distinguer les différentes phases par lesquelles passe le comportement du logiciel de commande. La difficulté de cette étape réside surtout dans le fait de relier ces phases aux signaux de contrôle (*1) hérités des protocoles raffinables du diagramme de contexte.

En effet, cette opération n'est, en règle générale, pas linéaire:

- Le concepteur est tout d'abord tenté de détailler du mieux qu'il le peut les interfaces composites apparaissant au niveau de la racine de l'arborescence des comportements pour être ainsi certain de ne pas manquer d'informations cruciales lors de l'analyse.
- Il opère ensuite à l'inverse, au fur et à mesure qu'est étayé le modèle des modes de marches du système, en écartant les signaux ne s'impliquant pas directement dans ces modes.
- Enfin, sont regroupés les signaux temporairement écartés de la modélisation dans de nouvelles instances de protocoles raffinables reléguées à une décomposition ultérieure.

Cette procédure s'applique à l'exemple du SDDS; l'identification des modes de fonctionnement de cette cellule part de la spécification opérationnelle de ce sous-système qui stipule:

- que le procédé de dosage et de distribution peut être interrompu à deux occasions:
  - d'une part, pour changer de recette
  - d'autre part, pour commander l'arrêt définitif de la production
- que tout changement de recette passe par une phase d'entretien des 4 cuves de la cellule et peut intervenir à condition que cet entretien soit achevé et après qu'ait explicitement été émis l'ordre de mise en application de la nouvelle recette.

Notre système se caractérise par 2 modes de fonctionnement, "entretien" et "production", alternativement actifs dans un processus itératif qui est interrompu dès lors qu'apparaît le signal provoquant l'arrêt de la cellule. (signal masqué par le protocole "interruption" présenté ci-dessous)

Ces modes sont symbolisés par le diagramme suivant:

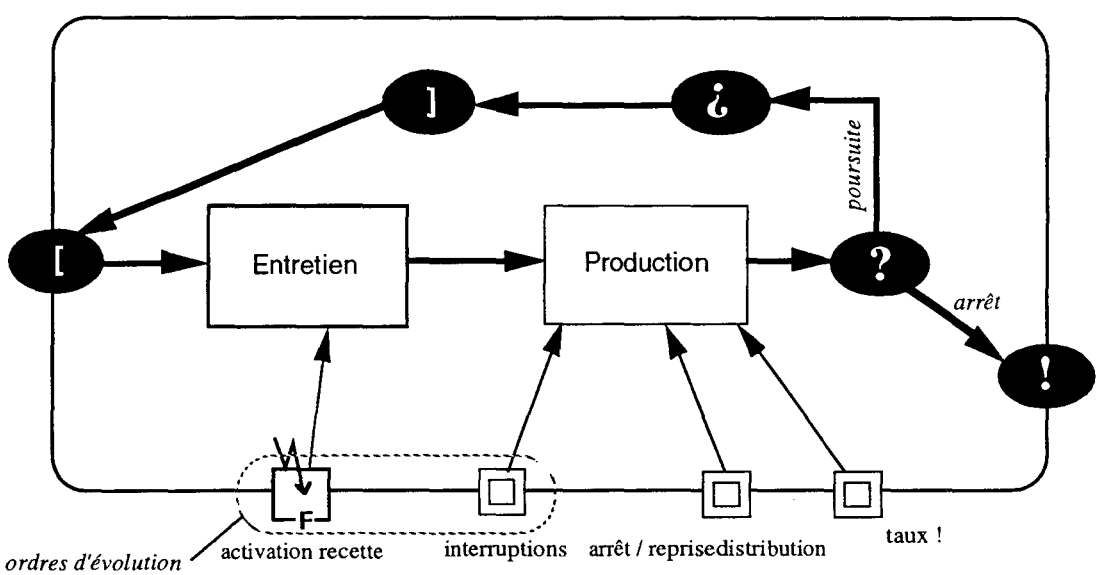


figure 85: Modes de fonctionnement du SDDS

Les deux comportements principaux de cette cellule ne se transmettent aucune information hormis celle qui consiste à activer, via un lien temporel, le mode "production" après que l'entretien soit terminé.

V.3.4.3 ATTACHEMENT DES INSTANCES DE PROTOCOLES DE NIVEAU 1

A chacun des modes précédents sont ensuite attachés l'ensemble des interfaces de niveau 1 du sous-système. Ces interfaces sont déduites du diagramme fonctionnel ayant servi de base à la décomposition. Comme il est indiqué à la fin du paragraphe précédent, seuls les protocoles raffinables participant à plusieurs modes réclament alors d'être décomposés.

Le mode "entretien" se voit ainsi associer la commande des vannes de rinçage et de vidange des cuves de l'installation, au même titre que le comportement "production" pilote les vannes de vidange, d'approvisionnement en ingrédients de base, de distribution du mélange et de remplissage des bouteilles. L'opération se poursuit ainsi jusqu'à ce que toutes les interfaces listées à la périphérie du diagramme de contexte soient prises en compte par les comportements identifiés dans cette étape.

On obtient en définitive le diagramme suivant:

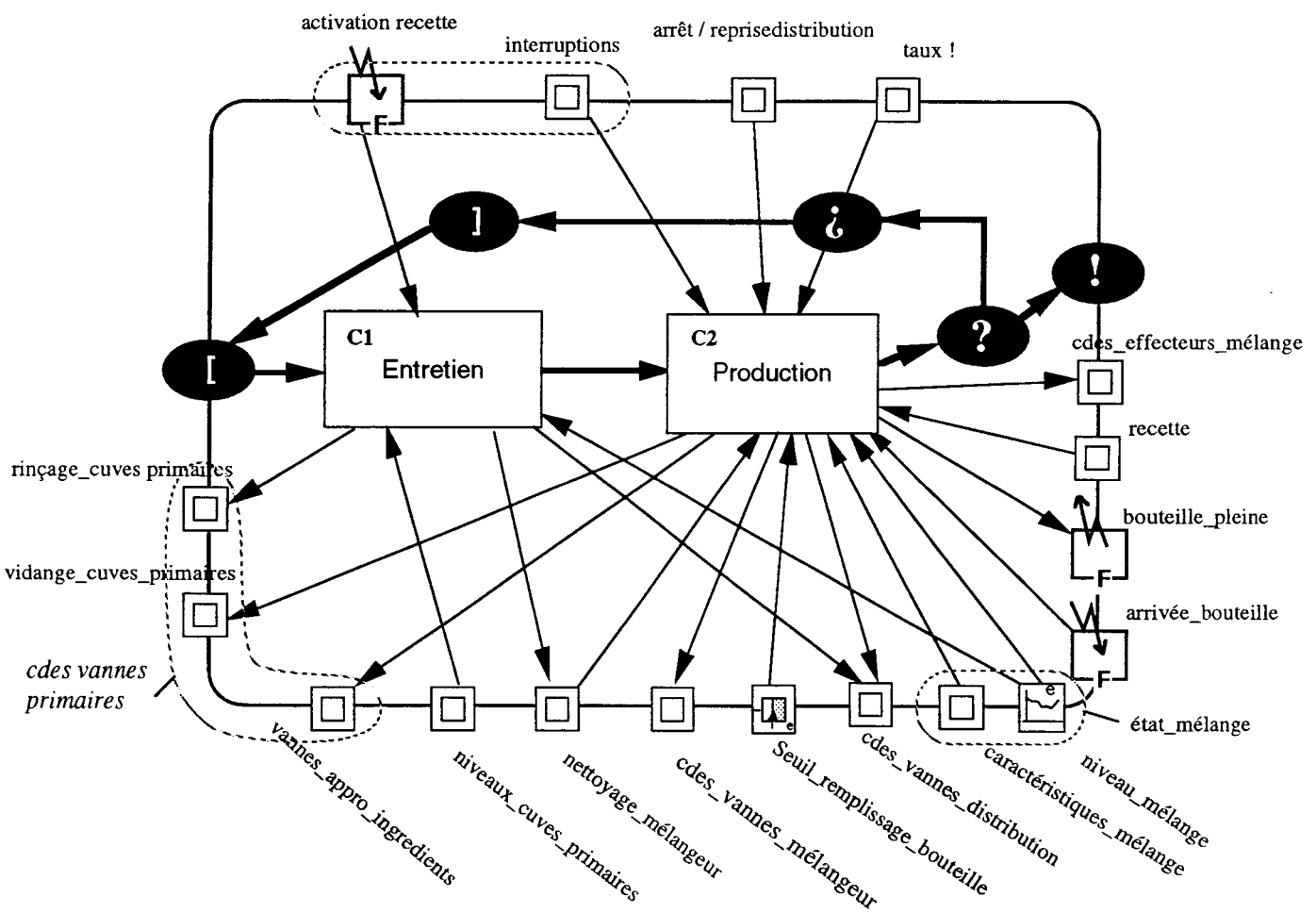


figure 86: Diagramme de niveau 1

où seuls les protocoles "cdes vannes primaire" et "état_mélange" ont nécessité d'être raffinés.

Le raffinage du protocole "niveaux_cuves_primaires" n'aurait en effet rien apporté puisque toutes les informations qu'il renferme sont partagées par les comportements "entretien" et "production". (cf. plan de raffinages de niveaux supérieurs)

Il en est de même du protocole "cdes_vannes_distribution" et à un niveau plus fin du protocole "vidange_cuves_primaires" dont les signaux sont également pris en charge par les deux modes de fonctionnements de la cellule.

V.3.4.4 VALIDATION DU NIVEAU 1

1) Couverture fonctionnelle:

Comme pour clore toute étape de conception (cf. V.2.2.2), la validation du premier plan de raffinage de l'application s'attache à évaluer la couverture fonctionnelle des différents comportement relevés durant l'analyse.

Cet exercice est trivial dans le cas de l'exemple:
C1 couvre en effet une partie des fonctions M1 et M2; C2 se projette quant à lui sur M1, M2 et M3. Le plan 1 assure donc potentiellement les trois fonctions du SDDS. Cette propriété reste toutefois à vérifier sur les plans suivants.

2) Traçabilité des interfaces:

- La procédure de validation (cf. figure 73) consiste également à vérifier la traçabilité des interfaces du modèle de comportement par rapport aux interfaces identifiées dans le modèle formel de spécification du besoin.

Dans l'exemple traité, la mise en évidence des modes de fonctionnements de la cellule à nécessité le raffinage des 3 protocoles ("ordre d'évolution", "cdes vannes primaires" et "état mélange").
Il est important de vérifier que les protocoles issus de ce raffinage correspondent bien aux interfaces relevées à un niveau M inférieur de la décomposition fonctionnelle.
On constate, après un rapide coup d'œil sur le diagramme d'analyse fonctionnelle, que c'est effectivement le cas des fonctions M1, M2 et M3 qui récupèrent à tour de rôle les données regroupées dans les 3 protocoles raffinables précédents.

Dans la mesure où le modèle fonctionnel d'une unité de traitement adopte précisément le point de vue de ses modes de fonctionnement, (cf. V.3.2.2) le premier plan de raffinage TOCCATA retranscrit alors totalement les fonctions du diagramme fonctionnel, ce qui nécessite une opération supplémentaire, celle de vérifier la correspondance entre les interfaces internes du modèle SADT et les interfaces entre modes véhiculées par des instances de protocoles TOCCATA.

3) Adéquation aux règles sémantiques TOCCATA:

- Il faut en outre veiller, dans cette étape de validation, au respect des règles sémantiques sur lesquelles s'appuie la décomposition des comportements.
On veillera en particulier à ce qu'aucune instance de protocole événementiel ne lie un mode à un autre.
On vérifiera également que l'orientation des liens opérationnels est bien respectée entre les plans 0 et 1 de description de l'application (une entrée pour le diagramme de contexte ne pouvant, en aucun cas, donner une sortie à un niveau inférieur)

Par ailleurs, toute instance de protocole raffinable représentant une interaction, c'est-à-dire au moins une entrée et au moins une sortie, devra être liée au comportement principal de l'installation par une flèche à double sens, de telle sorte que toute erreur d'interprétation soit évitée par la suite.

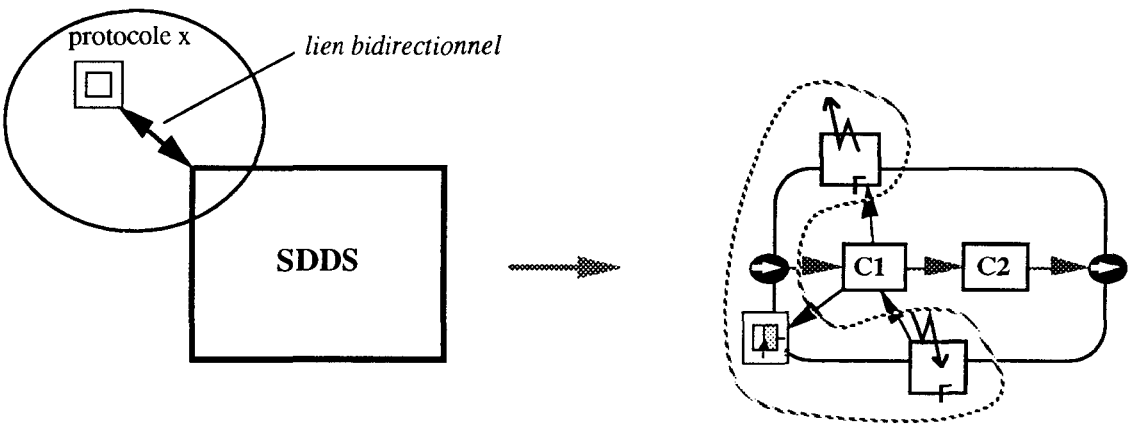


figure 87: Règle sémantique à respecter

Il va de soi que les vérifications de cet ordre sont à effectuer au fur et à mesure de la description des comportements.
C'est la raison pour laquelle l'outil ORGUE, supportant la méthode TOCCATA, vérifie en ligne tous ces aspects.

V.3.5 ETAPE 3: ANALYSE DÉTAILLÉE DES COMPORTEMENTS

Dans cette troisième étape, nous respecterons à la lettre la démarche que préconise TOCCATA pour concevoir une application logicielle.

Cette démarche tire profit de la sémantique accordée aux flux de données du système pour déterminer ses sous-comportements respectifs. Elle est itérative et prend fin dès que tous les sous-comportements d'une l'UT atteignent le niveau de granularité d'un **processus de commande**. (cf. III.3.1.2)

L'opération de raffinage d'un comportement, renouvelée autant de fois que nécessaire, se déroule en trois temps:

- dans un premier temps, sont raffinés les protocoles représentant les interfaces externes du comportement étudié et sont identifiés, en liaison avec ces protocoles, des granules de comportement élémentaires;
- dans un deuxième temps, sont évaluées les communications entre ces granules et estimés les sous-comportements à représenter;
- dans un troisième temps, sont mis en place des liens temporels entre ces sous-comportements.

V.3.5.1 RAFFINAGE DES PROTOCOLES / IDENTIFICATION DE GRANULES DE COMPORTEMENTS

Cette étape nécessite tout d'abord de procéder au raffinage des instances de protocoles liées au comportement intéressant l'étude.
Cette opération suit la décomposition des données du diagramme fonctionnel de l'UT.
Elle est semblable à celle présentée en V.3.4.1, excepté le fait que cette fois, tous les protocoles représentant des interfaces composites doivent maintenant être systématiquement raffinées.

Reprenons l'exemple du SDDS dans son mode "entretien" (cf. figure 86):

- correspond donc au protocole raffinable "rinçage_cuve_primaires", les deux instances de protocoles "rinceur_cuve_eau" et "rinceur_cuve_concent",
- correspond au protocole "vidange_cuves", les protocoles "vidange_eau" et "vidange_concent",
- correspond au protocole "nettoyage mélangeur", les protocoles "rinçage mélangeur" et "vidange mélangeur",
- correspond au protocole "cdes_vannes_distribution", les données continues "moteur_vanne_eau", "moteur_vanne_sucre" et "moteur_vanne_concent",
- enfin, correspond au protocole "niveaux_cuves_primaires", les mémoires d'entrées "niveau_cuve_e", "niveau_cuve_s" et "niveau_cuve_concent".

Toutes ces informations figurent dans les diagrammes M1 (au niveau des fonctions "rincer", "vidanger cuves" et "gérer niveaux") et M2 (au niveau des fonctions "vidanger cuve mélangeur" et "rincer cuve mélangeur") (cf. annexe 5).
Il leur est associé, de la même façon qu'en V.3.3.2, des instances de protocoles types.

A ces instances de protocoles, sont ensuite attachées des granules de comportements, s'accordant à représenter les mécanismes élémentaires prenant en charge les interfaces externes du mode en cours.

Ainsi, comme l'illustre la figure 88, tous les liens opérationnels suscités par la présence d'instances de protocoles externes au comportement étudié sont, par ce biais, identifiés et temporairement attachés à des **granules** caractérisant, dans une première approximation, les activités élémentaires du comportement.

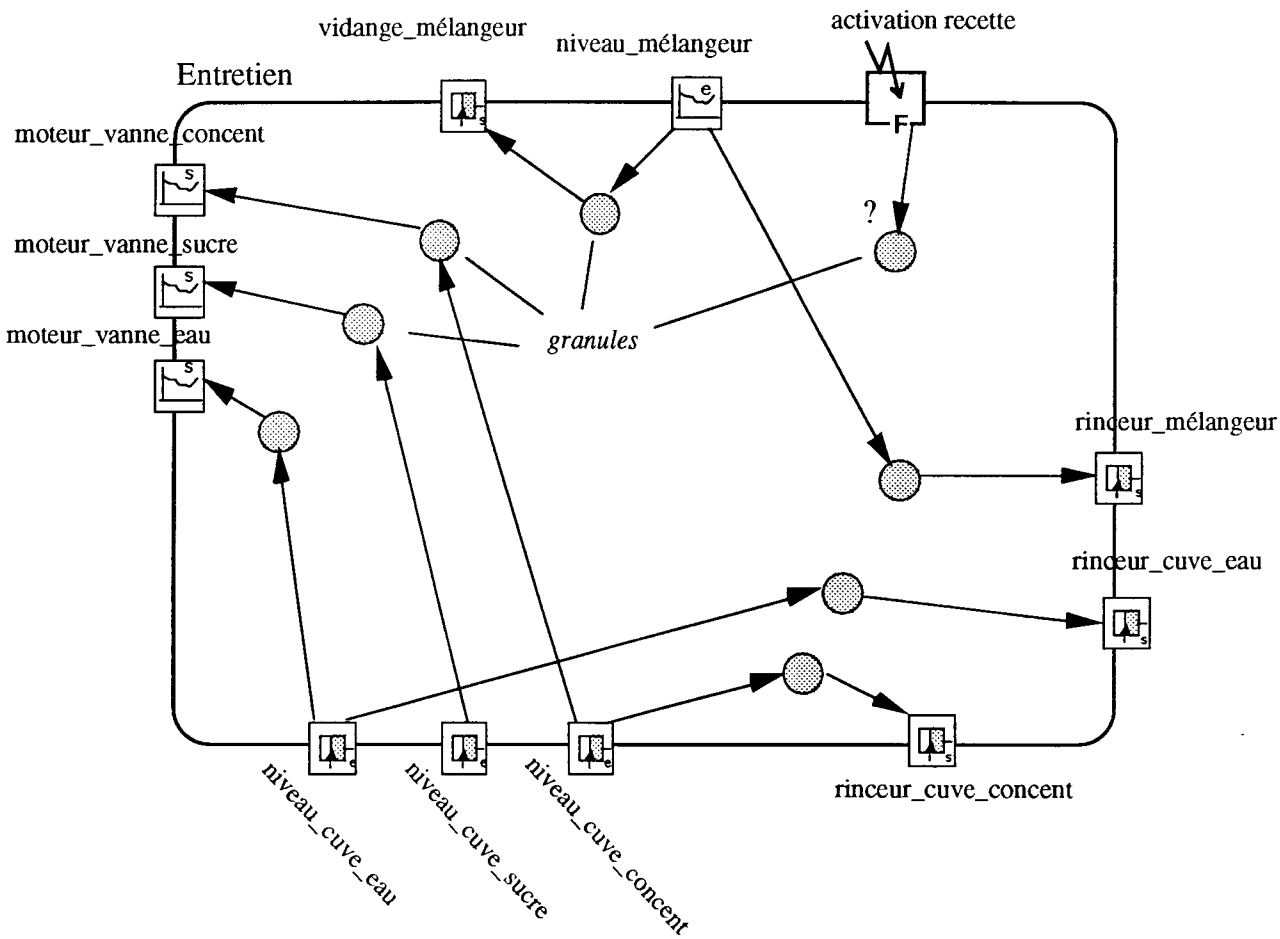


figure 88: Granules de comportement du mode "entretien"

Ces granules ne sont pas toutes nécessairement liées à des informations d'entrée ou de sortie. Nous ne pouvons donc pas encore apprécier totalement leur rôle. Leur nombre et leur qualité est, de ce fait, encore imprécis.

Certaines d'entre elles sont par ailleurs intimement liées:

Il convient donc d'examiner les interactions entre ces granules et d'adjoindre si nécessaire à cette description , tous les éléments permettant de traduire les fonctions du schéma SADT attachées aux interfaces externes.

La procédure détaillée au paragraphe suivant s'accorde donc à traduire les interfaces internes du sous-système, relatées par le modèle fonctionnel, par des instances de protocoles types permettant de dégager les sous-comportements attendus de la description.

V.3.5.2 EXAMEN DES INTERFACES INTERNES DU COMPORTEMENT

L'étude des interfaces internes d'un sous-système à un niveau de raffinage donné ne fait que très rarement intervenir des granules particulières de comportement. Elle met plutôt en relation des ensembles de granules entre eux, ensembles témoignant d'une certaine cohésion temporelle. (cf figure 89)

Elle part de l'inventaire des données d'interface entre fonctions du sous-système.

Elle a pour objet de traduire ces données par des protocoles de communication entre granules.

A cette occasion, il ne faut pas s'attendre à ce que le modèle des fonctions de l'UT donne tous le éléments permettant de mettre à jour ces communications. Cette procédure révèle en effet généralement les imperfections du modèle formel initial qui est loin d'être un modèle à états.

Certains signaux propres à l'exécution des processus liés au comportement étudié ne sont en particulier pas évoqués dans la description initiale du besoin. Ils n'apparaissent que lorsque sont attachés des éléments de solutions (exprimés ici sous la forme d'éléments de comportements) à cette description.

L'analyse respecte donc, jusqu'à un certain niveau, la description des flux de données du schéma fonctionnel mais ne s'appuie pas uniquement sur ceux-ci pour concevoir l'application recherchée. Il est besoin d'un autre point de vue pour modéliser le comportement, point de vue que la spécification opérationnelle du système contribue à exprimer.

Le concepteur à donc fort à faire à ce stade de la démarche. Il doit construire une solution en respectant à la fois les spécifications opérationnelle et fonctionnelle du logiciel portée par une unité de traitement.

Sa ténacité et son bon sens le mènera, après quelques essais infructueux, à un modèle d'application traduisant parfaitement le besoin.

Nous pouvons estimer, dans le cas de l'exemple qui nous préoccupe, que les éléments mis à la disposition du concepteur, permettent de dégager les interfaces internes présentées figure 89, occasionnant les regroupements de granules décrits à la page suivante:

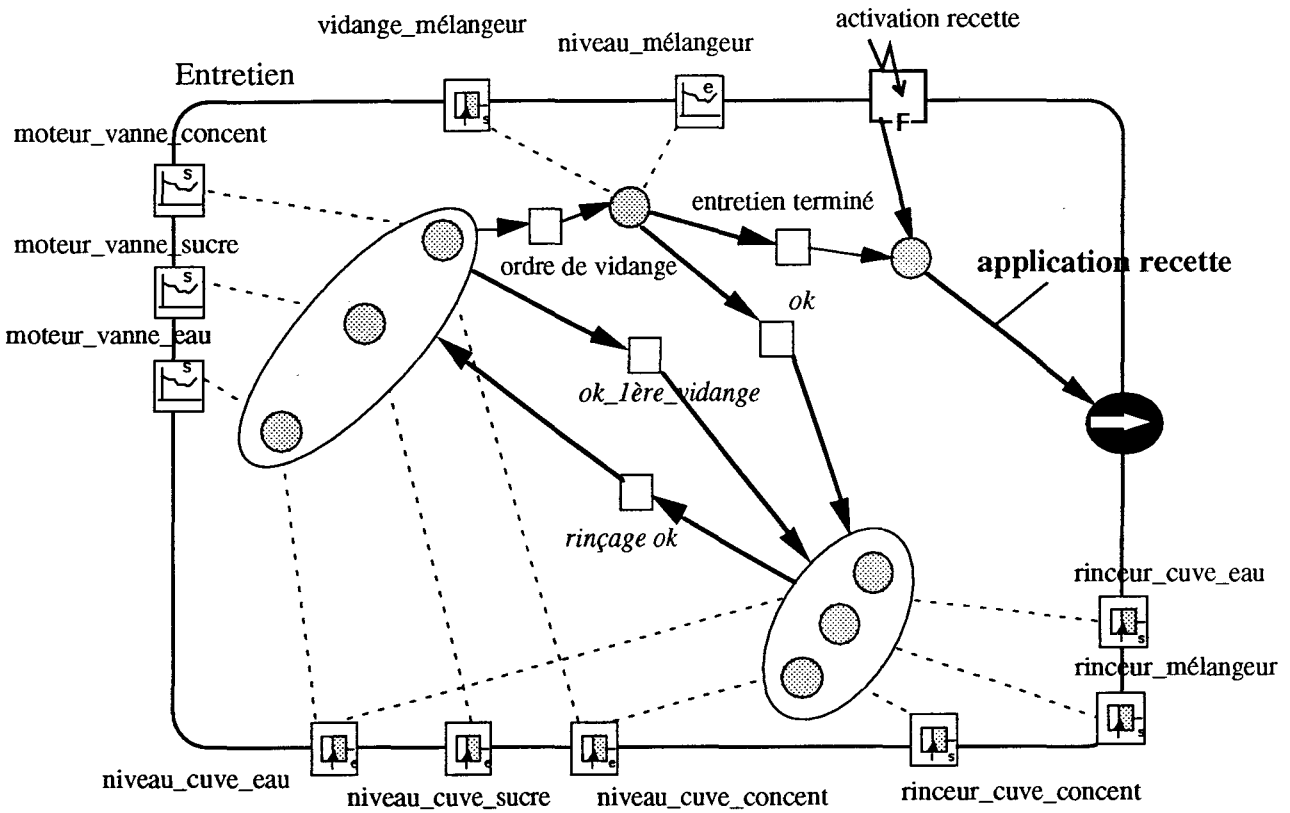


figure 89: Interfaces internes

Les interfaces "ok_1ère_vidange" et "ok_rinçage" sont directement issues des diagrammes fonctionnels M1 et M2.

Les interfaces "ordre de vidange"" et "entretien terminé" sont par contre le fruit de la conception. Ils ont pour objectif de mettre les granules encore isolées dans la description, en correspondance avec les éléments de spécification opérationnels de l'application.

V.3.5.3 IDENTIFICATION DE SOUS-COMPORTEMENTS / ETABLISSEMENT DES LIENS TEMPORELS

D'autre part, il serait inutile de séparer dans les faits la vidange des cuves primaires de la cellule de celle du mélangeur. Nous ne considérerons donc que deux sous-comportements caractéristiques: "vidange" et "rinçage" et insérerons ces derniers dans une boucle de comportements traduisant la sémantique temporelle des liens observés entres les granules élémentaires précédents.

Sont donc associés au sein d'un même sous-comportement les cinq granules de la partie supérieure de la figure 88, pour donner former le sous-comportement "Vidange". Cette opération nous permet de rester très proche du modèle fonctionnel initial puisque les trois protocoles internes effectivement prise en compte dans le modèle proposé figurent dans M1 et M2. (ok, ok 1ère vidange, rinçage ok)

L'opération se conclut par la description complète du plan de raffinage du mode "entretien".

L'opérateur "exit de boucle" se charge d'activer le comportement "production" en transmettant l'information "application recette".

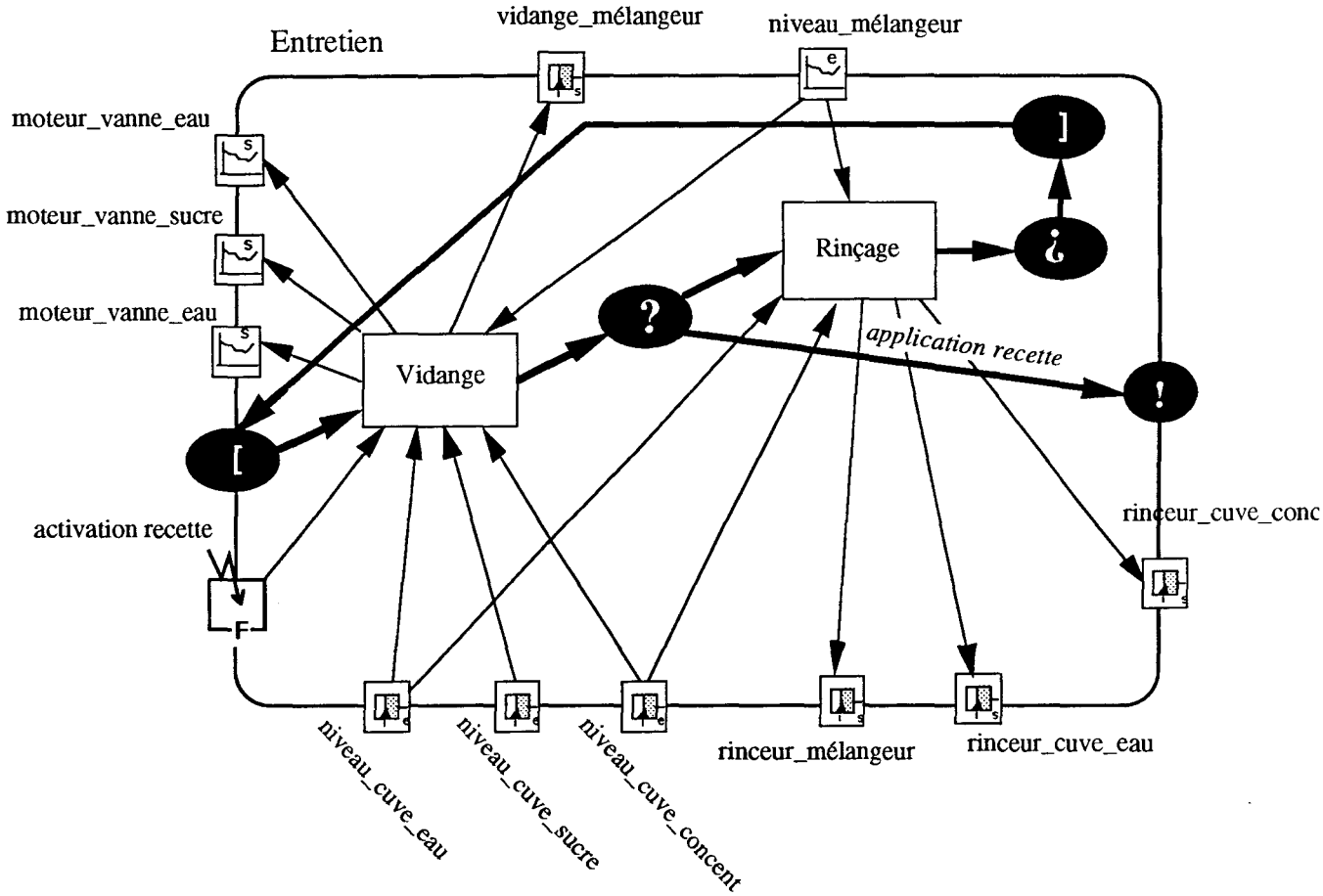


figure 90: Raffinage du mode entretien

Le cas du mode "entretien" est très simple. Il ne réclame pas un gros effort de créativité. C'est moins évident du comportement "production", qui ne laisse filtrer de sa spécification que peu d'indices sur ses constituants éventuels.

Une chose est sûre, son cas ne relève pas de l'impossible et voici pourquoi:

V.3.5.4 LES POINTS FORTS DE TOCCATA

L'intérêt de la méthode TOCCATA est de permettre une décomposition à un rythme progressif de l'application, de plus adapté à chaque difficulté à surmonter. Ce rythme est modulé par le taux de progression de raffinage du modèle fonctionnel que nous avons rapidement appris à quantifier en V.3.2.2.

Il n'y a donc, aucun danger, en appliquant les recommandations données au paragraphe précité, de voir le niveau de besoin auquel doit satisfaire chaque plan de raffinage de l'application, atteindre des proportions inquiétantes. Quant bien même cela pourrait se produire, il serait toujours possible de ralentir la cadence de progression de la conception en limitant la décomposition des protocoles raffinables d'un plan à un autre.

TOCCATA offre donc sur ce point une double garantie qui permet d'aborder la conception en toute sérénité.
Avoir comme fil directeur un modèle de décomposition dont on sait à priori qu'il ne mènera pas à l'insurmontable, apporte en effet un confort psychologique important dans cet exercice délicat d'analyse et de synthèse.

V.3.5.5 TECHNIQUES DE RAISONNEMENT ASSOCIÉES

Il est une technique à adopter lorsque la solution au problème ne vient pas rapidement à l'esprit, celle de rapprocher, autant que faire se peut, des ensembles de granules élémentaires du comportement étudié, des **rôles** distincts que jouent le sous-système.

Cette technique donne généralement de bons résultats, car elle attribue aux sous-comportements alors mis en évidence, une série de fonctions élémentaires nécessitant l'introduction d'un "organe" particulier de contrôle-commande.

Le raisonnement par analogie porte, lui aussi, très souvent ses fruits dès lors qu'il est couplé à la technique précédente;
En effet, il est quelques fois commode d'assimiler le problème qui nous est donné à résoudre, à un système par certains aspects analogues (*même procédé mais appliqué manuellement, procédé semblable utilisé à d'autres fins, automatisation similaire conduite à l'aide d'une autre technologie, etc*).

Cette assimilation donne des points de repères à la démarche de conception, ouvrant ainsi la voie à des types solutions éprouvées.

Appliquons cette technique au mode "production" de notre exemple:

Loin de prétendre réinventer le process et cherchant surtout à organiser ces éléments de comportement, ce mode de fonctionnement laisse apparaître, avec un peu de bon goût, de bon sens et d'imagination, trois rôles essentiels dans l'application de la recette de fabrication du sirop, rôles qui peuvent être mis en parallèle avec l'organisation du personnel d'un bar.

En effet, tout comme un barman à l'expérience du dosage d'un cocktail, la cellule a besoin d'un organe qui puisse appliquer la recette désirée.

De plus, un cocktail, une fois préparé, a besoin d'être acheminé au consommateur qui l'a commandé et cette opération qui est généralement laissée au soin d'un serveur. Nous retiendrons donc deux organes de ce type dans notre mode "Production".

Enfin, le niveau d'approvisionnement des ingrédients nécessaires à la mise en application de la recette doit, dans le cas du bar, comme dans le cas de la cellule SDDS, être géré parallèlement aux deux activités précédentes, ce qui conditionne l'introduction d'un troisième "acteur" indépendant dans le sous-système: un magasinier.

Ces trois rôles essentiels pour produire en quantité continue le sirop et le servir au fur et à mesure de l'arrivée des bouteilles vides sur la machine transfert regrouperont les granules de comportement identifiées comme précisé en V.3.5.1.

L'analogie avec le personnel d'un bar représente, dans notre exemple et de façon symbolique le savoir-faire réutilisé dans toute application. Ce savoir-faire est généralement connu de l'entreprise émettrice de l'offre. La conception se résume donc, dans la majorité des cas, à le formaliser.

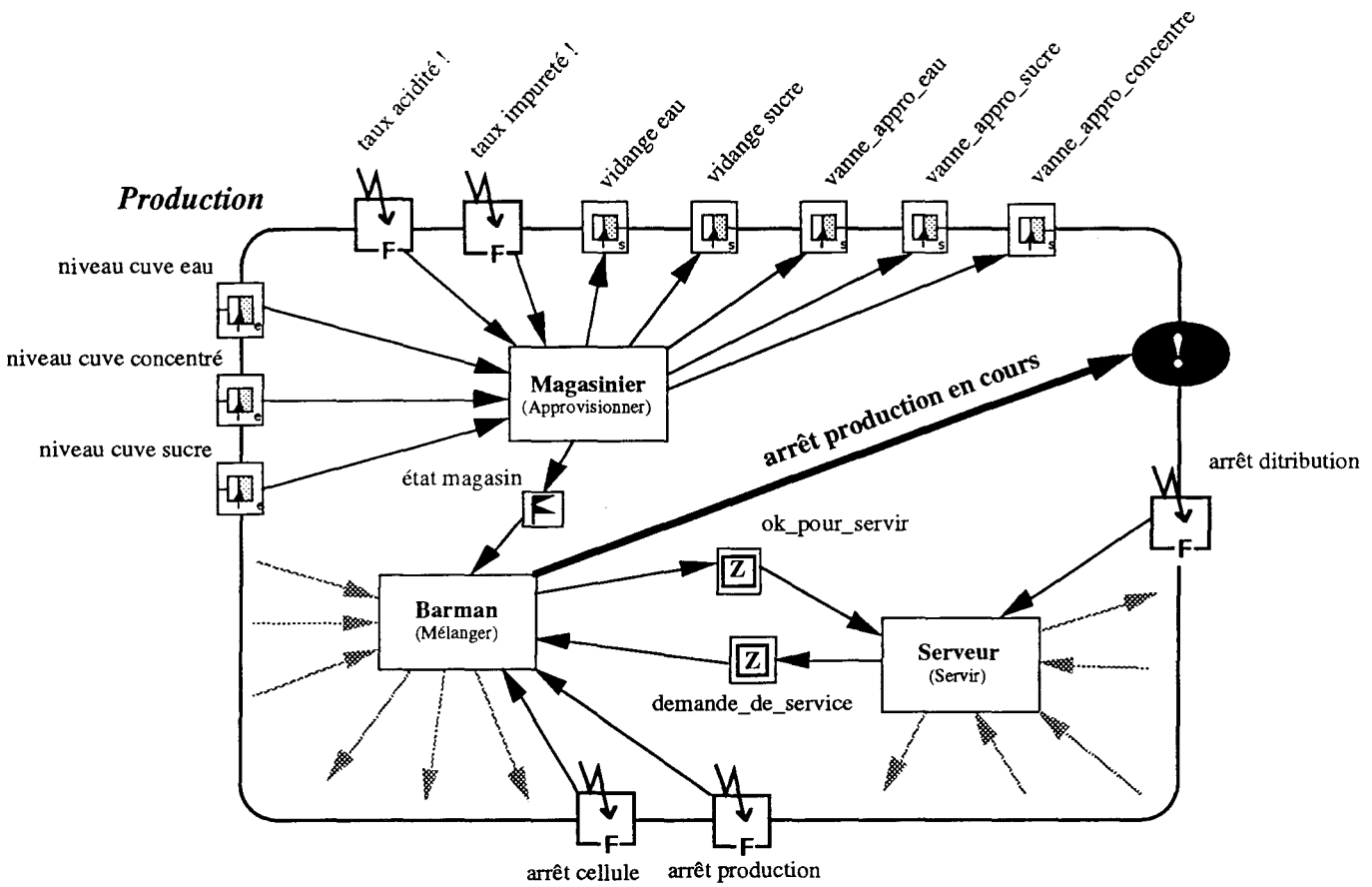


figure 91: Raffinage du mode production

Il leur sera par ailleurs attaché des signaux particuliers de contrôle relayés par le lien temporel "arrêt production en cours " ci-dessus.

L'examen des interfaces internes de ce mode de fonctionnement relèvera enfin trois instances de protocoles entre les sous-comportements "magasinier" "barman" et "serveur", interfaces permettant d'une part de tenir informé le barman de la disponibilité des ingrédients de base et d'autre part d'autoriser le remplissage des bouteilles lorsque le mélange est prêt.

V.3.5.5 VALIDATION DES COMPORTEMENTS

La validation de ces étapes de raffinage intervient niveau par niveau.

Elle permet d'évaluer, à chaque stade d'évolution de la décomposition:

- la couverture fonctionnelle des sous-comportements de l'unité de traitement.

Nous obtenons, à titre d'exemple, pour les comportements niveau 2 de la cellule SDDS, le tableau suivant:

	M11	M12	M13	M14	M21	M22	M23	M24	M25	M31	M32	M33
C11			▨		▨							
C12	▨					▨						
C21		▨	▨	▨								
C22							▨	▨	▨			
C23										▨	▨	▨

figure 92: Tableau de couverture fonctionnelle de niveau 2

Aucune fonction n'y est oubliée et aucun comportement ne s'y trouve isolé. La description formelle initiale s'avère par ailleurs très granulaire puisqu'une seule fonction est assurée par plusieurs comportements.

- le suivi des interfaces du système:

Les interfaces externes des sous-comportements sur lesquels porte cet acte de validation sont à l'origine de la décomposition menée à chaque étape. Ils ne peuvent donc pas être omis durant cette phase.

Tout l'effort doit alors porter sur le suivi des interfaces internes du comportement étudié, de telle sorte qu'aucune donnée échangée entre les fonctions SADT du niveau pris en référence pour la décomposition ne soit évincé du modèle de comportement.

Certaines données d'interfaces peuvent être reléguées à une étape décomposition ultérieure. C'est le cas des données échangées entre fonctions regroupées dans un même sous-comportement. (exemples: arrêt appro et alarme du diagramme M1 (cf. annexe 5)). Celles-ci font alors corps avec un sous-comportement particulier. Elles ne peuvent faire l'objet d'une telle vérification.

- le respect des règles de décomposition TOCCATA:

Il s'agit là des mêmes vérifications que celles exposées en V.3.4.4.

V.3.5.6 ET AINSI DE SUITE

L'opération de raffinage du comportement d'une unité de traitement se poursuit jusqu'à ce que soient identifiés des sous-comportements correspondant à des processus de commande.

Il n'y a pas de critère d'arrêt précis qui permette de juger du moment où sont définis les processus. Le concepteur constate simplement que la plupart des protocoles intervenant alors dans la modélisation sont élémentaires. Le nombre par ailleurs très faible de granules de comportement restant à raffiner lui indique que le problème est désormais à la portée de la réalisation.

V.3.5.7 VALIDATION DE L'AXE TEMPOREL

Une fois que s'achève l'analyse événementielle du comportement de l'UT, peut ensuite s'opérer la vérification de l'axe temporel du modèle de conception de l'application.

Cette vérification nécessite le déploiement, sur un même graphe d'état, de tous les liens temporels tracés entre sous-comportements de l'unité de traitement, de façon à former le modèle de processus du sous-système étudié. Elle a pour objet d'évaluer la pertinence de chacun des états relevés dans le modèle en comparant les résultats de la conception à la spécification initiale.

Elle donne, pour la cellule SDDS, le graphe des modes de fonctionnement détaillés suivants:

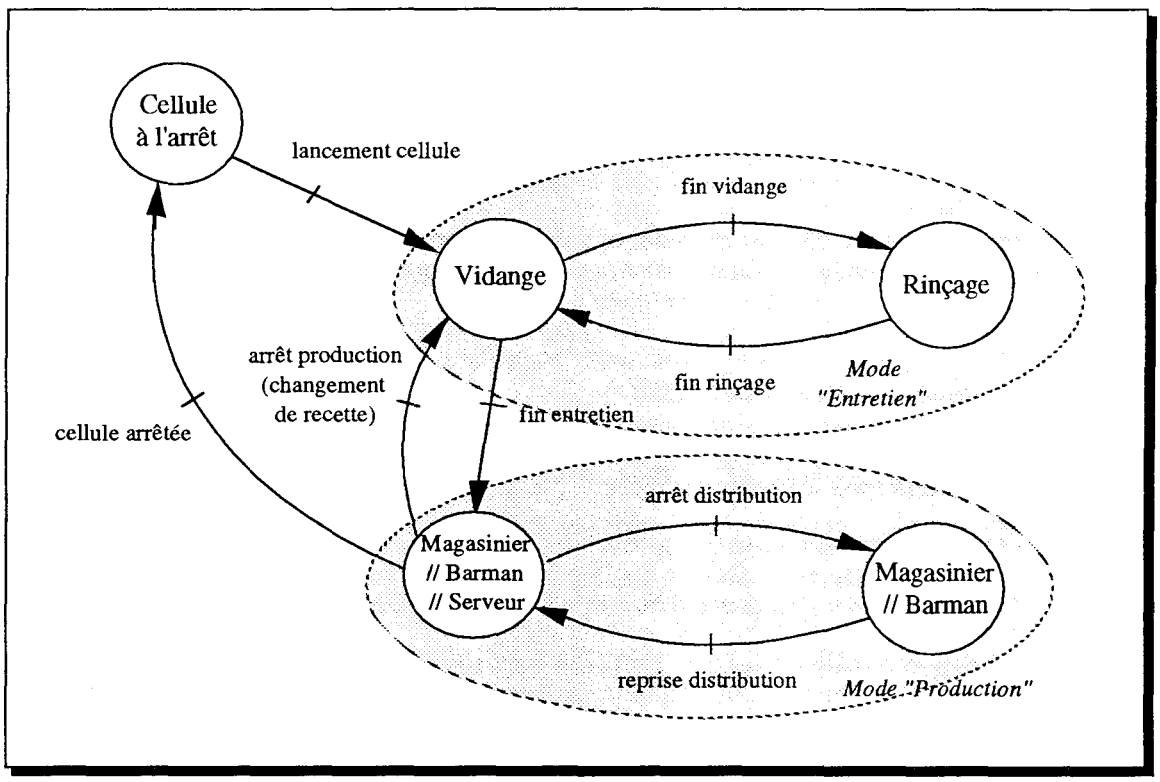


figure 93: Modes de fonctionnement détaillés de la cellule SDDS

L'arrêt momentané de la distribution du sirop fait en effet basculer le mode production dans un état latent qui n'invalide pas pour autant les processus "Magasinier" et "Barman".

On vérifie bien par ailleurs, dans la spécification opérationnelle de notre exemple, que cet arrêt n'est que momentané et permet, en cas de problème observé le tapis roulant, de ménager un certain temps à l'opérateur de la cellule afin de nettoyer les abords de l'emplacement de remplissage des bouteilles.

V.3.6 ETAPE 4: DESCRIPTION DES PROCESSUS DE COMMANDE

L'étape 3 nous a permis de définir les différents processus de commande de l'application. Ceux-ci réclament maintenant d'être détaillés sous leurs 2 aspects transformationnels et réactifs.

L'étape 4 commence tout d'abord par l'inventaire des types abstraits de commande permettant de concevoir le logiciel à l'aide d'éléments transformationnels réutilisables.

V.3.6.1 INVENTAIRES DES TYPES ABSTRAITS DE COMMANDE

Comme nous l'avons vu au chapitre III, le but recherché à travers la description distincte des aspects réactifs et transformationnels d'un processus est double:

- celui, d'une part, de déterminer l'organisation du programme exécutant le processus,
- celui, d'autre part, de concevoir le programme et ceux, à chaque fois qu'il l'est permis, à partir d'éléments génériques de commandes.

Pour atteindre ce deuxième but, il est souhaitable de commencer par passer en revue tous les processus de l'application pour en déterminer les besoins et en mesurer les éléments communs. Cette opération aboutit à la définition d'une série de types abstraits, référencés ou pas dans des bibliothèques déjà établies. (l'appartenance à une bibliothèque ne constitue qu'un mode de classement particulier des types abstraits, elle occasionne un gain de temps dans la recherche de ces derniers)

A ces types abstraits seront ensuite attachées des instances de machines abstraites supportant la gestion des interfaces propres aux différents processus identifiés dans le système.

V.3.6.2 VARIÉTÉS D'ÉLÉMENTS TRANSFORMATIONNELS GÉNÉRIQUES

Nous avions cité dans le chapitre III l'exemple de la régulation "tout ou rien" de température d'un four via sa résistance de chauffage et un capteur spécifique.
L'application de procédé de commande constitue un exemple de mise en œuvre d'un élément transformationnel générique. En effet, ce procédé de commande peut être reproduit plusieurs fois dans une même affaire, ou bien d'une affaire à une autre et mieux encore, d'un domaine d'application à un autre. C'est donc un objet de commande à fort potentiel de réutilisation qui possède en outre plusieurs champs d'application selon:

- qu'il s'adresse à tel ou tel type d'interfaces avec la partie opérative,
- ou intégre en sus d'autres types de fonctions en son sein.

L'objet réutilisable de base "Commande Tout ou Rien" se prête donc, au mieux de sa généricité, à la représentation ci-après:

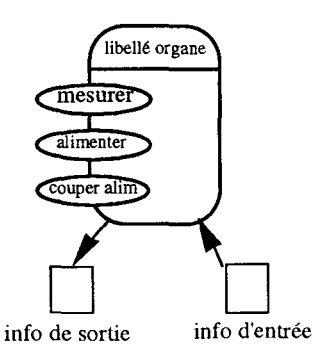


figure 94: Type abstrait de base

Et peut prendre, en considérant le premier axe de spécialisation cité page précédente (tel ou tel type d'interfaces), les trois formes suivantes:

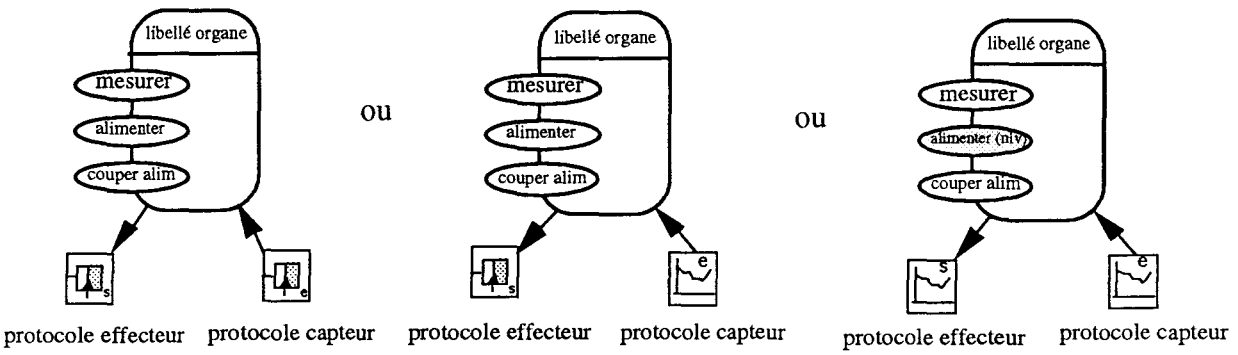


figure 95: Spécialisation d'un type abstrait

Dans le premier cas de figure, les 3 services de la machine se contentent d'acquérir ou de restituer des informations à la PO
Dans le second cas de figure, la machine effectue un pré raitement sur l'entrée capteur
Dans le troisième cas de figure, sont opérés un prétraitement sur l'entrée capteur et un post-traitement sur la sortie effecteur.

Aux variantes ci-dessus s'ajoute la possibilité d'adjoindre au type abstrait considéré d'autres types de fonction, comme par exemple ici la gestion de deux organes de même types en "Normal Secours":

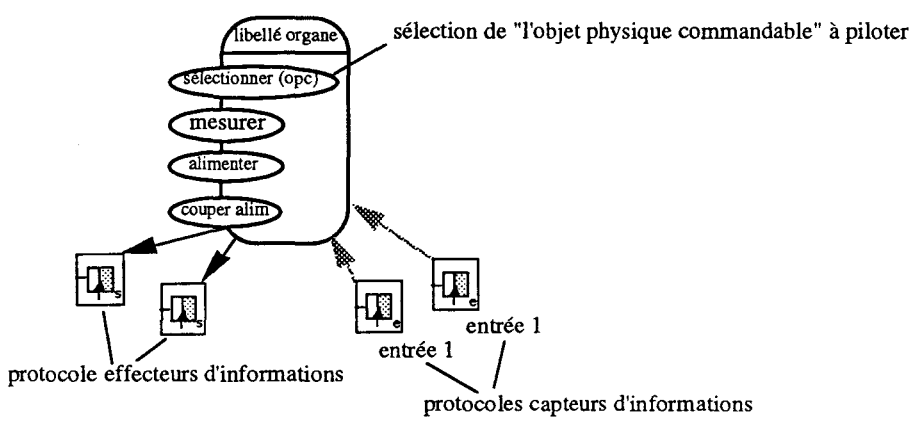


figure 96: Autre type de spécialisation

Ces types de machines abstraites ne font qu'intégrer des "architectures" de commande appliquées aux organes de la partie opérative.

Ils s'emploient à reproduire les procédures d'acquisition ou de restitution d'information (les actes de transformation) couplées à l'automatisation des processus. Il ne faut donc pas voir en eux le moyen de réaliser, au complet, tel ou tel sous-processus particulier (en considérant systématiquement la prise en compte de l'axe du temps) mais plutôt la manière d'opérer les commandes nécessaires à la mise en application d'une partition du procédé.

Il est néanmoins permis de sophistiquer ces commandes en introduisant une part de l'algorithme du processus dans ces machines.
Considérons par exemple la variante du type abstrait "Commande tout ou rien", évaluant d'elle-même le dernier seuil franchi par le fluide ou la température à réguler (seuil bas ou seuil haut).

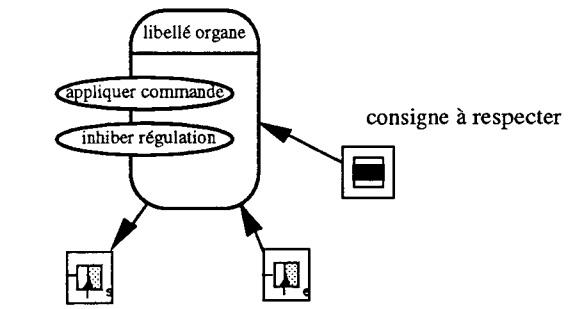


figure 97: Type abstrait "Régulateur Tout ou Rien"

Cette machine ne requiert alors que d'être régulièrement activée ou, mieux encore, activée au bon moment, c'est-à-dire lorsque l'un des seuils vient à être franchi par le signal témoin d'entrée de la régulation.
Nous verrons, dans le paragraphe V.3.6.4, comment faire correspondre un schéma d'implantation approprié à ce type de variante. Il va de soi qu'une telle machine abstraite réclame alors d'être accordée à un mode d'exécution particulier.

V.3.6.3 DESCRIPTION DE L'ASPECT TRANSFORMATIONNEL D'UN PROCESSUS

Une fois que sont listés les types abstraits précédents, sont ensuite instanciées des machines abstraites pour chacun des processus issus de l'analyse événementielle.

Correspond généralement à chaque organe de la partie opérative et pour chaque processus, une instance particulière de type abstrait. Cette instance encapsule les services ayant accès aux données de l'organe considéré et est soit appelée par le réacteur du processus, soit par une machine abstraite hiérarchiquement plus élevée qu'elle. (cf. III.4.1 et III.4.5)

La description reste simple tant que l'on respecte l'organisation de la partie opérative et en particulier les règles de composition dont émanent la définition des organes PO (composés d'objets physiques commandables) [CRU 91]; elle se complique inutilement dès que l'on s'éloigne de cette dernière.
S'il est donc un seul conseil à donner sur ce point, c'est bien celui de ne pas chercher à regrouper les ressources de l'application dans des machines abstraites ne manipulant pas les mêmes données et ceci dans le seul but de simplifier la représentation. Cette opération aurait pour conséquence de faire perdre à la solution recherchée son accessibilité à des éléments transformationnels génériques et contrarierait le processus de réutitisation cher à tout automaticien en mal de standardisation.

Les machines abstraites des 5 processus élémentaires de la cellule SDDS sont présentées à la fin de l'annexe 5. Le fait d'avoir traité cet exemple jusqu'à son terme montre bien comment des sous-procédés en apparence très distincts peuvent aboutir, en définitive, à la description de machines abstraites similaires en variétés restreintes.

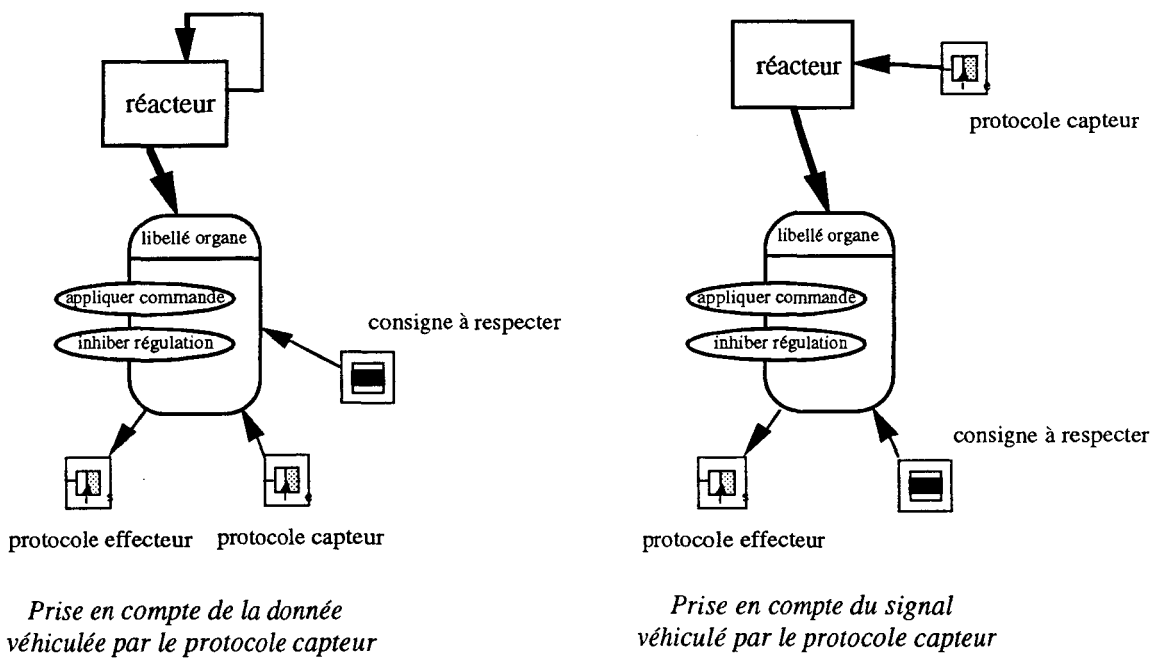
V.3.6.4 DESCRIPTION DE L'ASPECT RÉACTIF D'UN PROCESSUS

L'aspect réactif d'un processus s'emploie à représenter les informations témoignant de l'évolution de ce dernier. Le réacteur du processus correspond pour ce faire, avec les instances de protocoles caractérisant l'environnement de ces sous-systèmes élémentaires et véhiculant des informations soit fugitives, soit rémanentes.

De par leur caractère purement événementiel, les informations fugitives ne peuvent en effet qu'être prises en compte par un réacteur.

Représentant à la fois un signal et une donnée, une information rémanente aura dans certains cas, intérêt à passer par le réacteur du processus et dans d'autres cas, intérêt à être directement reliée aux machines abstraites qui lui font référence.
La configuration 1 privilégie l'aspect "signal" de l'interface. La configuration 2 prête au contraire attention à l'aspect "donnée" de l'information échangée.

Appliquons brièvement cette distinction à l'emploi de l'une des machines abstraites permettant la commande "tout ou rien" d'un organe de la partie opérative, par exemple la plus sophistiquée:
Celle-ci peut, du fait de la présence d'un protocole délivrant une information continue au processus, être cycliquement activée, ce qui, en vertu des deux modes d'exécutions les plus représentés dans le monde des automates (les modes 1 et 2 exposés en III.2.3), nous incite à relier le protocole capteur directement à la machine abstraite.
Celle-ci peut également, dans le cas où aucun protocole continu ne se trouve impliqué dans le processus, n'être activée qu'aux moments opportuns, en considérant alors le "signal" véhiculé par le protocole capteur.



Prise en compte de la donnée véhiculée par le protocole capteur

Prise en compte du signal véhiculé par le protocole capteur

figure 98: Distingo réacteur bouclé - réacteur non bouclé

Nous voyons ici que ce choix est conditionné par l'environnement opérationnel du processus.
Il conditionne lui-même les services à utiliser sur les machines abstraites de commande. (le service "mesurer" des machines présentées figures 95 et 96 est en effet inutile dans le deuxième cas)

C'est pourquoi les processus "Rinçage", "Magasinier" et "Serveur" emploient des instances de types abstraits spécifiques ne reproduisant pas ce type de service.

Présentation du processus "Magasinier":

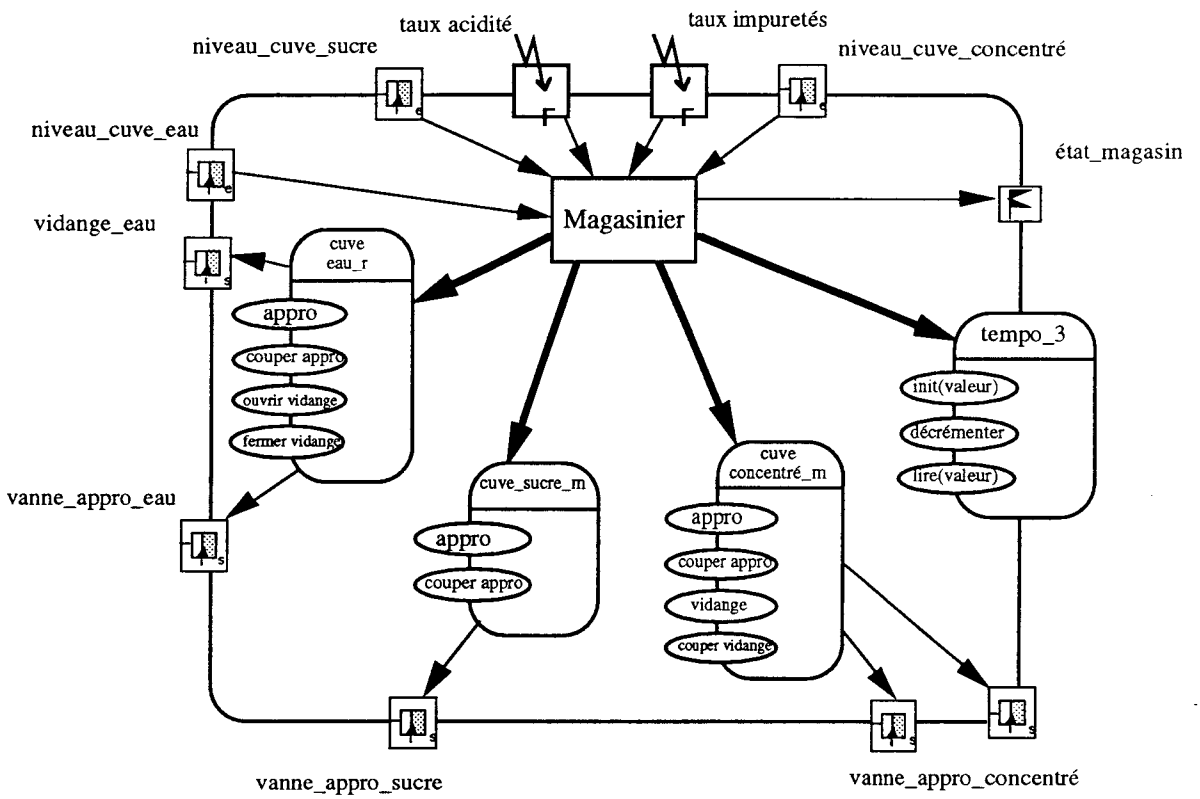


figure 99: Machines Abstraites du processus Magasinier

Dans cet exemple, le réacteur considéré fait appel, sur réception d'un signal de franchissement de seuil d'un niveau d'une cuve primaire, aux services de la machine abstraite appropriée disposant des commandes permettant la régulation tout ou rien de cette cuve.

Contrairement à la proposition de la figure 48 présentée à la fin du chapitre III, le corps du réacteur n'a pas ici à prendre cycliquement connaissance du niveau dans lequel se trouve l'information de retour de la boucle de la régulation, en comparant ce niveau à un seuil donné. Le franchissement du seuil active directement le réacteur qui fait alors appel aux machines abstraites de son choix.

Pour nous situer par rapport aux trois catégories de langages préposées en III.6.3 à la description détaillée des réacteurs de processus, nous nous plaçons ici théoriquement dans le troisième cas de figure, préférant à toute autre chose une description à l'aide d'un langage réactif.

V.3.7 QUELQUES PAS VERS L'IMPLANTATION

Notre but n'est pas ici de traiter de la conception détaillée ou du codage de l'application. Néanmoins, afin de mieux comprendre comment les objets issus de la phase présentée dans ce chapitre se traduisent par des éléments de logiciels programmés, la plupart du temps compilés puis "linkés" sur des unités de traitement de l'architecture logique, il paraît important de donner quelques précisions quant à l'utilisation des produits de sortie de cette phase.

En particulier, chaque processus se compose d'un réacteur donnant le corps d'un programme dans l'application.

Ce programme s'implante soit sur une tâche périodique offerte par le système d'exploitation de la machine de traitement (cas d'un réacteur bouclé), soit sur une tâche apériodique pilotée par des événements extérieurs.

Le réacteur exploite des machines abstraites se traduisant par des sous-programmes dont l'expression fait désormais appel une norme internationale (la norme CEI-1131 [CEI-92]).
Sur ce point, la traçabilité entre les éléments de conception et les éléments de réalisation a déjà évoquée en III.6.2. Il nous suffit donc d'y faire référence et de préciser toutefois que figure également dans cette norme des langages à "flots de donnée" incompatibles avec notre approche séparant les deux aspects réactifs et transformationnel d'un processus.
L'utilisation de ces langages (en particulier des langages à boîtes) sera donc à proscrire dans notre cas.

Il n'est fait par ailleurs aucunement référence, dans ce consensus international, aux langages réactifs qui s'inscrivent dans la perspective d'une exécution synchrone, telle que le propose intrinsèquement le mode 2 présenté en III.2.3.2.
Nous ne craindrons donc pas, à chaque fois que l'environnement de programmation ainsi que le processus le permettront, de nous écarter de cette norme pour profiter pleinement des avantages d'une telle approche.

L'alternative précédente ne concerne bien évidemment que les processus séquentiels.
Les processus continus verront leur programme s'implanter sur des tâches périodiques.
Les processus mixtes réclament idéalement un modèle d'exécution pseudo-périodique, comme celui retenu dans le mode 3 présenté en III.2.3.3. A défaut de pouvoir, sur la plupart des machines, utiliser ce dernier, (puisqu'il n'est que très rarement proposé), seul le mode 1 permet alors de les représenter.

V.4 CONCLUSION

L'originalité de la méthode proposée dans ce chapitre réside dans le fait qu'elle est basée sur une séparation de l'étude d'un sous-système en deux phases, respectivement consacrées à *l'analyse événementielle du comportement attendu de la commande d'un sous-ensemble de la partie opérative* et à *la caractérisation des éléments de logiciels permettant de définir un processus élémentaire dans ce sous-système.*

Ce phasage coïncide avec l'emploi consécutif de deux démarches, descendante puis ascendante, qui confèrent au processus de conception, la complémentarité exigée de toute bonne méthode d'analyse.

Il permet, en outre, au moment adéquat (c'est-à-dire lors de la phase ascendante), de composer l'application en envisageant la réutilisation d'éléments génériques de commande s'apparentant à des types abstraits d'automatique.

La méthode tire profit de l'expérience acquise dans l'élaboration de logiciels à fortes contraintes de temps en adaptant la technique d'organisation de la conception TOCCATA, et plus particulièrement la sémantique des liens opérationnels entre sous-comportements déduits de cette analyse, au domaine des systèmes d'automatisme.
Le respect, dans cette méthode, de la séparation entre l'étude des aspects réactifs et transformationnels des processus de commande se porte enfin en faveur d'une plus grande généricité des composants entrant dans la conception de ces processus.

VI CONCLUSION GÉNÉRALE

Nous avons pu constater que la particularité d'une méthode de conception d'un Système Appliqué de Contrôle-commande tient au fait qu'elle appuie la solution recherchée sur des éléments de structuration génériques en nombre fini pour un domaine donné.

Il faut donc, pour élaborer ces systèmes particuliers, retenir de leurs spécificités, des éléments susceptibles d'être exprimés par des solutions-types plus ou moins agrégées. Ces solutions sont symbolisés, dans cet ouvrage, par des catalogues de produits d'automatisme, intégrant les constituants d'une même famille de composants matériels. Les constituants ainsi répertoriés sont ensuite mêlés à un processus de réutilisation qui s'appuie sur un mode d'organisation astucieux des catalogues produits tirant profit des principaux concepts introduits dans la norme française Z99-001.

Cependant, la méthode de conception du système va bien plus loin que la définition de l'architecture matérielle du SCC puisqu'est également proposé de réutiliser des éléments logiciels de contrôle-commande génériques pour élaborer l'application recherchée.

Finalement, dans son souci de réutilisation, le concepteur est amené à identifier une couche support (1er candidat à la réutilisation), qui, dans l'attente d'une uniformisation des plateformes d'accueil pour logiciels d'automatisme, ne peut être définie qu'à partir du choix prélable d'un ensemble matériel.

La première grande originalité de la démarche proposée dans ce mémoire réside dans la continuité offerte entre la conception du matériel et la conception du logiciel d'un SCC.
La deuxième grande originalité de la démarche est dans la manière d'appréhender la modélisation d'un système, qui distingue dans celui-ci, ses architectures fonctionnelle, logique et physique et s'appuie sur un modèle de représentation de son logiciel dérivé des concepts de la méthode TOCCATA de CEGELEC, actuellement fort appréciée pour la conception d'applications à fortes contraintes de temps. Il nous a fallu en particulier proposer une nouvelle classification des protocoles utilisés dans cette méthode pour lui permettre de supporter l'analyse du comportement attendu d'un système pouvant admettre toutes sortes de caractéristiques temporelles.

La force de TOCCATA réside en grande partie dans la séparation des étapes d'analyse et de synthèse d'un problème posé, en distinguant l'étude des comportements d'un système, donnant l'aspect réactif de celui-ci, de l'organisation des composants dont émane l'aspect transformationnel des processus ainsi décrits.
Nous avons associé à ce fait la possibilité de réutiliser, de façon systématique, des constituants transformationnels peu dépendants de l'application à étudier et de réemployer des composants réactifs qui sont eux, plus spécifiquement dédiés au système en cours d'élaboration.

La démarche entreprise dans l'étude ayant donné lieu à la rédaction de ce mémoire a de plus permis d'identifier les principaux critères d'évaluation de la généricité d'une application en distinguant ses composants caractéristiques et leur appartenance à un domaine précis de modélisation du système. C'est en effet en recoupant ces deux points de vue que peuvent être objectivement caractérisés les éléments d'un SCC.
La modélisation conceptuelle des systèmes permet donc, non seulement de fixer les règles attachées à sa représentation informationnelle, mais surtout d'en déduire des spécificités qu'une analyse fonctionnelle ou organique classique ne peut montrer. C'est en appliquant en particulier la méthode NIAM [NIJ 81] et en transcrivant ses résultats dans un formalisme plus simple (le formalisme Entité/Relation Etendu) qu'ont pu être obtenus la plupart des concepts ne découlant pas directement de l'observation d'un système opérationnel.

Notre modélisation est compatible avec celle proposée dans la norme BasePTA, pour laquelle nous avons fortement œuvré et qui concerne les données d'un système implanté. Elle peut également être rapprochée des concepts de la récente norme internationale CEI 1131 qui, dans sa partie n°3, traite de la programmation d'un logiciel d'automate programmable.

Elle trouve application dans des projets en cours d'élaboration au sein de CEGELEC qui ont été initiés par la réalisation d'une maquette d'atelier appelé ADAM sensibilisant les concepteurs aux problèmes d'ingénierie système et à la gestion des activités relatives à un projet.

La méthode et les principes exposés dans cet ouvrage sont en grande partie repris dans la version 2 de la méthodologie MODAL, développée par CEGELEC, s'attachant à proposer des procédures, des méthodes, des guides et des règles de rédaction de documents liés aux différentes phases de développement d'un système de contrôle-commande industriel.

Un gros travail reste donc à faire, celui de promouvoir les concepts et principes exposés en substance, dans les chapitres précédents, et les appliquer à la mise en œuvre d'outils d'ingénierie et de conception liés au secteur contrôle industriel.

ABBRÉVIATIONS

AGA: Atelier de Génie Automatique

BasePTA: Base et standard d'échange pour le Poste de Travail de l'Automaticien

API: Automate Programmable Industriel

AR: Automate Régulateur

CIM: Computer Integrated Manufactering

ETF: Ensemble Topo-Fonctionnel

MC: MicroCalculateur

RAO: Réponse à l'appel d'offre

SAP: Système automatisé de production

SCC: Système de contrôle-commande

TOCCATA: Technique d'Organisation de la Conception par Comportements Abstraits et Types Abstraits

UT: Unité de Traitement

TERMINOLOGIE

Méthodologie: Agrégation de méthodes, de modèles et de techniques éprouvés

Méthode: Ensemble de règles bien définies qui conduisent, pour un problème, à la définition d'une solution correcte

Modèle: Interprétation explicite de la compréhension d'une situation, en termes d'entités et de relations entre elles.

ANNEXES

ANNEXE 1 - LE FORMALISME ENTITÉ RELATION ÉTENDU

Le formalisme Entité/Association s'appuie sur deux concepts de base: l'Entité et la Relation (cf. figure a1). Lors de la phase d'analyse d'un système, ces notions sont en effet, naturellement perçues.

A1.1 ENTITÉ

Une entité représente un objet ayant un intérêt pour le système à formaliser. Certaines entités sont concrètes; elles désignent, par exemple, des objets connus (c'est le cas d'un module électronique ou d'un bac). D'autres sont abstraites et peuvent désigner des concepts comme, par exemple, un repère site ou un équipement logique. Chaque entité est caractérisée par un certain nombre d'attributs.

A1.2 RELATION

Ce concept exprime des liens sémantiques pouvant exister entre les entités. Comme les entités, chaque relation peut porter des attributs la caractérisant. Les relations sont par ailleurs binaires, c'est-à-dire que dans une relation, n'interviennent que deux entités.

A1.3 CLASSES D'ENTITÉS ET CLASSES DE RELATIONS

Les entités (et les relations) de même nature, ayant une même structure et partageant une même définition, sont regroupées en classes ou types d'entités (et de relations). Chaque membre d'une classe est nommé occurrence ou encore instance de cette classe.

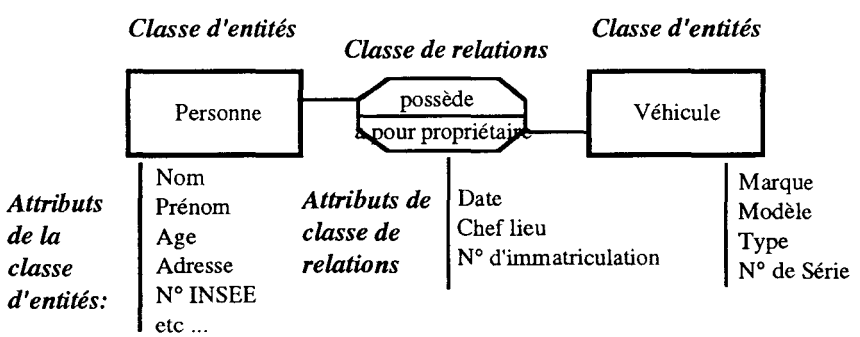


figure a1

A1.5 ATTRIBUTS

C'est une donnée élémentaire décrivant une entité ou une relation. Chaque Attribut a un libellé et un type précisant la nature des valeurs associées. Parmi les attributs associés à une entité, un ou plusieurs attributs caractérisent l'unicité d'une instance de cette entité. On parle alors dans ce cas d'identifiants qui sont bien évidemment obligatoires. A titre d'exemple, l'attribut "N° INSEE" associé à l'entité personne de la figure a1 constitue son identifiant absolu. D'autres identifiants secondaires caractérisent cette personne comme son nom et son prénom.

A1.6 TYPES DE RELATIONS

Dans le formalisme utilisé, nous avons retenu des relations, un typage plus fin que celui prévu initialement dans la modèle Entité/Association de base [CHE 76] afin de renforcer son aptitude à prendre en compte, d'une manière aussi fidèle et complète que possible, la sémantique du réel à modéliser. C'est la raison pour laquelle nous avons appelé ce modèle "Entité/Relation Etendu".
Dans des applications telles que celles que nous prétendons modéliser, de fortes relations structurelles apparaissent entre les objets et induisent des contraintes de dépendance listées ci-après.

A1.6.1 SPÉCIALISATION / GÉNÉRALISATION (RELATIONS ENTRE CLASSES)

Cette relation vise la factorisation des connaissances et le partage de la définition d'un type d'entités entre plusieurs classes. Cette définition est exprimée au niveau d'une classe générique appelée "super-classe" ou "classe générale". Elle traduit les notions de sous-types et sur-types d'entités, comme le montre la figure ci-dessous:

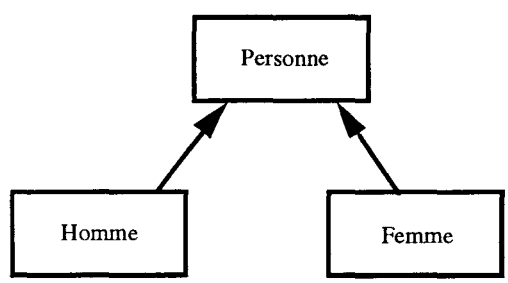


figure a1.2

Un Homme est en effet une personne, au même titre qu'une femme.

A.1.6.2 AGRÉGATION (RELATION ENTRE INSTANCES D'ENTITÉS)

Une relation de type Agrégation est une relation structurelle forte exprimant une sorte de dépendance entre deux instances de deux classes d'entités. Sur le plan sémantique, le cas le plus fréquent est celui de l'appartenance à un objet ou à un autre. On parle plus communément d'objets Agrégeant et Agrégés dans cette configuration particulière.

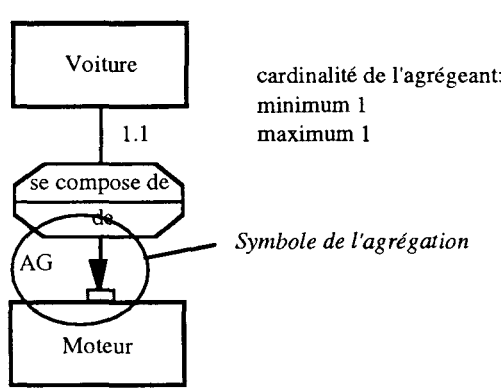


figure a1.3

Par rapport à une relation classique, l'agrégation a les propriétés suivantes:

- un élément agrégé n'est agrégé qu'à un seul agrégat. Un agrégat peut bien évidemment agréger plusieurs éléments,
- l'existence d'un élément agrégé implique celle de son agrégat. Par conséquent, la suppression d'un agrégat conduit à la suppression de ses éléments agrégés.
- un élément agrégé est identifié par rapport à son agrégat et par rapport à lui seul.

A1.7 CARDINALITÉS

La cardinalité d'une relation définit le minimum et le maximum d'occurrences de la relation, par rapport à une instance d'entité:

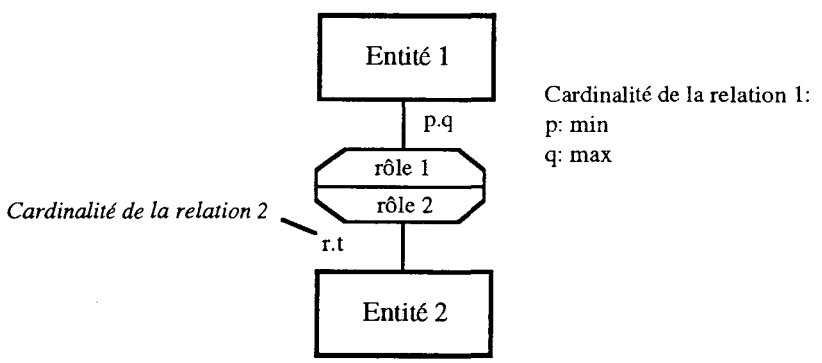


figure a1.4

On donne habituellement l'une des cardinalités suivantes à chacun des rôles d'une association:

- (0.1) une instance de l'entité ne participe ne peut participer plus d'une fois à la relation
- (1.1) une instance de l'entité toujours une et une seule fois à la relation
- (1.N) une instance de l'entité participe toujours au moins une fois à la relation
- (2.N) une instance de l'entité participe toujours au moins deux fois à la relation
- (0.N) une instance d'entité peut participer, ou pas, autant de fois que nécessaire à la relation

Dans certains cas particuliers, il se peut qu'une relation d'ordre relativement générale, doivent donner plusieurs associations distinctes lors de l'implantation, en respectant toutefois ce sens général et bien évidemment les mêmes règles de cardinalités. On parle alors d'association avec libellé paramétrable, en extension d'une association classique où, à priori, les libellés des deux rôles qui la compose, sont figés.

La notation de cette extension au formalisme de base Entité/Association est la suivante:

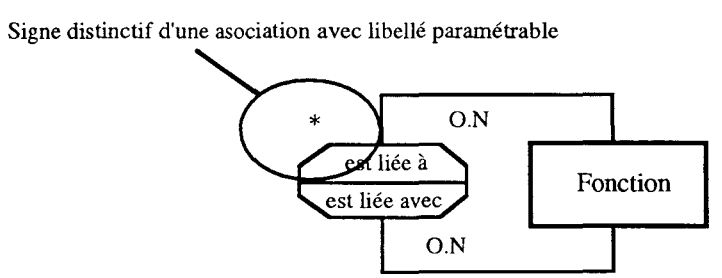


figure a1.5

A1.8 CONTRAINTES

Le modèle Entité/Association ne distingue à la base que des entités et des associations (assemblage de deux rôles distincts). Nous avons préconisé dans ce mémoire l'emploi d'un formalisme étendu autorisant le positionnement de contraintes à la fois sur des entités et sur des relations.

A.1.8.1 CONTRAINTES ENTRE ENTITÉS

Les objets d'un modèle conceptuel peuvent être en effet liés par un certain nombre de contraintes qui témoignent à la fois de leur caractéristiques propres et de l'exhaustivité de la description.

Prenons par exemple la relation de sous-type de la figure a1.2; il est évident qu'un homme ne peut être une femme et inversement. Il y a donc exclusion entre les deux sous-classes "Homme" et "Femme", exclusion qui se note de la manière suivante:

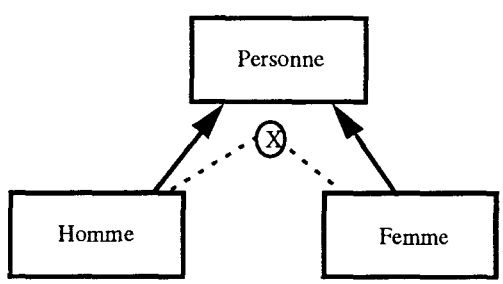


figure a1.6: Exclusion de sous-types

Par ailleurs, la description de l'exemple prend aussi ici un caractère exhaustif puisqu'un individu ne peut être que de sexe masculin ou féminin. Il y a donc, de surcroît, totalité entre les deux sous-types précédents. Une nouvelle contrainte peut alors figurer sur le modèle ci-dessus et compléter ainsi la définition de ces concepts:

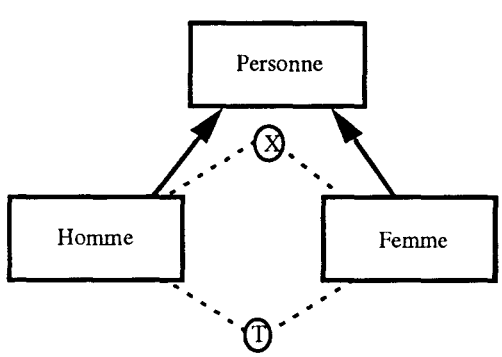


figure a1.7: Contrainte de Totalité entre sous-types

A1.8.2 CONTRAINTES ENTRE RELATIONS

Les relations entre entités peut également être liées entre elles par des contraintes. Il est par exemple utile de signaler qu'une entité ne peut participer à la fois à deux relations pourtant spécifiées au niveau de sa propre classe (sur le schéma conceptuel).

Nous noterons dans ce cas cette contrainte de la façon suivante:

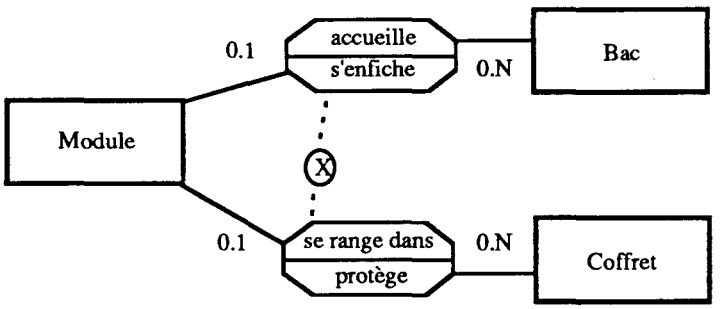


figure a1.8: Exclusion entre phrases

On parle ici d'exclusion entre phrases puisque la suite (entité 1 - relation - entité 2) forme une phrase (dans une grammaire régulière) comportant respectivement un sujet suivi d'un verbe (le prédicat rôle 1 ou rôle 2) et d'un complément.

D'autres contraintes sont évoquées dans les différents schémas présentés dans cet ouvrage, comme par exemple la totalité entre phrases qui se note de la même façon que ci-dessus, en remplaçant le symbole "X" par la symbole "T". Cette contrainte donne le sens suivant à la description de la figure a1.8: "n module doit être soit rangé dans un coffret, soit enfiché sur un bac".
Les cardinalités 0.1 sont donc toujours valable mais ne peuvent dans cas être considérées que séparément.

Contrainte procédurale:

Certaines contraintes entre relations occasionnent des traitements. Elles doivent tout de même être signalées au même titre que l'est fait une contrainte sur type (classe d'entités) ou une contrainte occurrentielle.
Nous les symboliserons par un "P", en remplacement du "T" ou du "X" précédents.

Exclusions entre deux phrases:

Notre modèle présente, dans son aspect matériel notamment, ce type de contrainte qui témoigne de l'exclusion complète entre deux phrases:

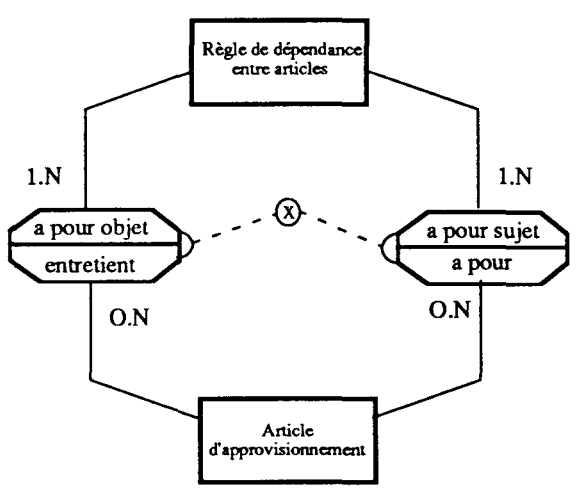


figure a1.9

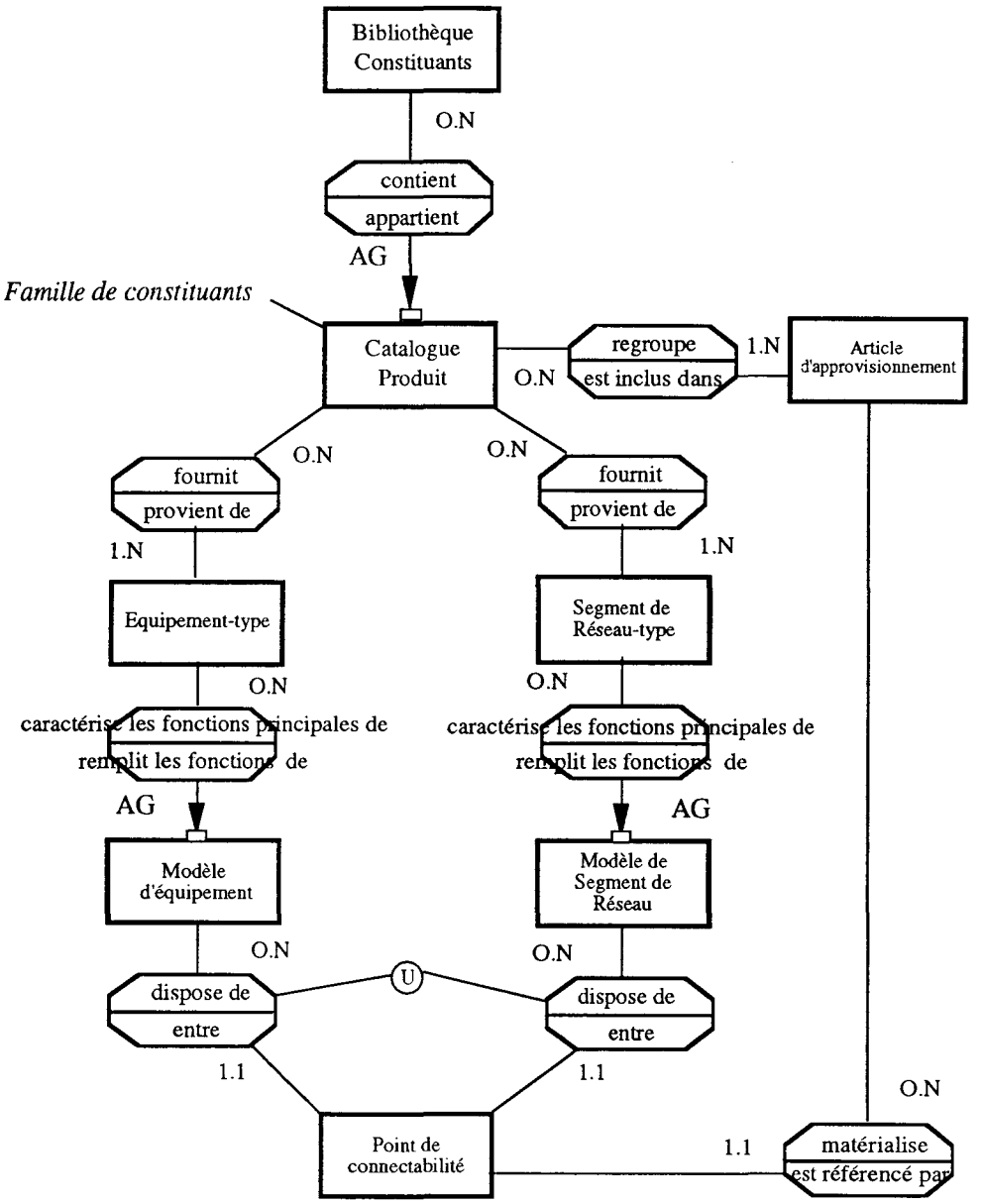
Cette exclusion ne peut toutefois porter que sur des relations engageant les mêmes objets.

La description qui est faite ici n'est pas exhaustive. D'autres configurations peuvent être envisagées mais toutes ne sont malheureusement pas licites. Attention donc aux erreurs conceptuelles occasionnées par un usage abusif de ces contraintes !

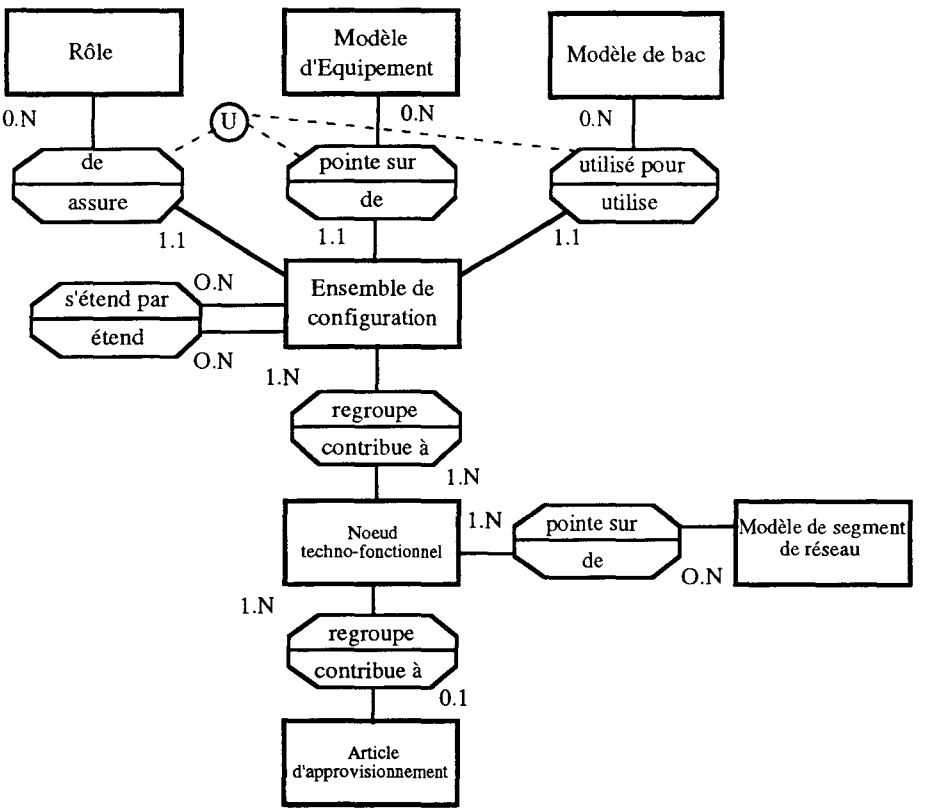
Pour plus d'informations sur le modèle Entité-relations et ses extensions possibles, se reporter à [CHE 76] et [NIJ 81].

ANNEXE 2 - BIBLIOTHÈQUES DE CONSTITUANTS MATÉRIELS

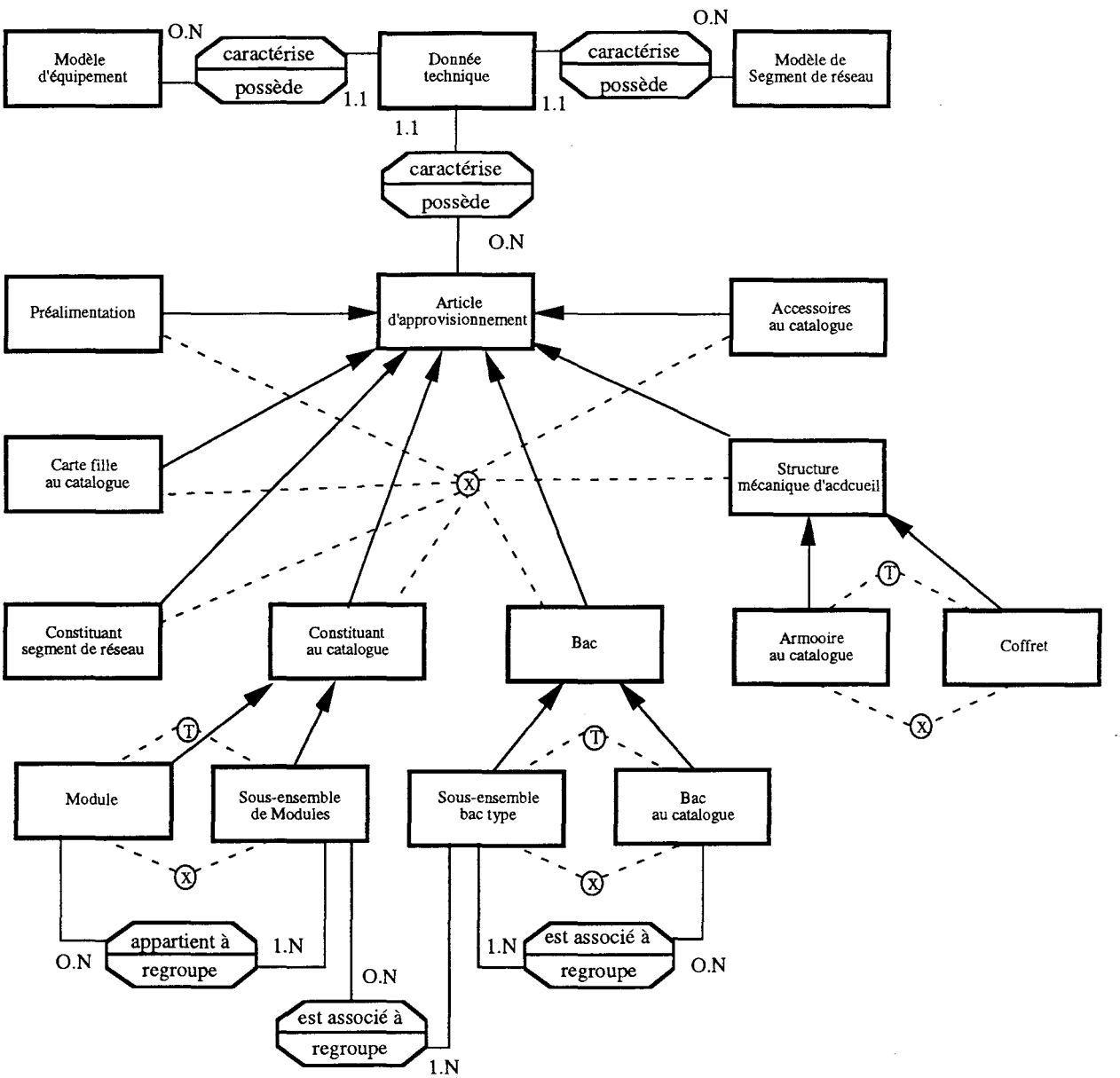
A2.1. SCHÉMA CONCEPTUEL D'UNE BASE CONSTITUANT



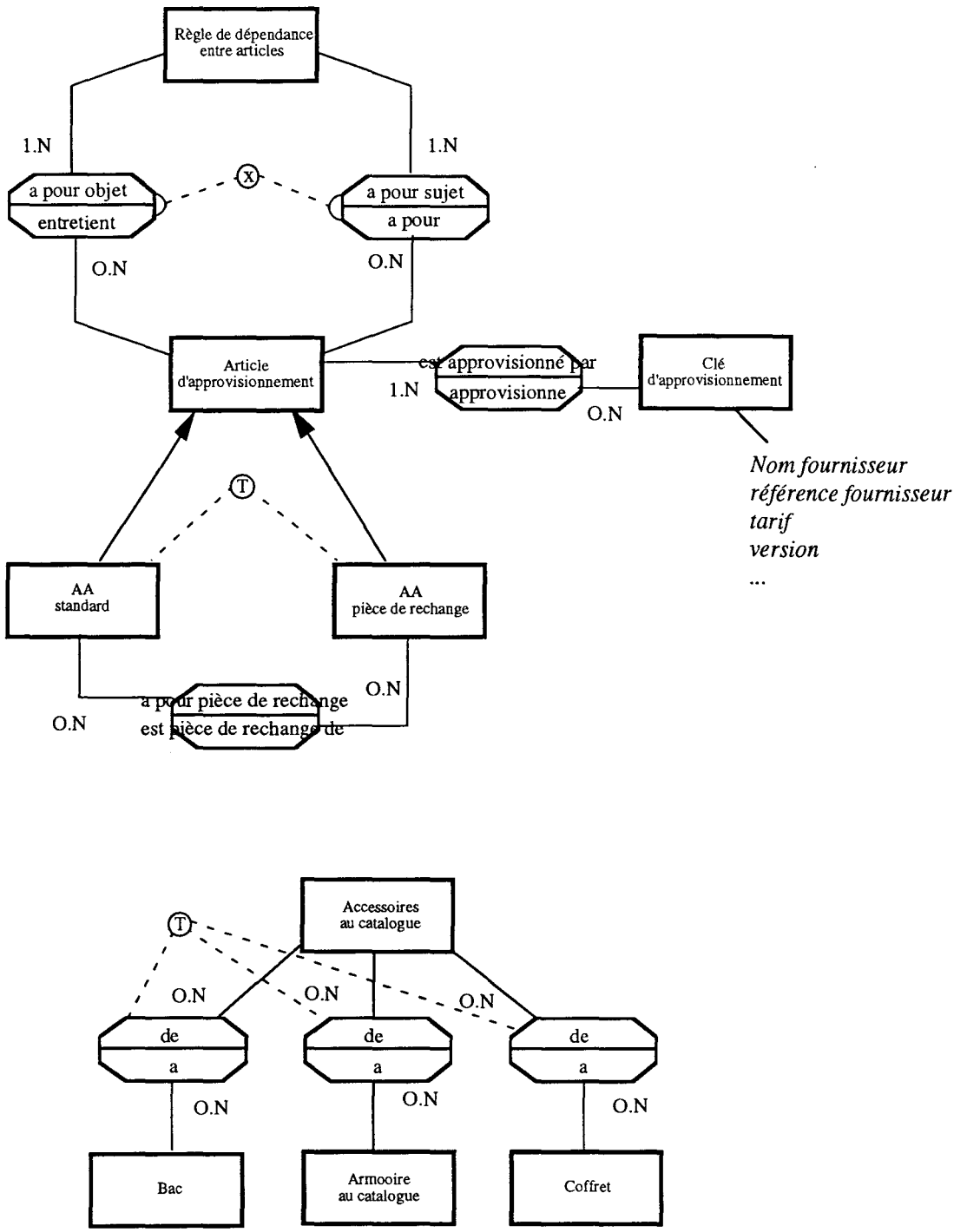
Catalogue produits - Types et modèles de produits



Ensembles de configuration - Noeuds Techno-fonctionnels



Types d'articles d'approvisionnement



Règles de dépendances entre articles (modules en particuliers) - clés d'approvisionnement

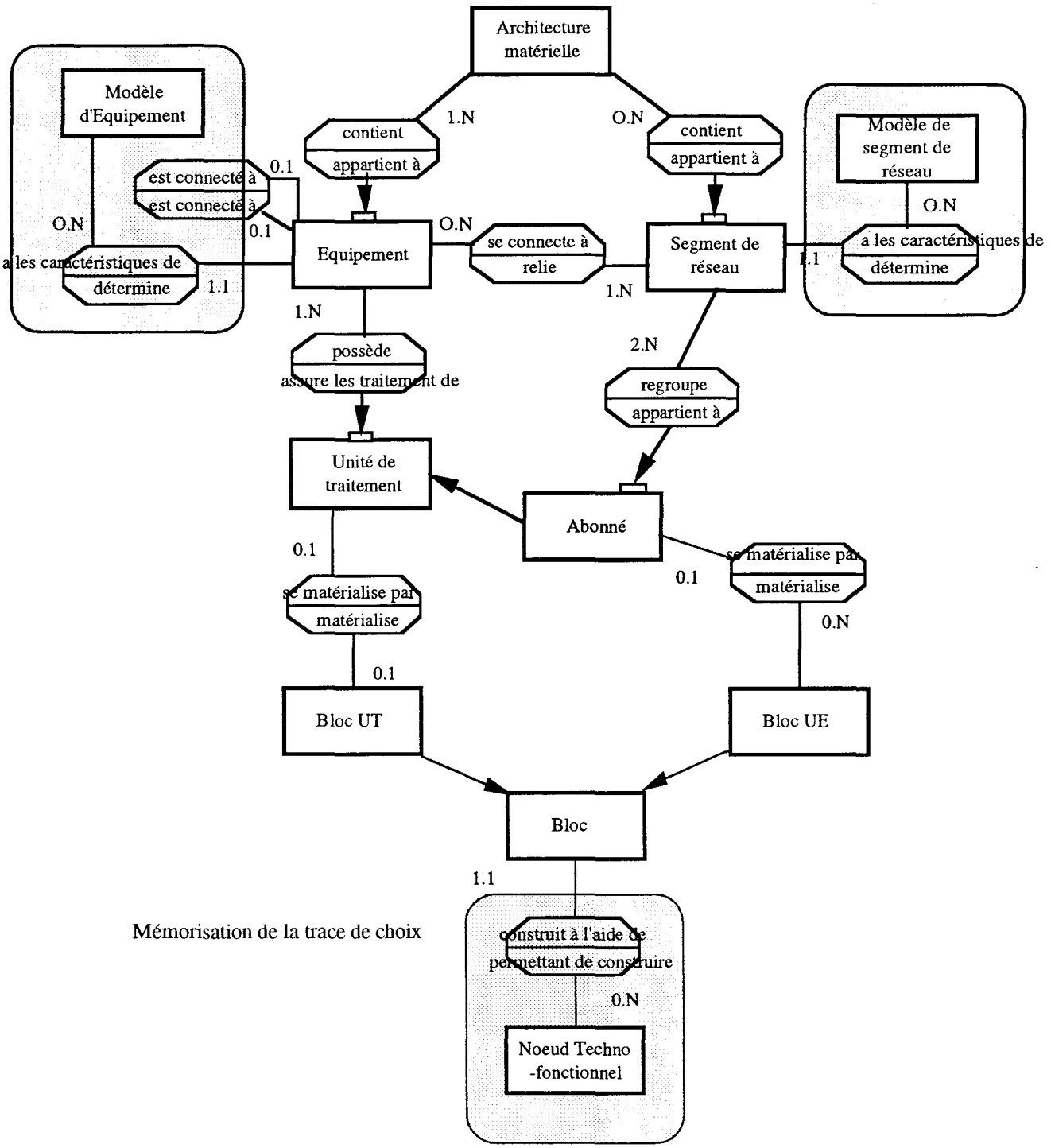
A2.2. PRÉSENTATION DE 2 FAMILLES DE CONSTITUANTS

Catalogue produits S	*Catalogue CE2000*	*Désignations*
SE 02		(S 251 + S 802A + UT 120)
SE 03		(2* S251 + S821 + 2*UT120 +2* UT132 + CR278)
S 802		Bac UT standard du C370
S 802A		Bac UT avec poutre insert CEM
S 803		Bac bi-UT (C370 + S600)
S 804A		Bac UT alim renforcée
S 804B		Bac UT alim renforcée avec poutre insert CEM
S 821		Bac UT double C370
S 822		Bac Bi-UT C370 fixation arrière
AL 122		Module d'alimentation 230/5-12-24 V
UT 120		Unité de traitement (UT121 + UT122 + PM112 + PM113)
UT 121		Processeur de tests
UT 122		module ALU
PM 112		module mémoire de l'UT, lié à UT122
PM 113		carte-fille de PM112 - mémoire programme
PM 114		carte-fille de PM112 - gestion de fichiers
UT 123		Réseau interne normal/ secours
PM 117		carte-fille de PM113 (2*64 ko EEPROM + 1* Mo RAM)
PM 118		carte-fille de PM113 (2*64 ko EEPROM + 2 Mo RAM)
PM 119		fille de PM113 (2* 64 ko EEPROM)
PM 120		fille de PM113 (64 ou 2*32 ko EEPROM)
S 851		Bac de préalimentation 220/24V 5A
S 852		Bac de préalimentation 220/24V 10A
S853		Bac de préalimentation 220/24V 20A
S 854A		Bac de préalimentation 115/220/24V 15A
S 855		Bac de préalimentation 24=/24V 10A
S 856		Bac de préalimentation 48=/24V 10A
S 859A		Bac de préalimentation 24=/5V 10A
S 251		Tête de bac UT
S 270		Unité d'échange N900
S 752		modem N900
S 763		boîte de raccordement N900
S x61		Cordon de raccordement S752, S763
S 933		Bretelle de raccordement S270, S752
S 930		T de raccordement N900
S 925		Charges externes N900
IR 129		Unité d'échange FIP
CR 128		Raccordement lien R
CR xx2		Guirlande de raccordement locafip
	AM 127	Bac CE2000
	AM126	Bac alimentation

	AL 119	module alimentation 5V - 30A
	AL 120	module alimentation 12V - 12A
	UT 129	Unité de traitement CE 2000
CR 298		Cordon de Raccordement alimentation
	LE 108	16 entrées TOR 48V
	LC 105	16 sorties logiques 48V avec point commun
	LD 106	16 sorties lampes 48 V
	AB 120	8 entrées ANA - thermocouples
	AB 121	8 entrées ANA - sondes à résistance
	AH 115	8 entrées ANA de haut niveau
	AH 111	8 sorties ANA de haut niveau
	AS 112	4 entrées-sorties MCA

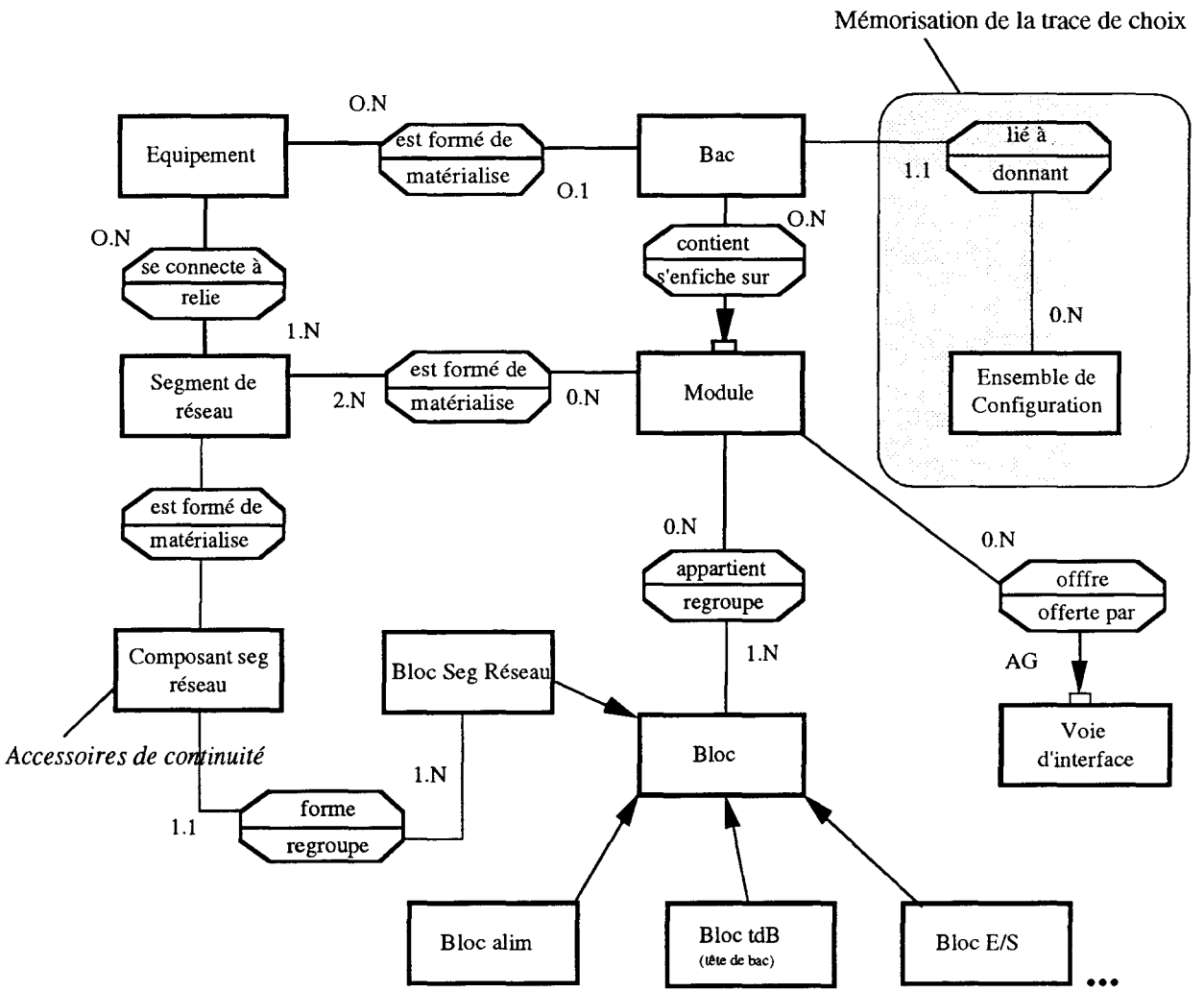
ANNEXE 3

MODÈLE CONCEPTUEL DE L'ARCHITECTURE MATÉRIELLE D'UN SCC APPLIQUÉ

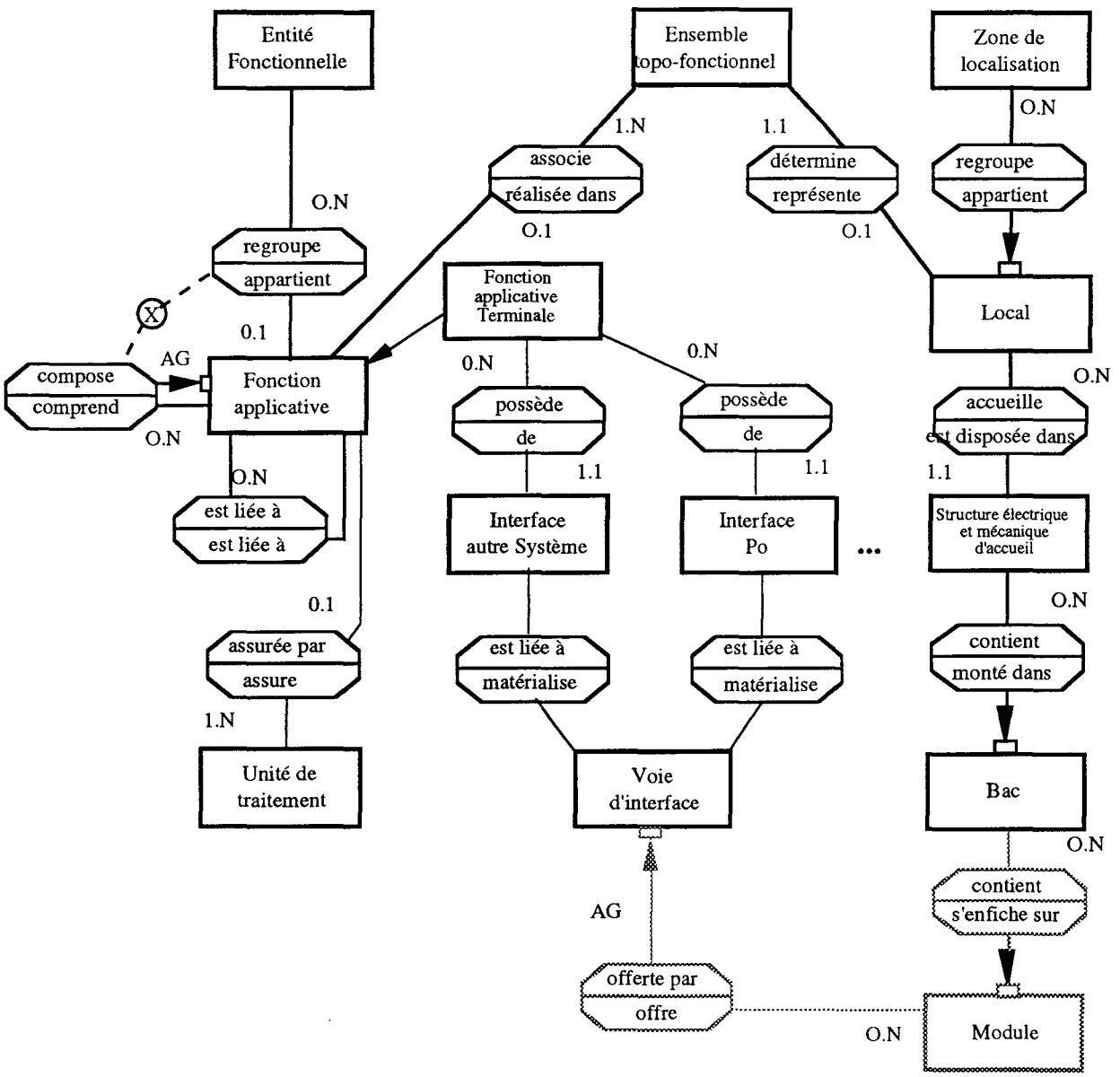


Mémorisation de la trace de choix

Architecture logique du SCC

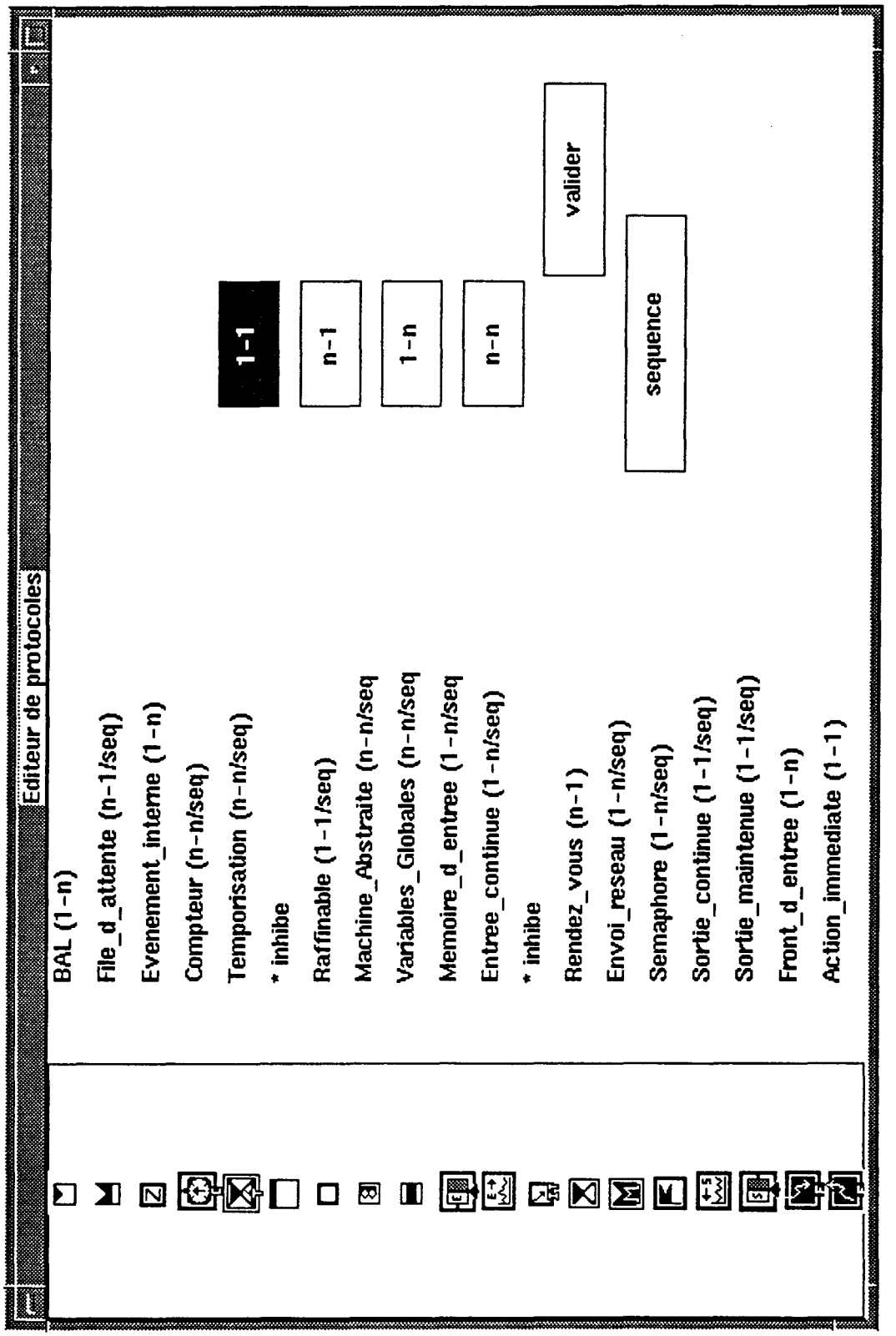


Architecture matérielle



Architecture Site - Arborescence Fonctionnelle

ANNEXE 4: DÉFINITION DES PROTOCOLES DE BASE TOCCATA



ANNEXE 5: SPÉCIFICATION ET CONCEPTION DE LA CELLULE DE PRODUCTION SDDS

A5.1 SPÉCIFICATION DU SOUS-SYSTÈME

A5.1.1 DESCRIPTION DU PROCÉDÉ

Le procédé de dosage et de distribution de sirop qu'est chargée de réaliser la cellule de production SDDS consiste à:

- approvisionner la cellule en ingrédients de base en maintenant un niveau suffisant dans les cuves primaires de cette dernière,
- réaliser le mélange de ces ingrédients, dans les proportions indiquées par la recette associée à chaque type de concentré de base en dosant avec précision et de façon continue, le remplissage de la cuve du mélangeur,
- chauffer le mélange à une température variant en fonction de la cadence d'arrivée des bouteilles sur le tapis roulant (qui donne la quantité de mélange à préparer par unité de temps),
- mélanger les ingrédients de base en adaptant la vitesse de rotation de l'agitateur également en fonction de cette cadence,
- servir les bouteilles se présentant sous l'orifice de remplissage dès leur apparition, à condition que le mélange soit, à la fois à température indiquée et en quantité suffisante dans le mélangeur.

A partir de l'information de débit qui a comme butée supérieure, le débit correspondant à une ouverture maximale de la vanne de distribution d'eau, sont déduites les quantités de mélanges instantanés en concentré et en sucre.

Le sous-système de contrôle commande étudié procède au contrôle du taux d'impureté de l'eau du mélange en analysant cette eau directement dans la cuve d'approvisionnement de cet ingrédient de base. Le sous-système commande, en cas de taux excessif, la vidange de la cuve, en interrompant provisoirement le dosage du sirop dans le mélangeur. Il est procédé, de façon similaire, au contrôle du taux d'acidité du concentré primaire.

Enfin, le concentré de base est déjà sucré en faible proportion et sa teneur en sucre est amené à varier au cours de l'approvisionnement. Il est donc tenu compte du taux de sucre mesuré dans le mélangeur à l'aide d'un capteur spécifique pour rectifier, ci-besoin, la proportion du dosage initialement donnée par la recette.

A5.1.2 MODES OPÉRATOIRES

Le SDDS a la charge de commander et contrôler le nettoyage des cuves avant tout changement de recette. Il gère aussi les changements de recette en vidangeant les différentes cuves de l'installation et préparant le mélange associer à tout nouveau concentré. Cette opération également est valable pour l'application d'un premier dosage, en relançant la processus après une interruption plus ou moins prolongée.

Les opérations à effectuer sont consécutivement les suivantes:

1) Vidanger chacune des 4 cuves
2) Rincer ces dernières
3) Vidanger à nouveau l'installation
4) Lancer l'approvisionnement automatique des cuves primaires
5) Une fois que le niveau de ces trois cuves a atteint leur seuil minimum, autoriser l'application de la recette, en considérant une cadence minimale d'arrivée des bouteilles sur la machine transfert
6) Entreprendre le remplissage de ces bouteilles dès leur apparition sous la goulotte du mélangeur en respectant les deux conditions notifiées en A5.1
7) Sur demande d'arrêt de distribution d'un opérateur de conduite, interrompre le processus de remplissage des bouteilles et maintenir le mélange prêt en attendant la reprise de distribution
8) Sur demande d'arrêt de la production, stopper l'application de l'ensemble du procédé et procéder à un changement de recette en nettoyant préalablement les cuves de la cellule
9) Sur demande d'arrêt de le cellule, couper totalement cette dernière.

Nota:

- La commande "arrêt cellule" fait office d'arrêt d'urgence sans reprise possible du processus.

- Afin de rincer les conduits de distribution en ingrédients de base, la vidange des cuves primaires en phase de nettoyage s'effectue en actionnant les vannes servant au dosage du sirop.

A5.1.3 DESCRIPTION DES INTERFACES

La liste ci-après décrit les interfaces entre le SCC et la partie opérative et le SCC et les opérateurs de conduite, que les commandes soient, dans ce dernier cas, indifféremment émises depuis le local de production ou en salle de commande.

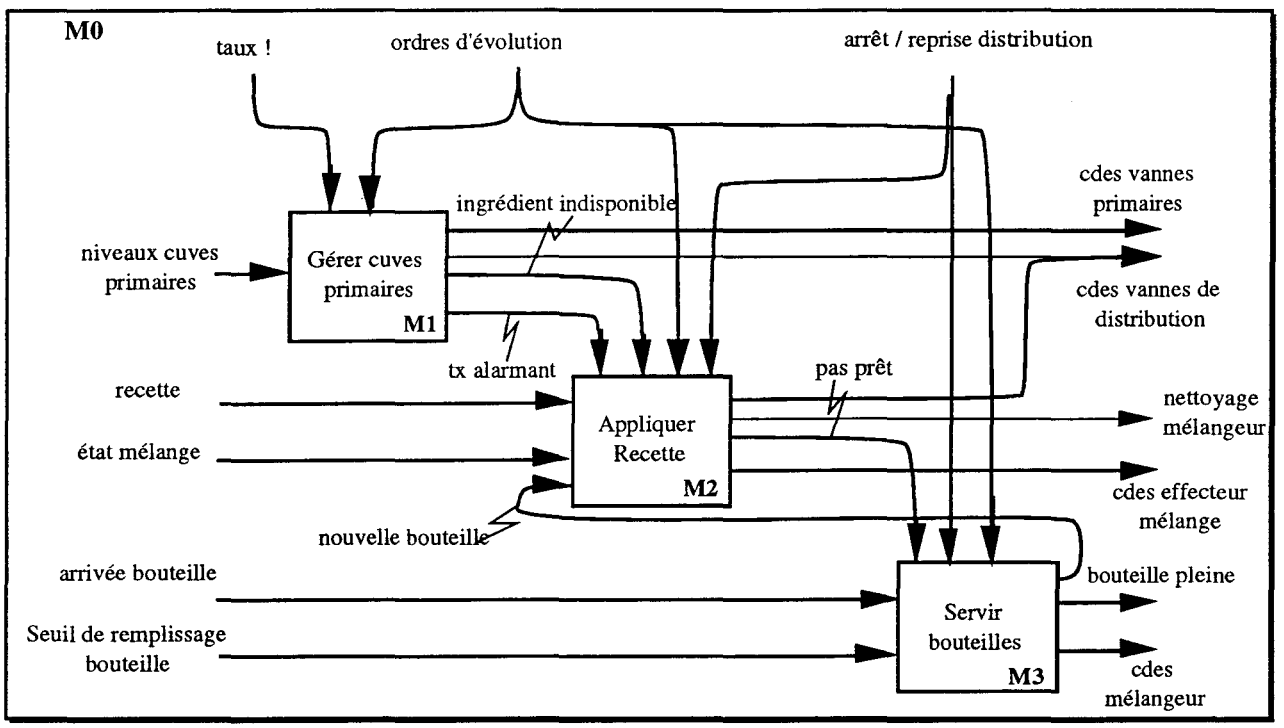
Nom Interface	*Type*	*Désignation*
Vanne_appro_concentré	Sortie TOR	Commande de remplissage de la cuve de concentré primaire
Vanne_appro_sucre	Sortie TOR	Commande de remplissage de la cuve de sucre
Vanne_appro_eau	Sortie TOR	Commande de remplissage de la cuve d'eau
Niveau_cuve_concentré	Entrée TOR	Seuil bas / Seuil haut du niveau de la cuve de concentré
Niveau_cuve_sucre	Entrée TOR	Seuil bas / Seuil haut du niveau de la cuve de sucre
Niveau_cuve_eau	Entrée TOR	Seuil bas / Seuil haut du niveau de la cuve d'eau
Taux_acidité	Entrée TOR (sur front)	Signalement de dépassement du taux maximum d'acidité permis dans la cuve de concentré
Taux_impuretés	Front d'entrée	idem pour le taux d'impuretés
Rinceur_cuve_concentré	Sortie TOR (temporisée *)	Commande de rinçage de la cuve de concentré
Rinceur_cuve_eau	Sortie TOR (temporisée *)	Commande de rinçage de la cuve d'eau
Vidange_eau	Sortie TOR (temporisée *)	Commande de vidange de la cuve d'eau en mode production
Vidange_concentré	Sortie TOR (temporisée *)	Commande de vidange de la cuve de concentré en mode la production
Moteur_vanne_eau	Sortie ANA	Commande de la vanne à débit variable de distribution en eau du mélange
Moteur_vanne_concentré	Sortie ANA	Commande de la vanne à débit variable de dosage en concentré
Moteur_vanne_sucre	Sortie ANA	Commande de la vanne à débit variable de dosage en sucre
Tx_sucre	Entrée ANA	Information capteur dans mélangeur
Rinceur_mélangeur	Sortie TOR (temporisée *)	Commande de rinçage du mélangeur
Vidange_mélangeur	Sortie TOR (temporisée *)	Commande de vidange du mélangeur
Temp_mélange	Entrée ANA	Mesure de température du mélange
Cde_résistance	Sortie TOR	Alimentation résistance de chauffage
Vitesse_moteur	Sortie ANA	Consigne transmise au régulateur de vitesse du moteur de l'agitateur
Niveau_mélangeur	Entrée ANA	Quantité de mélange en préparation

Arrêt_distribution	Entrée TOR (sur front)	Commande d'arrêt momentané de remplissage des bouteilles
Reprise_distribution	Entrée TOR (sur front)	Commande de reprise de la distribution
Arrêt_production	Entrée TOR (sur front)	Commande d'arrêt de la production, changement de recette
Arrêt_cellule	Entrée TOR (sur front)	Arrêt complet et définif de la cellule SDDS
Arrivée_bouteille	Entrée TOR (sur front)	Signalement de l'arrivée d'une nouvelle bouteille vide
Position_verseur	Sortie TOR	Commande de positionnement du bec verseur (pour remplir les bouteilles)
Cde_remplissage_bouteille	Sortie TOR	Commande de la vanne de distribution du mélange
Seuil_remplissage_bouteille	Entrée TOR	Niveau "seuil haut" de sirop dans la bouteille en cours de remplissage

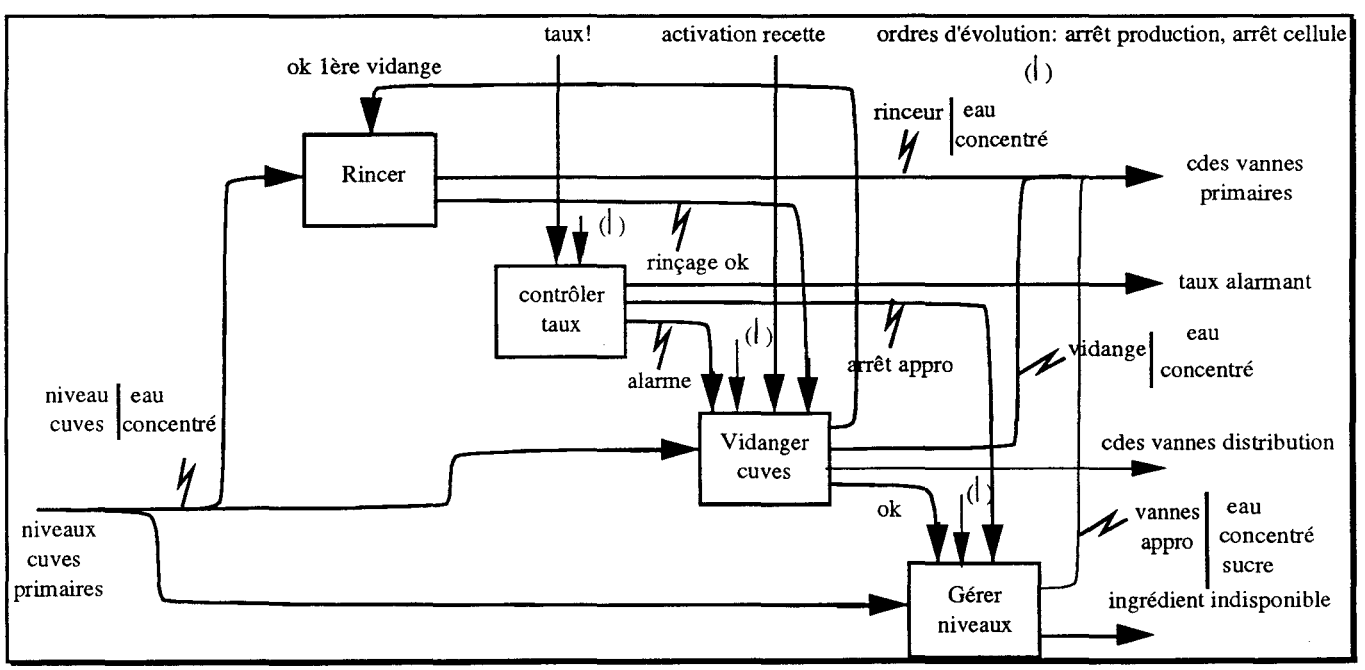
(*) Les informations dites "TOR temporisées" sont maintenues à leur état pendant la durée de la temporisation qui leur est associée

A5.1.4 DIAGRAMMES SADT

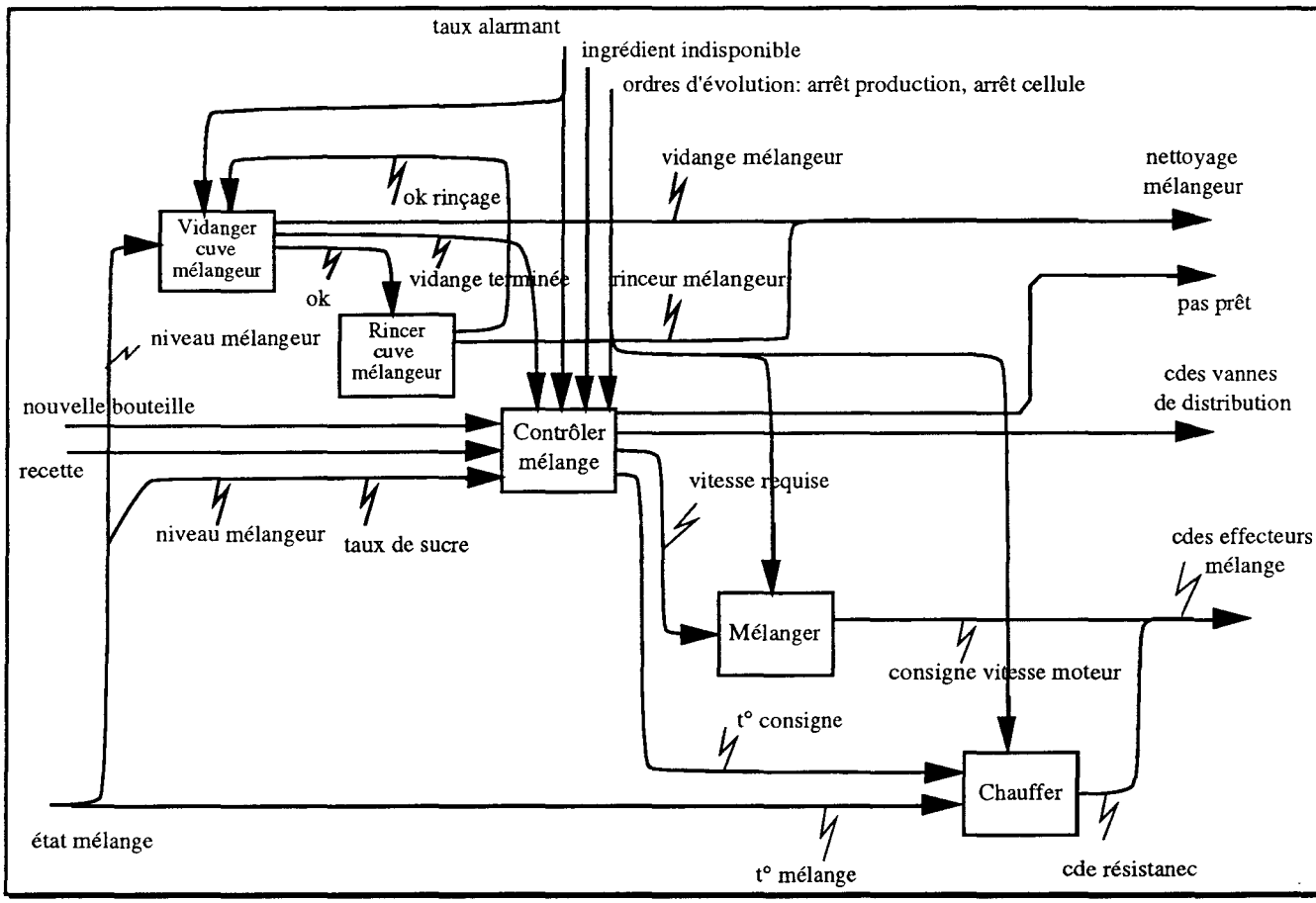
DIAGRAMME M0



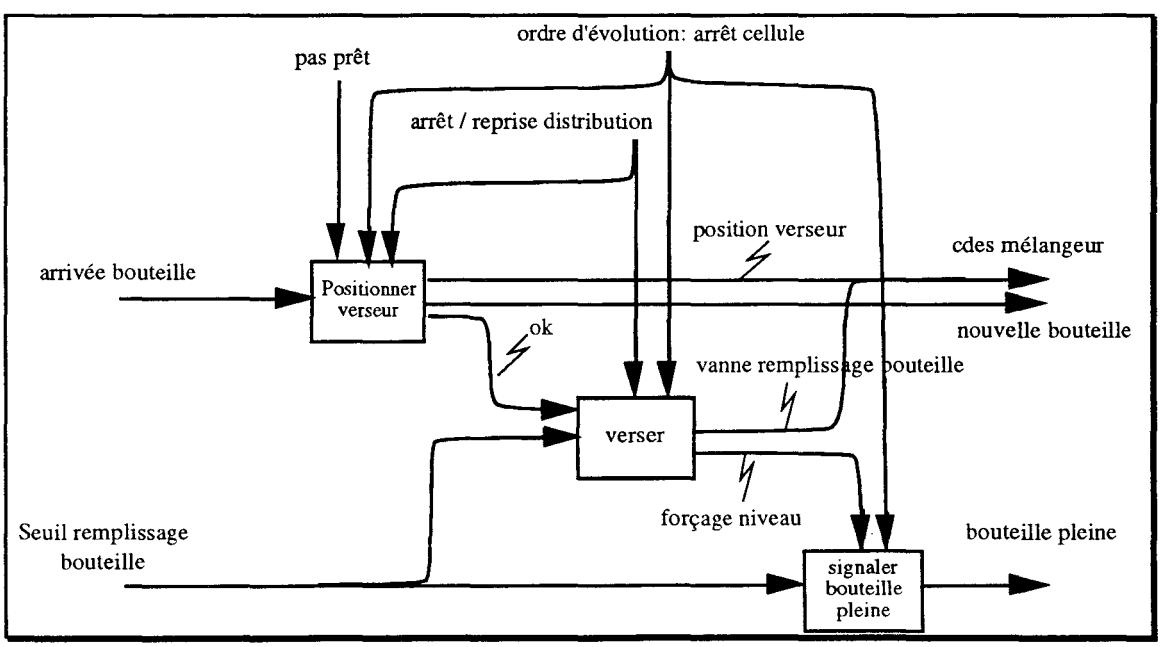
M1: GÉRER CUVES PRIMAIRES



M2: APPLIQUER RECETTE

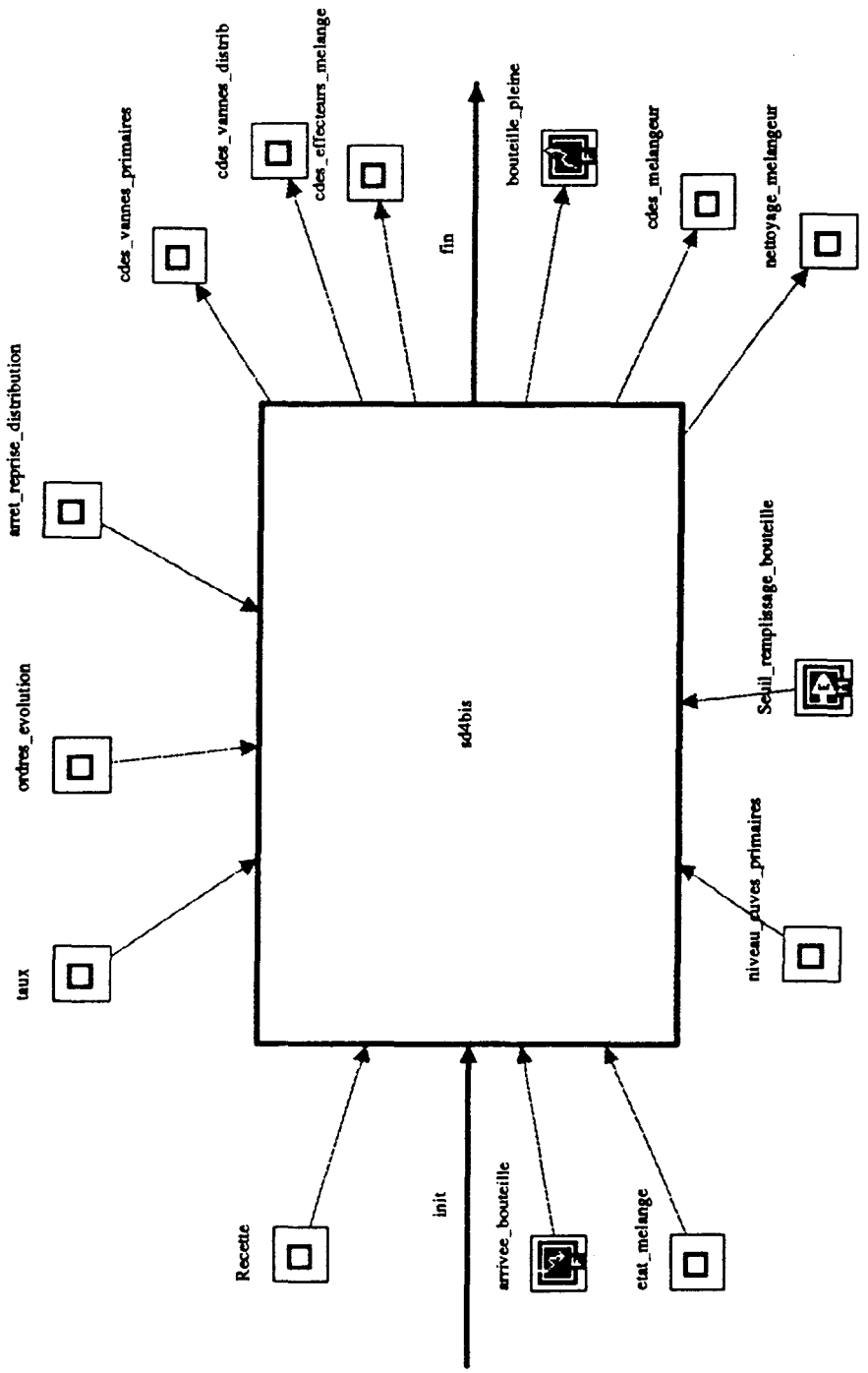


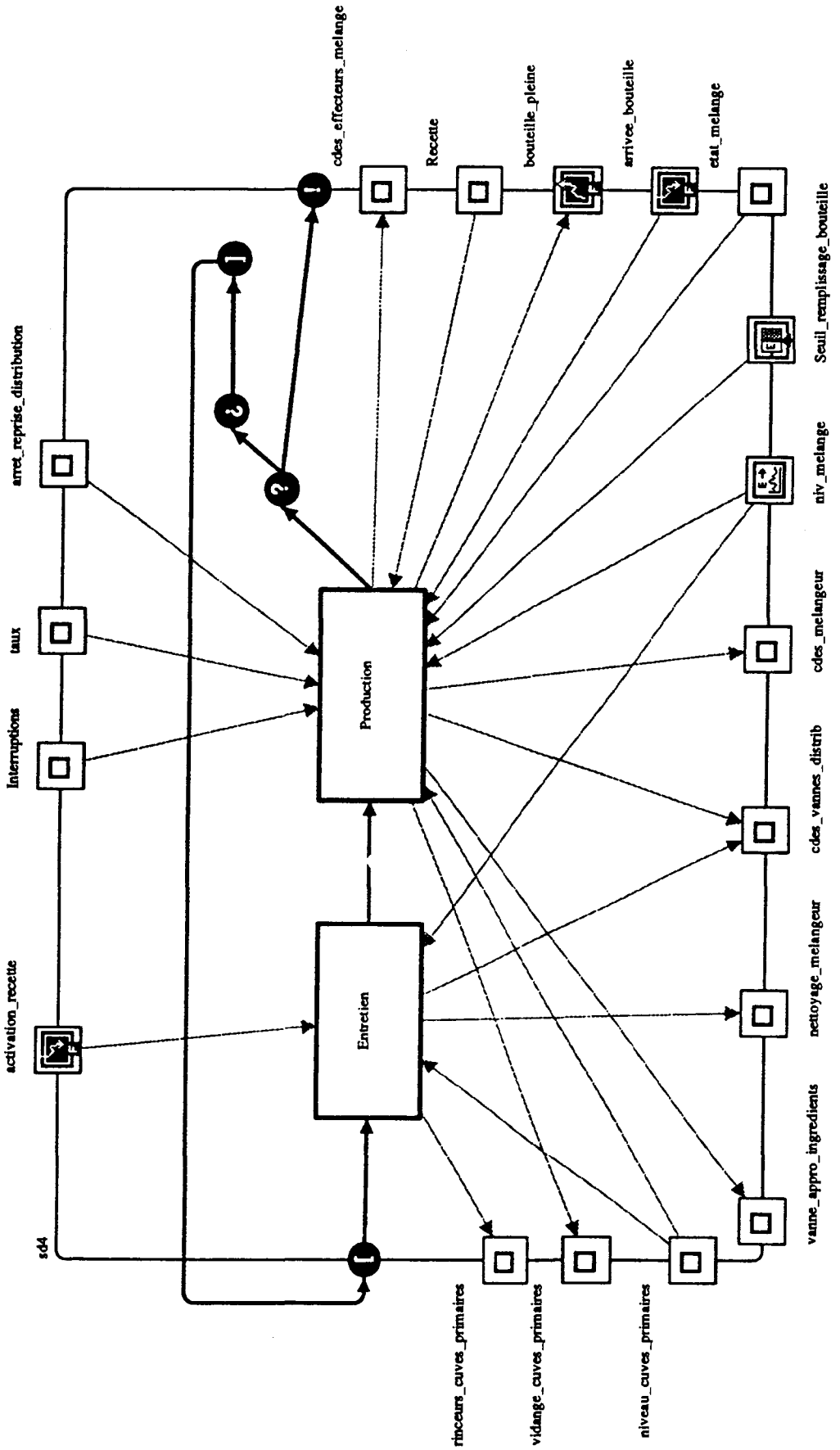
M3: SERVIR BOUTEILLES

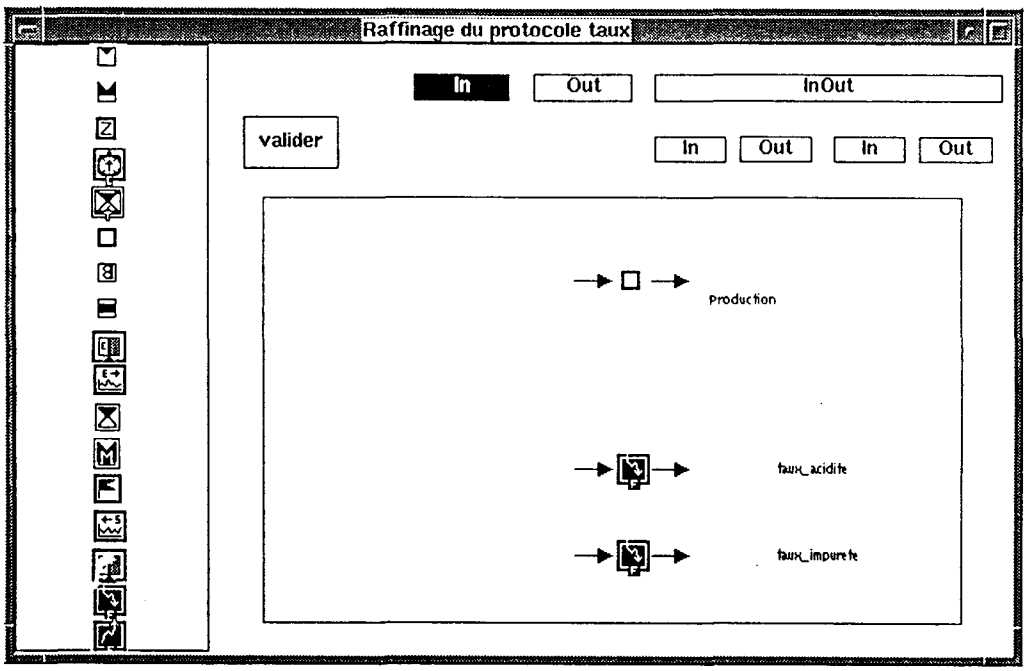


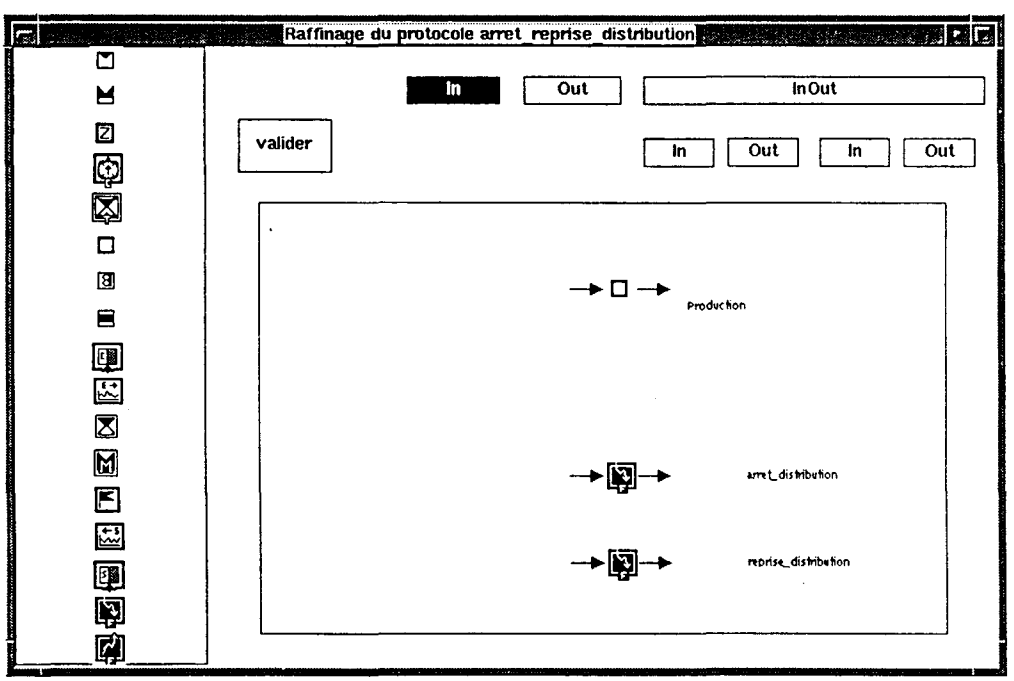
A5.2 CONCEPTION PRÉLIMINAIRE

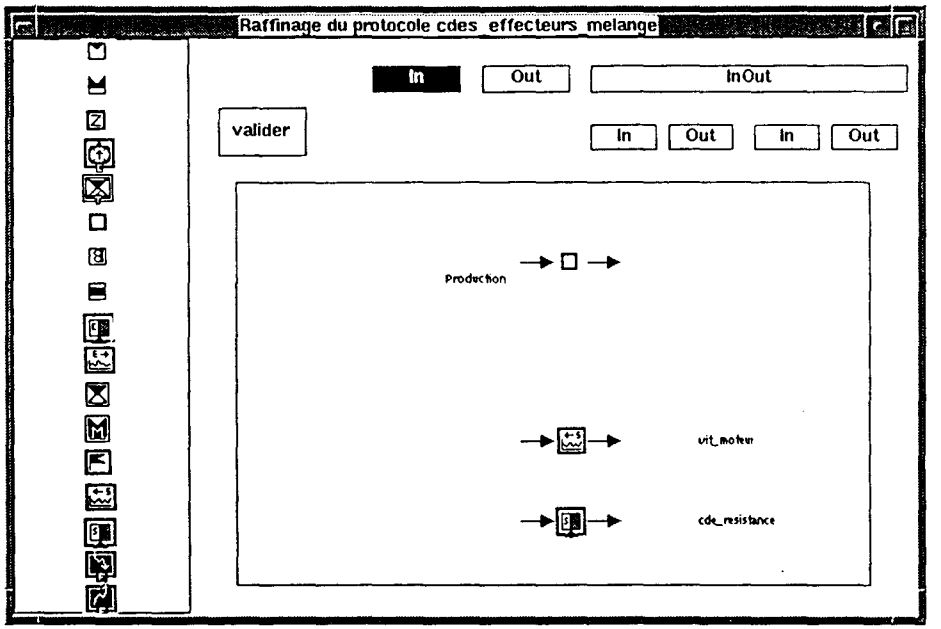
A5.2.1 DESCRIPTION DES COMPORTEMENTS

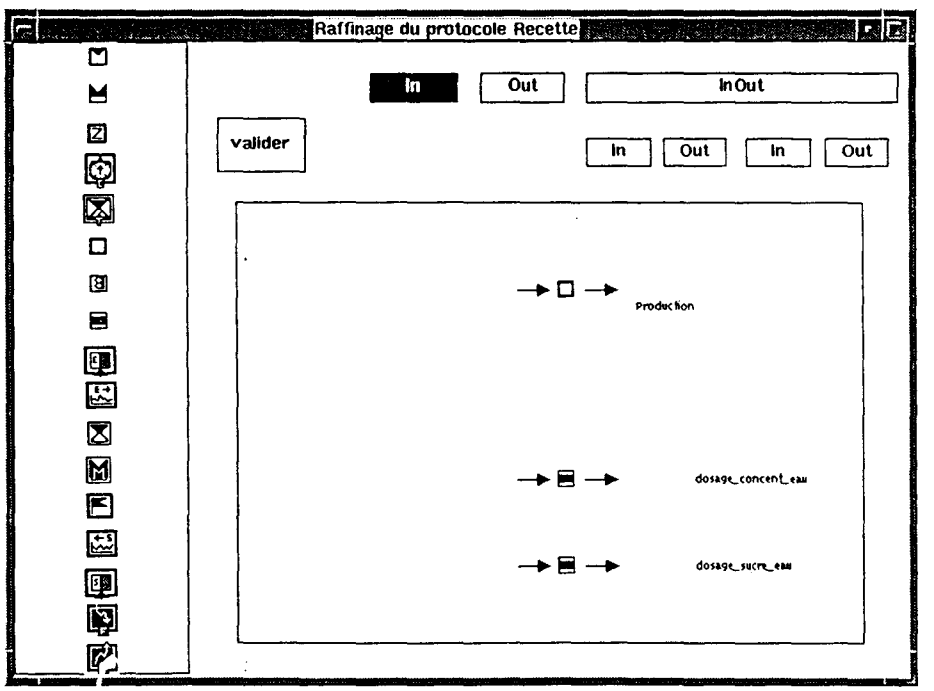


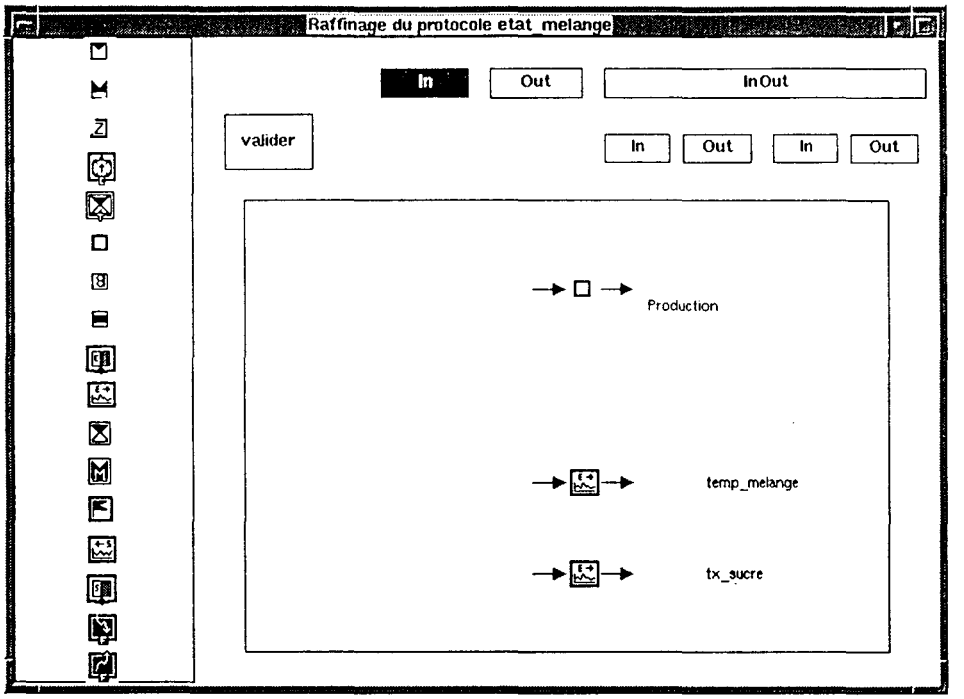

cdes_effecteurs_melange
Recette
bouteille_pleine
arrivee_bouteille
etat_melange
arret_reprise_distribution
taux
Interruptions
activation_recette
sd4
Production
Entretien
Seuil_remplissage_bouteille
niv_melange
cdes_melangeur
cdes_vannes_distrib
nettoyage_melangeur
vanne_appro_ingredients
rinceurs_cuves_primaires
vidange_cuves_primaires
niveau_cuves_primaires

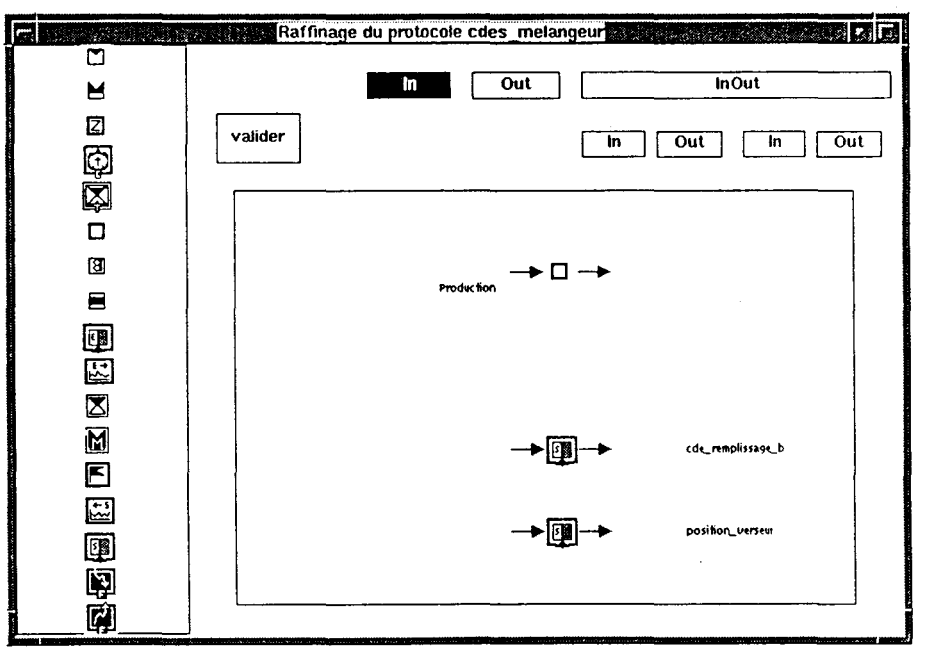

Raffinage du protocole taux
In
Out
InOut
valider
In
Out
In
Out
Production
faux_acidite
faux_impurete

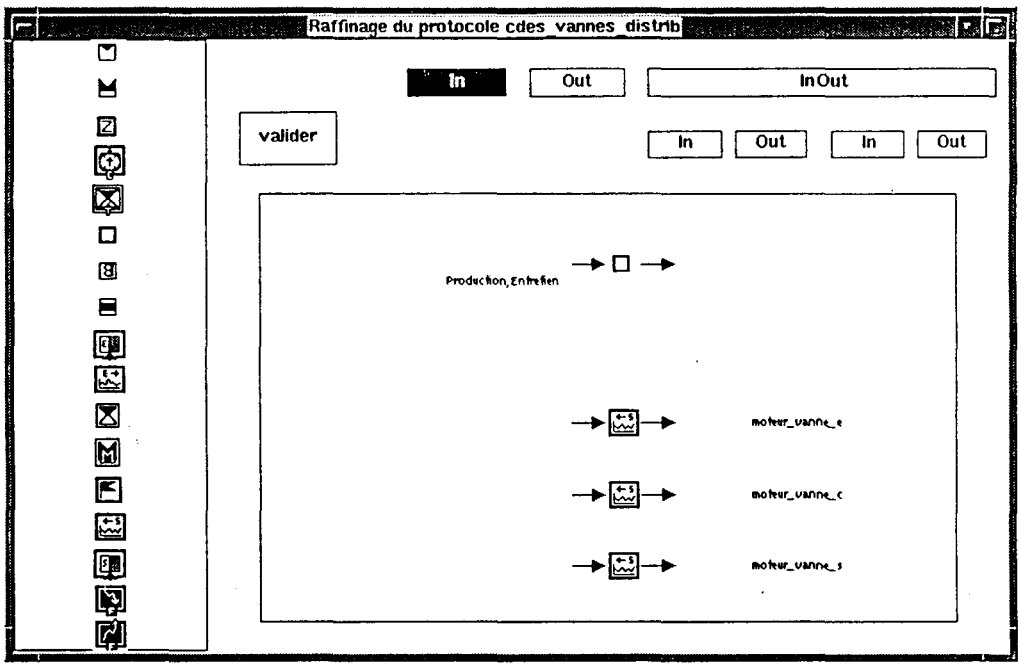

Raffinage du protocole arret_reprise_distribution
In
Out
InOut
valider
In
Out
In
Out
Production
arret_distribution
reprise_distribution

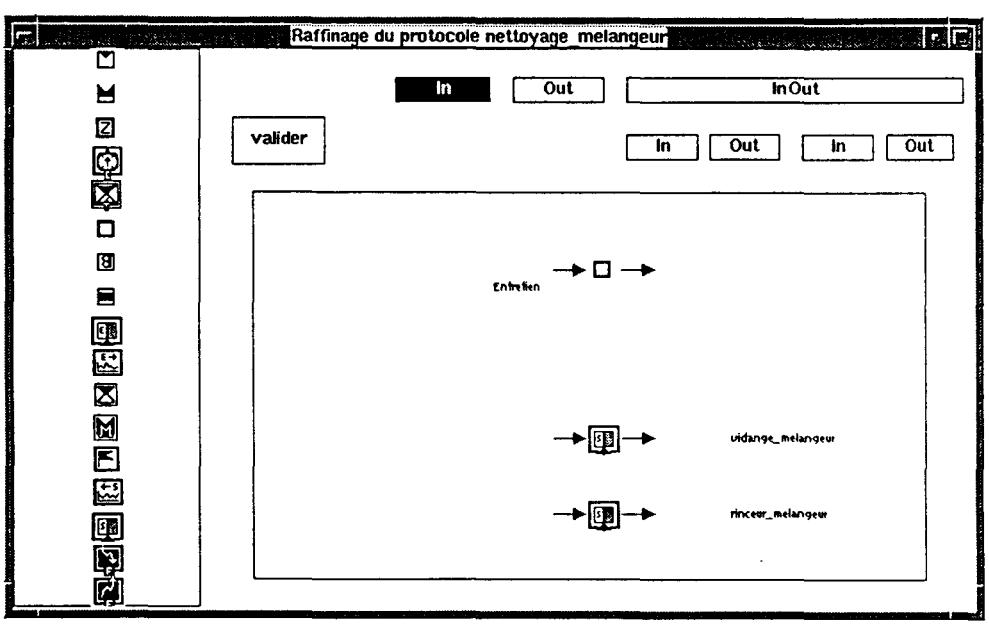

Raffinage du protocole cdes_effecteurs_melange
In
Out
InOut
valider
In
Out
In
Out
Production
vit_moteur
cde_resistance

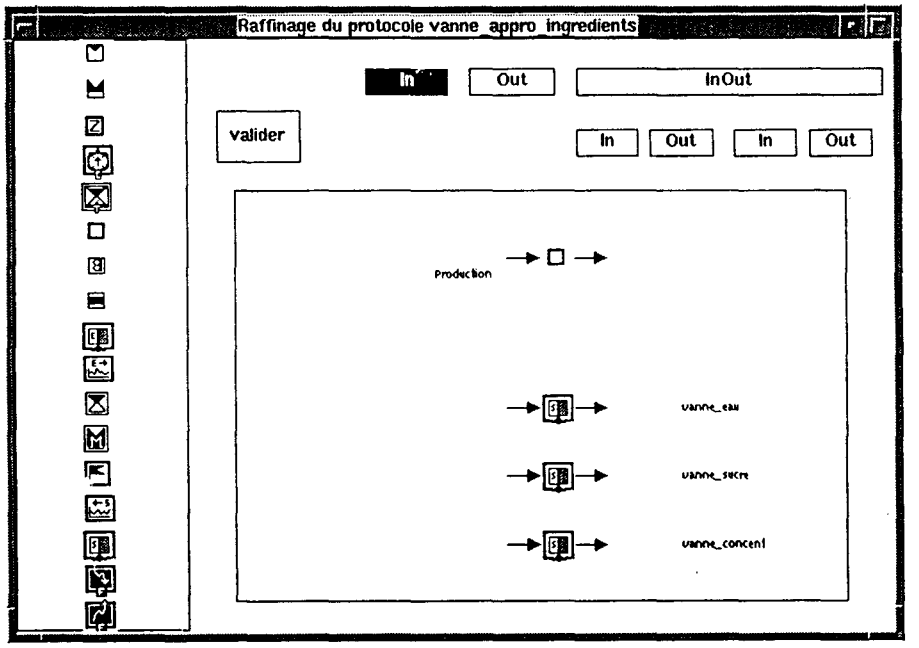

Raffinage du protocole Recette
In
Out
InOut
valider
In
Out
In
Out
Production
dosage_concent_eau
dosage_sucre_eau

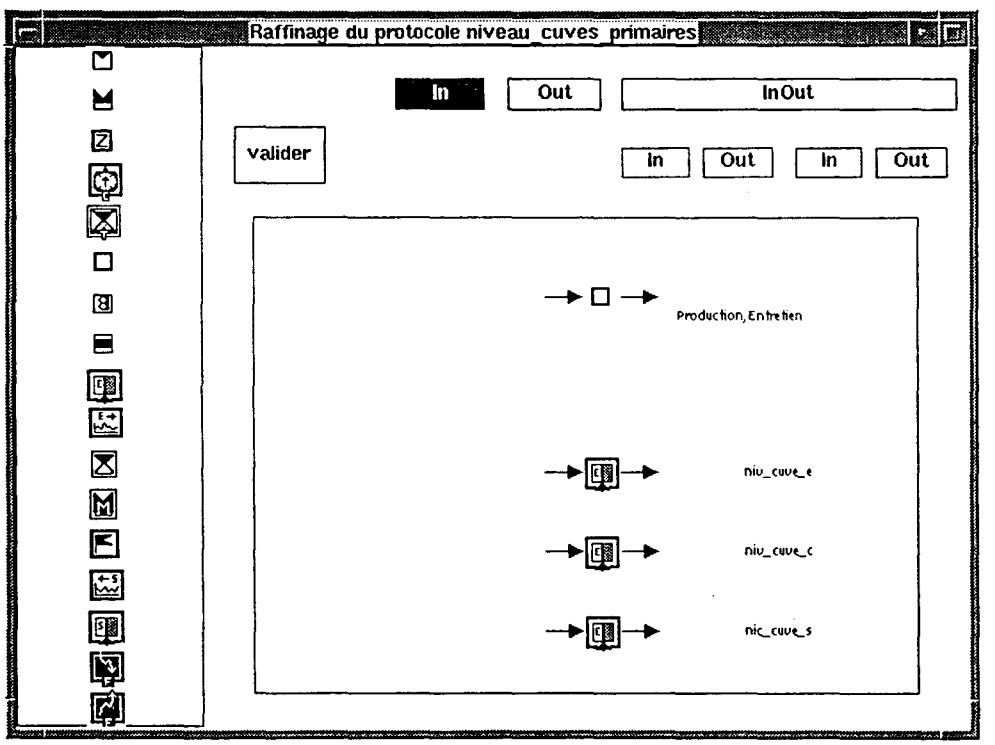

Raffinage du protocole etat_melange
In
Out
InOut
valider
In
Out
In
Out
Production
temp_melange
tx_sucre

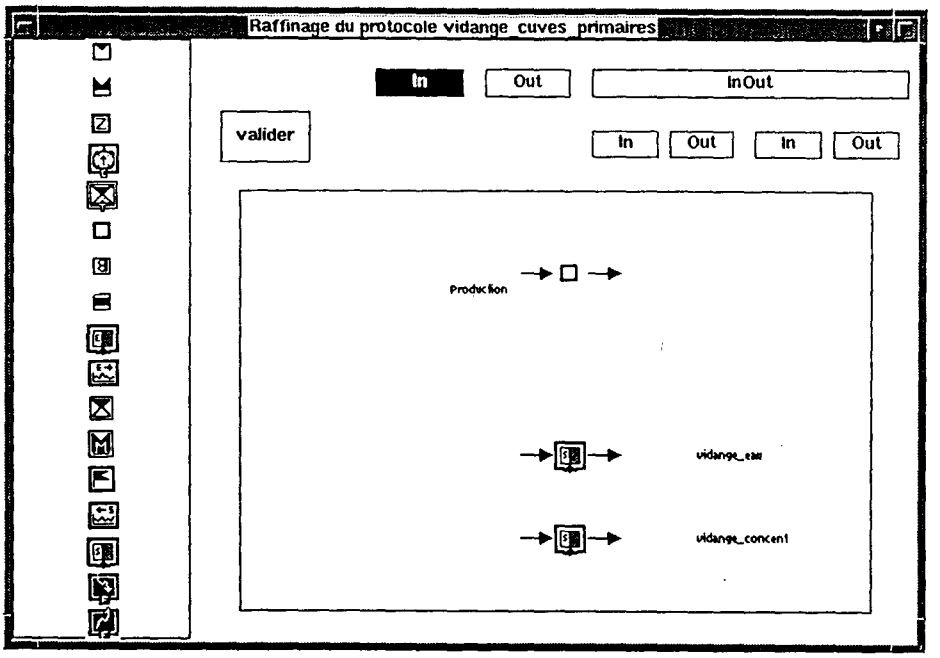

Raffinage du protocole cdes_melangeur
In
Out
InOut
valider
In
Out
In
Out
Production
cde_remplissage_b
position_verseur

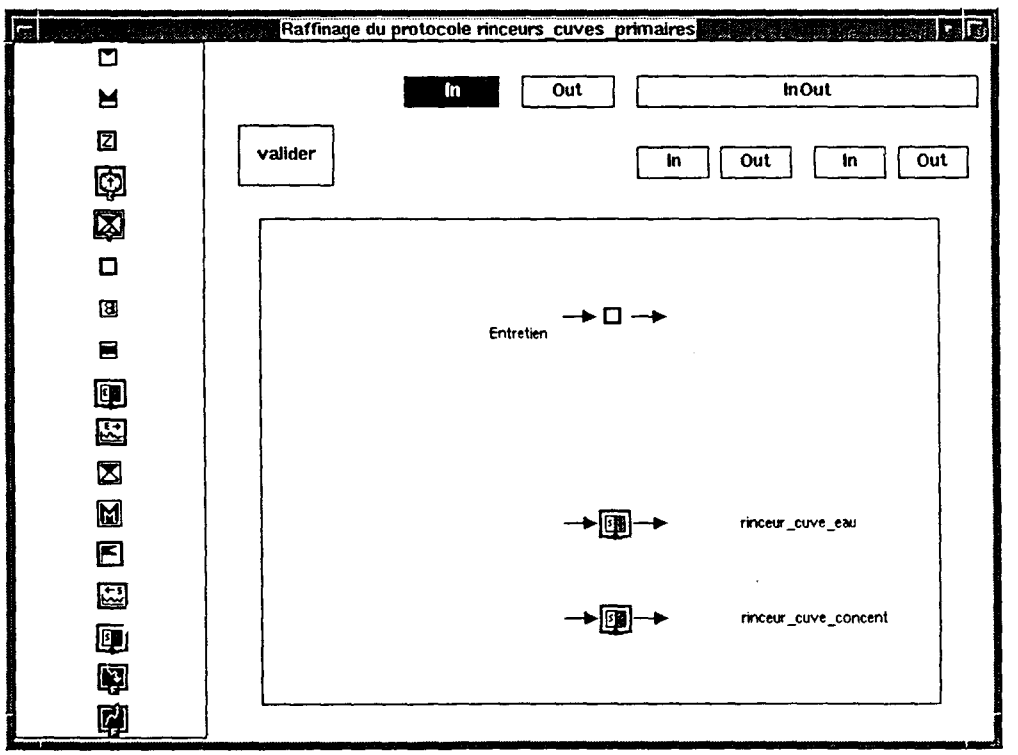

Raffinage du protocole cdes_vannes_distrib
In
Out
InOut
valider
In
Out
In
Out
Production_Entretien
moteur_vanne_e
moteur_vanne_c
moteur_vanne_s

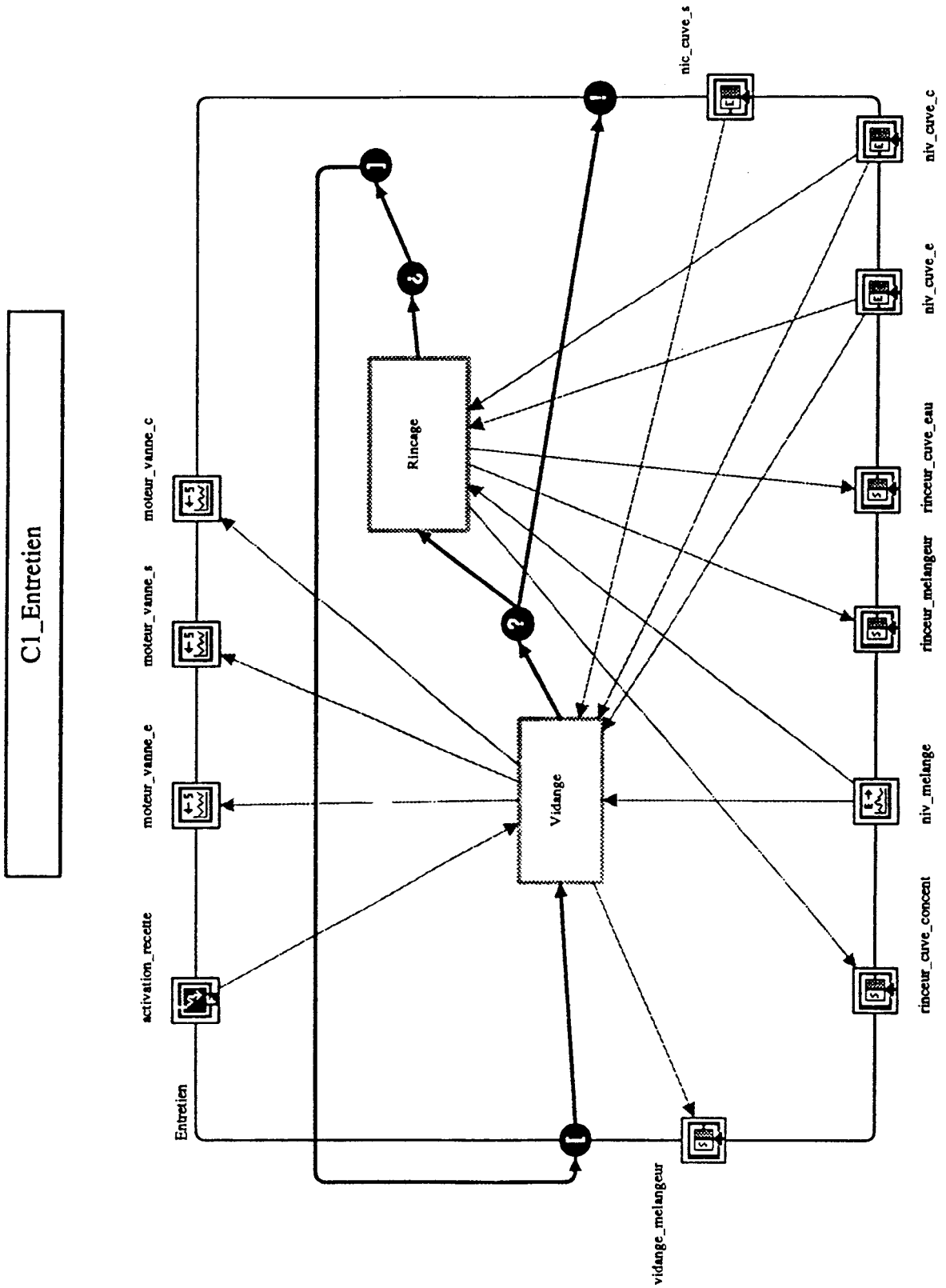

Raffinage du protocole nettoyage_melangeur
In
Out
InOut
valider
In
Out
In
Out
Entretien
vidange_melangeur
rinceur_melangeur

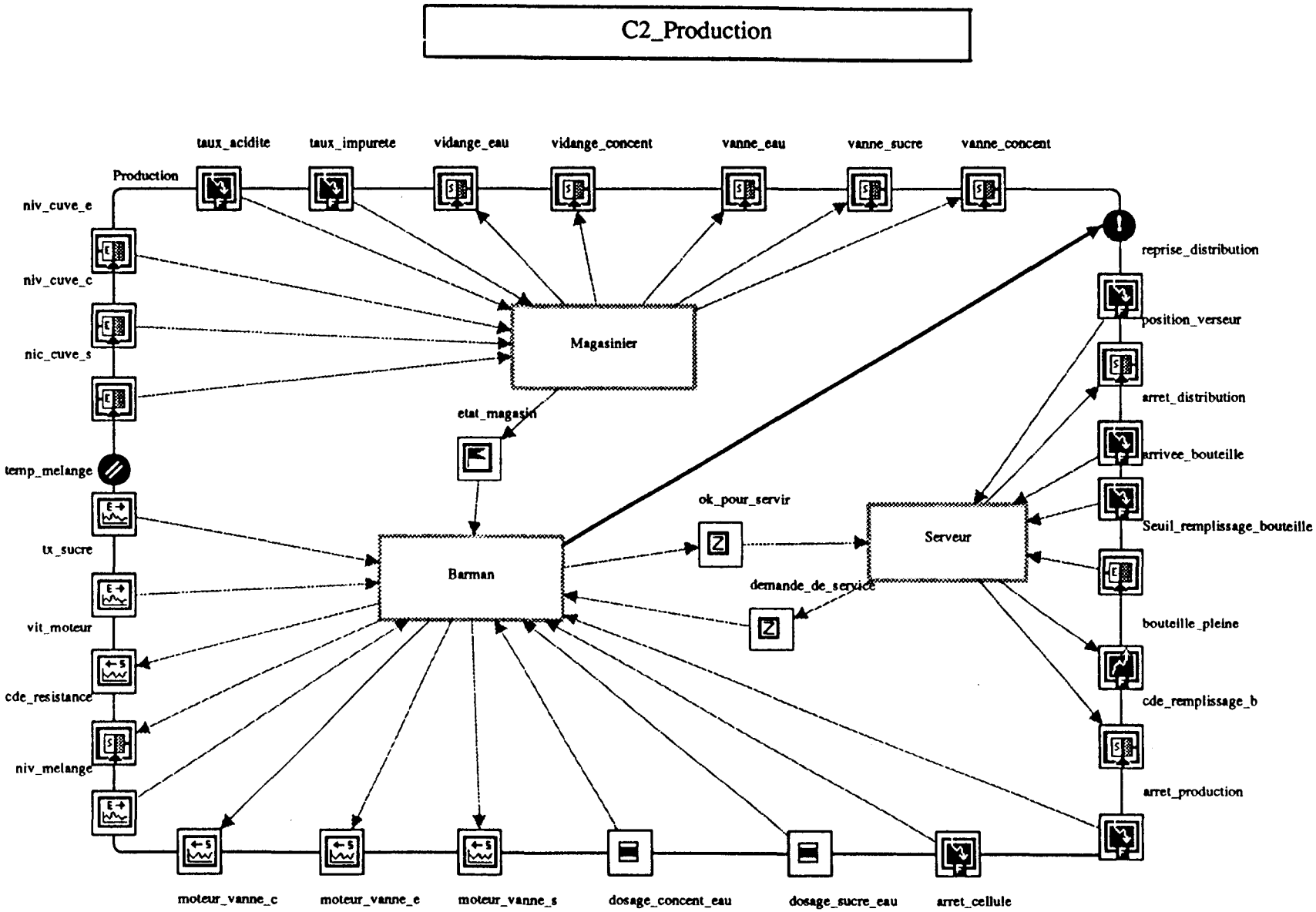

Raffinage du protocole vanne_appro_ingredients
In
Out
InOut
valider
In
Out
In
Out
Production
vanne_eau
vanne_sucre
vanne_concent


Raffinage du protocole niveau_cuves_primaires
In
Out
InOut
valider
In
Out
In
Out
Production,Entretien
niv_cuve_e
niv_cuve_c
niv_cuve_s


Raffinage du protocole vidange_cuves_primaires
In
Out
InOut
valider
In
Out
In
Out
Production
vidange_eau
vidange_concent

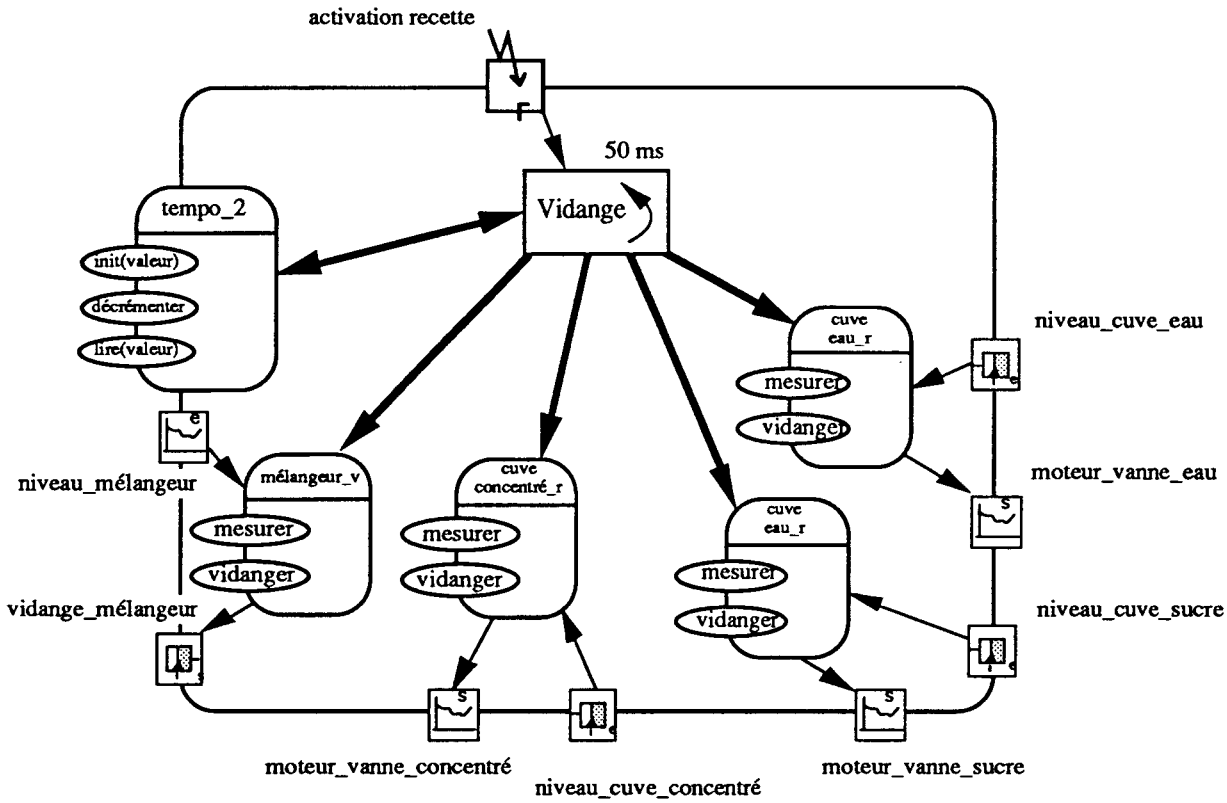

Raffinage du protocole rinceurs_cuves_primaires
In
Out
InOut
valider
In
Out
In
Out
Entretien
rinceur_cuve_eau
rinceur_cuve_concent


C1_Entretien
Entretien
activation_recette
moteur_vanne_e
moteur_vanne_s
moteur_vanne_c
Rincage
Vidange
vidange_melangeur
nic_cuve_s
niv_cuve_c
niv_cuve_e
rinceur_cuve_eau
rinceur_melangeur
niv_melange
rinceur_cuve_concent

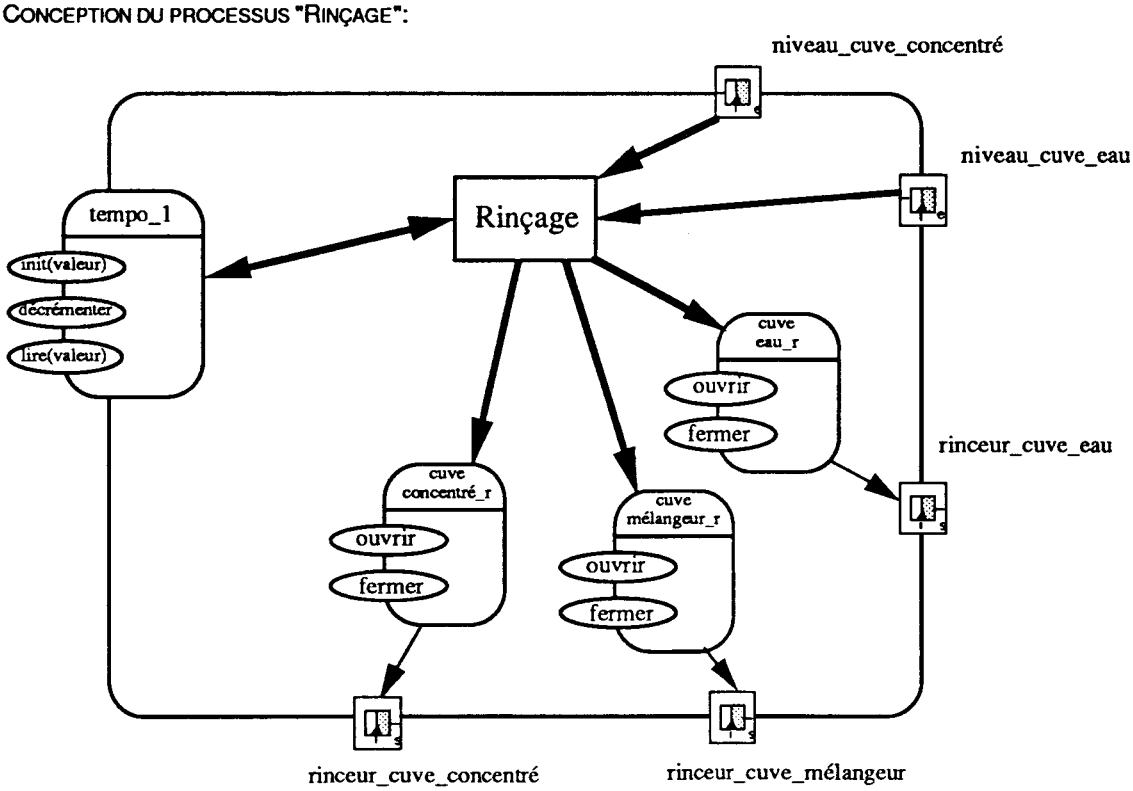

C2_Production
taux_acidite
taux_impurete
vidange_eau
vidange_concent
vanne_eau
vanne_sucre
vanne_concent
Production
niv_cuve_e
niv_cuve_c
nic_cuve_s
Magasinier
etat_magasin
temp_melange
tx_sucre
vit_moteur
cde_resistance
niv_melange
Barman
ok_pour_servir
demande_de_service
Serveur
reprise_distribution
position_verseur
arret_distribution
arrivee_bouteille
Seuil_remplissage_bouteille
bouteille_pleine
cde_remplissage_b
arret_production
moteur_vanne_c
moteur_vanne_e
moteur_vanne_s
dosage_concent_eau
dosage_sucre_eau
arret_cellule

A5.2.2 DESCRIPTION DES PROCESSUS

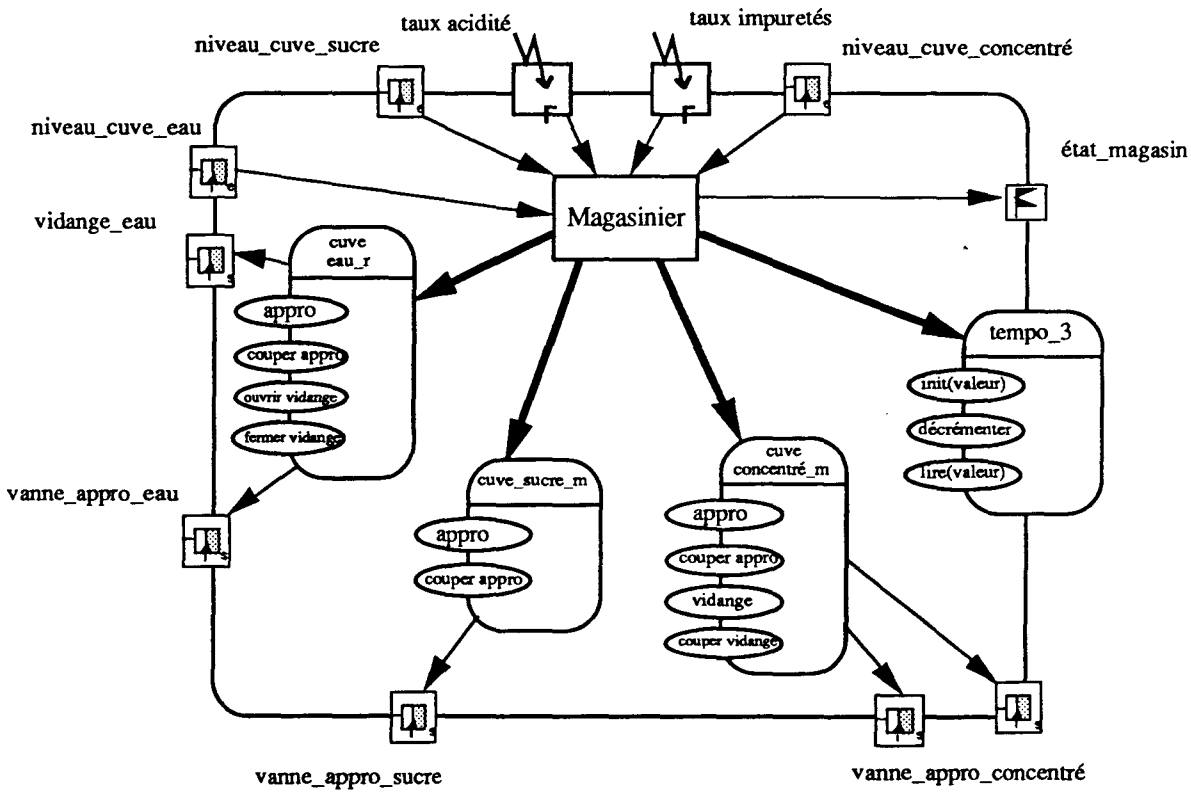
CONCEPTION DU PROCESSUS "VIDANGE":



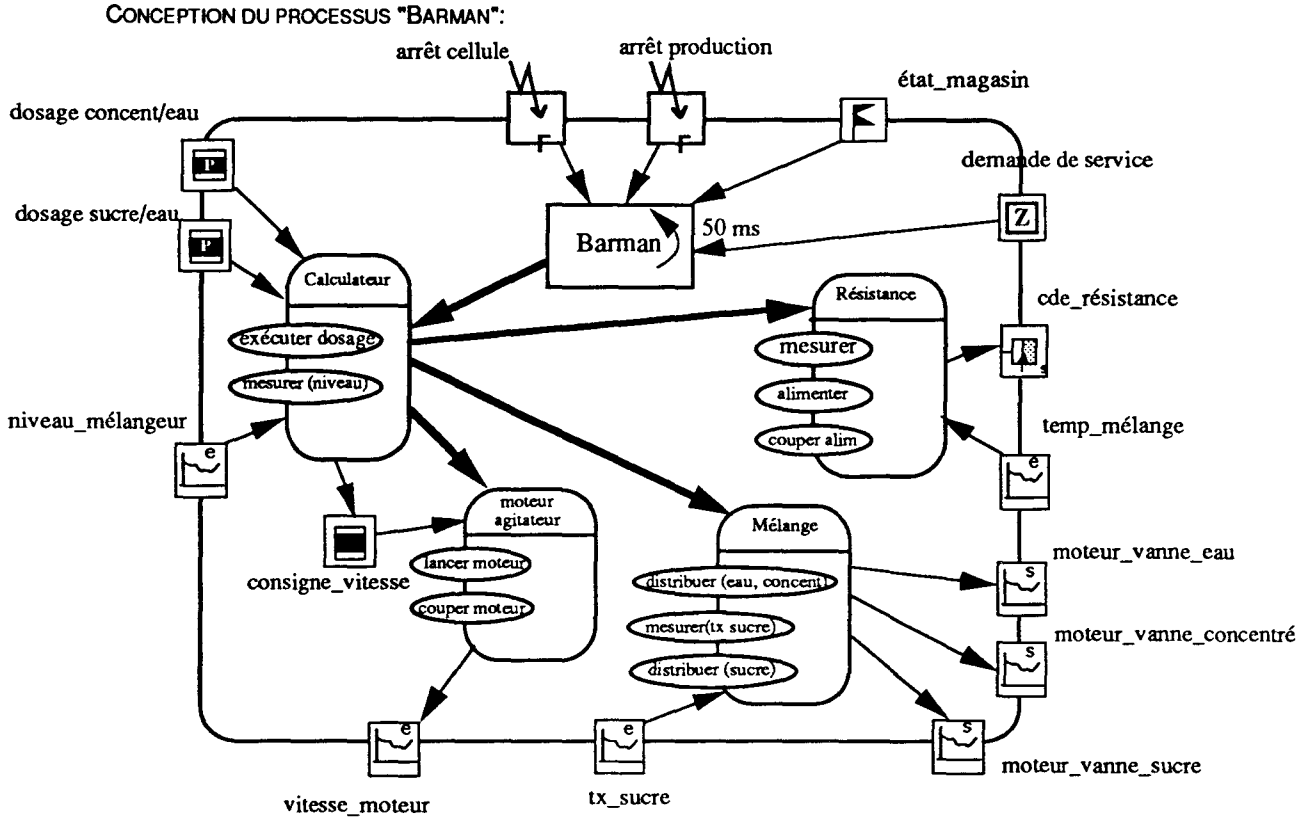
CONCEPTION DU PROCESSUS "RINÇAGE":



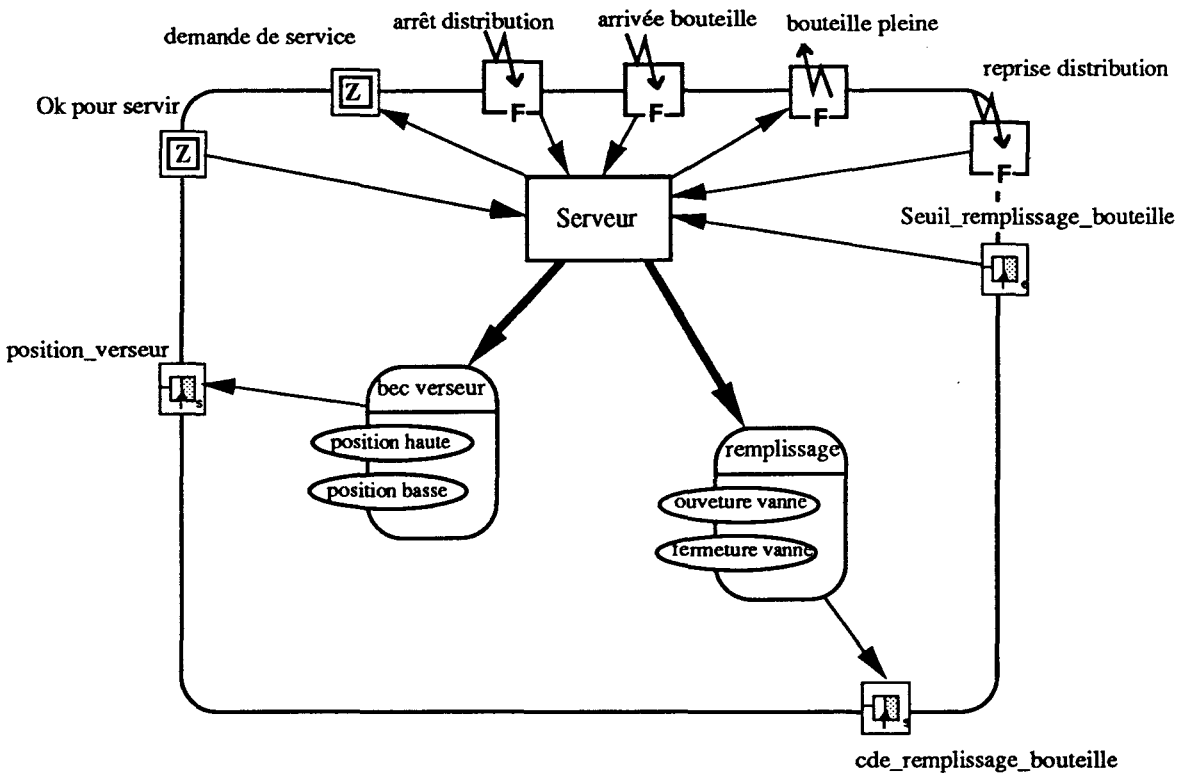
CONCEPTION DU PROCESSUS "MAGASINIER":



CONCEPTION DU PROCESSUS "BARMAN":



CONCEPTION DU PROCESSUS "SERVEUR":



RÉFÉRENCES BIBLIOGRAPHIQUES

[ALA 84]
P. ALANCHE, X. DE ROQUEMAUREL, D. MORIN
"Le Poste de Travail de L'Automaticien"
Texte des communications du colloque "Automatique Appliquée" - SEE - NICE 1984

[AMA 90]
S. AMAR, E. CASTELAIN, J.C. GENTINA
"Modélisation des moyens de production par langages orientés objet en vue de la conception de la commande d'un système de production flexible
CIM Productica 90, p 323, Bordeaux

[BAZ 69]
I. BAZOVSKY - Théorie et pratique de la sureté de fonctionnement - DUNOD

[BER 87]
Gérard BERRY, Philippe COURONNE, Georges GONTHIER
"Programmation synchrone des systèmes réactifs: le langage ESTEREL" - TSI janvier 87

[BOO 86]
G. BOOCH - "Object-Oriented Dévelopment"
IEEE Transactions on Sofware Engineering, Vol. 12, page 211-221, feb. 1986

[CAL 90]
J.P. CALVEZ - "Spécification et conception des systèmes: une méthodologie"
MASSON 90

[CEI-92]
Norme CEI 1131, partie 3: "Programmable controllers - Part 3: Programming langages.
juillet 92

[CHE 76]
P.P. CHEN - "The Entity-Relationship Model: Toward a Unified View of Data"
ACM Transaction of Database Systems, Vol 1, n°1 - Juin 76

[CHPP 87]
P. CASPI, N. HALBWACHS, D. PILAUD, J.A. PLAICE
"LUSTRE: A declarative language for programming synchronous systems", 14th ACM Symp. on Principles of Programming languages, Munich, janvier 87

[CIM 88]
Référence Architecture Specification
public domain report, ESPRIT I project 688, 1988

[CLA 83]
E. M. CLARKE, E. A. EMERSON, A. P. SISTLA
"Automatic Vérification of Finite State Concurrent Systems Using Temporal Logic Specifications: a practical Approach" - Department of Computer Science Report. Carnegie-Mellon University, 1983.

[COL 91]
G. COLOMBARI - "Le point sur le Grafcet: enrichissements et utilisations"
Acte des conférences Automation 1991
Journée du 14 mai 1991 - pp 35 à 46

[CRU 91]
D CRUETTE - "Méthodologie de conception des systèmes complexes à événements discrets: Application à la conception et la validation hiérarchisée de la commande de cellules flexibles de production dans l'industrie manufacturière" - Thèse de doctorat à l'USTLFA

[DEL 93]
D. DELFIEU, A.E.K. SAHRAOUI
"Expression and Verification of Temporal Constraints for Real-Time Systems"
COMPEURO 93 - Session A.5

[FRA 87]
JP FRACHET - "Une introduction au génie automatique: Faisabilité d'une chaîne intégrée d'outils pour la conception et l'exploitation des machines automatiques industrielles" - Thèse de doctorat ès Sciences Physiques - Université de NANCY 1

[G7 92]
P. BRARD, N. BOUTEILLE, G. COLOMBARI - G7 modèle d'exécution - 1992

[GAG 93]
A. DEVARENNE, A. FREAU, S. PICAUD (Alcatel Alsthom Recherche)
" A Generic Activity Manager for a Control Engineering Workbench"
COMPEURO 93, Computers in Design, Manufacturing and Production - A.4.4 - pages 288 à 294

[GAL 93]
D. DALARA, JM. FAVENNEC, B. IUNG, G. MOREL
"Distributed Intelligent Actuators and Sensors"
COMPEURO 93 - Proceeding du 24 au 27 avril, pages 79-85

[GMA 93]
JB. KOECHLIN, D. CHAMPALOUX
"A Conceptual Data Schema for Integrated Maintenance: A Map to Design Maintainable AMS"
COMPEURO 93 - Session B.2

[GON 87]
G. GONTHIER
"Sémantiques et modèles d'exécution des langages réactifs synchrones; Application à ESTEREL; Thèse d'Informatique, Université d'Orsay, 1987.

[GSA 82]
C. GANE, T. SARSON - Structured Systems Analysis: Tools and Techniques Prentice Halla New jersey, 1982`

[HAR 90]
D. HAREL
"Statemate: A Working Environment for the Developpement of Complex Reactive Systems"
IEEE Transaction on Software Engineering, vol 16, n°4, april 1990

[HAR 84]
D. HAREL
"A Visual approach to complex systems, CS84-05, Department of Applied Mathematics, the Weizmann institute of Science, 1984

[HAR 87]
D. HAREL, A. PNUELI, J.P. SCHMIDT, R. SHERMEN
"On the fomal semantics of statecharts"
Proc. 2nd IEEE Symposium on Logic in Computer Science (1987)

[HOA 87]
C.A.R. HOARE - "Processus séquentiels communicants" - MASSON

[IGL 89]
I.G.L. Technologie - SADT: Un langage pour communiquer
Editions Eyrolles - 1989

[ISO-1]
Norme ISO 9001 - "Systèmes qualité. Modèle pour l'assurance de la qualité en conception/développement, production, installation et soutien après vente" - déc. 1988

[ISO-2]
Norme ISO 9002 - "Systèmes qualité. Modèle pour l'assurance qualité en production et installation" - déc. 1988

[KOE 90]
JB. KOECHLIN
"Projet ELEGA: première réalisation mettant en oeuvre BASEPTA"
CIM 90 Congress BORDEAUX - 1990

[LGU 86]
P. LE GUERNIC, A. BENVENISTE, P. BOURNAI, T. GAUTHIER
"SIGNAL ; A Data Flow Oriented Language For Signal Processing"
IEEE-ASSP 34(2), avril 1986, 362-374

[LIS 74]
B. LISKOV, S. ZILLES - "Programming with Abstract Data Types SIGPLAN"
Noticesn 9.4, avril 74

[LIS- 2]
B. LISKOV, J.GUTTAG
"Abstraction and Specification in Program Development"
MIT Press, Cambridge

[MAC 85]
Contenu du cours IGL sur la méthode MACH - Méthode Appliquée de conception Hiérarchisée

[MEY 88]
B. MEYER - " Object oriented Software construction "
Editions PRENTICE HALL, 1988

[MIL 83]
R. MILNER: "Calculi for Synchrony and Asynchrony"
Lectures Notes, Aarhus University, 1981

[MOD 89]
MODAL - Méthodologie d'Organisation du Développement d'Applications Logicielle
1989

[MOD 93]
MODAL V2 - Extension de la méthodologie aux systèmes - 1993

[NIJ 81]
G. M. NIJSSEN - "An architecture for knowledge Base software"
(Australian Computer Society)

[PAN 91]
H. PANETTO - " Une contribution au génie automatique: Le prototypage des machines et systèmes automatisés de production"
Thèse Doctorat de l'Université de NANCY I - Jan 91

[PET 89]
R. DAVID, H. ALLA - "Du Grafcet aux réseaux de Petri"
Paris: Hermès 1989

[QUE 93]
H. GUEGUEN, Y. QUENEC'DHU, P. MEREAU, M. BAYART
"Automatisation des procédés mixtes continus et séquentiels"
Actes du Colloque A2RP des 20 et 21 jan. 93 - Session Productique

[RAY 91]
Michel Raynal - "Synchronisation et état global dans les systèmes répartis"
Collection de la DER EDF - 1991

[ROE 87]
M. ROESCH, C. LAURENT
Structuration de la partie commande d'un système automatisé
Electronique Industrielle n°119/1-2-1987

[ROS 87]
JP. ROSEN - " Méthodes de conception orientées objets "
Génie Logiciel, n°9, nov. 87

[SAD 89]
Guide d'auteur SADT
(MODAL 89)

[SAR 92]
Recommandation pour l'Utilisation de la méthode SA-RT
(MODAL 92)

[SYS 91]
Construction des systèmes d'exploitation répartis
INRIA - Collection Didactique

[TIX 89]
JM. TIXADOR - "Une contribution au Génie Automatique: la SPécification EXécutable des Machines et Systèmes Automatisés de Production"
Thèse de l'Université de Nancy I option "Production Automatisée" - 1989

[TOC 92]
"TOCCATA, une méthode de conception des logiciels temps réels"
Revue des Journées Logiciels - n°29 déc.92

[TOC 93]
Manuel de référence de la méthode TOCCATA
Document interne à CEGELEC - ref MODAL BCI 52 401 a-fr

[WOL 83]
WOLPER - "Temporal Logic can be more expressive"
Information and control 56, pp 72-99, 1983.

[Z99-1]
NF Z 99-001 - "Bibliothèque de composants - Formats et structures de données"

[Z68-92]
NF Z 68-901 - "Représentation des systèmes de contrôle et de commande des systèmes automatisés de production - Modèle conceptuel BasePTA"
AFNOR sept 92

BIBLIOGRAPHIE

[ALA 87]
P. ALANCHE, G. MOREL, M. ROESCH, P. SALVI, A. SFALON
Utilisation de composants logiciels génériques dans la conception des commandes des systèmes de production manufacturière GAMI - 2ièmes journées "Les outils de la Productique" - Janvier 1987

[ALA 88]
P. ALANCHE - "Automatiser l'automatisation"
Revue Automatique et Productique Appliquées - Vol. 1 - N°2 - 1988

[BEE 89]
D. BEECKMAN - CIM-OSA (Computer Intégrated Manufacturing open Systems Architecture)
International Journal of CIM - 2(2). 1989

[BEL 89]
A. BELHIMEUR - "Contribution à l'étude d'une méthode de conception des automatismes des systèmes de conduite des processus industriels"
Thèse de l'Université de LILLE I, 1989

[BIL 89]
A. BILLIONNET, M.-C. COSTA ET A. SUTTER
Les problèmes de placement dans les systèmes distribués
Technique et Science Informatiques, vol.8, n°4, 1989, P. 307-337

[BIL 92]
A. BILLIONNET
Placement de tâches à structure arborescente avec contrainte de charge
Technique et Science Informatiques, vol. 11, n°1, 1992, page 117 à 137

[BOU 88]
JP. BOUREY - "Structuration de la partie procédurale du système de commande de cellules flexibles dans l'industrie manufacturière"
Thèse de Doctorat d'Université de l'USTLFA, 1988, Lille

[BOUR 78]
R.J. BOURBINA, P.B. PATEL - Structured Analysis and Design for Telephony Sofware Development
International Conference on Communications, Vol. 3, Toronto, Canada, june 1978

[CEI 84]
Norme CEI 848 - "Etablissement des diagrammes fonctionnels pour systèmes de commande"

[CIM 90]
CIM: Productique et Intégrations
Actes du Colloque International CIM 90 - 12 au 14 juin

[CLO 86]
C. CLOUET, H. HABRIAS - "La méthode NIAM et les outils logiciels associés"
Actes de la Convention Informatique - 1986

[MRT 92]
"Le Génie Automatique pour la maintenance"
Rapport finale projet MRT

[GRA 92]
GRAFCET ' 92 - Théorie et applications
Recueil des Journées du G7 patronées par le Ministère de la recherche et de la technologie les 25 et 26 mars 1992

[HUV 93]
B. HUVENOIT, E. CRAYE, JP. BOUREY
"Implantation Oriented Methodologie in Design Control of Flexible Manufacturing Systems"
Proceedings of COMPEURO 93 - Session C.2

[ISO-3]
Norme ISO 9003 - "Systèmes qualité. Modèle pour l'assurance qualité en contrôle et essais finals" - déc. 1988

[LAV 93]
E. LAVARIERE, M. NAKHLE
"Le projet CP2M / Conception de Processus de Production Mixtes"
Colloque A2RP, 20 et 21 jan. 93 - Session Productique

[LON 89]
J. LONCHAMP
"De la conception des applications réparties en commande de procédés industriels"
TSI, 89, Vol. 8, n°5, pages 407-421

[MAS 89]
G. MASINI, A. NAPOLI, D. COLNET, D. LEONARD, K. TOMBRE
"Les langages à objets" - InterEdition - 1989

[MIL 81]
R. MILNER: "Calculi for Synchrony and Asynchrony"
Lectures Notes, Aarhus University, 1981

[PIE 90]
H. PIERREVAL
"Les méthodes d'analyse et de conception des systèmes de production"
Hermes 1990

[POP 93]
J.C POPIEUL, T. BERGER, L. DELPORTE
"An Object based formalism helping the design of Automated Manufacturing Systems"
FUCAM 93 - pages 132-141

[PLO LN]
G. D. PLOTKIN: "A Structural Approach to Operational Semantics
Lectures Notes, Aarhus University

[SYS PUF]
La systémique
Collection "Que sais-je" - Presses Universitaires de FRANCE

[TOC MOD]
Manuel de référence de la méthode TOCCATA
Méthodologie MODAL V1 - réf. CEGELEC BCI 52 401 a-fr

[TOK 88]
J. TOKAONO - " Une approche méthodologique pour la conception certifiée de commande des automatismes industriels répartis"
Thèse de Doctorat de l'Université de NANCY I - 1988

[YOU 89]
E. YOURDON - " Modern Structured Analysis"
Edition PRENTICE HALL - 1989

TABLE DES MATIÈRES

CHAPITRE I LES SYSTÈMES APPLIQUÉS DE CONTRÔLE-COMMANDE

CHAPITRE II MODÉLISATION DE L'ARCHITECTURE MATÉRIELLE D'UN SCC

CHAPITRE III CONCEPTS LIÉS À L'APPLICATION LOGICIELLE D'UN SCC

CHAPITRE IV DE LA DESCRIPTION DE L'ARCHITECTURE MATÉRIELLE DU SYSTÈME

CHAPITRE V .. A LA CONCEPTION PRÉLIMINAIRE DU LOGICIEL DE SES EQUIPEMENTS D'AUTOMATISME

ANNEXES

LISTE DES FIGURES

INTRODUCTION

CHAPITRE I

CHAPITRE II

CHAPITRE III

CHAPITRE IV

CHAPITRE V