INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE
Centre de Recherche en Informatique de Nancy

SC N 89/ 150 A

**THESE**

présentée à

l'INPL

pour l'obtention du titre de

**Docteur en Informatique**

BIBLIOTHEQUE UNIVERSITE SCIENCES

par

**Jean-Luc DELCUVELLERIE**

Ingénieur de

l'Ecole Nationale Supérieure d'Electricité et de Mécanique

---

# INGENIERIE DES SYSTEMES D'AUTOMATISATION DE PRODUCTION:

## CONNAISSANCES EN CONCEPTION DES ARCHITECTURES DE SYSTEMES INFORMATISES D'AUTOMATISATION

et

## OUTIL D'ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES D'ARCHITECTURES REPARTIES.

---

Service Commun de la Documentation
INPL
Nancy-Brabois

Soutenue publiquement le 1er Mars 1989 devant la commission d'examen.

N

| | | |
|---|---|---|
| | URGEAU | Ecole des Mines de Paris |
| Rapporteurs | ATON | CRIN-Nancy |
| | Yves QUENEC'HDU | SUPELEC-Rennes |
| Invités | Jean-Pierre ELLOY | ENSM-Nantes |
| | Jean-Bernard FRANCK | ESIA |
| | René LAVIE | BULL |
| | Jean-Pierre THOMESSE | INPL/CRIN-Nancy |

D 136 037115 5

136037115 5

**INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE**
**Centre de Recherche en Informatique de Nancy**

(T) 1990 DELCUVELLERIE, J.L

# THESE

présentée à
l'**INPL**

pour l'obtention du titre de
**Docteur en Informatique**

par
**Jean-Luc DELCUVELLERIE**
Ingénieur de
l'Ecole Nationale Supérieure d'Electricité et de Mécanique

UNIVERSITÉ BIBLIOTHÈQUE SCIENCES

---

# INGENIERIE DES SYSTEMES D'AUTOMATISATION DE PRODUCTION:

## CONNAISSANCES EN CONCEPTION DES ARCHITECTURES DE SYSTEMES INFORMATISES D'AUTOMATISATION
## et
## OUTIL D'ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES D'ARCHITECTURES REPARTIES.

Service Commun de la Documentation
INPL
Nancy-Brabois

---

Soutenue publiquement le 1er Mars 1989 devant la commission d'examen.

Membres du Jury:

| | MM. | |
|---|---|---|
| Président | Claude LAURGEAU | Ecole des Mines de Paris |
| Rapporteurs | Jean-Paul HATON | CRIN-Nancy |
| | Yves QUENEC'HDU | SUPELEC-Rennes |
| Invités | Jean-Pierre ELLOY | ENSM-Nantes |
| | Jean-Bernard FRANCK | ESIA |
| | René LAVIE | BULL |
| | Jean-Pierre THOMESSE | INPL/CRIN-Nancy |

*à ma femme Isabelle,*
*à Lucie ou Arthur,*
*et à mes parents.*

Les travaux présentés dans ce mémoire de thèse ont été effectués au Centre de Recherche en Informatique de Nancy. Je tiens à remercier M.J-P.Finance, Directeur du CRIN de m'avoir accueilli dans son laboratoire de recherche.

Je tiens à exprimer ma profonde gratitude à M.J-P.Thomesse pour avoir toujours su cadrer les travaux dans une direction qui s'avère aujourd'hui pertinente (les Réseaux Locaux Industriels et les Architectures de SAP), tout en m'accordant confiance, responsabilité et initiative pour orienter les investigations selon ma sensibilité. L'apport théorique de ce document réside en grande partie dans la formalisation d'idées échangées au cours de nombreuses réunions de travail toujours fructueuses. Les résultats ont été obtenus grâce à l'ouverture volontaire de son équipe sur le monde industriel.

Que les membres de cette équipe de recherche soient aussi remerciés pour tous les échanges et les confrontations d'idées, en particulier M.C.Verlinde dont le sujet voisin et l'approche complémentaire ont largement favorisé l'avancée de nos réflexions.

Les résultats présentés dans ce document ont pu être obtenus grâce à l'intégration de l'auteur au sein de la société ESIA, Engineering des Systèmes Informatisés d'Automatisation. Je remercie M.Balança, Président d'ESIA, d'avoir permis cette insertion réussie.

Je remercie M.M.Blanchard d'avoir initié ce projet de recherche sous sa direction au sein de la société VALORIS sous la forme d'un contrat CIFRE jusqu'au 1er Avril 1988. Je remercie également M.F.Billard, responsable Recherche et Développement de SGN, d'avoir intégré cette action dans le programme des recherches effectuées dans le groupe.

Le travail de recherche a été orienté dans cette direction originale grâce à la motivation de M.J-B Franck, responsable de la Direction Technique et Développements d'ESIA. Qu'il soit ici remercié pour son appui constant et sa participation active aussi bien sur le plan technique qu' organisationnel tout au long de ce parcours jusqu'à la rédaction de ce rapport final.

La découverte du métier de concepteur d'architectures de SAP est le fruit de l'implication de M.J.Richet, aujourd'hui ingénieur dans le département Techniques Logicielles et Communications, dans mon travail quotidien au sein de la société ESIA. L'analyse du contexte de travail de l'ingénierie, de la problématique et des différentes contraintes de la conception d'architectures, et de la définition des objectifs essentiels attendus de ces recherches, sont le résultat de sa capacité à communiquer son expérience.

Je tiens à exprimer ma profonde gratitude à Claude LAURGEAU, aujourd'hui professeur responsable de l'option Robotique à l'Ecole des Mines de Paris, d'avoir confié en 1986 à notre équipe de chercheurs du CRIN et du CRAN, alors qu'il travaillait à l'Agence de l'Informatique (ADI), la mission d'écrire le premier livre dédié aux Réseaux Locaux Industriels et d'organiser un séminaire sur ce même sujet. Le travail effectué dans le cadre de cette action m'a permis de découvrir le contenu de ce qui constitue le sujet de ce document. Je le remercie également d'avoir bien voulu présider le jury de thèse.

Je suis reconnaissant à MM.J-P.ELLOY, professeur à l'ENSM, et R.LAVIE, responsable du projet CIM-OSA au sein de la société BULL, J-B Franck, d'avoir bien voulu faire partie du jury de thèse. Je remercie tout particulièrement MM.J-P.HATON, Responsable de l'équipe Intelligence Artificielle du CRIN et Y.QUENEC'HDU, Responsable de la section Conduite de Procédés Industriels de SUPELEC-Rennes d'avoir accepté la charge d'être rapporteurs de cette thèse. Leurs spécialités respectives reflète le double éclairage que j'ai souhaité donner au sujet traité.

Je tiens également à remercier tous les membres de la commission SAP de l'EXERA ainsi que M.M.Desjardins, directeur technique de cette association, pour la confiance qu'ils m'ont accordée et pour l'apport enrichissant des discussions et réflexions menées au cours des réunions de travail.

Je tiens à remercier MM.M.Prido, L-G.Denoeud, J-M.Jolivet, G.Haverlant, B.Berterottière, J-L.Millet, N.Bommelaer, B.Roussel, S.Porte, A.Dussud, H.Vendomele, F.Klinckemaillie, M$^{me}$ M.Le Dall et tous mes collègues et amis de la société ESIA qui par leur participation, leur aide, leurs conseils et leur amitié au cours des différentes actions effectuées dans le cadre de ce programme de recherche, m'ont permis d'accomplir ce travail de façon efficace et agréable.

## Sommaire:

**Introduction**

**1ère partie**
**PROBLEMATIQUE**

chapitre 1: Définition du cadre de travail

chapitre 2: Place des Réseaux Locaux Industriels dans les Architectures de Systèmes Automatisés de Production (SAP)

chapitre 3: Contexte des Etudes d'Avant-Projet en Ingénierie

chapitre 4: Objectifs et démarche

**2ème partie:**
**CONNAISSANCES EN CONCEPTION D'ARCHITECTURE**

chapitre 5: Concepts de modélisation d'une Architecture
1. Modèles de conception des systèmes d'information
2. Modèles d'Architecture de SAP
3. Concepts utilisés en ingénierie

chapitre 6: Formalisation des concepts
1. Concepts de hiérarchie et de communication
2. Représentation des connaissances
3. Formalisation des modèles

chapitre 7: Tâches d'Analyse et de Conception
1. Découpage en tâches
2. Analyse du contexte
3. Structuration Fonctionnelle
4. Conception d'Architecture Opérationnelle
5. Définition de solutions

**3ème partie:**
**ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES D'UNE ARCHITECTURE**

chapitre 8: Objectifs, Définitions et Analyse des Méthodes de Sûreté de Fonctionnement

chapitre 9: Méthode proposée: Procédures d'Analyse et Mesures Correctives

**Conclusion**

**bibliographie**

**annexes:** 1. Exemple d'application: architecture de conduite et suivi de production de verre plat selon le procédé FLOAT-EQUARRI

2. Environnement de développement et présentation de la maquette

**table des matières**

# INTRODUCTION

L'objectif fixé à l'origine de ce travail de recherche était de contribuer à tirer le meilleur parti des nouvelles techniques de communication que constituent les Réseaux Locaux Industriels (RLI). Les enquêtes que nous avons effectuées auprès des utilisateurs ont montré que des progrès importants pouvaient être accomplis dans cette direction. Se contenter de remplacer les liaisons existantes par un câblage en réseau et quelques protocoles adéquats n'est pas suffisant. La complexité croissante des Systèmes Automatisés de Production (SAP) ne permet plus de construire un Système d'Information efficace sans réfléchir à l'organisation de son Architecture. Cette réflexion préalable trop souvent absente ou négligée doit être entamée dès la phase de conception. L'évolution des besoins et des techniques ne permettent plus de se contenter d'une approche empirique. Dans un contexte où, d'une part, l'intégration du SAP avec les autres fonctions études, méthodes, administratives, commerciales, financières, etc... est devenue un véritable enjeu pour l'entreprise, et d'autre part, de nouvelles technologies informatiques (calculateurs personnels, micro-systèmes, stations de travail, réseaux locaux, bases de données) permettent d'envisager de nouvelles organisations réparties, il est nécessaire de reconsidérer le processus de conception globale des Architectures Informatiques des SAP.

Dans ce but, étudier comment et dans quel contexte sont conçues ces architectures a constitué une part importante de notre travail. Ceci a été effectué à travers de nombreux contacts avec des entreprises utilisatrices de RLI et surtout grâce à l'intégration de l'auteur au sein d'une entreprise d'ingénierie informatique dans le cadre d'un contrat CIFRE. C'est sur l'analyse de cas industriels existants que les besoins ont été identifiés. Insistons sur la collaboration recherche-industrie sans laquelle cette démarche originale n'aurait pu être entreprise. Les résultats de ce travail préliminaire constituent la partie de ce document intitulée **problématique**. On trouvera ici notamment une présentation du contexte de travail dans lequel sont effectuées les études d'architectures, et une analyse des symptômes pouvant être attribués à des insuffisances au niveau de la conception de l'architecture. Cette première partie est complétée par l'exposé de la démarche suivie.

Les problèmes rencontrés dans une architecture sont essentiellement basés, soit sur l'absence d'une réflexion globale et cohérente sur l'ensemble de l'architecture, soit sur le manque de rigueur méthodologique de l'étude d'architecture. Le résultat attendu d'une étude d'architecture est la définition des matériels et logiciels de base réalisant le meilleur compromis technico-économique pour supporter les fonctions de traitement de l'information demandées par le prescripteur et pour garantir les possibilités d'évolution du système. L'examen des procédures de conduite et de gestion de projets industriels

d'automatisation montre que cette définition est nécessairement préalable aux étapes de développement des logiciels d'automatisme, de régulation ou d'informatique industrielle, et en particulier préalables à l'étape de spécification fonctionnelle. Or si ces étapes de réalisation sont bien couvertes par les méthodes et outils de génie logiciel, les concepteurs d'architectures ne disposent pas de méthode, ni d'environnement spécifiquement conçus pour cette tâche. Les méthodes employées sont donc propres à chaque spécialiste et fonction de leur expérience acquise. Généralement peu ou pas formalisées, elles ne font que rarement appel à des outils existants car ils doivent, pour être utilisables, faire l'objet d'adaptations aux caractéristiques particulières des avant-projets. Nous avons donc étudié comment faciliter l'expression des besoins et améliorer le passage à la définition de solutions, en gardant à l'esprit de ne proposer que des solutions applicables de façon réaliste dans l'industrie.

Notre contribution s'est pour cela focalisée sur l'étude des connaissances pratiques mises en oeuvre par les ingénieristes et de concepts théoriques issus de différents travaux de recherche. En effet, si la démarche couramment pratiquée apparaît comme empirique à travers les documents d'étude, elle n'en fait pas moins appel à une forme de savoir-faire dont le principal inconvénient est surtout de ne pas être transmissible de façon explicite. Notre contribution est donc basée sur l'identification et la formalisation des connaissances des spécialistes en conception d'architectures de SAP. Elle consiste de plus à formaliser une partie de ces connaissances de terrain à l'aide de concepts issus des travaux menés dans notre équipe de recherche, et à enrichir ces connaissances par des concepts tirés de méthodes et modèles existants, soit dans le domaine des méthodes de conception de différents types de systèmes d'information, soit dans le domaine de la modélisation des architectures de SAP.

L'étude de ces connaissances et concepts s'est effectuée selon une procédure comparable a posteriori aux études d'expertise mises en oeuvre dans le développement de Systèmes à Base de Connaissances. La seconde partie de ce document présente selon ce plan d'approche les résultats de cette étude des **connaissances méthodologiques en conception d'architectures**:

- concepts retenus dans la modélisation des architectures,
- tâches principales de la conception,
- formalisation de ces connaissances.

La mise au point de cette appproche méthodologique a été réalisée par l'étude approfondie de deux applications réelles:

- l'étude d'une architecture de base pour les applications de production de verre plat selon le procédé "FLOAT" à partir des différentes réalisations déjà effectuées par ESIA, (présentée en annexe),

- une étude également réalisée par ESIA sur un Avant-Projet de Conception d'Architecture d'Automatisation et de Gestion des Transferts et Mélanges de Produits pour une raffinerie européenne ("Tank Farm Automation Project: Preliminary System Design"), (présentée au chapitre 5.3).

L'élaboration d'une architecture est un processus complexe, largement itératif, permettant d'aboutir à une solution satisfaisante pour le prescripteur. Et pour être satisfaisante, cette architecture doit présenter différentes qualités dont un niveau de sûreté de fonctionnement suffisant. Nous avons approfondi la tâche d'**analyse de la robustesse aux défaillances d'une architecture**, car si l'analyse de la sûreté de fonctionnement est couramment pratiquée dans des domaines critiques tels que l'armement, l'aérospatial, l'aviation ou le nucléaire, cette tâche est encore loin d'être approchée de façon systématique dans le contexte avant-projet des systèmes automatisés de production. Les connaissances mises en oeuvre dans une maquette développée avec l'aide de techniques logicielles avancées, ont permis de montrer qu'une démarche originale, tenant compte des contraintes particulières du contexte de l'ingénierie, pouvait aboutir rapidement à un résultat démonstratif. Le contenu de la démarche d'analyse, ainsi que ces développements font l'objet de la troisième partie de ce document.

# 1ère partie:

# PROBLEMATIQUE

# chapitre 1:

# DEFINITION DU CADRE DE TRAVAIL: Conception d'Architectures Intégrées de Systèmes Automatisés de Production.

## INTRODUCTION

Ce chapitre présente le cadre du travail de recherche. Nous décrivons le domaine et le sujet en définissant les notions de Système Automatisé de Production et d'Architecture afin de préciser le vocabulaire employé. Cette présentation introduit l'interdépendance entre communication et architecture.

## 1. SYSTEME AUTOMATISE DE PRODUCTION

### 1.1. Définitions

Le terme **système** est un terme très général et désigne selon l'acception de l'Analyse Systémique, *"un ensemble d'éléments en interaction dynamique organisés en fonction d'un but"*. On trouvera dans [Mes 70], [Le M 77] et [De R 75] le développement détaillé des bases de cette théorie. On retrouvera tout au long de ce travail de nombreux concepts "système" assez généraux que nous avons tenté de respecter dans notre approche et que nous supposerons connus: notions d'environnement, d'interaction, de flux, de structure, de système ouvert, de finalité... pour ne citer que les plus significatives. L'idée essentielle que nous nous contenterons de rappeler pour l'instant est l'importance de la **synergie**, c'est-à-dire que *"le tout est plus que la somme des parties"*. La conséquence directe de cette notion est l'importance d'une **approche globale** (et non pas cartésienne), donc à la fois centrée sur les **interactions** entre les composants du système considéré, et intégrant les **objectifs** (la finalité) pour lesquels le système est conçu et organisé.

Notons donc les définitions de système et de sous-système:

| **système**: ensemble d'éléments en interaction dynamique organisés en fonction d'un but. |
|---|

Un "composant", "élément" ou "constituant" est lui-même un système qui peut à son tour être décomposé en éléments. Cette définition récursive s'arrête quand aucune décomposition supplémentaire ne peut être décrite, ou ne présente plus d'intérêt pour l'analyste-concepteur.

**sous-système**: élément constitutif d'un système.
synonymes: **composant, constituant**

Le terme **production** indique que le système a pour vocation de "transformer" ou "manipuler" des flux de **produits** pour augmenter leur valeur ajoutée. On utilisera le terme **produit** dans son sens le plus général pour désigner tant les produits manufacturiers (ou discrets) que les produits continus (fluides, gaz, énergie, etc...). La notion de valeur ajoutée [GRE 85] est également à prendre dans son sens le plus large. Elle recouvre des activités, parfois qualifiées de service dans la mesure où le produit n'est pas lui-même modifié: par exemple le transport (de marchandises ou de personnes), le prélèvement d'informations sur le procédé ou le stockage. Les concepts développés dans ce travail concernent essentiellement les industries de transformation, mais des applications largement basées sur le transport telles que la conduite des mouvements de produit en raffinerie ont été étudiées.

Remarque: *Notre cadre de recherche exclut donc les applications de gestion administrative, bureautiques, calcul scientifique, et les systèmes temps-réels non-concernées par la production (systèmes de pilotage embarqués par exemple).*

Le terme **automatisation** désigne le fait que cette activité de production est contrôlée par un système minimisant les interventions humaines. Le terme "minimisant" est employé à dessein puisqu'il ne s'agit pas d'éliminer ces interventions, mais de les limiter et surtout d'en améliorer la qualité et la fiabilité. Malgré l'utilisation du terme "automatisation", nous considérons en fait tout le système d'information lié au système de production et non pas seulement la "partie commande" au sens [GRE 85]: *"L'automatisation de la production consiste à transférer tout ou partie des tâches de coordination, auparavant exécutées par des opérateurs humains, dans un ensemble d'objets techniques appelé* ***partie commande****."*

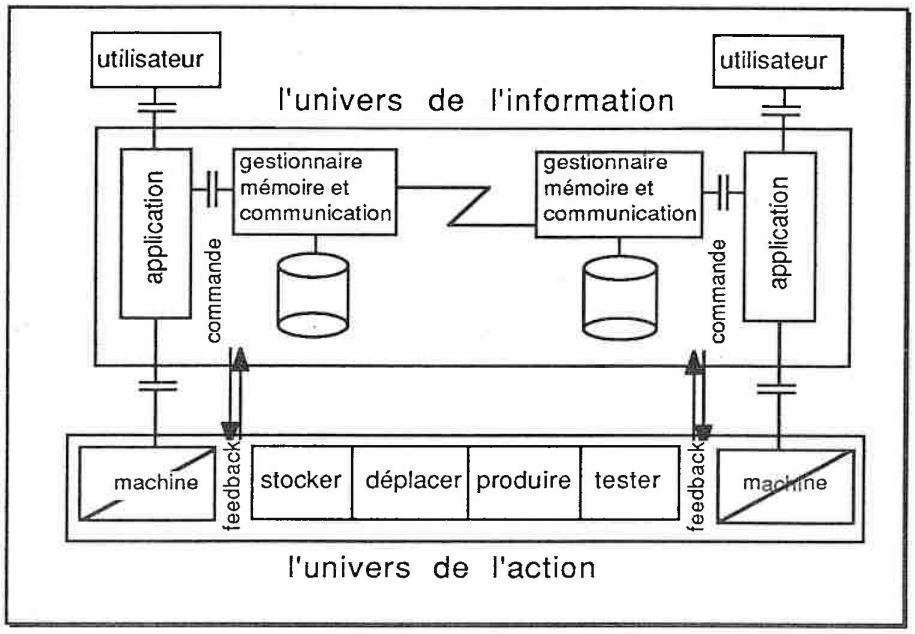
Nous résumerons cette présentation par la définition suivante:

**système automatisé de production**: système destiné à augmenter la valeur ajoutée de produits conformément à des objectifs de productivité et de qualité.

Un **SAP est constitué de deux sous-systèmes** associés respectivement aux **flux de produits** et aux **flux d'informations**, *"l'univers de l'action et l'univers de*

*l'information"* [Huy 85]. (cf. schéma). Différentes appellations sont couramment employées, mais généralement mal définies, pour désigner chacun de ces sous-systèmes:

- processus de production, processus physique, système physique, partie puissance ou partie opérative, moyens de production,
- système de contrôle, de conduite ou de pilotage, partie commande ou plus généralement système d'information, destiné à supporter l'*automatisation* au sens large.




*la vue du SAP dans le projet CIM-OSA [Huy 85]*

Remarque:

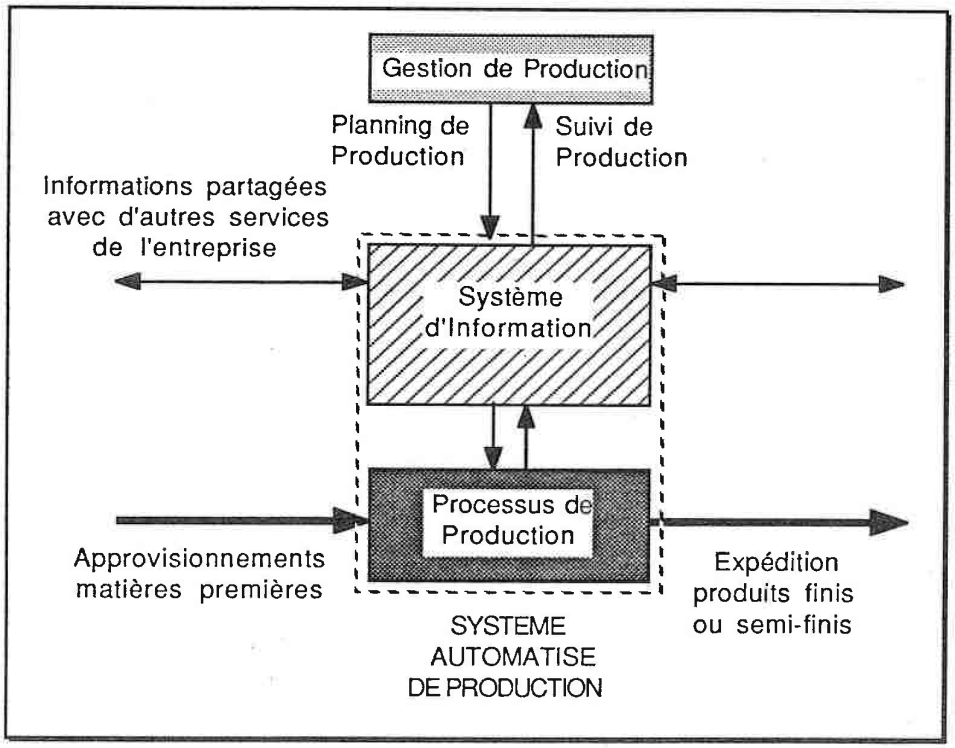
Nous réserverons l'emploi du terme "procédé" pour désigner la connaissance des recettes, gammes indiquant la façon de fabriquer les produits, c'est-à-dire l'enchaînement ainsi que le détail des opérations effectuées sur le produit.

Le terme processus désigne dans ce document l'ensemble des machines destinées à assurer la production conformément à un procédé de fabrication.

Nous retiendrons les termes **processus de production** et **système d'information** en les définissant de la manière suivante:

| **processus de production**: système destiné à effectuer des opérations de transformation, stockage et transport sur des flux de produits. |
|---|

**système d'information**: système destiné à effectuer des opérations de traitement, mémorisation et communication sur des flux d'informations.




*le Système Automatisé de Production dans son environnement*

Remarque: Nous utilisons le terme système d'information pour désigner de façon systémique tous les éléments transformant, communiquant ou mémorisant de l'information indépendamment des fonctions supportées, et non pas seulement sous son acception dans le contexte de l'informatique de gestion: "*artefact, objet artificiel, greffé sur un objet naturel ... conçu pour mémoriser un ensemble d'images de l'objet réel à différents moments de sa vie ...*" [Rol 86a], (acceptation restrictive pour ce qui nous concerne puisqu'elle ne reflète pas la notion de contrôle du comportement de l'objet naturel).

## 1.2. L'environnement du SAP: Production Intégrée par Ordinateur ou Computer Integrated Manufacturing (PIO/CIM)

Tout système se définit en termes d'**interactions** d'une part vis à vis de son **environnement** externe et d'autre part entre ses **composants**. Les paragraphes suivants décrivent l'environnement du SAP et les constituants de chaque sous-système. Cette description générale a simplement pour but de préciser le cadre de travail. Elle ne décrit pas le détail des fonctionnalités et des relations fonctionnelles supportées par le système d'information du SAP que nous analyserons dans la deuxième partie de cette étude.

### 1.2.1. Système ouvert

L'environnement du SAP est constitué soit par les autres services de l'entreprise, soit par des entités externes à l'entreprise avec lesquelles sont établies des relations clients/ fournisseurs. Les limites du système sont spécifiques à chaque application. En décrire exhaustivement de façon générique les fonctionnalités et les frontières se révèle être une tâche complexe au vu de la diversité des domaines d'application considérés. Il est cependant important de préciser pour chaque étude particulière les **interfaces** entre ce que les utilisateurs définissent comme le SAP et son environnement. Un SAP est un **système ouvert** [Le M 77], c'est-à-dire ayant des relations, des échanges, des interactions avec son environnement. Ces échanges correspondent à 2 types de flux: les **produits** et les **informations**. Cette ouverture du SAP sur son environnement est à la base de la notion d'intégration, intégration dont la mise en œuvre se matérialise par le Computer Integrated Manufacturing (CIM) ou en français Production Intégrée par Ordinateur (PIO).

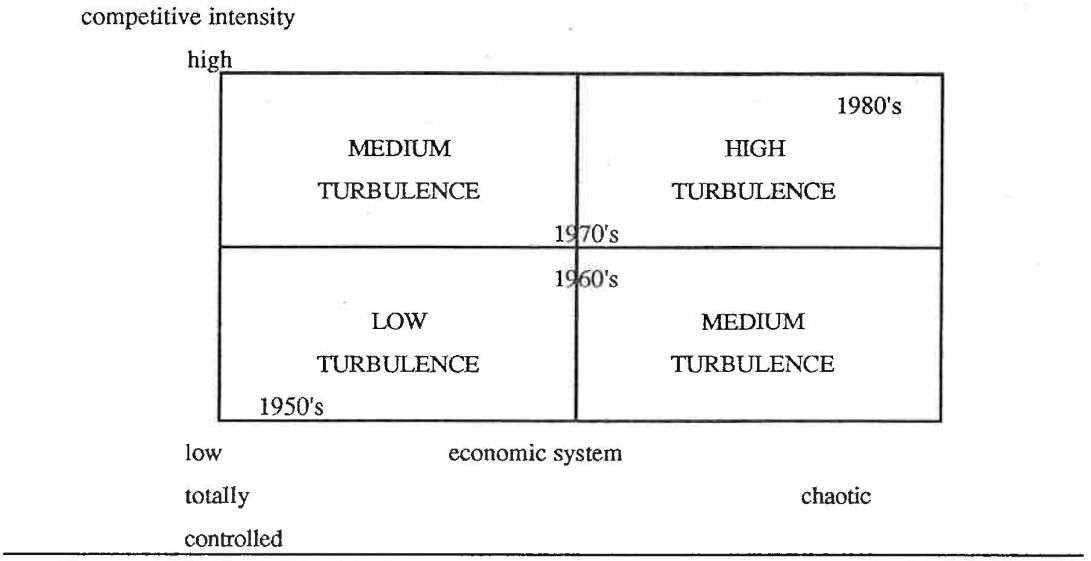
### 1.2.2. Intégration et CIM

La notion d'**Intégration** telle qu'elle est perçue dans le concept PIO-CIM, se réfère à la "***communication** entre les composants d'un système de telle sorte que les entités intégrées d'un système d'ordre supérieur se comportent comme un seul système*" [Huy 85]. Selon [Bur 87] , il faut plutôt parler de "ré-intégration" ou "d'intégration améliorée" dans la mesure où les entreprises sont déjà des systèmes intégrés. En fait les bénéfices attendus de l'intégration sont de promouvoir la synergie, c'est-à-dire d'essayer d'obtenir d'avantage de performances par des améliorations pensées en terme d'intégration que par des modifications individuelles.

La notion de CIM n'est pas nouvelle puisqu'elle a été introduite par Joseph Harrington dès 1973 pour désigner la nécessité de faire communiquer les entités d'une

entreprise. L'évolution de l'environnement conjoncturel international (cf schéma) a fortement contribué à la popularité du concept: confrontées à une concurrence mondiale aiguë dans un système économique perturbé [DEC 88], les entreprises doivent progresser en termes de :

- **performances** économiques (productivité et qualité),
- positions sur le **marché**,
- capacité de **réponse à la demande**.

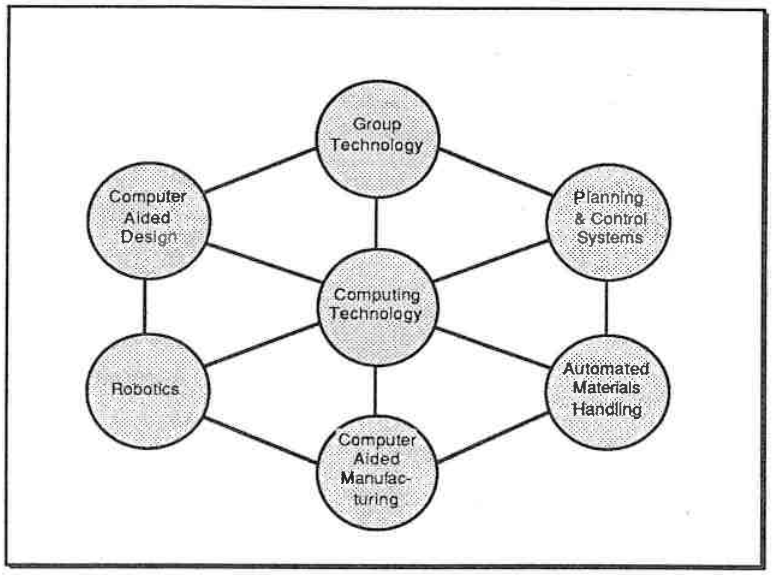



*l'évolution de l'environnement économique vu par DEC [DEC 88]*

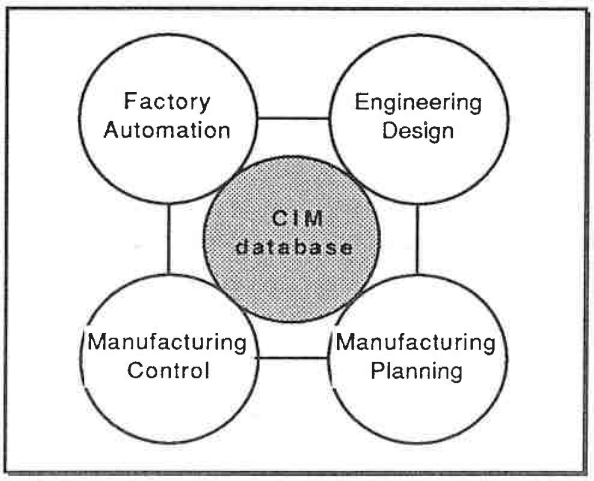
L'objectif du CIM est d'accroître productivité, flexibilité et évolutivité par une synergie basée sur la communication entre les fonctions de l'entreprise. Les différents schémas présentés ici illustrent cette approche [Plo 87] [Gun 86]. Derrière cet objectif global se cachent en fait différentes acceptions du terme CIM. Savolainen propose dans [Sav 88] quatre catégories de définitions:

- étroite (narrow): limitée à la cellule flexible de production intégrée,
- moyenne (medium): intégrant la CAO/FAO,
- large (broad): intégrant le traitement des données d'affaires (business data), et
- stratégique (strategic): décision stratégique et organisationnelle de l'intégration de toute l'entreprise dans son environnement.

On retrouve ces différentes définitions dans l'examen de différents schémas proposés dans la littérature spécialisée. A un premier niveau, les fonctions directement liées à la production sont intégrées autour de la technologie informatique.

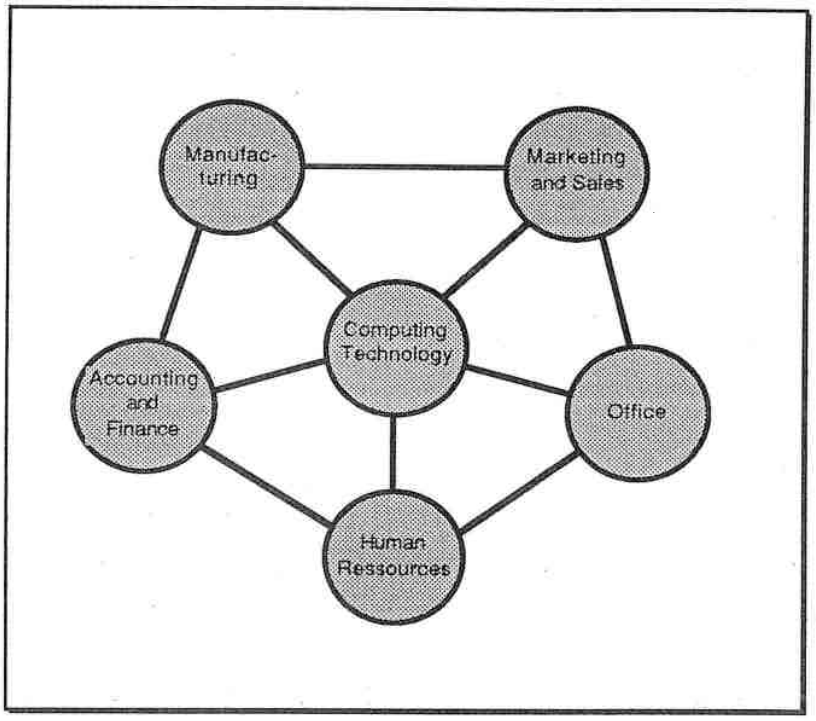



*les fonctions de base du CIM [Gun 86]*




*les fonctions de base du CIM [Plo 87]*

Un second niveau d'intégration concerne toutes les fonctions de l'entreprise.




*le CIM dans l'entreprise [Gun 86]*

Le CIM est maintenant étendu au delà de l'entreprise pour intégrer les relations avec les fournisseurs et les clients.

Dans la pratique, partant du fait que de nombreuses fonctions sont déjà ou peuvent être automatisées et informatisées, cela consiste à interconnecter les équipements informatiques grâce aux produits réseaux normalisés bientôt disponibles tels que MAP et TOP. Mais le simple fait d'échanger des informations ne suffit pas et la nécessité de faire coopérer les fonctions de l'entreprise conduit à **reconsidérer la structure et l'organisation**. Le CIM n'est pas un but en soi mais bien un moyen d'accroître la compétitivité de l'entreprise. Le projet européen ESPRIT n° 688 CIM-OSA (Open Systems Architecture) ("AMICE") a pour but de définir un cadre ("framework") architectural permettant l'élaboration de telles organisations. Nous analyserons cette approche en détail dans la seconde partie de cette étude.

# 2. LES CONSTITUANTS DU SAP

## 2.1. Le processus de production

Le **processus** est constitué d'un ensemble de machines permettant d'effectuer des opérations physiques de différentes natures sur des flux de produits en respectant un procédé donné. Les équipes "procédé" (mécanique, chimique, ...) sont généralement responsables de la définition du processus. Une classification de ces constituants, structurée selon ce point de vue, sera présentée au chapitre "modélisation des processus de production".

Selon une approche plus automaticienne, on peut aussi voir les constituants du processus physique à travers les dispositifs élémentaires de l'instrumentation. Ces dispositifs permettent d'effectuer la transformation d'informations (et d'énergie) en actions physiques et réciproquement d'élaborer des informations à partir de grandeurs physiques. On parlera alors de partie opérative [ADE 82] [Fra 88] composée au niveau élémentaire d'actionneurs et de capteurs.

## 2.2. Le système d'information

Le système d'information (SI) d'un SAP a longtemps été, du point de vue fonctionnel **destiné à élaborer les informations à destination du processus,** d'une part à partir d'**objectifs de production** (plans d'action, valeurs de référence) et d'autre part à partir des **informations recueillies sur l'état du processus** (feed-back). "*L'information distribue la matière et l'énergie dans l'espace et dans le temps*" [Ber 86]. C'est la fonction conduite automatique, contrôle ou pilotage. Mais ce système d'information est de plus en plus sollicité pour "informer" les opérateurs ou utilisateurs, c'est-à-dire **recueillir et synthétiser des informations sur l'état ou l'historique des événements relatifs à l'exploitation du processus**. Ces informations ne sont pas directement utilisées pour définir selon des procédures déterministes les consignes pour contrôler le comportement futur du processus. Mais elles renseignent différents acteurs ou fonctions de l'entreprise, internes ou externes au SAP, soit pour une prise de décision, soit pour des traitements particuliers. Le contrôle-qualité ou le suivi des coûts de production sont des exemples de telles fonctions.

D'un point de vue opérationnel, le système d'information est constitué de dispositifs matériels et logiciels de traitement (automates, régulateurs, calculateurs), de mémorisation (mémoires, disques, bandes) et de communication (lignes, réseaux locaux) supportant les

différentes fonctionnalités du SAP. La particularité des SI des SAP est liée aux caractéristiques suivantes:

- **distribution géographique** des dispositifs de traitement imposant l'emploi de systèmes de communication dits "locaux",
- **hétérogénéité des équipements informatiques** dûs à la multiplicité de fournisseurs spécialisés,
- **fonctionnement "temps-réel"**, c'est-à-dire nécessitant une réaction (on parle de système réactifs) aux événements issus du processus de production dans un délai compatible avec la dynamique du procédé industriel considéré et tenant compte de la durée de validité des données.

Ces SI sont donc des systèmes complexes si l'on considère que *"la complexité d'un système est d'autant plus grande que le nombre de composants du système et d'interactions entre ces composants est élevé"* [Bel 88] . Cette complexité est augmentée par la diversité des composants du SI, diversité des fonctions, diversités des techniques, et diversité des équipements. La complexité du SI rend la conception d'un tel système elle-même très complexe car il n'est pas possible de considérer immédiatement tout le système au niveau de détail requis pour éventuellement l'implanter.

### 2.3. Les hommes

Bien que le Système *Automatisé* de Production soit souvent associé à la limitation des interventions humaines, tous les industriels s'accordent pour témoigner du besoin de la présence de l'homme dans l'exploitation de l'usine pendant encore de nombreuses années. La mécanisation et l'automatisation (la robotique en est l'exemple le plus médiatique) permettent de supprimer certaines tâches fastidieuses. Le travail de l'homme se trouve modifié. Les opérateurs sont ainsi éloignés du processus et placés devant des consoles de conduite. Leurs interventions sur le système sont des interventions d'entretien et de maintenance. De nombreux choix ou décisions ne peuvent encore être pris par la machine seule. Aussi convient-il de ne pas exclure les opérateurs de la description du système mais de les considérer comme partie prenante du SAP, quelque soit leur niveau d'intervention.

# 3. ARCHITECTURE

## 3.1. Définitions

Le terme Architecture concerne à l'origine deux aspects distincts:
- la **structure** d'un système et
- "l'**art de construire**" [Huy 85] comprenant la conception, le développement et la maîtrise d'oeuvre d'un projet.

Très souvent employé sous des acceptions différentes, ce terme a peu à peu perdu sa véritable signification. L'analogie entre le développement des systèmes d'informations et l'architecture au sens le plus classique concernant la construction immobilière, et l'analogie entre le métier de concepteur et celui d'architecte ont permis à [Mc L 87] et surtout [Zac 87] de rechercher le sens original de cette notion. Mais comme le souligne Zachman dans son introduction, *"it probably is not reasonable to expect reconciliation or commonality of definition to emerge from the professional data processing community itself."*

La difficulté vient en partie du fait que l'architecture d'un système d'information **peut** et **doit** être décrite suivant les **points de vue des différents partenaires** et **différents niveaux d'abstraction** en fonction de l'avancement de la spécification. Différents modèles de cette architecture doivent refléter chacun de ces points de vue. Pour la conception d'une maison ou d'un immeuble, on passera de la représentation à l'aide d'un plan de masse avant d'aborder les plans de détail. On utilisera des plans en vue de dessus ou en perspective pour mettre en avant différents aspects de cette future construction. Les responsables de l'électricité ou de la plomberie devront représenter ce qui concerne leur prestation. L'architecture du système d'information devra de même être décrite à un niveau global, puis détaillé, décrite suivant l'aspect flux ou l'aspect données, décrite selon l'aspect fonctionnel ou du point de vue des fournitures matérielles. L'importance de cette notion de point de vue conduit à ne pas parler "d'architecture" en général, mais "d'architecture de système" ou "d'architecture de référence".

Afin d'éclaircir ces notions, nous proposons les définitions suivantes:

| |
|---|
| **structure d'un système**: ensemble des composants et des relations ou interactions que l'on peut expliciter entre les composants d'un système. |

Le niveau minimum de description d'une structure est le graphe dont les sommets constituent les composants et les arcs les relations entre ces composants.

Par conséquent l'on dira qu'un système n'est pas structuré quand on ne peut pas expliciter ces relations de façon détaillée. En revanche, un système "désorganisé" ou organisé sans logique apparente, mais dont on peut décrire les composants et les relations possèdera donc une structure, et sera donc dit structuré. "Structuré" ne signifie pas "bien structuré".

L'**organisation** désigne soit la structure, soit le mode d'organisation de cette structure [Lus 77]. Ce terme est trop général et pas suffisamment employé dans le domaine ici étudié pour être retenu. Nous utiliserons de préférence le terme architecture:

**architecture d'un système**: structure d'un système conforme à certaines conventions relatives à la conception et à la réalisation du système.

Ces conventions proviennent soit d'un savoir-faire, soit de réglementations professionnelles. On dira donc qu'un système est "architecturé", ou qu'il possède une architecture, si et seulement si sa structure est conforme à des conventions de conception. Cette définition intègre donc la notion de "l'art de construire" tout en désignant la structure d'un système.

**conventions**: règles et modèles qui, dans un domaine particulier, définissent et / ou limitent la nature des composants ainsi que les relations ou interactions autorisées entre les composants d'un système.

La différence entre règles et modèles est simple: un modèle est une "photographie" et des règles définissent comment passer d'un modèle à un autre.

**modèle d'un système**: représentation abstraite de la structure et / ou du comportement d'un système existant ou en projet selon un point de vue donné.

Un tel modèle représentant les particularités d'un système donné est appelé un modèle "spécifique" ou "particulier".

**règles**: procédures identifiées de transformation d'un modèle à un autre modèle ou d'un modèle au système réel.

La notion "d'architecture d'un système" diffère de la notion "d'architecture de référence". Une architecture d'un système est la structure d'un système particulier bien identifié: on parlera par exemple de l'architecture du système informatique de l'entreprise "Untel". Alors que l'architecture de référence est ne sont que des conventions générique, c'est-à-dire ne concernent que les caractéristiques générales communes à tous les systèmes du domaine considéré, ceci pour permettre l'élaboration d'architectures de différents systèmes particuliers.

**architecture de référence**: ensemble des conventions régissant la conception de systèmes dans un domaine particulier.

Le terme "référence" indique que les conventions sont adoptées par les concepteurs: ils font "référence" aux conventions pour élaborer leur système. Afin que ces conventions soient adoptées par toutes une profession, elles font parfois l'objet d'une normalisation.

Une architecture de référence rassemble donc à la fois un ou plusieurs modèles de référence et des règle de référence. On fait trop souvent l'assimilation "architecture de référence" - "modèle de référence" car les règles, au sens ou nous les avons définies, sont rarement définies ou normalisées. Ceci donne toute son importance (et sa difficulté) à la certification des normes.

**modèle de référence**: représentation abstraite et générique de la structure et / ou du comportement de systèmes selon un point de vue donné.

Un modèle de référence permet d'élaborer un modèle spécifique par "instanciation". Le modèle d'un système particulier est déduit du modèle de référence.

Le modèle OSI de l'ISO est un exemple de la normalisation d'un modèle de référence des systèmes de communication. L'instanciation de ce modèle s'effectue en choisissant les normes de chaque niveau pour réaliser un "profil fonctionnel" (pouvant faire l'objet d'une "norme fonctionnelle" ou d'une standardisation de fait: exemple les propositions de norme MAP ou FIP).

### 3.2. Architecture et système de communication

Dans la suite de notre travail, nous désignerons de façon limitative par "architecture" ou "architecture de SAP", **l'architecture du système d'information du SAP** et non pas celle du processus de production. Nous supposerons que la structure du processus de production est définie au préalable, ou du moins que nous n'intervenons pas sur cette structure en tant que concepteur. Cela ne signifie pas qu'il n'y ait aucune interaction entre la conception de ces deux sous-systèmes du SAP. Des itérations peuvent et, dans certains cas, doivent être effectuées. La situation, si elle n'est pas souhaitable dans son principe, est cependant telle que cette hypothèse est vérifiée dans la pratique: les responsabilités sont très souvent bien distinctes et l'organisation du planning est très séquentielle. Le processus de production est d'abord défini et le système d'information est ensuite élaboré. Ceci n'empêche pas le concepteur d'analyser la structure du processus de production, non seulement pour prendre connaissance avec le projet, mais aussi parceque la structure du processus est largement déterminante vis à vis de la structure du système d'information.

Toujours en ce qui concerne les conventions de terminologie, nous avons choisi de parler d'**Architecture Opérationnelle** pour désigner l'architecture du système d'information opérationnel résultat de la conception et modélisant la solution. Afin de limiter les risques de confusion, nous parlerons plutôt de **Structure Fonctionnelle** pour désigner le niveau fonctionnel, le terme structure ayant une connotation plus abstraite ou moins "matérielle" que le terme architecture. Nous détaillerons plus avant ces différents termes dans le développement des concepts utilisés en conception.

La définition d'un système d'information suppose la définition de ses différents constituants: les ressources informatiques (matérielles et logicielles), de leurs caractéristiques, et de leur organisation. Le cadre de ce sujet de recherche est issu du problème de l'**Intégration des Réseaux Locaux Industriels dans les SAP.** Il apparaît de façon évidente que l'on ne peut parler d'intégration de réseaux sans parler de conception d'architecture. En effet la définition générale que nous avons donnée de la notion d'architecture introduit la notion de relation. Or les relations entre les constituants d'une architecture d'un système d'information sont constituées:

- soit par les **relations d'appartenance** (relations ensemblistes hiérarchisées entre un constituant du système et ses propres constituants à un niveau inférieur),
- soit par les échanges d'informations.

C'est donc sur l'aspect **communication** que nous avons porté nos efforts, donc sans approfondir d'autres points importants comme la définition des dispositifs de traitement (problème de dimensionnement des ressources calculateurs) ou de mémorisation (problème de répartition des bases de données) des informations. Cette approche est certes restrictive, mais ce choix est expliqué par:

- la nouveauté d'un sujet peu exploré ayant conduit à un travail de défrichage important,
- la nécessité de limiter le champ d'investigation pour aboutir à de premiers résultats significatifs,
- l'importance de la communication dans la structure d'une application répartie de conduite de SAP.

Insistons encore sur ce dernier aspect déjà mis en évidence et exploré en détail dans les travaux de recherche menés au CRIN sur la structuration des applications réparties par l'étude de la communication [Jul 81] [Per 85] [Tho 80] [Der 83] [Zak 84] [El F 85] [Lon 87]. L'acquis de ces recherches, renforcé d'une part par l'importance croissante des communications dans une perspective CIM, et d'autre part par le développement des solutions RLI, justifient ce choix de traiter l'aspect communication.

Le chapitre suivant étudie les relations existant entre la mise en oeuvre de réseaux locaux industriels et la conception d'architectures de SAP.

## CONCLUSION

De ce chapitre consacré à la définition du cadre de travail, nous retiendrons que nous nous intéressons à la partie système d'information du système automatisé de production. Ce système d'information est le lieu privilégié de l'intégration et de l'ouverture du SAP avec l'ensemble des fonctions de l'entreprise dans une perspective CIM. Si la communication est un aspect essentiel de cette intégration, les efforts de conception doivent d'abord porter sur l'architecture du système d'information avant de s'intéresser aux techniques de communication. Une telle perspective passe par une approche globale du SI, et non pas limitée à certaines de ses fonctions (automatismes, supervision, gestion de production, suivi de fabrication, etc...).

# chapitre 2:
# PLACE DES RESEAUX LOCAUX INDUSTRIELS (RLI) DANS LES ARCHITECTURES DE SAP

## INTRODUCTION

Depuis l'annonce de l'initiative MAP [MAP 88] et TOP [TOP 86], des travaux considérables ont été réalisés par les commissions de travail, les centres de recherche et de développement autour des systèmes de communication en milieu industriel. Les projets européens tels que CNMA [CNM 87] et FIP [FIP 88] ont montré la nécessité de compléter l'offre Réseaux Locaux Industriels par des solutions adaptées aux différents problèmes de l'entreprise. Alors que la disponibilité commerciale de produits conformes à ces différents normes est annoncée comme imminente ou à court terme, il est intéressant de faire le point sur les principaux problèmes pratiques rencontrés aujourd'hui par les utilisateurs. Ceci permet d'évaluer dans quelle mesure ces produits répondent aux besoins et quelles sont les difficultés qui subsisteront ou apparaîtront. Au fur et à mesure que des progrès techniques sont accomplis et que les utilisateurs acquièrent confiance et maturité, surgissent de nouvelles difficultés ou de nouveaux besoins. Nous essayerons donc de mettre l'accent sur l'évolution passée et à venir des besoins induits par l'utilisation de ces nouvelles techniques de communication.

Le premier paragraphe montre quels sont les apports des RLI aux premiers besoins ressentis par les utilisateurs d'architectures réparties de contrôle-commande. Le second paragraphe souligne les limites de l'évolution des architectures en matière de structure de données et de contrôle dûes aux insuffisances en services de haut niveau. Le troisième paragraphe explicite quels sont les besoins en conception d'architecture induits par la mise en oeuvre de solutions réseaux.

Cette synthèse est le fruit de plusieurs enquêtes et actions de recherche menées par l'équipe Informatique Industrielle du Centre de Recherche en Informatique de NANCY dans le domaine des RLI dont notamment: une enquête ADI sur l'état de l'art des RLI en 1986 [Tho 86], les travaux menés au sein du Club FIP, des contrats de recherche Université-Industrie sur la modélisation et la conception des architectures de SAP avec la société ESIA [Del 89] en particulier, une enquête sur l'offre RLI au niveau européen en 1987 [] et un projet de formation RLI-Architectures dans le cadre du projet européen COMETT [COM 87] et l'animation du groupe de travail Systèmes Automatisés de Production de l'EXERA*. (Association de EXploitants d'Equipements de mesure, de Régulation et d'Automatismes).

# 1. L'APPORT DES RLI

## 1.1. Remplacer le point à point

Historiquement, c'est le besoin de faciliter et d'améliorer la répartition des architectures de SAP qui a donné sa raison d'être au réseau local industriel. Avant la disponibilité des produits réseaux locaux, le développement des automates programmables et commandes numériques, des micro- et minicalculateurs a permis de mettre en oeuvre des architectures de commande réparties. Les utilisateurs se sont vite aperçu que pour élaborer des systèmes complexes, les réseaux "maillés" construits autour de **liaisons point à point** présentaient des inconvénients majeurs:

- coûts de câblage très élevés et complexité inflationniste des borniers de connexion,
- faible modularité des applications due à la difficulté de partager des ressources,
- multiplicité des protocoles entre équipements hétérogènes,
- surcharge de la capacité des unités de traitement par la gestion des communications,
- évolutivité insuffisante: toute modification ou extension entraînant des problèmes de connexion ou de câblage vite difficiles à résoudre.

Tous ces inconvénients vont rapidement inciter les utilisateurs à ne plus se poser la question: "*Faut-il utiliser un RLI?* ", mais à essayer d'en tirer le meilleur parti. Concrètement deux besoins ont fortement favorisé la pénétration des réseaux dans les entreprises.

### 1.téléchargement

Une part importante des premières applications dans le domaine manufacturier a concerné le téléchargement de paramètres, fichiers ou programmes. Les programmes d'usinage des machines à commande numérique remplaçant les rubans perforés en sont un exemple important, besoin renforcé par le développement des systèmes DNC et de CFAO. Les débits et les contraintes de temps de réponse de la plupart des applications de ce domaine restent généralement faibles:

- messages courts à délai de transfert de l'ordre de la seconde,
- fichiers et programmes en téléchargement off-line.

**2.simplification du câblage**

Dans le domaine des procédés continus, le réseau est venu remplacer tout ou (plus souvent) partie du système de câblage traditionnel dans des installations existantes en cours d'extension ou de "revamping". Les distances importantes entre les équipements et les salles de conduite justifient rapidement le coût d'une solution réseau. Les distances maximales autorisées par un réseau constituent d'ailleurs un des critères de choix. Le nombre d'informations à remonter vers le niveau conduite-suivi atteint fréquemment plusieurs milliers de variables TOR ou analogiques à rafraîchir chaque seconde.

**3.extensibilité**

Lié à la simplification du câblage et de la connectique, la possibilité d'étendre l'application par une "simple" adjonction de calculateurs ou automates sur un réseau existant constitue un besoin largement exprimé par les utilisateurs. Mais cet avantage ne peut être effectif que s'il est possible de connecter une gamme variée d'équipements en provenance de différents fournisseurs.

## 1.2. Interconnecter des équipements hétérogènes

Malheureusement les premières solutions réseaux proposées par les fournisseurs de systèmes étaient fermées. Le développement de passerelles ne s'avère pas une solution satisfaisante tant du point de vue technique que du point de vue économique. La diversité des équipements, la volonté d'une indépendance vis à vis des fournisseurs a donc créé un besoin d'interconnexion de matériels hétérogènes. L'offre de systèmes ouverts est restée au départ très limitée et l'utilisateur trouvait difficilement les logiciels de communication ("drivers", "handlers", "protocoles", etc...) lui permettant de connecter ses équipements. La plupart du temps, il fallait, en collaboration avec le fournisseur, développer au coup par coup les interfaces manquantes, qui étaient ensuite récupérées pour être mises au catalogue. Le principal problème de l'utilisateur est donc devenu la communication d'équipements a priori incompatibles. Deux démarches se sont jusqu'à aujourd'hui développées en parallèle:

- la normalisation de fait, c'est-à-dire la disponibilité de nombreuses interfaces compatibles avec un produit réseau donné,
- la normalisation officielle au sein des organismes nationaux ou internationaux.

La voie de la normalisation étant, du point de vue des utilisateurs, perçue comme longue et sinueuse, un certain nombre d'entreprises indépendantes des constructeurs de matériels informatiques, et en particulier sur le marché français (Aptor, Cerci, Compex, Gixi), ont suivi dès le début des années 80 une démarche pragmatique consistant à proposer un système simple, efficace, bon marché et surtout immédiatement disponible.

D'autres fournisseurs de matériels ont suivi ce mouvement en "ouvrant" leurs produits, pour faciliter la nécessaire connexion d'autres équipements.

De l'autre côté les organismes de normalisation ont entrepris la définition de protocoles. De longs travaux infructueux pour les utilisateurs (PROWAY [PRO 80]), n'ont pas bénéficié du support d'entités officielles nationales responsables telle que les Agence de Télécommunications. L'initiative MAP soutenue par de nombreux utilisateurs rassemblés derrière General Motors, et surtout le ralliement général des constructeurs de matériels, ont finalement permis de créditer la normalisation au niveau international. Cette démarche est sur le point d'aboutir avec la sortie de la version 3.0 et surtout l'annonce de produits compatibles à brève échéance.

Mais dans l'intervalle, la question est devenue pour l'utilisateur :
*"quel réseau choisir?"* , avec pour corollaire l'interrogation:
*"Faut-il attendre les produits conformes aux normes ?"*.
Les applications n'attendent pas et donc ont été choisis des systèmes non normalisés mais disponibles, ayant fait leurs preuves et offrant de nombreuses possibilités d'interconnexion entre les matériels les plus diffusés parmi lesquels on trouve les automates programmables, les machines à commande numérique, ou les micro et mini-calculateurs. En plus de ce critère primordial de disponibilité commerciale, les prix doivent rester raisonnables devant les coûts des équipements connectés. Le succès commercial de certains réseaux témoigne de la prépondérance des critères de coûts et de disponibilité des interfaces pour une grande variété d'équipements sur les critères de performances pures. La confiance et la pérennité estimée du fournisseur (références industrielles, collaboration avec des laboratoires de recherche) sont également des critères de décision souvent cités.

Après cette période transitoire, la convergence des produits réseaux existants et des équipements d'automatisation vers les standards de communication et la baisse des coûts des produits compatibles doit apporter une réponse satisfaisante au problème de l'interconnexion d'équipements hétérogènes. Sous réserve que la compatibilité aux standards puisse être certifiée.

### 1.3. Supporter différents flux d'informations

Le support de communication multipoint que constituent les RLI permet de supporter des échanges de différents types de messages:

- consignes,
- états,

- informations de suivi de production,
- alarmes,
- programmes,
- paramètres,
- fichiers.

Derrière la diversité apparente de ces types de messages se dessinent **trois principaux types de flux** liés à une organisation traditionnelle en niveaux [Gal 84]. (cf. figure ):

1. Entre les capteurs-actionneurs et les automatismes de premier niveau, le flux est essentiellement périodique. Les informations qui transitent sont des objets simples représentant la "photographie" ou l'état à un instant donné de grandeurs ou de dispositifs physiques. C'est surtout à ce niveau que les contraintes de temps (déterminisme des temps d'accès et respect des délais de transmission) sont les plus élevées.

2. Entre les fonctions de premier niveau, et avec le deuxième niveau de coordination, le flux est plus ou moins aléatoire dans la mesure où les instants d'échanges dépendent de la durée des opérations et des instants de lancement. On voit apparaître en plus de données synthétiques relatives au process, de plus longs messages correspondant aux programmes système ou applicatifs destinés à être téléchargés au démarrage ou lors des changements de fabrication. Ces échanges peuvent être effectués en temps masqué, c'est-à-dire "off-line" par rapport au fonctionnement du processus de production, donc sans contrainte de temps importante. Des messages courts de synchronisation de début ou de fin de cycle, échangés entre les automatismes sont les plus contraints en terme de temps de transfert.

3. A partir du deuxième niveau, les instants ou durées d'opération sont de plus en plus aléatoires (comme l'arrivée d'une commande client par exemple) et les messages échangés sont souvent des fichiers ou des données plus volumineuses (planning de production, données de CFAO, synoptiques) nécessitant des débits importants mais des contraintes de temps assez larges.

Ces flux sont actuellement ou seront respectivement pris en compte par :
- les bus de terrain ("fieldbus") tel que FIP,
- les réseaux propriétaires (X-Bus: Modbus, Jbus, etc...) ou MAP-EPA et
- MAP (version complète à 7 couches).

Avec le développement des premières applications, l'avancement des procédures de normalisation et des travaux de recherche, il apparait clairement que même à moyen terme, un seul réseau ne pourra supporter tous les besoins de communication d'une application.

## 1.4. Quels protocoles de communication ?

Ces flux nécessitent l'utilisation de protocoles adaptés à leurs caractéristiques. La nature périodique des flux d'informations d'états entre capteurs, actionneurs et automates nécessite la mise en oeuvre de mécanismes différents de ceux prévus pour les flux associés aux traitements dans les niveaux supérieurs [Gal 84]. Les contraintes particulières de respect des contraintes de temps ont provoqué de longs débats sur la bonne ou mauvaise adaptation des protocoles de niveau 2 (Medium Access Control en particulier) : accès aléatoire ou jeton, norme IEEE 802.3 ou 802.4 ? Au delà des querelles de spécialistes, qu'en pensent les utilisateurs ? Les performances ne constituent pas un critère de choix intrinsèque. Le débit instantané, le comportement sous forte charge, le déterminisme des temps d'accès sont certes limités dans les produits disponibles actuellement mais en général suffisant pour de nombreuses applications. Dans de très nombreux cas l'accès aléatoire ne pose aucun problème. Il a de plus été démontré [Le L 84] que les normes basées sur des protocoles à jeton montrent leurs limites en ce qui concerne les temps de réponse et le déterminisme en cas de perte de jeton.

Cela ne signifie pas que de bonnes performances soient inutiles. Mais pour l'instant les applications envisagées sont quelque peu timides et surtout tiennent compte des possibilités des produits. On ne définit pas l'application en faisant abstraction des produits disponibles. En d'autres termes, on ne déduit pas le choix du réseau en fonction des besoins de communication, un compromis est toujours nécessaire et il faut donc limiter l'expression de ses besoins. Cependant des réseaux à haut débit (Full-MAP à 10 Mbits/s), à faible délai de transmission (MAP-EPA avec un temps de transfert de l'ordre de 20 ms) et des fieldbus permettant de rafraîchir 2000 variables en 100 ms vont permettre d'envisager de supporter beaucoup plus d'informations en respectant de meilleurs temps de réponse. Et comme dans de nombreux domaines nouveaux, les utilisateurs ne tarderont probablement pas à exploiter pleinement ces nouvelles possibilités.

## 1.5. Conclusion

Les RLI apportent donc une réponse satisfaisante aux problèmes de communication en termes:

- d'économie de câblage et de connexion,
- de facilité et de "propreté" d'interfaçage d'équipements hétérogènes,
- de performances de communication dans la mesure où ils restent peu sollicités,
- évolutivité des applications.

Mais quel que soit le type de réseau, il apparaît aujourd'hui clairement que l'utilisateur désire voir le réseau local comme une "boîte noire". Il est essentiellement concerné par les performances globales "vues de l'extérieur". Par rapport aux sept couches de protocoles normalisées dans le modèle OSI [OSI 79], seuls:

- la **couche physique** pour le câblage et la connectique, et surtout
- les services offerts dans la **couche application** pour l'interface avec les programmes de l'application

doivent être connus par l'utilisateur.

## 2. INFLUENCE DES RLI SUR LES ARCHITECTURES

Après avoir examiné la situation sous l'angle communication, il est nécessaire d'examiner plus finement quelle est l'utilisation des RLI, autrement dit de poser la question: *"des Réseaux Locaux, pour quoi faire ?"* . La communication n'est en effet pas un but en soi mais bien un moyen permettant à des applications de **collaborer** avec d'autres applications implantées sur d'autres calculateurs. Nous allons voir que la structure des architectures a encore relativement peu évolué avec l'emploi des RLI. En fait la confiance accordée au réseau local reste faible: on conçoit les applications avec des éléments connectés d'un niveau d'**autonomie** suffisant, d'une part pour qu'une défaillance du réseau n'entraîne pas l'arrêt de toute l'installation, d'autre part comme nous l'avons vu pour que les échanges d'informations soient peu contraignants.

### 2.1. Sûreté de fonctionnement

Le réseau est encore vu comme un **mode commun** du système, et si la fiabilité des communications, c'est-à-dire le risque de perte de messages ou de non détection de message erroné, est satisfaisante grâce aux protocoles élaborés aux niveaux 2, 3, et 4, on craint plus la défaillance physique d'un communicateur ou du câblage lui-même. Ainsi très souvent dans certaines applications, toutes les données nécessaires sont échangées en début de cycle de fabrication pour que la production ne soit pas interrompue par une éventuelle défaillance du réseau. Si on peut estimer que le taux de défaillance et les temps de réparation des éléments de communication assurent un taux de disponibilité supérieur à celui des machines de production, la défaillance des éléments de communication peut dans certaines applications (processus continus en particulier) provoquer des conséquences graves au niveau de la sécurité des installations ou des personnes. Quand les contraintes de sécurité ou de disponibilité sont importantes, une solution **redondante** est choisie pour le câble et/ou les communicateurs. Concernant cet aspect sûreté de fonctionnement, jusqu'à présent **rien n'est précisé dans les projets de normalisation**. Il est donc probable qu'en l'absence de normalisation, certains constructeurs proposeront des adaptations ou des extensions aux normes avec toutes les difficultés que cela suppose au niveau interopérabilité.

### 2.2. Quelle répartition ? Traitements, contrôle, données ?

L'analyse des applications existantes avec cette modélisation permet de constater que dans les architectures réseaux actuelles, les liaisons de contrôle-synchronisation ne sont peu ou pas véhiculées par les réseaux locaux. Seuls des flux de données sans couplage fort passent par le réseau. Les architectures sont distribuées (ou réparties) mais ne sont pas décentralisées: en effet les traitements sont encore conçus de façon autonome pour que le contrôle et les données ne soient pas répartis mais centralisés.

L'évolution des structures peut être résumée par les étapes suivantes:

| traitements | contrôle | données |
|---|---|---|
| - locaux séquentiels mono-tâche, | local, | locales, |
| - multi-tâches monoprocesseur, | centralisé, | centralisées, |
| - sur des unités parallèles distribuées, | centralisé, (situation actuelle) | serveurs centralisés, |
| - distribués | distribué, | distribuées. |

## 2.3. Besoins en services de haut niveau

### 2.3.1. Coopération

Malgré la répartition des traitements, les architectures restent donc basées sur des structures de contrôle hiérarchisées et des structures de données centralisées. La raison en est simple: n'ont été jusqu'à présent résolus que les problèmes de **communication** et d'**interconnexion**. En effet on s'est occupé de la façon de transmettre des messages sans considérer la signification (la sémantique) et l'utilisation de ces messages. Les services actuellement normalisés de façon stable et complète s'arrêtent aux niveaux réseau et transport. La possibilité d'élaborer de nouvelles architectures est directement liée à la disponibilité de services réseaux évolués. Cela suppose une définition précise de ces besoins: échange de fichiers, transmission de consignes, synchronisation de tâches distantes, scrutation périodique de données, partage de données réparties, etc... De tels services permettront de concevoir de véritables applications réparties, non seulement communicantes mais aussi **coopérantes** (l'"interworking" ou interopérabilité par opposition à l'"internetworking" [McC 88]). Actuellement les réseaux disponibles ne fournissent que des services de transfert de fichier (FTAM dans la version 2.1 de MAP) ou des émulations de terminaux. Cela signifie peu de coopération et beaucoup de duplication d'informations. Aussi le développement d'applications réparties passe encore par le développement de fonctionnalités supplémentaires au sein même de l'application.

Après avoir été longtemps délaissée, la couche application fait l'objet de travaux encourageants dans le cadre des deux projets de normalisation MAP et FIP.

### 2.3.2. Services de Messagerie Temps-Réel

Les services MMS [Ell 88], [ISO 87] (Manufacturing Message Specifications) de la couche application de MAP recouvrent la gestion:

- de domaines,
- de variables et de types,

- d'événements,
- d'équipements virtuels de production,
- de contexte,
- de programmes,
- de journaux,
- de sémaphores, et
- de communication opérateur.

Les fonctionnalités attendues de la couche application sont proches de celles d'un **système d'exploitation réparti** (doit-on parler de "couche 8" ?). Et de même que très peu d'applications nécessitent le développement d'un système d'exploitation spécifique pour les calculateurs, des fonctionnalités d'un système d'exploitation réseau standard devraient énormément simplifier le développement d'applications véritablement réparties.

### 2.3.3. Bases de données temps-réel réparties

La distribution du contrôle doit être accompagnée de la **distribution des données**. Par exemple rapprocher les données des utilisateurs qui les consultent le plus fréquemment tout en permettant des consultations distantes plus occasionnelles suppose des services plus évolués que le transfert de messages ou de fichiers. Une "modularisation" des systèmes de traitement de l'information nécessite l'exploitation **cohérente** de données communes. La notion de **base de données répartie temps-réel** suppose:

- des performances suffisantes pour satisfaire les contraintes du procédé, mais surtout,
- une gestion explicite de la notion de temps:
- durée de validité des données,
- fréquence (ou pseudo-fréquence ) de mise à jour,
- la garantie de la cohérence spatiale et temporelle des données.

Une illustration de ces concepts est donnée par les travaux sur la couche application de FIP [FIP 88]. Ils définissent des services périodiques/apériodiques (MPS) de lecture et écriture, mise à jour de variables réparties et périodiquement rafraîchies, de synchronisation, de validité de transmission et de cohérence sur des listes de variables.

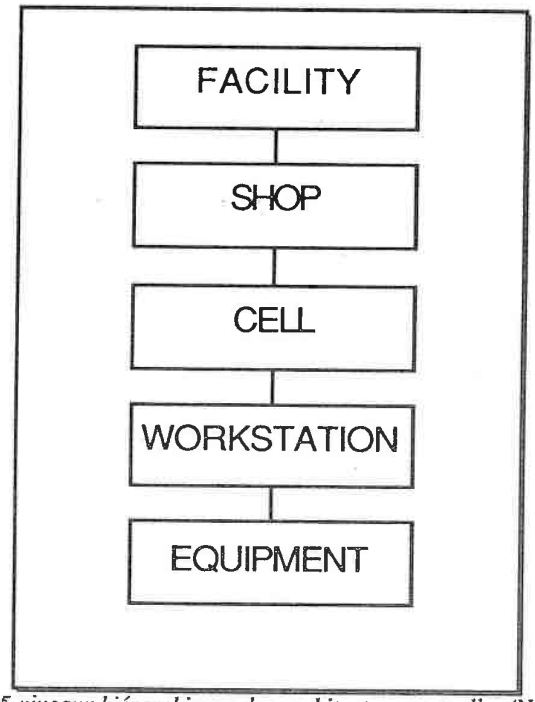
Ces services ne constituent cependant qu'un premier pas, les systèmes d'exploitation temps-réel et les structures de bases de données réparties faisant encore l'objet de recherches théoriques importantes[1]. De tels systèmes de gestion de bases de données réparties et a fortiori à caractère temps réel devraient constituer un enjeu

important dans le développement des applications industrielles réparties et l'application du concept CIM [Le L 88].

### 2.4. De nouvelles architectures

Examinons de plus près quelles sont les évolutions prévisibles en matière d'architecture.

De plus en plus de fonctions informatisées ("assistées par ordinateur") sont disponibles et doivent être intégrées au système d'automatisation de la production. La conduite automatique, le suivi de production, le contrôle-qualité, la maintenance, la gestion de production doivent partager des informations issues du process et des informations issues de l'environnement extérieur. Ces fonctions dont les objectifs sont différents ne peuvent être mises en relation de dépendance hiérarchique. Au contraire, elles doivent pouvoir être intégrées de façon modulaire. **Le schéma traditionnel d'architecture** [Wil 87], [Mc L86] **en niveaux hiérarchisés ne fournit pas une telle structuration.**



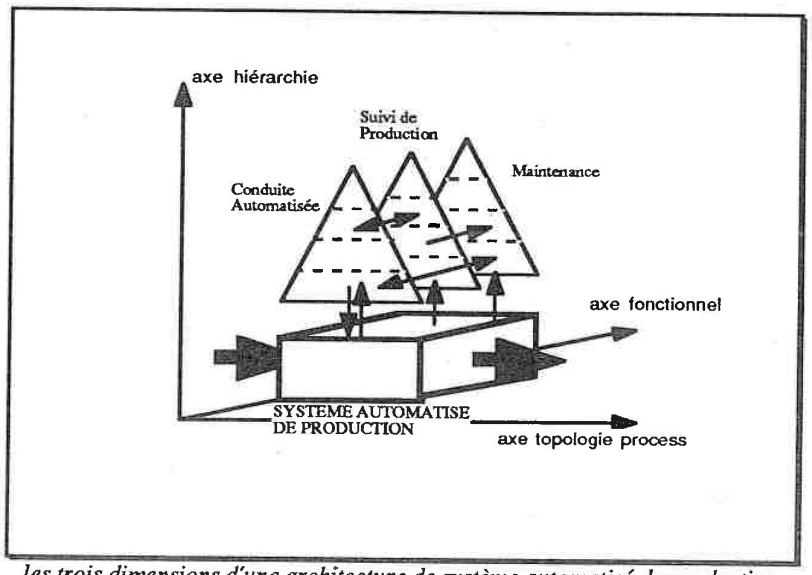

*les 5 niveaux hiérarchiques des architectures actuelles (NBS)*

Nous proposons de représenter dans un nouveau modèle d'architecture les "**plans fonctionnels parallèles**" (eux-même hiérarchisés) capables de partager des informations [Del 88]. Ce schéma apporte donc une dimension fonctionnelle supplémentaire aux deux dimensions traditionnelles:

- **topologie du processus** (c'est-à-dire selon la décomposition du processus en sous-systèmes) et

- **niveaux hiérarchiques.** (cf. schéma).

Si actuellement ces différentes fonctions doivent être regroupées au sein de mêmes calculateurs pour éviter de trop nombreux et contraignants échanges, une implantation à terme sur des équipements distincts, éventuellement spécialisés (intelligence artificielle, serveurs de base de données, calcul parallèle, traitements graphiques) facilitera le développement, l'évolutivité et la maintenabilité des applications. De façon analogue à l'évolution en informatique traditionnelle, cette approche permet de concevoir des architectures issues des fonctions et de l'organisation des entreprises, et non pas calquées sur les systèmes informatiques.




*les trois dimensions d'une architecture de système automatisé de production*

## 3. BESOINS EN CONCEPTION D'ARCHITECTURES INDUITS PAR LA MISE EN OEUVRE DE SOLUTIONS RESEAUX

Si l'on suppose ces services disponibles à court terme, le problème de l'utilisateur devient : "*Comment maîtriser les flux d'informations de mon application ?*", "*Comment assembler les différents produits réseaux disponibles ?*" et "*Quels sont les services qui correspondent à mes besoins ?*" Ceci suppose une bonne définition préalable de ces besoins et de la structure de l'application, autrement dit suppose la définition de l'architecture des applications de SAP.

Nos travaux de recherche [Del 89], [Ver 89] nous ont logiquement amené à étudier comment étaient conçues les architectures des SAP. Une solution d'architecture est généralement le résultat d'une succession de développements ponctuels liés aux décisions successives d'investissement. Chaque sous-ensemble est le résultat d'une proposition en réponse à un **appel d'offre**, mais une réflexion globale sur la cohérence de l'architecture est rarement effectuée. Dans de rares cas une **étude préalable** est entreprise à cet effet. Mais quelque soit la procédure, nous avons pu constater que les méthodes de travail n'étaient pas ou très **peu formalisées**, en partie à cause de l'importance des données économiques, commerciales et stratégiques. L'absence de méthodologie dédiée spécifiquement à la conception d'architecture et intégrant les contraintes de délais et de coûts de ces pré-études conduit la plupart du temps les concepteurs à se baser sur leur expérience, heureusement importante, pour **raisonner essentiellement en terme de solutions**. Trop souvent les sources d'insatisfaction résultent d'une approche pas assez globale mais au contraire trop partielle privilégiant tantôt le point de vue des automatismes, tantôt celui de l'informatique, ou de la gestion de production, etc... La difficulté consiste à concilier ces ponts de vue souvent contradictoires.

### 3.1. Symptômes de défaut de conception d'une architecture

Il nous a paru intéressant d'analyser les principales sources d'insatisfaction, les principaux signes de non-qualité d'une architecture relatifs à un défaut de conception. Pour chacun des symptômes significatifs, nous avons voulu vérifier l'existence de causes que l'on peut affecter aux difficultés de conception et non à la gestion de projet, au développement ou à la mise en oeuvre. Ce travail est délicat car il est pour des raisons évidentes très difficile d'obtenir des informations précises sur les cas à problèmes. De plus la complexité inhérente aux architecture de SAP et à leur processus de développement rend délicate l'identification précise des causes exactes d'insatisfaction. Ce travail n'a donc pas pu être bien formalisé mais il permet d'illustrer les difficultés auxquelles sont confrontés les concepteurs.

### 3.1.1. dépassement budgétaire et les retards à la mise en oeuvre

Le dépassement budgétaire et les retards sur les dates contractuelles sont un des principaux symptômes. Une étude générale sur les difficultés des études d'ingénierie [Bru 88] cite le retard au démarrage comme un indicateur intéressant de la performance du projet. Sur les différentes études analysées, l'auteur constate que *"c'est un phénomène qui touche massivement les projets actuels"*. *"Les pertes de production engendrées par ces retards peuvent atteindre une valeur significative du montant de l'investissement."* Fort heureusement ces symptômes sont surtout révélateurs d'une maîtrise insuffisante de la gestion de grands projets: sous-estimation des ressources nécessaires, manque de suivi d'avancement.

Mais un découpage inadapté au développement et surtout à la mise en oeuvre (tests, intégration et démarrage) accentue les difficultés éventuelles. Les dépassements peuvent aussi être dus au choix de techniques ou produits mal appropriés ou à un dimensionnement inadapté conduisant à des "acrobaties" coûteuses en développement. Ces erreurs s'expliquent généralement par une analyse trop rapide des besoins et des performances requises par les composants matériels et logiciels de l'architecture. Il est toujours difficile de déterminer la part de responsabilité des partenaires d'un projet. "Les engagements sont-ils non-tenus par le fournisseur ou les spécifications sont elles insuffisantes ?" La définition précise des prestations constitue donc un "garde-fou" à la fois pour le client et son fournisseur. Du point de vue de l'ingénierie, il est en effet important de pouvoir se "border" c'est-à-dire de délimiter précisément le contenu des prestations afin d'éviter des dérives techniques entre le contenu de l'offre sur laquelle est passée la commande et le résultat des spécifications fonctionnelles précédant la réalisation. Ce risque est d'autant plus grand que le rédacteur de la proposition est rarement le chef de projet responsable du contrôle de ces spécifications fonctionnelles. Celui-ci peut donc difficilement refuser les desiderata du client à ce niveau. Si la solution proposée par l'ingénieriste doit préserver l'évolutivité, il faut aussi être capable de délimiter ce qui était prévu dans la prestation.

### 3.1.2. Inadaptation à l'exploitation

Les difficultés d'utilisation de l'application par les opérateurs sont souvent le fruit d'une mauvaise prise en compte des conditions d'exploitation. La définition des dialogues utilisateurs est souvent mise en cause: il est en effet très difficile d'évaluer la qualité des interfaces homme-machine sur le papier, a fortiori pour de nouvelles applications. La structure globale de l'application: centralisée, répartie, doit également être cohérente avec l'organisation du travail d'exploitation. La définition des solutions techniques occulte trop souvent l'analyse de l'organisation et des fonctions. La définition des modes d'exploitation ne repose sur aucune méthode globale et rigoureuse au niveau des études

des architectures. Les Guides des Modes de Marche (GEMMA) [ADE 84] restent peu utilisés et leur contribution est limitée aux automatismes séquentiels.

### 3.1.3. Disponibilité insuffisante

Un des principales faiblesses d'une architecture se traduit par une disponibilité insuffisante ou une charge de maintenance trop importante. L'évaluation de la sûreté de fonctionnement d'une architecture complexe est difficile à réaliser en phase d'avant-projet. Le concepteur se retrouve en effet démuni, ne pouvant, faute de temps et d'informations suffisantes faire appel aux méthodes classiques utilisées en fiabilité. En découlent des risques d'erreur en ce qui concerne:

- les choix de répartition des fonctions sur les équipements informatiques répartis: structure trop fortement maillée conduisant à des relations de dépendance entre les sous-systèmes ;
- les spécifications de fiabilité et de disponibilité des composants matériels: débouchant sur la question classique "faut-il doubler un automate ou un réseau de communication ?",
- la définition des modes d'exploitation par les opérateurs en cas de défaillance d'un sous-système.

Faute de méthode et d'outil adéquat le repérage des éléments critiques est difficile à réaliser.

### 3.1.4. Coûts d'évolutivité prohibitifs

Un autre symptôme est la charge de travail que représentent les évolutions de l'application par modification des programmes mais surtout par adjonction de fonctionnalités devant partager des données avec l'application existante. Très souvent l'analyse des besoins est limitée au court terme et conduit à des architectures ne préservant pas l'avenir de l'entreprise. Le rôle de l'ingénieriste est de veiller à la prise en compte des évolutions prévisibles dans le contexte de l'application. Ces besoins se traduisent le plus souvent par des possibilités d'intégration, d'ouverture, le choix de produits conformes à des standards et une structuration la plus modulaire possible.

## CONCLUSION

Cette situation n'est bien-sûr plus tolérable dans une perspective CIM où l'architecture devra garantir flexibilité et évolutivité des systèmes. Si les utilisateurs que nous avons rencontrés prennent conscience de ce problème, force est de constater qu'aucun programme de recherche ne s'est consacré directement à ce problème de conception préalable tout à fait différent du problème du développement et de la réalisation des applications (qui concerne l'axe génie logiciel). En effet aucune méthode ne prend en compte de façon globale et intégrée toutes les composantes à la fois techniques et non-techniques du problème. En plus d'une représentation fonctionnelle adéquate destinée à la répartition des fonctions et l'identification des flux d'informations et de leurs caractéristiques, il est nécessaire d'étudier et de prendre en compte de nombreuses et diverses données du problème. Dans le simple but d'illustrer cette diversité, nous citerons avant de les étudier en détail dans la seconde partie de ce document:

- les installations existantes,
- les contraintes d'environnement, les contraintes liées à la topologie des installations, aux procédures, règlements et autres normes,
- les objectifs et stratégies exprimés dans le cadre du schéma directeur de l'entreprise,
- les ressources financières et humaines consacrées au projet,
- la portée des choix d'architecture sur les coûts et délais de développement et de mise en service des systèmes,
- le dimensionnement des systèmes de communication,
- l'adéquation des services de communication avec les besoins d'échanges d'informations et de coopération entre les applications,
- la robustesse aux défaillances de l'architecture pour atteindre les objectifs de sécurité et de disponibilité.

Nous rappellerons ici rapidement l'évolution que nous avons perçue et les efforts qui restent à faire.

L'évolution de la situation est marquée par les étapes suivantes:

- la simplification des connexions par le remplacement du point à point par le réseau et la simplification des communications par l'emploi d'un seul protocole au lieu de n protocoles spécifiques,
- l'apparition des concepts de "services de niveau application",
- des échanges de données temps réel mais dans des architectures très centralisées,
- la disponibilité de normes, services et produits.

Mais pour faire passer les idées dans les faits, pour que cette disponibilité se traduise par des applications industrielles, il faudra bien sûr le temps nécessaire pour que ces concepts parviennent à maturité, la difficulté de faire évoluer le parc des applications existantes et le temps nécessaire au développement de méthodes et d'outils tirant parti de ces nouvelles possibilités. Restent à résoudre des problèmes tels que:

- comment choisir une architecture plutôt qu'une autre, intégrant tous les aspects évoqués dans le chapitre précédent et
- comment intégrer les différentes méthodes et outils existants qui ne résolvent souvent qu'un des aspects d'un problème CIM ?

C'est à ces questions que nous proposons d'apporter notre contribution.

# chapitre 3:

# CONTEXTE DES ETUDES D'AVANT-PROJET EN INGENIERIE

## INTRODUCTION

Nous avons vu dans le chapitre précédent en quoi la bonne intégration des RLI dans les applications de Contrôle-Commande de SAP supposait un travail important de définition de l'architecture. Ce chapitre synthétise l'étude que nous avons faite du **contexte de travail de l'ingénieriste** chargé d'effectuer de telles études d'architectures. Précisons que dans ce document nous désignerons de façon limitative par ingénierie, la partie ingénierie automatique et informatique industrielle, c'est-à-dire la fonction d'une entreprise ou d'un service interne chargé de la conception d'applications d'automatisation et d'informatisation de production. Ceci ne recouvre donc que les systèmes de commande et le système informatique, et suppose définis par ailleurs les autres éléments de l'ingénierie productique [Bla 85] :

- le système de production proprement dit,
- le génie civil et
- la définition des procédures d'exploitation et de maintenance.

L'objectif est d'établir un constat de situation pour **identifier** les **besoins** des utilisateurs et les **contraintes** en matière de méthodes de conception. Nous nous sommes efforcés de considérer de façon globale l'ensemble des paramètres techniques et économiques du problème. Ce chapitre reprend, en les détaillant de façon plus approfondie, certaines des conclusions déjà publiées dans le rapport OURAI [Del 87] .

## 1. ETUDES D'AVANT-PROJET

Les prestations de base effectuées en informatique industrielle peuvent être classifiées suivant leur position par rapport à la décision d'investissement en:

- **études préalables** à une décision d'investissement,
- travaux de **développement** de logiciels d'application entrepris après cette décision.

Nous conviendrons de considérer la première catégorie de prestations comme des prestations d'ingénierie, et de considérer la seconde comme des prestations de société de service.

Dans presque tous les cas, puisque le développement suppose un contrat comprenant la définition des équipements informatiques qui supporteront les logiciels , la liste des fournitures doit être établie au préalable. Habituellement deux procédures permettent de définir une architecture :

- procédure par **appel d'offres** lancé par le client: l'architecture est alors le résultat de la proposition,
- procédure par **étude préliminaire** de définition d'architecture.

La différence essentielle entre ces deux procédures est pour l'ingénieriste la non-rémunération directe des études relatives à l'élaboration d'une réponse à un appel d'offre. Le nombre moyen de sociétés consultées pour un appel d'offre étant d'une dizaine, la proportion moyenne des propositions retenues dépend ensuite des performances commerciales de l'entreprise. Mais le **risque de non-rémunération** n'est donc pas négligeable. La rédaction du document d'appel d'offre peut être établie par le client (un service d'ingénierie interne) ou faire appel à une ingénierie tierce qui en principe et pour des raisons évidentes d'impartialité ne sera pas consultée.

Les études préliminaires sont établies soit par l'ingénierie interne, soit par une société extérieure, la prestation s'apparentant alors à une prestation de conseil. Notre travail a donc consisté à étudier ces prestations d'avant-projet auxquelles nous associerons d'autres prestations qui peuvent contenir une partie étude d'architecture:

- études de faisabilité,
- diagnostic,
- audit,
- rédaction de cahier des charges,
- schéma directeur,
- etc...

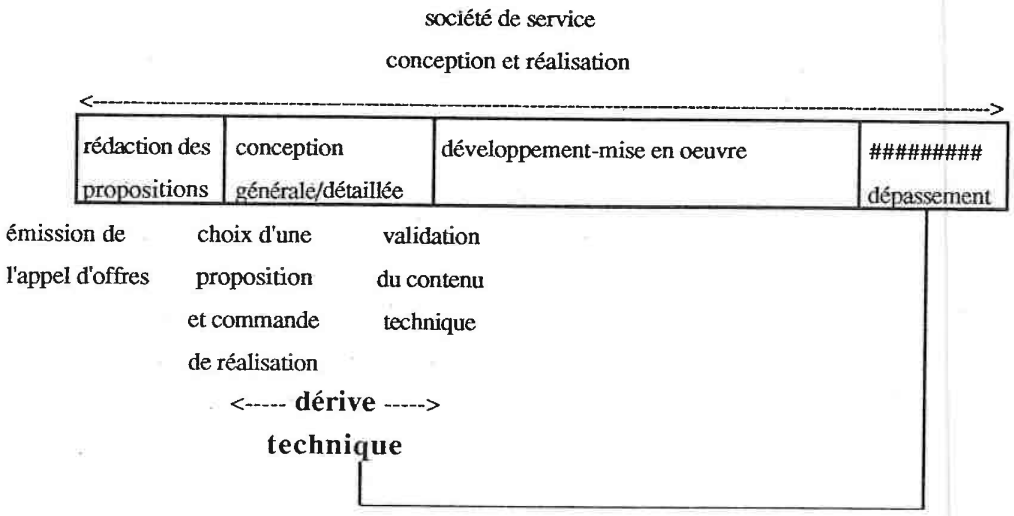
## 2. CONTRAINTES LIEES AUX APPELS D'OFFRES

Une différence importante entre appel d'offres et étude préliminaire d'architecture est liée à l'**étendue du domaine** sur lequel porte l'étude. En effet si nous avons défini le SAP comme le système global recouvrant l'ensemble des fonctionnalités attachées au processus de fabrication, les appels d'offres portent rarement sur un champ aussi large. La plupart des applications concernent un sous-ensemble du SAP: un atelier, une cellule ou une ligne de fabrication particulière, mais rarement tout le processus entre la réception des matières premières et l'expédition des produits finis. De même les fonctions du système d'information sont souvent limitées à un sous-ensemble fonctionnel: par exemple la conduite sans s'occuper de maintenance et de suivi de production; ou encore seulement certains niveaux comme la supervision, les automatismes ou l'ordonnancement.

En revanche, l'étude préliminaire d'architecture a pour vocation de reprendre la réflexion à un niveau global à un niveau intermédiaire entre le schéma directeur de l'entreprise et l'étude détaillée d'un sous-ensemble pour appel d'offre. Cette nuance est à prendre avec précaution car il existe des procédures d'appel d'offre portant sur un SAP complet et des études d'architecture (certes mal nommées) limitées à un sous-ensemble du SAP.

Les difficultés liées à cette procédure sont plus facilement perceptibles si on en détaille le déroulement. Dans la procédure classique, la commande est passée à partir d'une proposition retenue parmi les différentes offres. Cette proposition repose sur un cahier des charges fourni par le client et quelques entretiens de clarification. Des choix techniques sont donc effectués sans études fonctionnelles détaillées. Entre cette commande aux implications juridiques et le résultat des études fonctionnelles définissant le contenu technique détaillé des prestations peuvent donc apparaître des dérives importantes. Les risques inhérents à cette procédure sont donc importants:

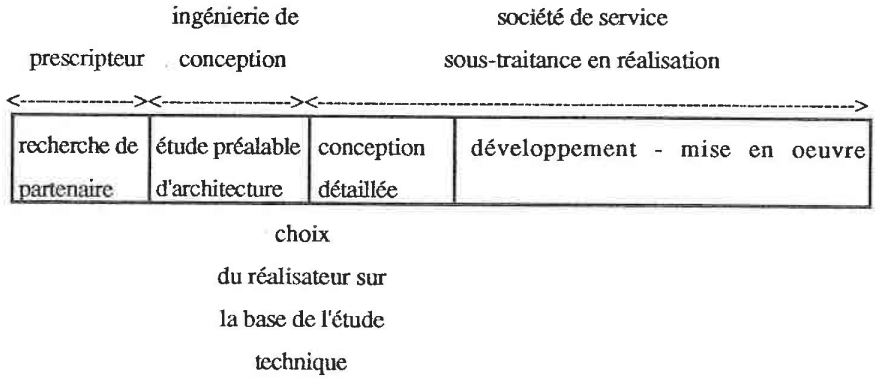
- difficulté d'estimer la charge de travail que représentent les développements,
- dérive technique liées à des difficultés non-détectées dans les propositions,
- sous-dimensionnement de la proposition du fait de la pression commerciale liée à l'appel d'offre.




*la procédure traditionnelle d'appel d'offres*

Afin de limiter ces risques, une autre procédure est de plus en plus suivie par les utilisateurs. Elle consiste à diviser la procédure en deux phases:

- une première phase porte sur l'étude technique de l'application par un partenariat entre une ingénierie et le client (le résultat de cette étude est une définition de solution et une évaluation budgétaire de cette solution),
- une deuxième phase porte sur la réalisation de cette solution sans remise en cause par une société de service.




*la procédure d'appel d'offres après étude préalable d'architecture*

Cette procédure permet de minimiser les risques de dérive technique en retardant les choix de solutions et facilite l'évaluation de la charge de travail par une connaissance plus approfondie de l'application. La solution est définie en rapport avec les besoins et contraintes budgétaires du client, et non pas artificiellement alignée sur les offres concurrentes.

## 3. DELAIS

Les délais accordés à ces études sont généralement très **courts**, en particulier pour les réponses à appels d'offre. La durée moyenne est de l'ordre du mois, et peut atteindre 3 à 6 mois pour les études préliminaires, mais elle dépend beaucoup de la complexité des applications. Cette rapidité est dûe au fait que la définition de l'architecture matérielle et logicielle conditionne tout le développement de l'application. En effet la commande des matériels et logiciels de base doit être effectuée suffisamment tôt pour que les délais de livraison ne retardent pas le planning du développement de l'application. Cette contrainte est très importante car elle interdit d'attendre la fin de la phase d'analyse fonctionnelle pour définir l'architecture.

Cette interdiction est également renforcée par le **coût important** que représente une analyse fonctionnelle menée suivant les méthodes et outils du génie logiciel informatique ou automatique. Ce coût est en effet estimé à 30 % du coût total de développement d'un projet. Il n'est bien sûr pas envisageable pour une entreprise d'ingénierie d'investir autant dans la définition préalable d'une architecture si l'étude n'est pas financée directement ou si l'obtention du contrat de réalisation de l'application est trop aléatoire. Cet objectif est d'ailleurs cité parmi les principes directeurs de l'étude préalable définie dans MERISE: "*éviter de dépenser en pure perte un coût d'étude couvrant de façon exhaustive la spécification de solutions, alors que la décision de les mettre en oeuvre n'est pas acquise.*" [Tar 85]

De plus si une analyse fonctionnelle complète peut paraître dans l'absolu l'approche idéale (puisqu'elle définit de façon précise, formelle et détaillée l'ensemble des fonctions de l'application), l'étude des informations pertinentes nécessaires à la définition d'une architecture nous a montré (cf chapitre sur les données de base) qu'une telle analyse fonctionnelle classique, d'une part fournissait beaucoup trop d'informations, et d'autre part ne prenait pas en compte des composantes importantes du problème (des données non-techniques en particulier). La difficulté de ce type d'étude consiste donc à bien poser le problème sans rentrer dans des détails qui n'auraient pas d'incidence sur les choix d'architecture. Autrement dit la tâche du concepteur est d'abord de **repérer** les **données pertinentes essentielles**. Toutes les données ne concernent pas la conception de l'architecture, mais interviendront sur les phases de développement du logiciel par exemple. Cette notion de pertinence est à rapprocher de la notion de point de vue déjà évoquée dans le chapitre de "Définition du cadre de travail".

## 4. EXPRESSION DU BESOIN DU CLIENT

Comme dans tout étude de conception, l'expression du besoin est souvent délicate. La **formulation** par le client est presque toujours **imparfaite**, c'est-à-dire plus ou moins incomplète, ambiguë, ou incohérente puisqu'informelle. Ce phénomène est normal si l'on considère que le premier métier du client n'est pas de spécifier son besoin et de connaître toutes les finesses des outils de spécification, mais d'exploiter et gérer ses installations. Le rôle de l'ingénierie est donc d'assister son client dans la spécification de ce besoin afin d'**assurer la qualité de cette spécification.** Il n'est bien sûr pas question de remplacer le client dans l'expression du **contenu** des besoins.

Le travail de collecte d'informations exige une expérience et une polyvalence importante pour établir le dialogue avec tous les interlocuteurs qui sont concernés par la définition d'un SAP: chef de projet, automaticiens, mécaniciens, concepteurs des produits, exploitants, équipes de maintenance et entretien. Or suivant les métiers, suivant les secteurs d'activité ou les particularités de l'entreprise, un vocabulaire différent peut désigner des concepts analogues ou un vocabulaire identique peut désigner des concepts différents. Les travaux de réflexions menés par les utilisateurs issus de différents secteurs professionnels rassemblés dans le groupe SAP de l'EXERA ont montré la nécessité de constituer sinon une normalisation du vocabulaire, du moins un glossaire commun.

Une des principales difficultés à la bonne communication client-ingénieriste est liée à la **confusion fréquente entre la spécification des objectifs et la description des moyens** nécessaires [Sou 85] . Cette constatation n'est pas nouvelle[Lus 71]: "*Trop souvent ces derniers (les clients) au lieu d'analyser la structure de leur entreprise, l'éventail de leurs exigences, l'articulation de leur système de valeurs, en un mot: de définir clairement leurs besoins, se contentent de comparer les spécifications techniques des matériels qu'on leur présente, spécifications d'ailleurs très voisines, sans se demander si d'autres outils, mieux adaptés à leurs besoins, ne pourraient être créés.*" Une approche fonctionnelle explicite telle que celle prônée par les normes X50-151 AFNOR sur la rédaction de Cahiers des Charges Fonctionnels doit clarifier cette situation. La compréhension client-concepteur est d'autant plus importante dans les cas où la nouveauté nécessite une analyse plus approfondie que les cas déjà connus résolvables par analogie à des réalisations précédentes.

## 5. EVALUATION D'UNE ARCHITECTURE

Plus encore que les difficultés d'expression et de communication, le **manque de critères objectifs d'évaluation** d'une architecture complique la tâche du client et de l'ingénieriste. *Peut-on dire qu'une architecture est "bonne" ou "mauvaise"* ? Dans la mesure ou une architecture est un compromis entre des besoins et des contraintes, une telle question n'a pas de sens. Plus précisément, si certaines applications ne donnent pas satisfaction, est-on capable de dire s'il s'agit d'un véritable problème d'architecture et non pas un problème de qualité de programmation ou des matériels informatiques, ou encore un problème de formation par exemple ? Cela suppose d'être capable de définir ce **qu'est un problème d'architecture.** Or pour l'ingénieriste la réponse à cette question signifie savoir justifier une solution, savoir argumenter et se démarquer face aux offres concurrentes. Plus concrètement il faut pouvoir justifier le prix d'une architecture dans un environnement concurrentiel et pouvoir lutter contre le phénomène du "moins-disant" fonctionnel que cachent souvent les solutions les plus économiques sur le papier. Pour le client cela signifie savoir qualifier, différencier et donc choisir parmi les différentes propositions celle qui répond le mieux à son besoin. Comme une architecture est toujours le résultat d'un compromis entre les besoins et les moyens disponibles, on ne peut pas parler de solution optimale dans l'absolu. Il faut définir un ensemble de critères d'évaluation conformes aux objectifs et à la prise en compte des besoins et contraintes propres à chaque application.

La définition de tels critères est de plus un préalable indispensable si on souhaite expliciter les enseignements des applications réalisées pour accroître le savoir-faire des équipes de conception.

## 6. METHODES DE TRAVAIL

L'examen des méthodes de travail utilisées en ingénierie (sociétés d'ingénierie ou services internes des utilisateurs finaux ou des fournisseurs de matériels d'automatisation), montre que ces **méthodes** sont **peu formalisées** mais se traduisent essentiellement sur le terrain par :

- une organisation du travail (plan d'étude),
- une standardisation plus ou moins poussée des documents produits,
- parfois par l'emploi ou l'adaptation de méthodes issues des techniques de génie logiciel (fréquemment des diagrammes de flux ou SADT, Grafcet ou organigramme).

D'une façon générale les méthodes de travail s'apparentent plus à celles employées dans les différents domaines des prestations de conseil ou "consulting" (organisation ou stratégie) qu'aux méthodes plus formelles du développement logiciel. Cette situation résulte du fait qu'une partie importante des informations déterminantes dans la définition d'une architecture sont des données non-techniques tant économiques que stratégiques très difficiles à formaliser.

De plus le constat dressé dans [Bin 78] s'applique encore tout à fait : "*... la plupart des analystes de systèmes n'ont eux-même pas une idée très claire de la structure de leur tâche. Ils sont habituellement surchargés et se débrouillent avec leur travail quotidien car dans la plupart des sociétés, il y a toujours un arriéré de travail dont les priorités sont définies par leur degré d'urgence. Ainsi ils suivent une série d'objectifs à court terme plutôt qu'une méthodologie d'analyse de systèmes.*" Effectivement les experts en conception ont rarement l'occasion de prendre du recul par rapport à leur métier. Et très peu d'entreprises consacrent des budgets à l'étude et la formalisation de méthodes d'avant-projet.

La principale conséquence de la très faible utilisation de méthodes formelles pour ces études est le **manque d'explicitation du "quoi"** et du **"pourquoi"**, le raisonnement se faisant trop souvent directement en terme de solutions matérielles, c'est-à-dire d'équipements informatiques (le "comment"). Ce phénomène est accru par le fait qu'il est très rare que le client n'évoque pas des éléments de solutions dans l'expression de son besoin.

Cette faible utilisation de méthodes rigoureuses constitue un frein à la **réutilisation quasi nulle** des études précédentes. Dans le meilleur des cas, la réutilisation est limitée

au plan de l'étude et à la structure des documents, ainsi qu'à la présentation de la société et de ses références.

Enfin le travail de conception reste **non formalisé**, confiné individuellement dans le cerveau des spécialistes. Leur expérience ne peut donc pas être ni capitalisée, ni partagée. Il existe actuellement peu de formations à la conception des architectures de SAP.

## 7. COMPLEXITE CROISSANTE DES ARCHITECTURES

En fait si jusqu'à présent le manque de méthodologie explicite n'a pas fortement été ressenti, cela est dû à l'accélération récente de l'accroissement de la complexité des applications étudiées, au sens où nous l'avons définie dans le premier chapitre. En effet tant que la dimension des applications était relativement limitée (dans la mesure où ces applications concernaient seulement des sous-ensembles du SAP et étaient plutôt isolées: "îlots d'automatisation"), le problème de la conception d'architecture se résumait à un problème de dimensionnement de la capacité de traitement des calculateurs, de débit des systèmes de communications ou de capacité d'entrées/sorties. L'évolution actuelle et à venir des applications SAP complique le problème de la conception. Cette évolution tient essentiellement à trois facteurs:

- développement de **nouvelles fonctionnalités** *(contrôle qualité, maintenance assistée, etc...),*
- **intégration** du SAP avec tous les autres fonctions de l'entreprise *(concept CIM),*
- et essor de la **répartition** autorisé par l'augmentation du rapport performances/coûts des matériels informatiques *(micros, stations de travail, automates, etc...)* et la disponibilité de systèmes de communications locaux *(RLI).*

Une approche **globale** du système et de ses interactions internes et externes (approche systémique) et **structurée** (une approche descendante en particulier) permet de maîtriser convenablement cette complexité. Or on constate que les délais de mise en oeuvre ont tendance à conduire à une définition et une réalisation séparée des différents niveaux fonctionnels: pour respecter la date de démarrage de la production, les automatismes de conduite sont développés en priorité, donc indépendamment des niveaux supérieurs qui sont par la suite difficilement greffés sur l'architecture. Comme trop souvent des équipes différentes (automaticiens et informaticiens) sont chargées des développements, la cohérence de l'ensemble est alors difficile à garantir.

Le concepteur a donc besoin d'**outils intellectuels** (les méthodes et leurs supports) lui permettant:

- de mieux gérer la **masse** et la **diversité** des informations,
- d'avoir une **approche globale** de l'application et
- de mieux maîtriser cette nouvelle **complexité**.

## 8. BESOINS

Sans rentrer dans le détail de l'étude bibliographique du chapitre consacré aux concepts d'architecture, il faut souligner que nous n'avons découvert aucune recherche spécifiquement dédiée à ce problème de conception d'architecture dans le domaine productique en contexte d'ingénierie avant-projet. Ceci ne signifie pas, comme nous le verrons, qu'aucun élément présent dans d'autres domaines de conception ou dans différents modèles d'architecture ne puisse être utilisé ou transposé, mais que l'ensemble du problème n'a pas à notre connaissance été abordé dans sa globalité. L'intérêt d'une méthode plus rigoureuse est cependant d'autant plus ressenti que dans le domaine voisin du développement informatique, les méthodes formalisées en génie logiciel et leurs supports, les ateliers logiciels sont de plus en plus reconnus et utilisés dans l'industrie.

## CONCLUSION

A travers cette présentation du contexte de travail de l'ingénierie dans lequel sont effectuées les études d'architecture, nous retiendrons la nécessité d'améliorer l'expression des fonctionnalités à l'aide d'un formalisme adapté aux délais courts, la nécessité de ne s'intéresser qu'aux données pertinentes parmi la masse d'informations relatives à un projet complexe et le besoin de suivre une approche globale. En conclusion, nous pouvons synthétiser quels sont les besoins des utilisateurs vis à vis de notre programme de recherche:

- **accroître le couple qualité-productivité des études d'architecture,**
- **fournir des critères d'évaluation,**
- **capitaliser le savoir-faire des spécialistes en conception d'architecture.**

# chapitre 4:

# OBJECTIFS ET DEMARCHE

## INTRODUCTION

Ce chapitre synthétise à partir de la problématique détaillée dans les chapitres précédents, les objectifs de ce travail de recherche et la démarche suivie. Nous montrons ici en quoi un travail de recherche à caractère méthodique rejoint une démarche de type acquisition de connaissances. La convergence entre ces deux approches est présentée et justifiée. Le dernier paragraphe précise le plan de présentation des deuxième et troisième parties de ce mémoire, plan précisément choisi pour mettre en évidence cette convergence.

## 1. OBJECTIFS

L'objectif initial de ce travail de recherche est de contribuer à combler le vide méthodologique concernant spécifiquement la conception des architectures de SAP. Un principe de base fixé dès le début de cette étude était la réutilisation maximale de tout ce qui répondrait au problème posé (méthodes et outils existants). En analysant le détail de la problématique et du contexte de travail, nous nous sommes aperçus que le manque d'application de méthodologie rigoureuse ne signifiait pas l'absence de savoir-faire, d'expertise, de différents concepts, bref de connaissances en conception d'architectures, mais plutôt un manque d'explicitation et de formalisation de ces connaissances. Aussi cette étude a consisté à:

**étudier les différentes sources de connaissance utilisées ou utilisables par les ingénieristes dans le processus de conception d'architecture en phase d'avant-projet pour contribuer à la définition d'une méthode et d'outils d'assistance.**

Ceci afin :

- **d'accroître le couple qualité-productivité des études d'architecture,**
- **de faciliter la définition et l'utilisation de critères d'évaluation,**
- **de capitaliser le savoir-faire des spécialistes en conception d'architecture.**

Techniquement, ces connaissances méthodologiques doivent apporter une réponse aux **besoins** suivants:

- susciter une approche globale du problème,
- faciliter le repérage des données de base pertinentes essentielles du problème,

- fournir un support méthodique (organisation, modèles, outils) facilitant la gestion de la complexité,
- proposer une démarche suffisamment souple pour être applicable aux grands et petits projets.

## 2. DEMARCHE SUIVIE

Le constat du manque de formalisation et de support informatique des méthodes de conception d'architecture de SAP étant établi, deux types d'approches pouvaient être suivies:

- élaborer une nouvelle méthode de conception ,
- étudier les procédures mises en oeuvre par les ingénieristes.

Les paragraphes suivants présentent les justifications du choix d'une approche intermédiaire.

### 2.1. Caractéristiques des connaissances en conception d'architectures de SAP

**L'absence d'une théorie générale** et de modèles mathématiques rend difficile la première approche. La complexité du domaine considéré interdit de garantir l'optimalité d'une architecture puisque les différentes décisions de choix ne peuvent être prises simultanément mais seulement selon une approche progressive, c'est-à-dire pas à pas. Les décisions prises à un niveau donné contraignent celles prises aux niveaux suivants [Bie 88]: *"Il est difficile de prévoir l'impact d'une décision prise à un haut niveau d'abstraction, telle que par exemple les fonctionnalités souhaitées, sur les décisions de plus bas niveau, telles que le coût du matériel supportant cette fonctionnalité. Il est de plus impossible de décider dès le départ si une fonctionnalité est rentable. La procédure de conception doit donc plutôt viser une solution satisfaisante qu'une solution optimale."*

Les **données** disponibles à ce stade d'avant-projet, à la fois **incertaines, incomplètes** et **imprécises**, peuvent difficilement être manipulées par une méthode formelle telle qu'on peut l'envisager au stade des spécifications. La difficulté de mettre entre les mains des concepteurs de tels outils conduit également à délaisser cette approche dans l'état actuel des recherches. Cet aspect conduit les concepteurs à suivre des procédures de **raisonnement qualitatif**: par exemple qualification des variables pertinentes ayant une portée significative sur les choix d'architecture, qualification des ordres de grandeur de ces variables, etc...

On peut de plus constater que malgré l'absence de méthode formelle, et en dépit de certains cas d'échec ou d'insatisfaction que nous avons analysés, de nombreuses architectures complexes donnent des résultats satisfaisants. Ceci nous a conduit à poser l'**hypothèse de l'existence d'une forme de savoir-faire** que nous nous proposons précisément d'identifier et d'analyser. Quand le seul savoir-faire nous est apparu comme étant insuffisant sur certains aspects, nous avons introduit d'autres formes de connaissances adaptées au contexte dont nous avons analysé les contraintes. Le

principe de l'étude d'une méthode et du développement d'outils correspondant a donc été de recenser et d'intégrer trois formes de connaissances:

- les **connaissances théoriques** s'appuyant sur les **méthodes** existantes dans des domaines adjacents à la conception d'architecture de SAP,
- les **connaissances formalisées** dans les **modèles d'architecture de SAP,**
- les **connaissances pratiques** issues du **savoir-faire** et de l'expérience acquise sur le terrain par des **spécialistes** de l'ingénierie des systèmes informatisés d'automatisation.

Ces deux premières formes de connaissances ont permis d'enrichir les connaissances pratiques.

La démarche retenue est donc bien intermédiaire par rapport aux deux orientations présentées au début de ce paragraphe puisque nous avons étudié les procédures mises en oeuvre par les ingénieristes et nous les avons enrichies et consolidées par des connaissances plus théoriques ou mieux formalisées pour aboutir à une nouvelle méthode de conception.

### 2.2. Expertise et intelligence artificielle

Parmi les domaines d'application privilégiés des techniques d'intelligence artificielle et des systèmes à base de connaissances sont classiquement cités [Zwi 87], [Bar 87] , [Bar 88], [Kim 88]:

- la conservation du savoir-faire d'une entreprise,
- support adéquat de la modélisation du raisonnement qualitatif,
- l'augmentation de la productivité et de la compétitivité des experts,
- la résolution de problèmes techniques complexes difficilement modélisables,
- comportant beaucoup de données, en général mal définies, incomplètes et/ou contradictoires.

Les bénéfices concrets retirés du développement sont aussi [Rub 87] :

- la réduction des tâches répétitives,
- la réflexion sur la nature des connaissances et des processus de décision,
- l'amélioration de la qualité des décisions,
- l'adaptation à la mise à jour rapide des connaissances.

*"De plus, loin d'être difficiles, ces nouvelles techniques ont un fort contenu intuitif, et sont la plupart du temps plus proches de la façon de penser de l'ingénieur que les traditionnelles méthodes numériques basées sur des modèles mathématiques."* [Bar 87]

L'objectif de nos travaux n'a pas été exprimé au départ comme le développement d'un "système expert d'aide à la conception". Un Système Expert n'est qu'une technique parmi d'autres dont le choix ne doit pas être fait a priori sans avoir étudié la nature du problème posé.

Quelque soit le domaine considéré, l'étude d'une méthodologie de conception suppose l'acquisition et la formalisation de connaissances, qu'elles soient scientifiques ("deep knowledge") ou empiriques ("heuristique")[Mon 88]. Le développement des techniques de l'Intelligence Artificielle et des Systèmes-Experts en particulier a conduit à approfondir les recherches théoriques sur les méthodes d'acquisition et les modèles de formalisation des connaissances. Afin de souligner le fait que notre travail a consisté en l'étude de connaissances, nous avons choisi de présenter nos résultats selon ce type d'approche même si notre démarche a été sur le terrain plus empirique.

## 3. DEVELOPPEMENT DE SYSTEMES A BASE DE CONNAISSANCES (SBC)

La suite de ce chapitre expose donc:
- en quoi notre travail s'apparente à une identification de connaissances,
- quelles sont les principales idées (méthodes, modèles) relatives à l'acquisition de connaissances et au développement de SBC,
- le plan de présentation retenu.

### 3.1. Systèmes-experts et conception

Les outils de conception tant dans les domaines classiques de la conception de produits, que de la conception d'installations de production, de bâtiments, ou celui du génie logiciel, ont surtout porté sur l'amélioration et l'automatisation de l'édition graphique, parfois avec une aide à l'évaluation des structures. Ces outils sont souvent qualifiables de Dessin Assisté par Ordinateur DAO plutôt que de véritable CAO. Le développement des systèmes-experts permet aujourd'hui d'envisager une nouvelle génération d'outils de conception dépassant l'aide à la modélisation interactive pour prendre en compte les processus même de la conception et ainsi formaliser, puis capitaliser l'expertise des concepteurs. Une telle approche a été ainsi étudiée par [Mon 88] dans le domaine du génie civil pour la conception thermique de logements. Selon cet auteur le SE doit pouvoir apporter:
- une aide à la conception,
- une optimisation des ressources disponibles,
- une vérification des solutions,
- plusieurs représentations des connaissances,

- plusieurs alternatives à une même spécification.

Le SE doit, pour devenir opérationnel, être de plus:
- interactif,
- graphique,
- permettre à l'utilisateur de maîtriser les décisions,
- rendre transparent pour l'utilisateur les détails de l'implantation, et
- être utilisé fréquemment, c'est-à-dire être soit très général pour pouvoir traiter de nombreux cas, soit au contraire très spécifique pour résoudre un problème type.

Tous ces objectifs et arguments peuvent être directement appliqués au domaine que nous étudions. En effet la conception peut être considérée comme un problème générique dont l'objectif est de *"construire un système satisfaisant une spécification donnée"* [Boy 88] . Ces auteurs expliquent qu'un système de conception quel qu'il soit se caractérise par ses entrées, les buts à atteindre, les contraintes du domaine, les opérateurs permettant de changer d'état.

*Les **entrées**:*
- *la spécification du système en donnant ses caractéristiques, fonctions, contraintes, composantes, budgets, etc...*
- *des tests analytiques standards sur le système conçu ou projeté,*
- *les composantes possibles, leurs propriétés et dépendances;*

*les **buts**: produire un système qui rassemble les spécifications;*

*les **contraintes**:*
- *les tests peuvent être nombreux et difficiles à sélectionner ,*
- *les tests peuvent être coûteux et irréversibles,*
- *les tests peuvent être peu fiables et imprécis;*

*les **opérateurs**:*
- *sélectionner les formes globales,*
- *sélectionner les détails des composantes,*
- *inférer les propriétés du système désiré à partir des spécifications établies,*
- *rassembler les composantes en sous-systèmes,*
- *vérifier les spécifications (caractéristiques, contraintes, coûts),*
- *faire les tests analytiques afin de prédire les comportements,*

- *faire évoluer et mettre à jour le système conçu à un instant t, enregistrer les raisons des décisions prises et des dépendances des détails, et maintenir la satisfaction des contraintes,*
- *créer, représenter et utiliser de nouvelles composantes et des nouvelles contraintes,*
- *observer les marches à suivre pour mettre en oeuvre des procédures efficaces,*
- *considérer des critères non-économiques comme la sécurité, la protection de l'environnement, l'esthétique, etc...*

## 3.2. Recueil d'expertise

Ce paragraphe précise quelles ont été nos sources de connaissances et comment se situe la formalisation de ces connaissances par rapport aux méthodes utilisées pour le développement de systèmes à base de connaissances. **Notre contribution, cherchant à formaliser concepts et procédures, revient à élaborer le ou les modèles conceptuels relatifs au processus de conception d'architecture.** Nous devons d'abord nous doter d'un outil de conceptualisation formelle des connaissances. Et ensuite l'utiliser pour décrire les connaissances en conception d'architecture. En d'autres termes nous "concevons un système de conception" à base de connaissances.

### 3.2.1. Sources de connaissances

Deux sources de connaissances sont généralement disponibles [Dor 88] :
- connaissances exprimées directement par un groupe d'**experts**,
- connaissances extraites de **documents** de nature quelconque: rapports, publications, schémas, compte-rendus de réunion, etc...

Dans notre travail, la **première source de connaissances** a été fournie par les experts responsables des études de conception d'architecture. Ces connaissances ont été étudiées:
- par entretien avec ces concepteurs,
- par examen des documents relatifs à des études effectuées par ESIA: rapports d'études de faisabilité, diagnostics des causes d'insatisfaction d'architectures, différents cahiers des charges et réponses à appel d'offres,
- en assistant ou participant au déroulement de certaines de ces études.

En parallèle une étude bibliographique a constitué la **deuxième source de connaissances** provenant des **méthodes de conception** employées dans des domaines voisins du génie logiciel informatique ou automatique, de la conception des systèmes d'informations et de l'analyse de systèmes.

La **troisième source de connaissances** provient des **modèles d'architecture de SAP** proposés par différents centres ou projets de recherche, ainsi que par des organismes de normalisation.

Certains concepts et procédures pertinents sont bien sûr ici mieux formalisés et donc plus faciles à identifier que ceux relatifs au savoir-faire implicite des experts

### 3.2.2. Développement de systèmes à base de connaissances

Nous rappelons dans ce chapitre les principales approches théoriques développées dans le domaine général de l'acquisition de connaissances et le développement de systèmes à base de connaissances, ceci afin de préciser la position de notre démarche.

#### Acquisition de connaissances

Les premiers travaux effectués sur "l'extraction" des connaissances ont surtout porté sur le processus de communication entre l'expert et le "cogniticien" chargé de l'implantation directe des connaissances dans le formalisme d'implantation (on parle de développement par prototypage rapide). Ainsi les premières méthodes ont été constituées d'un ensemble de conseils basés sur les techniques d'entretien (reformulation, incompréhension, contradiction [Mah 87]). L'analyse linguistique du discours de l'expert et l'utilisation de différents concepts en anthropologie ou en taxonomie scientifique constituent une première approche de systématisation de l'acquisition des connaissances: méthode KOD, "Knowledge Oriented Design" [Vog 88].

D'un point de vue pratique, l'expertise brute (non-formalisée) est recueillie [Mah 87]:

- par **transmission** des connaissances à un "**cogniticien**":
  - par **observation** en situation (et étude de scénarios ou cas types),
  - par **interview** dirigée (suivant ou non un guide d'entretien résultat synthétique informel de de l'expérience acquise par les "cogniticiens": voir par exemple la "Decision Network Method" de [Dor 88]),
- par **transfert** direct des connaissances par l'**expert**: familiarisé aux techniques de l'IA ou utilisant un générateur de SE utilisable par des non-spécialistes,
- ou par **apprentissage** ou "**empathie**": l'analyste raisonne en essayant de répondre à la question "comment ferais-je pour résoudre le problème si j'étais à la place de l'expert ?" en étudiant des exemples et contre-exemples.

Actuellement les travaux portent sur le développement d'outils "automatisant" la phase d'acquisition [Boy 88]. Les auteurs concluent sur le fait que les systèmes d'acquisition actuels ne sont pas très "intelligents" et sont plutôt des systèmes sophistiqués d'édition de connaissances, encore du domaine de la recherche.

Comme nous l'avons déjà précisé, notre démarche n'a pas été définie a priori comme une démarche d'acquisition de connaissances. Nous n'avons donc pas utilisé de méthode d'acquisition de connaissances basée sur des techniques d'entretien, ni outil d'acquisition. L'approche "cogniticienne" est en effet d'autant moins satisfaisante qu'elle est typiquement associée à la notion de "prototypage rapide". N'ayant pas fait de choix a priori sur la technique de développement à mettre en oeuvre, cette approche n'était pas envisageable. Notre expérience en ce qui concerne les développements de SBC nous a montré que cette démarche n'était pas satisfaisante dans son principe. Ce constat, confirmé par plusieurs études, fait l'objet du paragraphe suivant.

### Développement et conceptualisation

Les publications sur le développement de systèmes de bases de connaissances ("knowledge base systems") convergent sur la nécessité de suivre une approche plus structurée que le prototypage rapide. Le prototypage apporte certes des réponses à certains défauts de l'approche traditionnelle: [Cho 86]

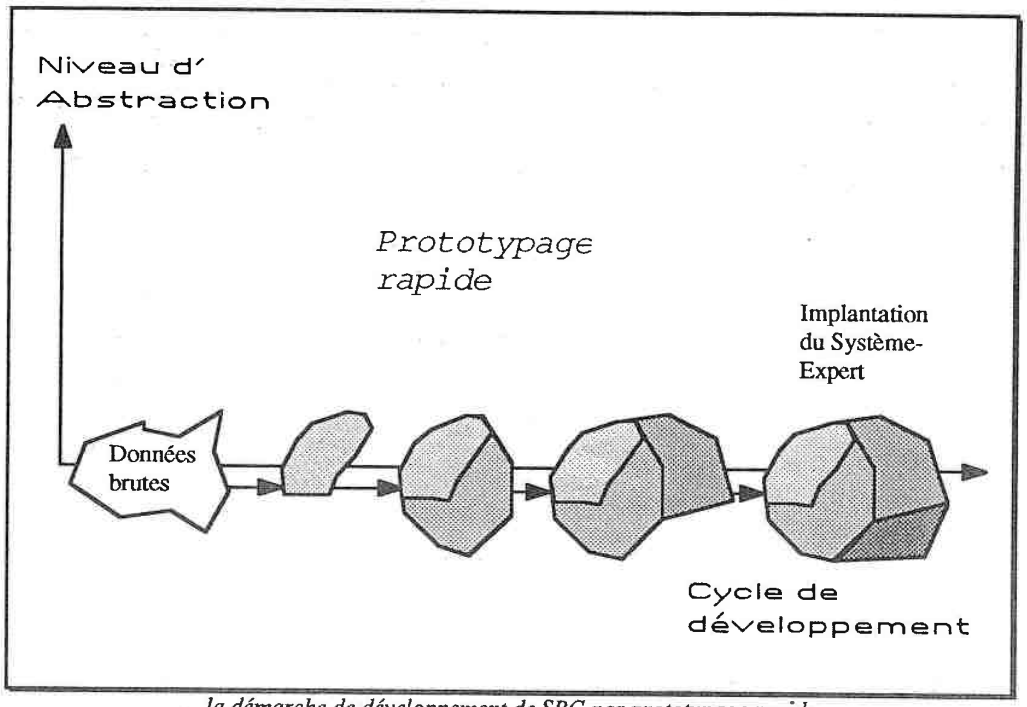
- l'étude de la distribution des erreurs et des efforts consacrés à leur correction montre que les spécifications en sont de loin la principale cause;
- le prototypage consolide la spécification en permettant de vérifier le comportement réel du produit en cours de développement;
- le prototype facilite la communication entre le client et le réalisateur en obtenant rapidement des réactions sur la conception du produit;
- permet de spécifier progressivement un problème que l'on ne sait pas décrire complètement et rigoureusement a priori.

Dans le domaine du développement des SBC, la difficulté de spécification rend l'approche prototype particulièrement adaptée à cet aspect du problème. Cependant cette approche induit de sérieux problèmes dans la maîtrise du développement d'une application dont les symptômes sont bien résumés par [Abe 88]:

- *la base de connaissances peut devenir incontrôlable de par sa taille;*
- *les déductions deviennent incontrôlables et lentes, et*
- *le système peut ressembler à un patchwork de programmes et de structures disparates;*
- *une efficacité qui se dégrade lorsque les problèmes sortent du cadre des exemples sur la base duquel le système a été développé;*
- *ces systèmes peuvent fréquemment produire des résultats contradictoires;*
- *les systèmes développés sans méthode deviennent rapidement des cauchemars au niveau de la maintenance car l'inorganisation des connaissances rajoute un niveau de complexité non contrôlable .*

Les difficultés rencontrées dans la transcription directe des connaissances en un système expert par prototypage évolutif ou "incrémental" résident autant si ce n'est plus dans une **compréhension trop superficielle des concepts** que dans l'analyse du langage de l'expert. [Sch 88], [Vog 88], [Abe 88], font une analyse similaire des causes de cette faiblesse limitant le développement des Systèmes-Experts: *"l'organisation des connaissances et les fondements de la démarche de l'expert ne sont pas mis en évidence. Le prototypage rapide revient à vouloir imiter le comportement de l'expert sans chercher à comprendre pourquoi ce comportement a lieu et quels sont les concepts utilisés."* [Abe 88]

La représentation graphique de ce phénomène est explicité dans le schéma suivant:



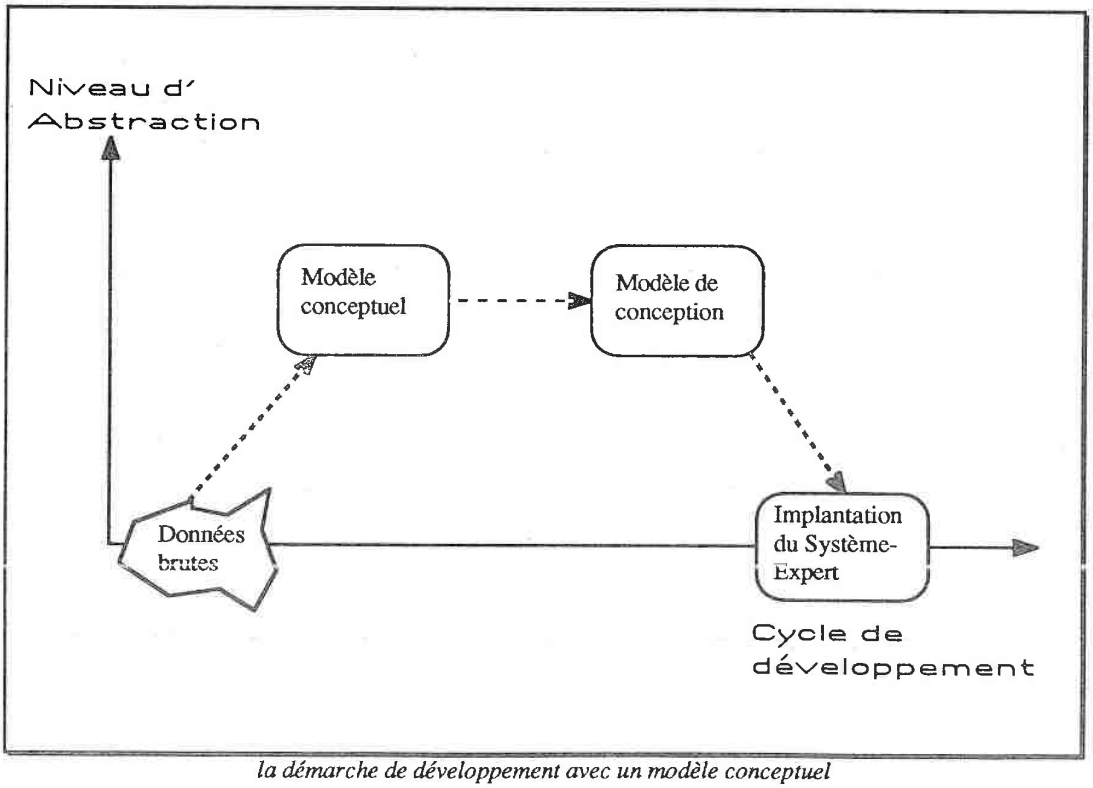

*la démarche de développement de SBC par prototypage rapide*

Il convient donc de réserver le prototypage aux domaines où il a fait ses preuves:

- le prototypage "**exploratoire**" [Cho 86], c'est-à-dire démonstratif, correspondant à l'élaboration d'une maquette dont les fonctionnalités sont réduites, mais **facilitant la définition des besoins et des caractéristiques fonctionnelles** souhaitées (au niveau des interfaces utilisateur en particulier);

- le prototypage "**d'expérimentation**" [Cho 86], permettant de s'assurer au niveau de la conception générale ou détaillée de la **faisabilité des parties critiques** du système et de l'adéquation de la technique de programmation du système-expert avec le problème posé.

Par contre le prototypage "évolutif" encore appelé prototypage "rapide" ne semble pas constituer une approche satisfaisante pour le développement de projets industriels. Les solutions proposées pour pallier les défaillances de cette approche, consistent donc naturellement à proposer des niveaux supérieurs d'abstraction par rapport à la formulation brute des connaissances (formulation aussi appelée "modèle pratique"). Ces niveaux sont basés sur des **modèles formels** dits **"conceptuels"** ou **"cognitifs"**, ceci selon un processus tout à fait comparable aux approches suivies dans le développement des logiciels informatiques traditionnels: par exemple modèles conceptuels des données et traitements dans MERISE. De tels modèles permettent de construire l'architecture du futur système en sous-ensembles stables et représentatifs de l'ensemble du problème selon une structure efficace: les classes d'objets utilisés sont identifiées, les catégories de règles également, et le développement ultérieur permettra ensuite de "remplir" en profondeur chacun de ces sous-ensembles. Pour l'instant les travaux publiés sur ce sujet traitent surtout de modèles conceptuels permettant l'expression de la structure implicite des connaissances et une classification des concepts, indépendamment des contraintes de réalisation et d'implantation. Les modèles de "conception" ou "de traitement de l'information" sont encore peu formalisés et correspondent souvent directement au formalisme de représentation des connaissances utilisé par le SE.




*la démarche de développement avec un modèle conceptuel*

## 4. PLAN DE TRAVAIL

Notre contribution consiste donc dans sa première partie à établir la partie déclarative des connaissances: les modèles "conceptuels" ou "cognitifs" génériques de conception d'architecture de SAP, modèles constitués des concepts jugés pertinents ou enrichissants. C'est au niveau de ces modèles que s'effectue la convergence entre une approche orientée méthode et une approche orientée connaissances: le modèle est d'abord côté méthode le formalisme de représentation de la réalité étudiée, mais également du point de vue connaissances l'expression du point de vue et du savoir-faire du spécialiste concepteur.

La deuxième partie de la contribution consiste à expliciter la partie procédurale constituants les traitements et transformations des modèles: les différentes tâches effectuées par les concepteurs pour obtenir une solution finale d'architecture de SAP.

Le plan de travail suivi pour souligner cette convergence peut donc être rapproché de ceux donnés dans [Dav 88] ou [Mah87] pour l'acquisition de connaissances expertes:

**Analyse globale:**

- se familiariser avec le domaine (définition de la problématique, études de cas),
- définir la finalité et les objectifs à atteindre,

**Analyse structurelle et détaillée:**

- identifier les données de base, les concepts manipulés et organiser les connaissances,
- identifier les tâches mises en oeuvre dans la résolution,
- définir les modèles de représentation des connaissances,

et pour la partie que nous avons prototypée:

**Transcription:**

- constituer la base de connaissance

et **Synthèse / Validation** des connaissances.

La deuxième partie présente donc les connaissances méthodologiques de conception d'architecture de SAP suivant le plan suivant:

**Concepts manipulés:**

- données de base,
- modélisation du processus de production,
- entités fonctionnelles,
- relations structurelles: hiérarchie et communications.

**Modèles de représentation des connaissances:**

- structure fonctionnelle,
- architecture opérationnelle.

**Procédures:**

- tâches,
- documents produits.

## CONCLUSION

Après ces quelques éléments sur le développement des systèmes à base de connaissances, la similitude apparaît plus clairement entre l'approche que nous avons suivie pour étudier les connaissances méthodologiques et l'approche de type acquisition de connaissances pour le développement de systèmes dits "experts". La convergence entre ces deux approches provient d'une part de la difficulté de modéliser les SAP et le savoir-faire des concepteurs, et d'autre part du rôle catalyseur joué par la disponibilité d'environnements basés sur des techniques d'intelligence artificielle permettant le développement d'outils méthodologiques.

Il est cependant important de souligner que tout le parti que l'on pourrait tirer d'une approche de type acquisition de connaissance n'a pu être exploité dans le cadre de ce travail de recherche puisque cette approche n'a pas été suivie d'entrée mais constatée a posteriori.

## CONCLUSION de la 1ère PARTIE

Nous avons vu dans cette première partie consacrée à la problématique, quelles étaient les spécificités de la conception d'architecture par rapport au développement d'une application. Ces spécificités sont donc liées au domaine particulier présenté dans le premier chapitre sur "le cadre de travail", aux fonctionnalités nouvelles attendues du SAP et à l'utilisation de nouvelles techniques informatiques présentées dans le chapitre sur la "place des RLI dans les architectures de SAP", et aux particularités du "contexte des études d'avant-projet en ingénierie" présentées dans le troisième chapitre. ces caractéristiques particulières seront prises en compte dans la deuxième partie de ce document pour définir ce que doivent offrir les modèles et tâches utilisés pour la conception. Cette deuxième partie "connaissances en conception d'architecture" est donc présentée en distinguant clairement, selon l'approche traditionnelle suivie dans les études de systèmes à base de connaissances, ce que nous appellerons:

- les "concepts de modélisation", c'est-à-dire les objets (statiques) manipulés par les analystes-concepteurs pour modéliser l'architecture aux différents stades de la conception,
- les "tâches d'analyse et de conception", c'est-à-dire les différents axes d'investigations parcourus "successivement" ou plutôt de façon itérative pour parvenir à la définition d'une solution.

## 2ème partie:

# CONNAISSANCES EN CONCEPTION D'ARCHITECTURES

# chapitre 5:
# CONCEPTS de MODELISATION d'une ARCHITECTURE

## INTRODUCTION

Nous examinons dans ce chapitre quels sont les concepts, c'est-à-dire les objets ou les éléments de connaissance ainsi que les relations entre ces éléments, utilisés ou utilisables par les industriels pour décrire le Système d'Information (SI) d'un SAP. Trois sources d'informations sont étudiées:

- les modèles d'analyse et de conception de SI employés dans le domaine du génie logiciel,
- les modèles d'Architecture de SAP,
- les concepts utilisées pour la représentation des Architectures par les ingénieristes.

Il s'agit donc de montrer en quoi les concepts sont adaptés ou non à la modélisation en phase d'Avant Projet. Il n'aborde pas l'aspect méthode, au sens des procédures mises en oeuvre pour appliquer ces concepts car cet aspect fera l'objet du chapitre suivant "Tâches de Conception".

Le lecteur souhaitant se doter d'un outil de représentation adapté à la conception des systèmes d'information de SAP, trouvera ici une liste des concepts que devrait supporter cet outil. Si cette liste ne correspond pas directement à un modèle offert par un outil existant, il ne faut pas conclure qu'un développement spécifique soit indispensable. Dans l'état actuel de la technologie logicielle, les coûts de développement d'un tel outil pour parvenir au stade opérationnel, même limité à un usage interne, interdisent de tels investissements pour la plupart des entreprises. L'utilisation d'un produit issu du génie logiciel pourra donc être envisagé si l'on consent à effectuer sérieusement:

- les adaptations possibles au niveau de l'outil informatique lui-même,
- la définition de conventions d'utilisation pour pallier les lacunes éventuelles et les limites d'adaptation.

Si sur le plan fonctionnel une telle solution n'est pas idéale, le rapport fonctions / prix reste tout à fait positif en l'attente de produits logiciels suffisamment ouverts et paramétrables pour autoriser le développement d'outils spécifiques par l'utilisateur final [Xer 88]. Une étude de l'adaptation de différents types d'environnements est jointe en annexe.

Le lecteur désireux de ne pas rentrer en première lecture dans le détail de ces modèles pourra se reporter directement à la "synthèse des concepts retenus" au paragraphe 1.3.

# 1. LES MODELES DE CONCEPTION DES SYSTEMES D'INFORMATION

De très nombreux modèles ont été définis pour faciliter l'analyse et la conception des logiciels. Les concepts utilisés sont le reflet de toute l'expérience acquise dans le domaine du Génie Logiciel. Ce paragraphe ne prétend pas détailler toutes ces approches: les nombreuses références bibliographiques permettront au lecteur intéressé d'approfondir chacune d'elle, on pourra également consulter [Kel 87], [Man 87], [Yau 86] et [Lud 87] pour des études comparatives de méthodes employées dans le domaine des systèmes temps-réels. L'objectif est ici de situer quels sont les modèles adaptés à la représentation des Architectures.

Afin de faciliter le positionnement de ces modèles et méthodes, nous avons repris différentes propositions de classification. Une première sélection a donc été effectuée en ne retenant que les modèles proposant des notations graphiques synthétiques et des outils informatiques supports. La seconde sélection est basée sur l'adaptation des modèles à la phase d'Avant Projet.

## 1.1. Classification

Les modèles utilisés dans les différentes méthodes d'analyse et de conception cherchent à modéliser soit le système informatique lui-même, soit le système objet de l'informatisation-automatisation. Cette division correspond à la classification donnée par [Rol 86] des méthodes de conception informatique en 2 grandes familles suivant leur *"paradigme fondateur"* :

- Les méthodes **cartésiennes** sont basées sur la décomposition du SI en le considérant comme un système de traitement de l'information. L'étude est effectuée en divisant en éléments simples avant de rassembler à nouveau ces éléments. Le SI est considéré comme une boîte noire définie par les sorties qu'il doit produire. Elles suivent généralement une approche fonctionnelle, conduisant à approcher le système par les traitements qu'il doit exécuter et les flux de données échangées plutôt que par les données qu'il doit gérer.

- Les méthodes **systémiques** sont basées sur la compréhension du système objet à informatiser (une organisation par exemple). Elles considérent le SI comme un modèle de la réalité, et non pas comme un processeur de traitement de l'information. Elles suivent généralement une approche de modélisation conceptuelle des données, celles-ci étant supposées constituer une structure plus stable que celle des procédures de traitement.

Le tableau ci-dessous [Rol 86] illustre cette classification par quelques exemples parmi les plus connus:

| **méthodes cartésiennes:** | **méthodes systémiques:** |
|---|---|
| SADT [Ros 77], | MERISE [Tar 84], IDA [Bod 83] |
| SA/SD (Analyse/Conception Structurée) [DeM 78], | modèles ERA entités-relations-attributs [Che 76], |
| machines d'états finis, | modèles relationnels (au sens B.de D.) [Cod 79] |
| PSL/PSA [Tei 77], | |
| Réseaux de PETRI [Bra 83], Grafcet [Bla 79], | |

*classification des modèles suivant leur paradigme fondateur*

Une autre classification [Dec 88] peut être établie directement à partir des concepts modélisés constituant le coeur de ces méthodes.

La première catégorie est basée sur la représentation des **fonctions** ou activités, donc des traitements par des boîtes noires et la représentation des échanges d'informations par des liaisons.

La deuxième catégorie est basée sur la représentation des **données** et des relations entre ces données.

Et la troisième est basée sur la représentation de la **dynamique**, par les notions d'état ou d'événement.

| espace des: | exemples: |
|---|---|
| **1 traitements** | |
| activités | SADT [Ros 77] |
| modules | SD [You 79] |
| | Modules Communicants [Lon 87] |
| flux de données | SA (DFD) [DeM 78], [Ward 86], [Bru 88], [Gom 86] |
| **2 données** | |
| | Entité/Relation [Che 76], IA [Nij 77] |
| | Structure de données [Jac 75] |
| **3 dynamique** | |
| états | diagrammes d'états [Hat 88], statecharts [Har 87] |
| événements | Réseaux de Petri [Bra 83], Grafcet [Bla 79] |
| externe | USE [Was 85] |

*classification des modèles suivant les points de vue*

Ces modèles emploient des notations correspondant à différentes *"vues"* [Dec 88] ou points de vue [EPO 87] dont l'importance est liée au domaine d'application: Système d'Information, contrôle de processus, système interactif, etc...: *"ces vues peuvent être comparées aux projections différentes pour décrire un objet solide en géométrie descriptive"*.

L'approche **objet**, qui possède notamment l'originalité d'intégrer les aspects statiques et dynamiques, n'est pas mentionnée dans les classifications citées. Elle se situe à la frontière entre les approches systémiques et cartésiennes. L'approche objet repose sur *"une décomposition structurale (et non pas fonctionnelle) au sens où elle décompose à la fois les objets et les fonctions associées à ces objets, et assujettit les secondes aux premières"* [Vog 88]. En tant que modélisation des objets naturels, cette approche peut être qualifiée de systémique. Pour la **spécification des besoins**, l'approche objet permet en effet une modélisation naturelle et rigoureuse garantissant une compréhension approfondie de la réalité du système ou de l'organisation à informatiser. Employée en **phase de conception** , elle rejoint l'approche modulaire cartésienne dans la mesure où les objets sont des éléments modulaires et génériques de transformation d'information, plus que des modèles de la réalité [Boo 86]. (cf à ce sujet les méthodes OOD et HOOD évoquées au paragraphe 1.2.2. )

## 1.2. Adaptation à la modélisation des architectures de SAP en phase d'avant-projet

Avant de détailler quelques uns de ces modèles, insistons sur le fait que les points de vue qu'ils représentent ne s'opposent pas mais sont complémentaires. Ils sont d'ailleurs dans la plupart des méthodes de conception informatique mis en oeuvre parallèlement pour intégrer cette complémentarité. Nous les avons séparés dans ce chapitre afin de mieux mettre en avant leurs spécificités et leur adéquation avec nos besoins.

### 1.2.1. Notations et support informatique

D'un point de vue pratique, on constate que la plupart des méthodes sont supportées (ou destinées à l'être) par des Environnements informatiques de type Génie Logiciel. L'environnement matériel retenu est le poste de travail personnel: station de travail ou micro-ordinateur. Si la puissance disponible à l'utilisateur doit être suffisante, le succès de ce type de matériel est surtout dû au besoin d'interfaces utilisateur conviviales basée sur des possibilités graphiques avancées.

La **description graphique** d'un système informatique complexe s'avère en effet moins fastidieuse à réaliser qu'avec des notations textuelles, ceci pour des raisons évidentes de concision. Les notations graphiques sont traditionnellement un des outils de travail de l'ingénieur (schémas logiques, diagrammes, plans...). L'intérêt de la représentation graphique en tant qu'outil de communication a été de plus largement étudié et démontré par les ergonomistes et psychologues. Les deux dimensions de la représentation graphique le rende plus efficace que la représentation textuelle linéaire [Pag 80] : *"Un avantage supplémentaire des dessins est qu'ils sont bi-dimensionnels, alors que le texte est linéaire, c'est-à-dire uni-dimensionnel. Les éléments d'un système, et sa structure générale sont au-moins bi-dimensionnels. C'est pourquoi un outil graphique est plus lisible qu'un outil textuel, peut être interprété de façon plus précise par le cerveau et constitue une description plus réaliste d'un système."* L'image tient une place prépondérante dans la communication et dans la transmission de la connaissance en général [Savoy 88]: *"Pour certains psychologues, le texte lui-même est d'abord transformé en image mentale avant d'être compris et mémorisé. L'usage de diagrammes et de schémas favorise la compréhension d'une idée."* L'examen des documents d'étude d'architecture et les discussions menées avec des spécialistes au sujet de la définition d'une architecture ont montré qu'une représentation graphique sur papier ou sur un tableau est inévitablement employée pour permettre la compréhension réciproque des

interlocuteurs. Interrogés sur leurs besoins en outils, ces concepteurs mentionnent toujours un outil graphique, qu'il soit associé à un formalisme particulier ou non.

Si l'avantage du graphisme est déjà perceptible en utilisation manuelle, un **support informatisé** donne toute sa raison d'être à des notations graphiques:

- facilités d'édition et de modification,
- standardisation,
- vérification des règles syntaxiques et sémantiques.

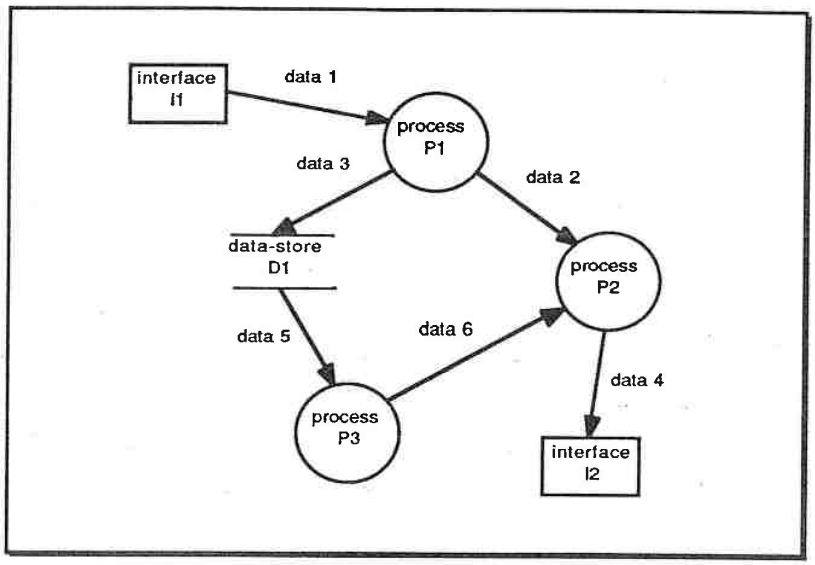
Ce type de notation ne correspond pas toujours au même degré de formalisme. Pour ne retenir que des exemples connus, les diagrammes SADT ne sont pas basés sur un support aussi rigoureux que les Réseaux de Petri. La classification par [Dec 88] des méthodes employant des notations graphiques dans la catégorie "semi-formelle" par opposition aux méthodes "formelles", nous semble donc comporter une part d'arbitraire, et n'être qu'une convention de vocabulaire trop large pour être pertinente. Si l'on peut bien-sûr souhaiter un formalisme le plus rigoureux possible, nous retiendrons avant tout le besoin général de graphisme exprimé par les utilisateurs concernant toute méthode d'analyse et de conception.

### 1.2.2. Adaptation des concepts

#### 1.2.2.1. Modèles de représentation des flux et traitements

La modélisation par les **flux de données** (Diagrammes Flux de Données et dérivés des méthodes Structured Analysis (SA) déjà citée) est bien adaptée à la conception d'architecture dans la mesure où:

- elle met en avant les interrelations fonctionnelles d'échanges d'information (les flux d'information) nécessaires à la conception du système de communication ,
- elle peut facilement être mise en œuvre à différents niveaux de détail et
- elle permet une approche descendante adaptée aux applications complexes par décomposition hiérarchisée des "process".

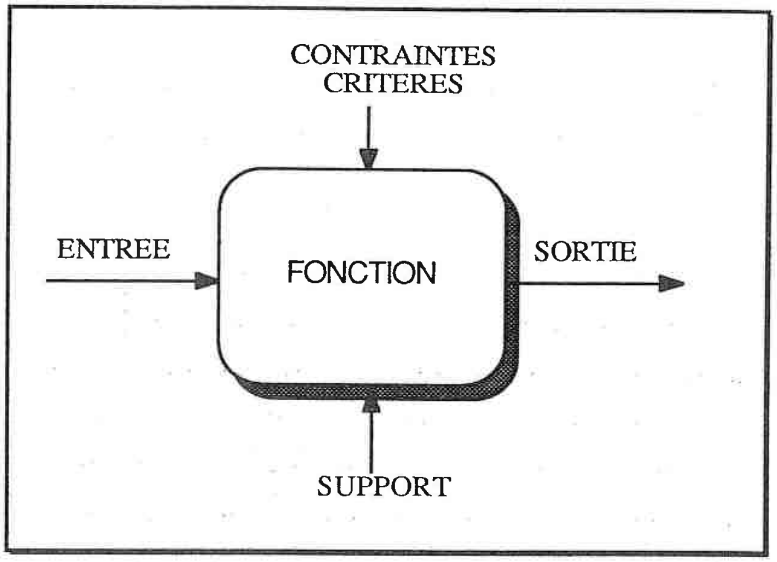



*exemple de diagramme flux de données*

De nombreuse variantes et extensions existent, intégrant notamment l'aspect "temps-réel" grâce à la représentation explicite des flux et fonctions de contrôle et l'utilisation de diagrammes d'état. [War 88]

Son **principal inconvénient** est de ne pas tenir compte explicitement de la structure du système ou de l'organisation à informatiser. Pour un système de production, la seule représentation du processus se matérialise par les entrées / sorties supportées par les capteurs / actionneurs. Ces E/S sont décrites de façon isolée, donc aucune structure de la réalité informatisée ne peut être représentée. Ceci conduit à des adaptations sur l'utilisation des notations: par exemple la représentation des flux de produits avec les notations des flux d'informations. Comme le fait remarquer [Gre 82], les spécifications fonctionnelles, orientées vers la description en terme de fonctions du système d'information, ne réservent pas une part explicite à la description de l'organisation souvent disséminée dans les documents et l'esprit des utilisateurs.

Les actigrammes **Structured Analysis and Design Technique** (SADT) [Ross 77] sont conceptuellement proches des diagrammes de flux de données. Egalement utilisés dans la méthode IDEF (voir le chapitre sur les modèles d'architecture de SAP), ils correspondent à la partie "Structured Analysis language" de la méthode SADT, à ne pas confondre avec son homonyme de la partie "Structured Analysis" basée sur les diagrammes de flux de données (DFD) de la méthode SA/SD (Structured Design) ou les diagrammes de flux contrôle (DFC) de la méthode SA/RT (Real Time).




*la représentation des activités ou fonctions de IDEF0 et SADT*

L'utilisation du modèle SADT-IDEF0 est assez largement répandue parmi la communauté automaticienne française pour l'analyse fonctionnelle des systèmes de commande des SAP (cf les nombreux modèles basés sur ce formalisme dans le paragraphe 2.1.2. de ce même chapitre).

Remarque: l'échec des datagrammes est à la hauteur de ce succès, comme en témoigne l'utilisation du modèle entité-relation dans la méthode IDEF, (cf paragraphe "IDEF-ICAM" 2.1.2.1). A ce propos [Vog 88] souligne la fausse dualité entre données et activités: *"les actions préexistent aux objets et non l'inverse. Poser le comportement avant l'entité n'a aucun sens."*

Les raisons du succès des modèles d'activités ou de flux de données sont intéressantes à analyser :

- **simplicité** et nature intuitive des concepts de base,
- **lisibilité** des notations graphiques,
- intégration forte de la **hiérarchie de décomposition** des "process ou activités" dans les notations et la procédure d'analyse,
- existence d'**outils informatiques support**, même si l'utilisation manuelle reste importante,

et pour la méthode SADT:

- importance accordée à la **communication** (cycle auteur-lecteur trop rarement appliqué rigoureusement dans la pratique).

Les **limites** de l'utilisation de ces méthodes, en ce qui concerne l'étude du système d'information du SAP, semblent cependant soulignées par la fréquente adaptation "libre" des notations et les nombreuses adaptations ou extensions apportées au formalisme et aux règles d'utilisation. L'application manuelle de ces méthodes ne constitue pas la seule explication. Parmi ces limitations nous relèverons les suivantes:

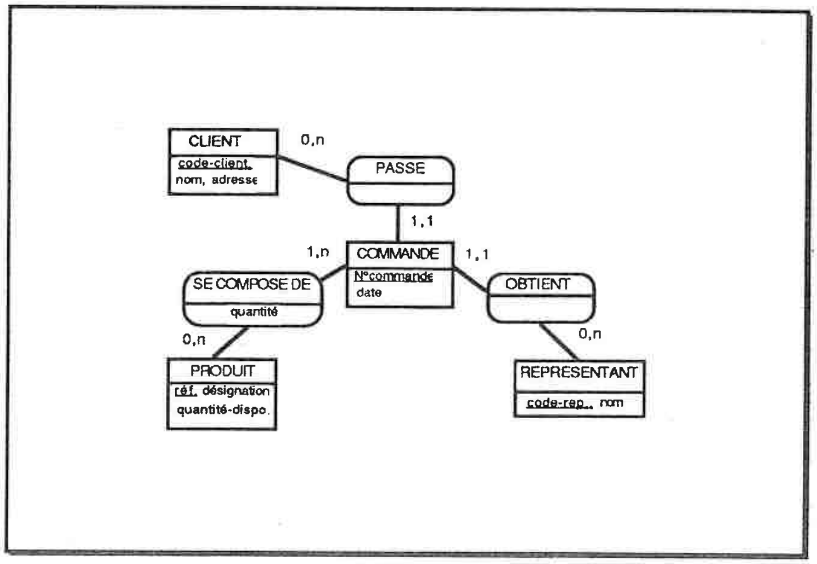
- la **difficulté de définir la frontière** entre les activités SADT: même si la décomposition est basée d'après Ross sur la définition du contrôle de l'activité, la multiplicité des liaisons de contrôle rend cette aspect délicat à maîtriser, d'où pour un même problème autant d'analyses différentes que d'utilisateurs de la méthode.
- **absence de représentation explicite des relations requête / compte-rendu** nécessitant l'utilisation de liaisons simples, ce qui ne permet pas de garantir l'absence de bouclage (une activité contrôlée contrôlant indirectement son contrôleur: risque de dead-lock). Cette remarque est également valable pour les DFDC et leurs dérivés.
- **pas de distinction** dans SADT **sur les types de liaisons** entre celles supportant les données et celles supportant les signaux de contrôle-synchronisation ("triggering") contrairement aux extensions des DFD par Ward-Mellor ou Hatley. La sémantique des liaisons n'est définie que par la position des points de connexion ("stubs") par rapport aux boîtes ("blocks") représentant les activités.
- **séparation complète** dans SADT de la représentation des structures de **données** et de **traitement**. Contrairement aux DFD, il n'est pas possible de représenter l'écriture de données par une fonction dans un fichier ou une structure de données.
- **confusion entre la notion d'activité et la notion de fonction** au sens process telle qu'elle est utilisée par les DFD: en effet rien n'oblige à associer une activité (transformation de données dans une certaine situation) à une fonction particulière, une fonction peut très bien supporter plusieurs activités dans des situations différentes. La notion de mécanisme, absente des DFD et précisément introduite pour définir le support de l'activité, est d'ailleurs généralement omise quand il y a confusion entre fonction et activité. Or du point de vue de la conception du système d'information (et de la communication en particulier), il est plus intéressant de travailler avec la notion de fonction et celle d'interface (E/S) d'une fonction, structurellement stable.
- les contraintes de **lisibilité** pouvant empêcher la représentation globale d'un système complexe en une seule vue conservant un niveau de détail suffisant. La "mise à plat" de plusieurs niveaux sur un seul schéma assimilable à un plan

architectural d'ingénierie ou la représentation des arbres de hiérarchie de décomposition témoignent cette limitation.

La **principale difficulté à laquelle on se heurte en essayant d'appliquer SADT est l'utilisation des résultats de cette spécification pour les étapes suivantes.** Aucune procédure systématique de transformation n'est proposée dans la méthode pour passer du modèle SADT à une autre forme de représentation plus proche de la solution. L'expérience montre que cette représentation des besoins, parfaitement générale au sens où elle demande le minimum de choix de structuration à l'analyste, est trop éloignée de la recherche de solutions opérationnelles. D'ailleurs Ross, dans les référence déjà citées, précise lui-même le rôle du modèle :*"la seule fonction de SA est de relier, structurer et communiquer des unités de pensée exprimées dans n'importe quel autre langage, ... est seulement concernée par la décomposition ordonnée et bien structurée du sujet"* et situe l'apport de SADT au niveau *"requirements definition, encompasses all aspects of system development prior to actual system design"*. Pour passer de l'expression des besoins à une architecture opérationnelle, il faut passer par une étape d'organisation de la structure fonctionnelle. Or SADT n'offre pas les concepts de représentation de la structure fonctionnelle d'un système d'information tels que ceux offerts par les DFD: process, différents flux de données et de contrôle, stockage de données. Le concepteur essayant de faire de la conception fonctionnelle avec un langage **d'expression et d'analyse des besoins** aboutit donc à une impasse. Cette vocation de généralité l'empêche de supporter un processus de conception particulier, étant donné l'absence de certains **concepts spécifiques de conception** tels que ceux rencontrés dans les architectures de SAP.

### 1.2.2.2. Modèles de représentation des données

Des formalismes orientés **données** tels que les diagrammes entités-relations facilitent la définition des principales structures de données manipulées par le système. Ce formalisme ne permet cependant ni de représenter une vue globale structurelle du système étudié (malgré certaines possibilités de décomposition hiérarchisée), ni d'expliciter clairement et explicitement les flux de données (les accès en lecture ou en écriture ne sont pas représentés). Nécessitant un niveau de détail élevé, elles peuvent difficilement s'appliquer aux études d'avant-projet sans être allégées. Nous avons vu que l'on ne dispose rarement ni du temps, ni de toute l'information nécessaires à ce type d'approche plus adaptée à la spécification détaillée.



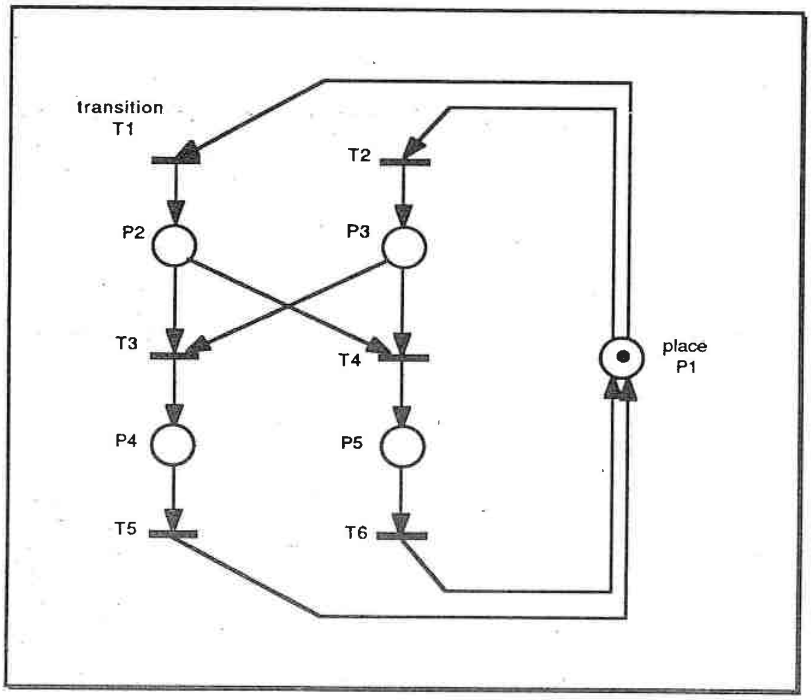

*exemple de modèle entité-relation-attributs*

Dans les SAP, les premiers niveaux(*) constitués des fonctions d'automatismes en contact direct avec le processus de production, ainsi que les fonctions de coordination sont encore peu basés sur l'emploi de bases de données. En revanche les fonctions nécessitant de nombreuses données sont de plus en plus basées sur de tels systèmes de gestion de base de données (SGBD). Le développement de SGBD répartis et adaptés au temps-réel devrait conduire les concepteurs à utiliser de façon plus fréquente une approche orientée données éventuellement simplifiée, par exemple sans préciser toutes les contraintes sur les relations entre les entités.

(*) la notion de niveaux sera étudiée en détail dans le chapitre suivant.

### 1.2.2.3. Modèles de représentation de la dynamique

Les formalismes orientés **événement** ou **états** (Grafcet, Réseaux de Petri, Machines d'Etats Finis) sont utilisés pour la spécification détaillée de la structure de contrôle des applications. Le niveau de détail élevé et la complétude des spécifications nécessaires à ce stade du développement d'une application informatique ne permet pas d'envisager leur utilisation en étude d'avant-projet. La représentation explicite des flux de contrôle entre les fonctions du système est à ce niveau de l'étude suffisante, car il n'est pas nécessaire dans une analyse fonctionnelle externe de détailler le contenu de chacune des fonctions (les différents états) et leur comportement (séquences de changements d'états).




*exemple de Réseau de Petri*

#### 1.2.2.4. Modèles de représentation orientée objets

L'approche **objet** semble prometteuse dans la mesure où:

- elle permet de calquer la structure de l'application informatique sur la structure de l'organisation ou du processus à informatiser;
- elle intègre de façon cohérente les aspects données et traitements.

La limitation à l'emploi de cette approche est liée à sa relative jeunesse. Les langages "orientés objets" sont beaucoup plus développés que les méthodes de spécification fonctionnelle basées sur cette approche. Deux orientations semblent se dessiner:

- une approche **développement** (et non conception) **des logiciels** liée à l'emploi du langage ADA en particulier: la méthode OOD (Object-Oriented Development) de Booch [Boo 86] ou la méthode HOOD [HOO 87] sont des exemples de cette première approche;
- une approche **conception de systèmes à base de connaissances** liée à l'emploi de système experts basés sur une représentation des connaissances à base d'objets: les exemples cités au paragraphe 2 "représentation des connaissances" du chapitre 6 illustrent cette seconde approche.

De plus parmi les produits de conception recensés, encore peu d'environnements de génie logiciel supportent actuellement ces approches orientées objets, malgré quelques annonces commerciales. Nous verrons dans les paragraphes 2 et 3 suivants que la structure de la réalité informatisée n'est pas le seul critère de structuration. Une représentation trop axée sur ce critère pourrait donc rendre délicate l'utilisation d'autres critères, basés par exemple sur la fonctionnalité du système d'information.

## 1.3. Synthèse des concepts retenus

Quels sont les modèles utilisés par ces concepteurs ? L'examen des méthodes employées sur le terrain confirme cette orientation vers l'emploi de méthodes fonctionnelles orientées traitements ou flux, et employant des notations graphiques. Nous n'avons pas identifié d'autre méthode originale spécifiquement dédiée.

En plus des raisons techniques, le choix de méthodes traitement / flux s'explique aussi par le profil des spécialistes chargés de la conception. Les concepteurs d'architecture sont le plus souvent automaticiens et informaticiens (informatique industrielle) de formation. Ils sont donc familiarisés avec les méthodes fonctionnelles du développement logiciel temps-réel: schémas fonctions/flux pour les structures statiques et formalismes états / transitions pour l'aspect dynamique.

En fonction:

- de la culture méthode des concepteurs,
- de la pratique courante au sein de l'entreprise,
- des particularités des applications étudiées,

on retrouve donc logiquement des approches basées sur des adaptations ou extensions de formalismes de type DFD ou SADT , la plupart du temps sans support informatique. Bien que des outils fassent leur apparition dans les entreprises, ils ne sont encore pratiquement pas utilisés en phase d'avant-projet. Le Grafcet constitue un exemple intéressant d'utilisation rigoureuse d'un outil de spécification. Mais son emploi est limité à la spécification des automatismes séquentiels et n'est que peu utilisé au cours de la conception d'une architecture de SAP.

Le tableau suivant résume et compare les principaux concepts présents dans les deux modèles les plus utilisés:

| | **actigrammes SA (SADT)** | **diagrammes flux de données et contrôle (SA)** |
|---|---|---|
| **traitements** | activity | process |
| **flux** | data flow | data flow |
| | Ø | control flow (triggering) |
| **rôle des données** | input | input |
| | output | output |
| | control | control |
| | mechanism | Ø |
| **mémorisation** | Ø | data store |
| **décomposition** | hiérarchisée | hiérarchisée |

Si l'on considère les trois grandes caractéristiques de la structure des Systèmes Automatisés de Production que sont le parallélisme, la communication liée à la répartition et la hiérarchie, l'utilisation de formalismes de type "blocs-diagramme" est effectivement la plus naturelle. Les formalismes orientés "données" et ceux orientés "représentation de la dynamique" ne permettent pas de représenter de façon suffisamment synthétique ces trois caractéristiques primordiales au niveau d'une conception à la fois globale et préalable.

Les besoins croissants d'intégration du SAP dans un contexte CIM conduisent malgré tout à donner une plus grande importance à l'aspect données. Dans les cas où la prise en compte de cette dimension complémentaire rend l'utilisation du symbolisme "data-store" des diagrammes flux de données insuffisant, des modèles entités-relations simplifiés (sans détailler les contraintes sur les relations) seront utilisés pour décrire la structure de base des relations entre les principales données.

Les principaux concepts utilisés dans ces méthodes et dont l'utilisation est jugée pertinente par les concepteurs d'architecture sont donc:

- **les fonctions de traitement de l'information du système selon une approche "boîte-noire",**
- **les structures de stockage de l'information,**
- **les différentes interactions fonctionnelles:**
  - **les relations de décomposition hiérarchisée de fonctions en sous-fonctions,**
  - **les différents flux entre les composants du système.**

## CONCLUSION

Nous avons donc examiné dans ce chapitre quels étaient les concepts issus de modèles proposés dans différentes méthodes de génie logiciel qui soient adaptés à l'analyse et la conception des architectures des SAP en phase d'avant-projet. Des concepts supplémentaires propres aux SAP ne sont en revanche pas supportés par ces méthodes dont la vocation générale interdit de telles spécificités. Le chapitre suivant présente les différents modèles d'architecture informatisée de SAP prenant en compte ces spécificités, dont notamment la notion de niveaux hiérarchiques classiquement utilisés dans le domaine du contrôle-commande.

## 2. LES MODELES D'ARCHITECTURE DE SAP

La difficulté de développer des systèmes de Conduite de Procédés Industriels a conduit de nombreux auteurs, centres de recherche et organismes de normalisation à élaborer des modèles d'Architecture ayant l'ambition d'être aux SAP ce qu'est le modèle OSI-ISO aux systèmes de communication, c'est-à-dire un "Modèle de Référence". Le but est de faciliter la structuration en fournissant un schéma standard d'organisation. Une des caractéristiques que l'on retrouve dans tous ces modèles est le concept de niveaux hiérarchisés, concept d'ailleurs souvent rapproché de la notion de couches dans le modèle OSI. Nous examinons dans ce chapitre quels sont les différents concepts proposés et comment est formalisée cette structure hiérarchisée dans les modèles publiés les plus significatifs.

Nous examinons essentiellement les modèles dont le champ d'application est suffisamment large et non pas focalisé sur certaines tâches telles que la planification, les systèmes de transport ou le contrôle de robots. En effet si certains concepts sont communs, les objectifs de ces modèles sont différents des nôtres. Nous étudions dans ce chapitre l'aspect modèle, mais il faut noter que l'aspect méthode est très peu développé dans ces publications.

Le lecteur désireux de ne pas rentrer en première lecture dans le détail de ces modèles pourra se reporter directement à la "synthèse des concepts retenus" au paragraphe 2.2.

### 2.1. Les principaux modèles d'architecture de SAP

Avant d'aborder la présentation de ces différents modèles d'architecture, quelques mots d'introduction sur l'origine de ces modèles. Les auteurs de ces publications appartiennent essentiellement à des équipes de recherche universitaires ou privées travaillant sur les sujets de l'automatisation de la production, la gestion de production ou l'organisation de l'entreprise. La plupart de ces travaux sont basés sur l'observation des structures des applications de systèmes automatisés. Ils constituent de ce point de vue une première formalisation du savoir-faire acquis par les industriels dans la mise en œuvre de moyens de contrôle-commande de procédés industriels depuis les débuts de l'automatique et l'informatique industrielle.

Tous ces travaux sont récents: les publications recueillies faisant état de la notion de modèle d'architecture sont postérieures au début des années 80. L'avancement de ces travaux trouve aujourd'hui un nouveau souffle au sein de projets de recherche en coopération comme les projets européens Esprit consacrés à l'automatisation ou au CIM, et au sein d'organismes de normalisation tels que l'ISO TC 184 / SC5 / WG1: "Reference Models Working Group", le DIN-KCIM (Kommission für Computer Integrated Manufacturing) allemand ou le groupe AMI (Architecture, Modèles et Interfaces) du CONEP (Comité d'Orientation de la Normalisation Européenne en Productique).

Notons enfin que les travaux consacrés à ce sujet sont en nombre croissant, en partie grâce à l'acceptation de l'importance du concept CIM d'intégration du SAP dans l'entreprise.

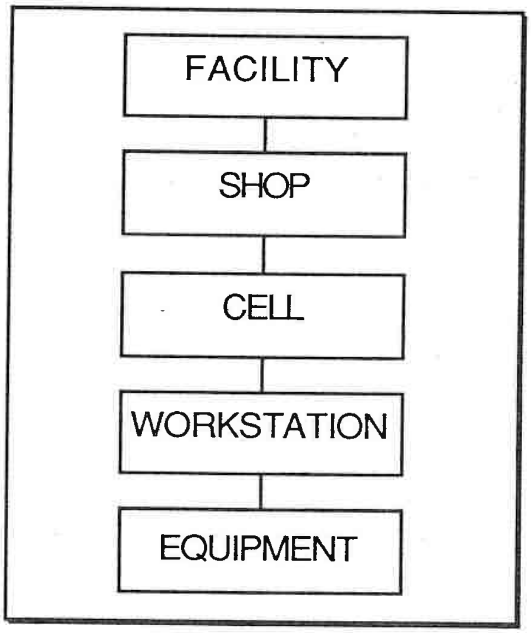
La première partie est consacrée aux modèles "originaux" ou spécifiques employant des notations propres, alors que la seconde partie est consacrée aux modèles basés sur des adaptations du modèle SADT-IDEF0.

### 2.1.1. Modèles spécifiques

#### 2.1.1.1. le Hierarchical Control Model du NBS

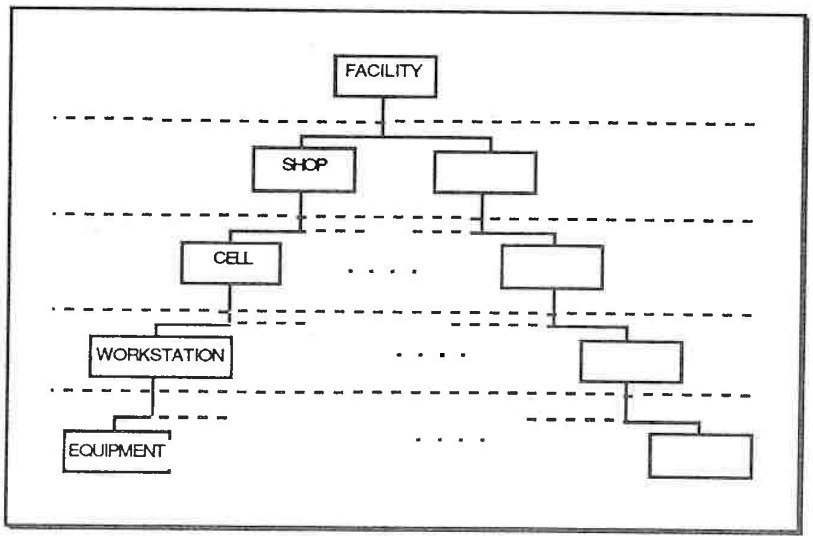
références: [Bro 87] , [Jon 86], [McLe 87]

Le modèle de contrôle hiérarchisé "Hierarchical Control Model" élaboré au sein du Automated Manufacturing Research Facility (AMRF) du National Bureau of Standards (NBS) de Gaithersburg est un des plus connu, à tel point que les nombreuses références à ce modèle en font presque un standard de fait. Basé sur la hiérarchie typique des organisations complexes, 5 niveaux sont proposés : Facility, Shop, Cell, Workstation et Equipment. La structure est basée, comme l'indique les termes retenus pour chaque niveau, sur une décomposition du processus de production en différents niveaux de détail.




*les 5 niveaux proposés dans le modèle NBS-AMRF*

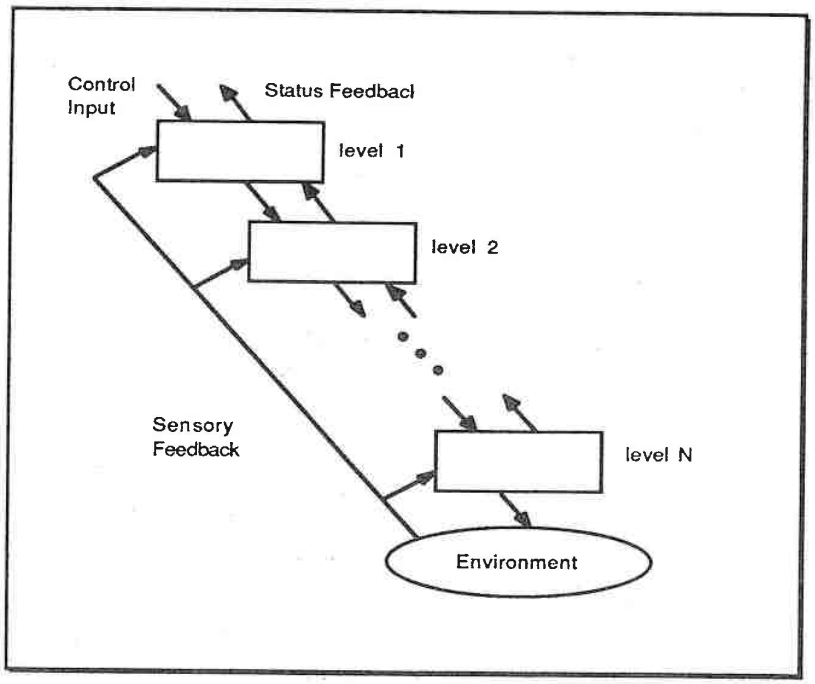
Cette structure est arborescente suivant une relation hiérarchique associée à un échange d'information bidirectionnel: *"control input / status feedback"* .




*arborescence de la hiérarchie de contrôle*

Bien que les flux de contrôle (control input/status feedback) dans cette structure soient strictement limités entre deux niveaux hiérarchiques adjacents, des données peuvent être partagées au sein d'un ou plusieurs niveaux. Toutes les données sont gérées par un système de gestion de données distribué complètement séparé. Le concept de "mémoire

commune" est utilisé pour transmettre vers et recevoir en provenance des fonctions de contrôle par l'intermédiaire d'un réseau de communication.




*relations "control input" et "status feedback" inter-niveaux*

Il est important de noter que la définition de ce modèle n'est pas directement dédiée au processus de conception, c'est-à-dire l'aspect méthode, mais à la définition et l'élaboration de composants standards, c'est-à-dire l'aspect réalisation.

A chaque niveau correspond l'horizon de planning, c'est-à-dire une période de temps dans laquelle un module est responsable de la planification et de la mise à jour des objectifs locaux.

Pour chaque niveau, les fonctions sont définies de la façon suivante:

Le niveau **facility** comprend les fonctions:

- manufacturing engineering (CAD),
- information management (fournit les informations nécessaire à la gestion pour l'estimation des coûts, la comptabilité des temps de fabrication, la facturation client, la comptabilité d'inventaire, la gestion des commandes, la gestion du personnel et de la paie),
- production management élabore les ordres pour le niveau inférieur, détermine les capacités de production, les données de qualité et de performance.

Le niveau **shop** est composé d'un task manager responsable de la planification des capacités, de l'ordonnancement des ordres en séries de fabrication, l'affectation de tâches aux cellules, l'allocation des ressources et le suivi de la réalisation de ces consignes. Le ressource manager est responsable de l'allocation des ressources de production, en particulier de l'affectation dynamique des stations en cellules virtuelles.

Le niveau **cell** est responsable de la planification et la coordination des cellules composées de différentes stations.

Chaque **workstation** est un ensemble intégré de machines de manutention, de mesure et de transformation qu'il faut coordonner. Une "station de travail" typique comprend par exemple un robot, une machine outil, un stock tampon de produits, un poste de préparation, et un calculateur de contrôle.

Le niveau **equipment** est constitué par les automatismes de contrôle des machines de production. Les deux principales fonctions des contrôleurs de ce niveau sont:

- transformer les commandes des stations en séquences de tâches simples compréhensibles par les contrôleurs fournis par les fournisseurs d'équipements,
- suivre l'exécution de ces tâches via les capteurs attachés au hardware.

Ce niveau a pour vocation d'homogénéiser le niveau de service des contrôleurs fournis par les différents fournisseurs.

Notons qu'après de nombreux et important travaux sur ce sujet et une acceptation assez large du modèle proposé, Mac Lean remet en cause l'approche "modèle de référence" dans une de ses dernières publications [McLe 87] et prône plutôt une *"philosophie d'interface standard"* permettant de construire des architectures adaptées à chaque organisation.

### 2.1.1.2. le modèle de WILLIAMS

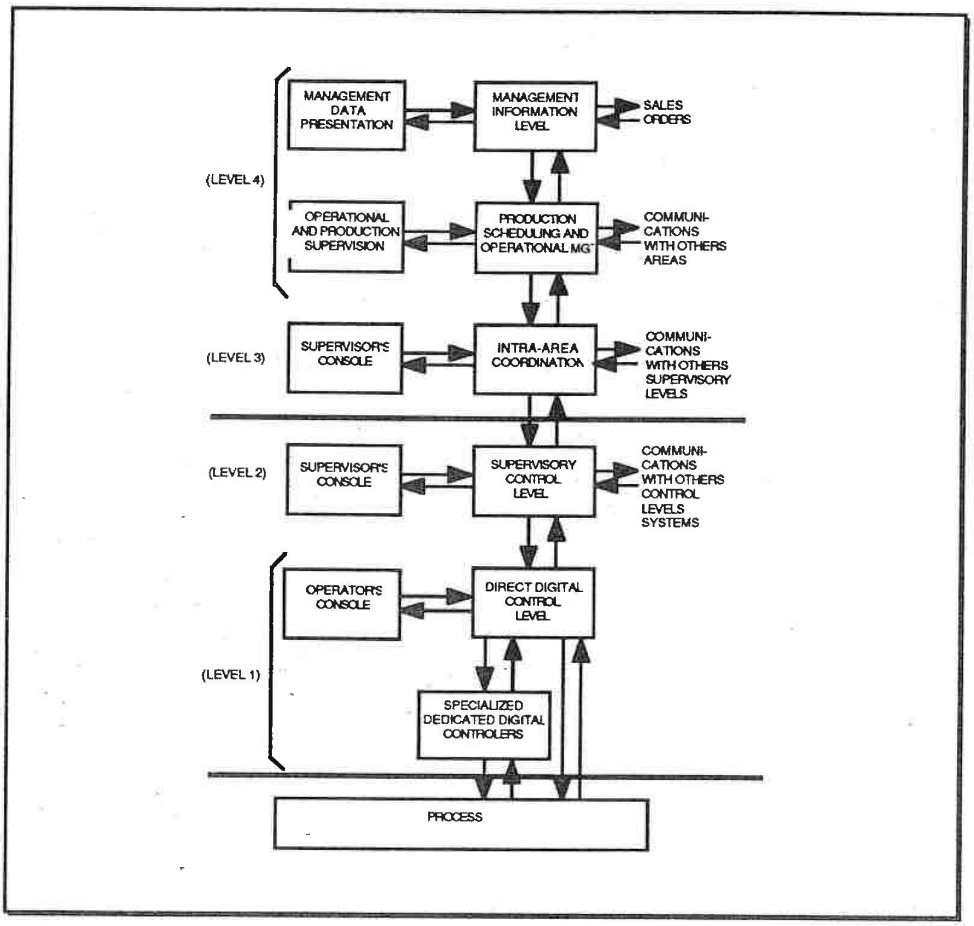
références: [Wil 87]

Les travaux de Williams s'inscrivent dans le cadre des travaux d'élaboration d'un modèle de référence menés par l'université de Purdue aux Etats-Unis pour la proposition d'une démarche de normalisation d'un modèle de référence en relation avec l'ISO/TC184/SC5/WG1. Les publications mentionnées sont relatives à un constat d'évolution des technologies et des architectures justifiant l'établissement d'un modèle de référence, plutôt qu'à la définition d'une méthodologie de conception.

Le modèle est issu d'une large étude des architectures utilisées dans des secteurs divers de l'industrie et de l'analogie avec le modèle OSI. Il intègre essentiellement l'organisation hiérarchique caractéristique à la fois des domaines process continu et manufacturier (dont un comparatif est synthétisé). Le modèle proposé possède 4 niveaux, dont certains sont décomposés en sous-niveaux:

4 - "management information" et "inter-area coordination",
3 - "intra-area coordination",
2 - "supervisory control",
1 - "direct digital control" et "specialized dedicated digital control".

Ce modèle assez classique intègre des liaisons verticales command / feedback et communications horizontales entre fonctions d'un même niveau ou avec les opérateurs. L'auteur semble admettre implicitement l'existence d'un autre axe fonctionnel: "*les différents niveaux hiérarchiques supportent différentes fonctions* ("duties")" .




*les 4 niveaux proposés dans le modèle de l'université de Purdue*

#### 2.1.1.3. la grille GRAI

références: [Dou 85], [Dou 87], [Bre 84], [Dou 84]

La méthode GRAI (Graphes à Résultats et Activités Interreliés) est une méthode d'analyse et de conception des systèmes de gestion de production développée à Bordeaux par le laboratoire GRAI. Bien que non-dédiée à la représentation des architectures de SAP, les concepts et modèles proposés n'en sont pas moins intéressants quant à la globalité de l'approche.

Le système global est décomposé en fonction de la **nature** des **activités**: information, décision, ou physique, conformément à l'esprit de l'approche systémique.

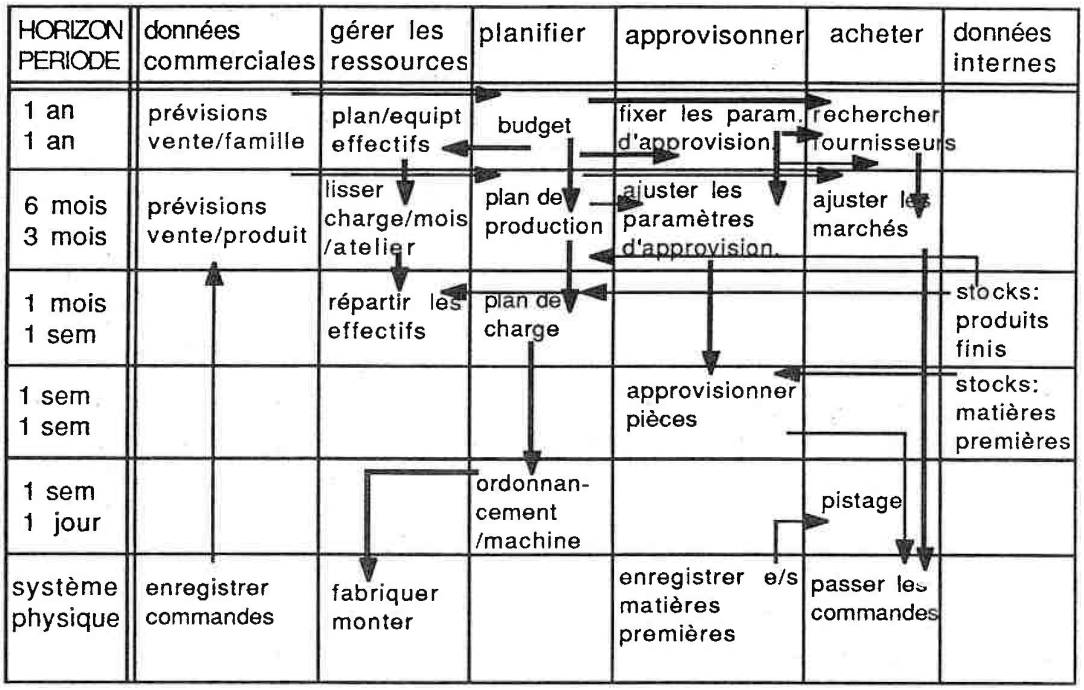
La "macro-structure" est utilisée par le modèle conceptuel GRAI pour décrire la structure du système de décision (et non pas celle du système d'information). Trois critères de décomposition sont utilisés suivant :

- les aspects **temporels** d'**horizon** (intervalle de validité des décisions) et de **période** (intervalle de révision des décisions),
- la **nature fonctionnelle** des activités, sans hiérarchie entre les différentes fonctionnalités.

Cette macrostructure est décrite au moyen d'un tableau baptisé "**grille** GRAI" permettant de représenter les centres de décision dans ces **deux dimensions** : une dimension temporelle "verticale" hiérarchisée et une dimension fonctionnelle "horizontale"

Les liaisons entre les centres de décision sont constitués par des informations de deux types:

- cadre de décision ("decision frame"): objectifs liés à la notion de performance requise pour la coordination des centres de décision,
- information de service complémentaire pour la prise de décision.




*exemple de grille GRAI*

Les centre de décision peuvent être décomposés hiérarchiquement en suivant la structure hiérarchique du processus physique de production, mais cette dimension horizontale, "orthogonale" aux deux dimensions précédentes, n'est pas explicitée dans les grilles GRAI.

La description formelle des centres de décision est réalisée au moyen des "réseaux GRAI" permettant de décrire les activités et leurs supports.

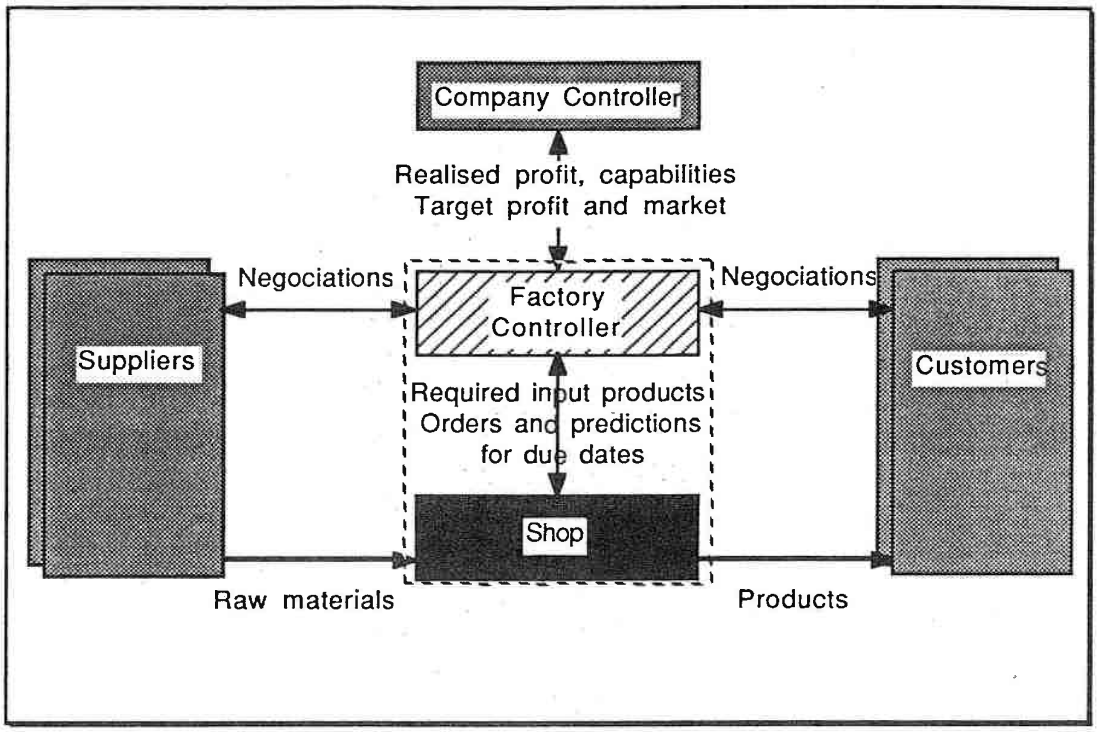
#### 2.1.1.4. le Reference Model de BIEMANS

références: [Bie 88], [Bie.a 86], [Bie.b 86], [Bie.B 86]

La démarche suivie dans les travaux de Biemans au sein du Computer Aided Manufacturing Centre de Philips aux Pays-Bas est très proche de nos objectifs puisqu'elle concerne la conception du "système de planification et de contrôle de production" (MPCS). Au fil des différentes publications les modèles et la méthode se sont à la fois affinés et généralisés pour ne plus seulement concerner les problèmes de transport. La voie suivie est de reproduire l'approche couches et services telle qu'elle est définie dans le modèle OSI, allant jusqu'à employer un langage de spécification de protocoles (Lotos) pour formaliser les spécifications d'architecture.

La démarche de conception en 5 étapes a pour objectif essentiel de retarder le plus longtemps possible les décisions de conception spécifiques à l'application. Elle propose de:

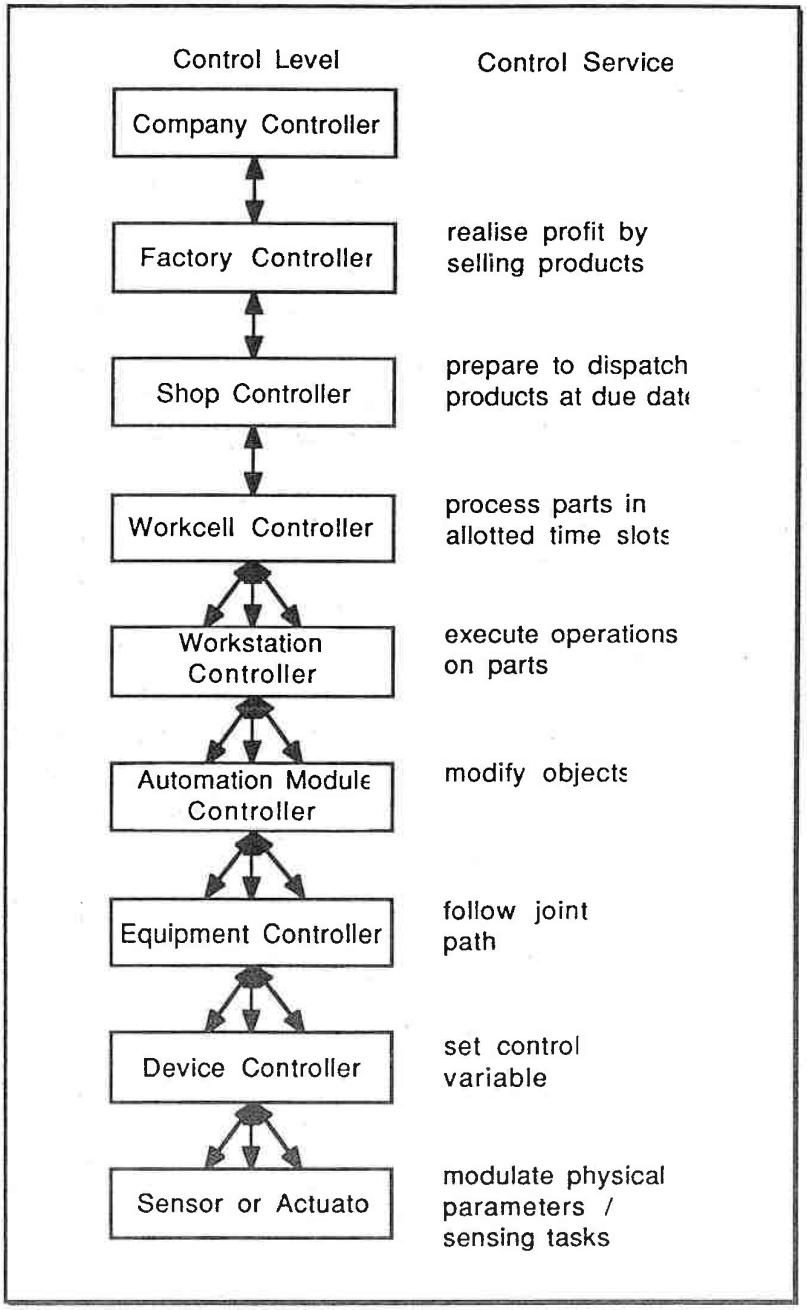
- définir les limites du système en tant que boîte-noire,
- décomposer cette boîte en une organisation hiérarchisée de composants coopérants,
- spécifier le comportement externe de ces composants (E/S),
- spécifier la "description générative" (modèle interne) définissant les relations entre les E et les S de chaque composant,
- déterminer quels composants sont implantés sur quels systèmes physiques.




*modèle global de l'entreprise CIM par Biemans*

Le modèle (dans sa dernière version publiée en 88) est basé sur la notion de service décomposable en contrôleur (qu'il n'est pas nécessaire de décomposer) et services de niveau inférieur (décomposables de façon itérative selon le même processus). Les aspects élaboration de l'architecture opérationnelle (physique) et la dimension fonctionnelle "horizontale" (autre que le contrôle: maintenance, qualité,...) ne sont pas abordés. Les niveaux proposés sont company, factory (externes au MPCS) et pour le MPCS:

- shop (gestion stocks et approvisionnements),
- workcell (objets vus par leur nom, quelles opérations, où et quand), -
- workstation (ordre des modifications physiques),
- automation module (caractéristiques physiques modifiées, états de l'équipement),
- equipment (valeurs des variables de contrôle),
- device (signaux de contrôle et de mesure),
- sensor/actuator.




*les 8 niveaux du modèle CAM-Centre Philips proposé par Biemans*

Si 8 niveaux sont proposés, tous ne sont pas obligatoirement utilisés selon la complexité de l'application. Les échanges d'informations sont verticaux (commande et feed-back) et définissent les relations hiérarchiques de contrôle (les échanges d'informations horizontaux, "peer to peer" semblent tolérés dans le modèle mais aucune précision n'est donnée sur ce type de liaison). Un modèle interne permet de décrire chaque entité (décomposition des commandes, synthèse des compte-rendu et modèle des

### 2.1.1.5. ESPRIT 932 Knowledge Based Real-Time Supervision in CIM

références: [932 86], [Mey 87]

Ce projet européen traite du développement d'un contrôleur de cellule ("workcell controller"). Plus précisément le projet spécifie les besoins d'une architecture de contrôle capable d'utiliser la technologie des systèmes experts pour la planification, l'assurance qualité et la maintenance préventive dans un environnement CIM. L'architecture retenue pour la modélisation du SAP est tirée de celle établie par le NBS.

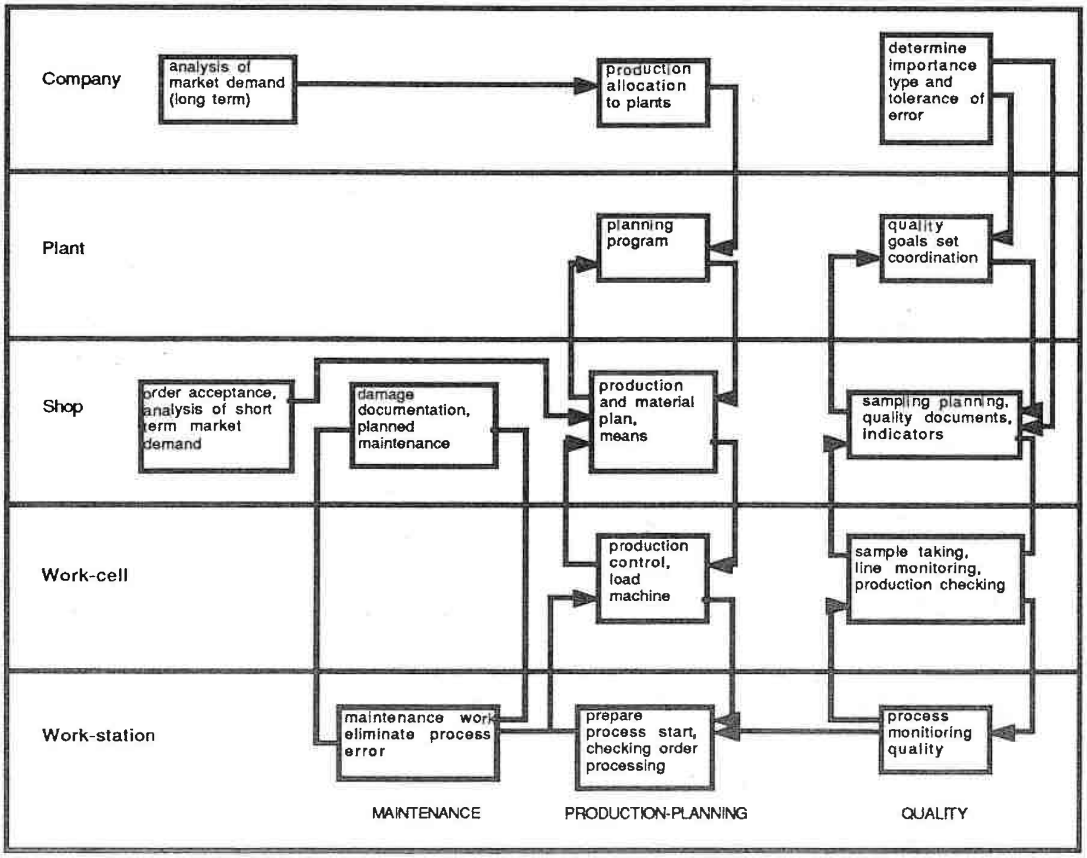
| hierarchical level | time horizon time period | mission | scope |
|---|---|---|---|
| COMPANY | years | corporate management | formulates policy places budgets |
| FACTORY | years month | plant management | responsible for overall planning and execution |
| SHOP / | month | production | schedules production, delivers management |
| WORKCENTER | day | management | informations by monitoring and supervising |
| WORK CELL | day minute | production supervision | coordinates production flows, integrates individual stations into an automated system |
| WORK STATION | minute second | process control | performs logic necessary to convert input from lower level to output commands up and down |
| AUTOMATION MODULE | milli-second | machine control | basic interface with plant floor, sensing and control devices respond to upper level commands |

*le modèle d'architecture repris par le projet Esprit 932*

Les niveaux où le contrôleur générique doit s'insérer sont les niveaux "shop" et "workcell", car ce sont les niveaux où l'intégration entre la gestion de production et la conduite du processus est insuffisante. Il s'agit de combler le vide entre le planning de production et l'ordonnancement en temps réel tenant compte de l'état du processus.

Un modèle global de représentation des architectures SAP est proposé sous la forme:

- d'une extension de SADT incorporant la notion de boucle de contrôle (SADT / CONTROL LOOP) alliée
- aux niveaux proposés par le modèle du NBS complétés par
- la dimension fonctionnelle horizontale des grilles GRAI.



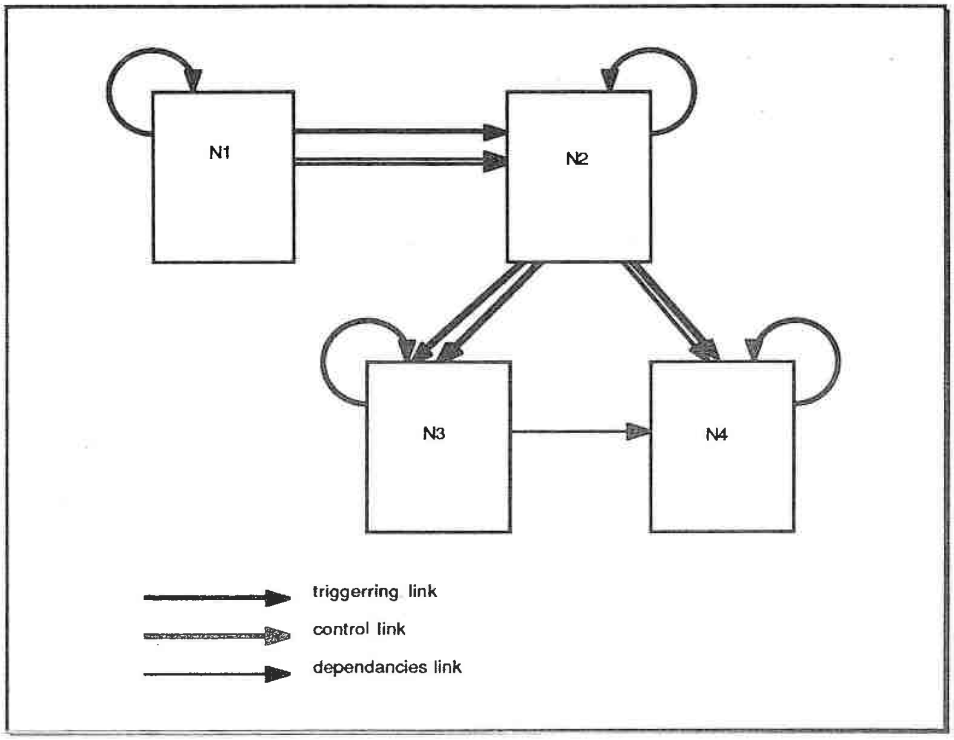

*l'adaptation NBS / SADT / GRAI utilisée par Esprit 932*

En fait les symboles utilisés semblent plus proches de la simple notion de block-diagrammes organisés de façon hiérarchique car ce qui subsiste du formalisme SADT n'apparait pas clairement:

- les liaisons typiques de ce modèle sont modifiées,
- la décomposition hiérarchisée ne semble pas utilisée, et
- le nombre et la disposition des boites ne sont pas respectés.

Le modèle CSM Conceptual Supervision Model proposé par le LIA (Laboratoire d'Intelligence Artificielle, [Aye 88]) vise un cadre certes plus restreint que la modélisation globale d'une architecture SAP, mais explicite plus en détail que les autres modèles les différents types de liaisons autorisées entre les composants. Le système de supervision est défini comme la fonction de contrôle des interactions et des synchronisations des différentes activités d'une usine. 3 types de liens sont définis entre les unités de supervision, correspondant à ceux proposés dans nos travaux précédents [Del 88]:

- liens de "**control**" formés de "decision frames" et définissant une relation hiérarchisée entre les composants,
- liens de "**triggering**" ou déclenchement permettant de spécifier par quel composant est donné l'ordre d'activation-synchronisation,
- liens de "**dependancies**" correspondant à des échanges de données nécessaires à l'accomplissement d'une activité.




*Le CSM Conceptual Supervision Model Esprit 932*

La distinction entre contrôle et déclenchement-synchronisation est intéressante car en effet les deux rôles de définition des objectifs et d'activation peuvent être donnés par des entités distinctes. On peut d'ailleurs considérer que ces deux notions correspondent à deux niveaux de spécification de l'architecture, le premier définissant les relations de

contrôle fonctionnel de l'architecture, le deuxième définissant des choix d'organisation du contrôle dynamique de l'architecture.

Ces différents liens peuvent supporter différents types de messages entre deux unités:

| type de liaison 1 -- 2 | type de message | type d'information (sémantique) | sens 1 -- 2 |
|---|---|---|---|
| control | information | decision frame | --> |
| | | change of structural objectives | --> |
| | | reporting | <-- |
| | alarme | alarm | <-- |
| | | change of decision frame | --> |
| | | relaxation of objectives request | <-- |
| | urgence | decision frame request | <-- |
| | (emergencies) | report request | --> |
| trigger | normal mode | synchro | --> |
| | | auto-synchro | auto |
| | alarme mode | triggering in perturbation | --> |
| | auto-trigger | auto | |
| dependencies | information | periodic orders or constraints (o/c) | --> |
| | alarme | periodic o/c in perturbation mode | --> |
| | | conflict between objectives and o/c | <-- |
| | urgence | o/c request | <-- |

*les différents types de liaisons et de messages du modèle CSM*

Chaque message est spécifié par différents attributs:
- identificateur,
- type de message,
- type de liaison,
- origine,
- destination,
- contenu: information, début de validité, date d'émission, durée de validité.

### 2.1.1.6. l'Architecture CIM / OSA

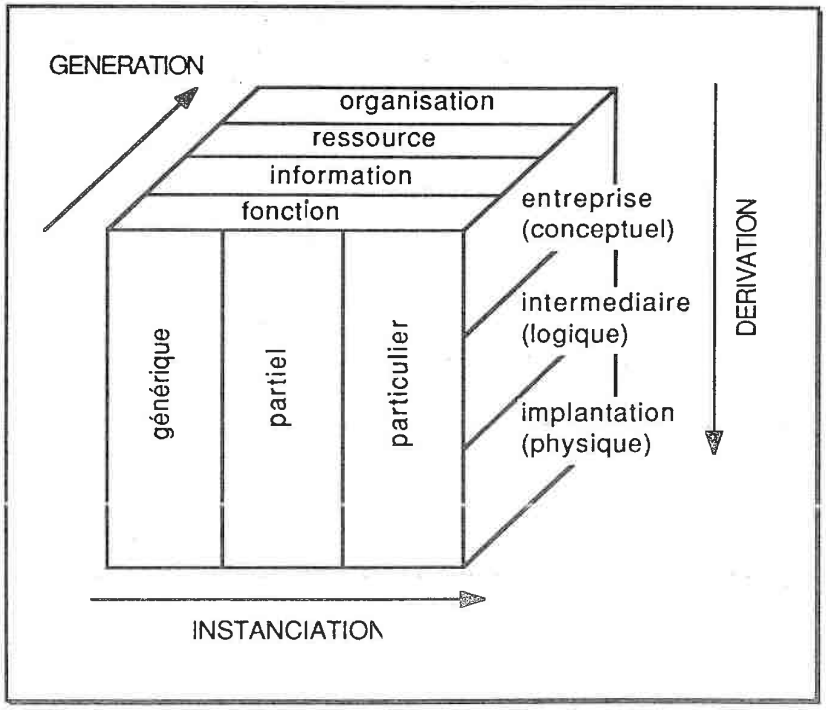
références: [Lav 88], [CIM 88]

Le contexte étudié par le projet AMICE (Architecture Manufacturing Integrated Computer European) ESPRIT Project n° 688 est nettement plus large que celui qui nous intéresse puisqu'il couvre l'ensemble des services de l'entreprise.

La notion d'Open System Architecture (OSA) réfère aux possibilités de connexion des systèmes CIM à d'autres environnements, modélisés par un plan structuré (l'architecture) contenant les règles et la définition des propriétés des composants du système. Le but est de fournir un ou des modèles de référence permettant de mieux décrire, concevoir et construire les architectures intégrées.

Une architecture est représentée dans un repère à 3 dimensions (modèle en cube, cf schéma):

- la dimension de l'*"instanciation"* : allant du "générique" au "particulier",
- la dimension de la *"dérivation"* : allant du conceptuel "entreprise" à l'"implantation"
- la dimension de la *"génération"* : modèle d'Entreprise allant du niveau "fonction" au niveau "organisation" en passant par le niveau "information".



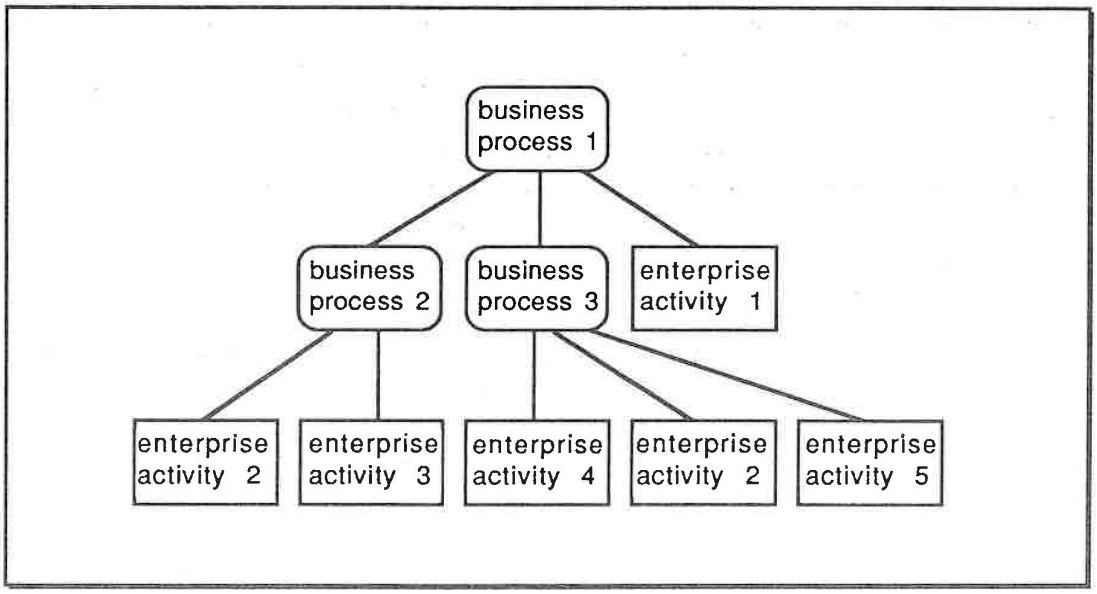

*les différentes dimensions du modèle CIM-OSA*

Tous ces aspects ne sont pas parvenus au même stade d'avancement. Les modèles d'Entreprise selon les *"points de vue"* ou *"sous-modèles"* **fonction** et **information** sont les mieux définis. Chacun de ces sous-modèles est divisé en modèle de "**référence**" et en modèle de "**particulier**". Le passage du niveau référence au niveau particulier se fait par instanciation des types de référence .

Le **point de vue fonction** décrit l'entreprise en termes :

- d'**activités** avec les entrées / sorties (physiques ou informatives) et les ressources nécessaires,
- la façon dont les activités sont associées en " **business process**" avec les structures de contrôle (règles définissant les flux d'actions),
- les événements déclencheurs ou conditionneurs des activités ou process.

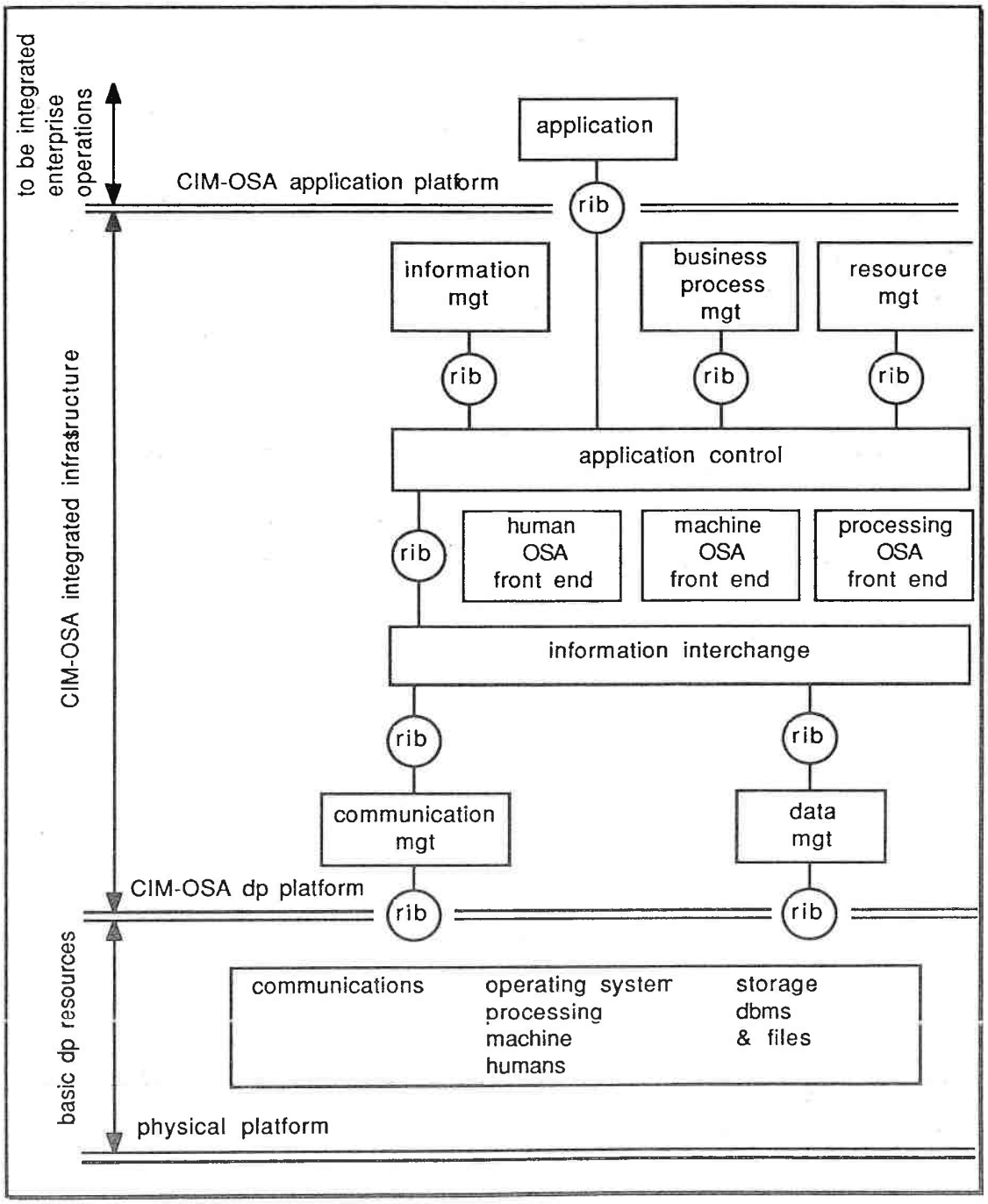
Une activité est le niveau élémentaire de description et ce niveau de détail doit être suffisant pour que les ressources soient identifiables.




*modèle des fonctions CIM-OSA*

Le **point de vue information** décrit les informations en entrées et sorties des activités par leur contenu et format. Un schéma "conceptuel" utilise actuellement le modèle classique ERA Entité-Relation-Attribut. Le schéma "externe" décrit comment l'information est saisie ou présentée aux utilisateurs. Le schéma "interne" décrit comment l'information est accédée et stockée physiquement dans le système informatique. Ce modèle pourrait évoluer prochainement vers une modélisation objet.

L'autre aspect le plus avancé est la définition d'un cadre ou infrastructure d'intégration baptisé IDPE Integrated Data Processing Environment qui vient se superposer aux ressources informatiques techniques de base de communication, système et base de données.



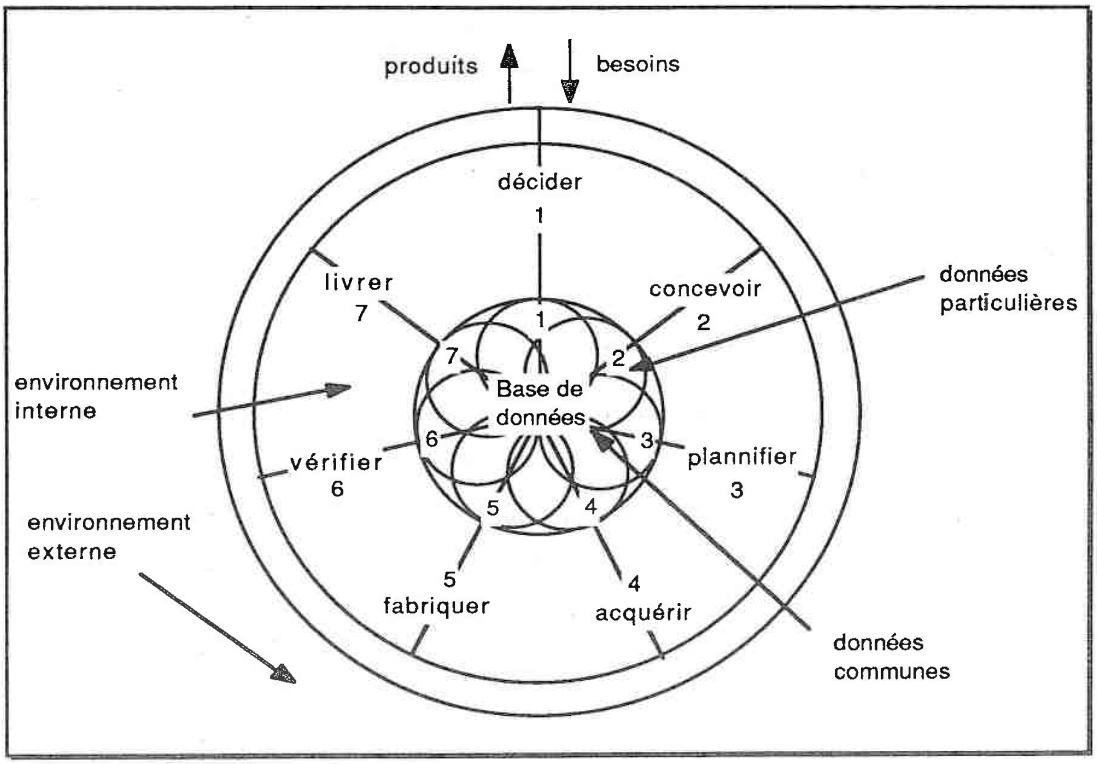

*le modèle IDPE Integrated Data Processing Environment*

(rib: reference boundary interface)

### 2.1.1.7. la méthode CAM-I (Computer Aided Manufacturing - International)

références: [Dou 86]

La méthode adoptée par le projet Factory Management du CAM-I concerne la conception des systèmes de GPAO. Cette méthode est basée sur un modèle conceptuel des activités fonctionnelles et une méthode de conception SSAD (Structured Systems Analysis Design) issue des travaux de Gane-Sarson basés sur les diagrammes de flux de données, proches des méthodes SA/SD déjà évoquées, aux conventions de notation près (les cercles étant remplacés par des carrés à coins arrondis). De nombreux outils de génie logiciel supportent maintenant ces notations.




*le modèle conceptuel des activités fonctionnelles du CAM-I*

Le modèle conceptuel fait bien apparaître la dimension fonctionnelle du SAP, chaque fonction intervenant à chaque niveau de la hiérarchie allant du niveau direction générale aux unités de fabrication:

- niveau direction générale,
- niveau usine,
- niveau atelier,

- niveau centre de coûts et centre de service (outillage, maintenance, magasins, contrôle de fabrication, manutention)
- niveau unités de fabrication.

Les 7 fonctions sont organisées autour d'une base de données propres ou partagées.

### 2.1.2. Modèles de type SADT-IDEF0

Les modèles présentés dans ce paragraphe n'emploient pas de formalisme ou de schéma de représentation propre mais reprennent ou adaptent les notations SADT-IDEF0.

#### 2.1.2.1. la méthode IDEF de l'ICAM Program et le COMPOSITE VIEW Model de BRAVOCO-YADAV

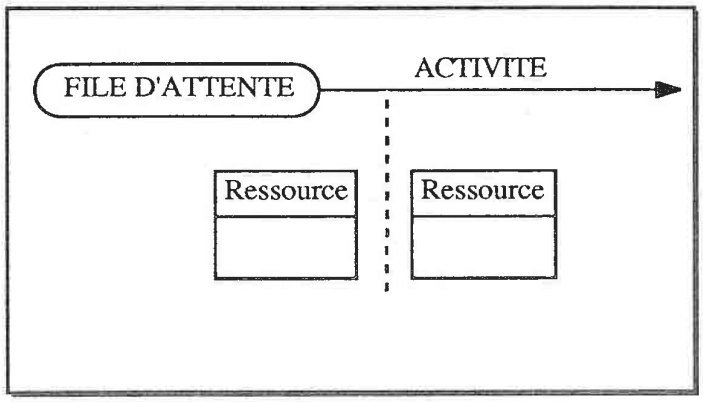
références: [Bra.a 85], [Bra.b 85]

Le programme Integrated Computer Aided Manufacturing de l'U.S.Air Force a identifié un besoin de mieux communiquer et analyser la fabrication pour les personnes impliquées dans l'amélioration de la productivité. Pour satisfaire ce besoin l'ICAM a développé la méthode IDEF (ICAM Definition) basée sur trois modèles IDEF0, IDEF1 et IDEF2.

IDEF0 est un **modèle fonctionnel** basé directement sur les actigrammes SADT pour définir la structure du système de production (flux de produits et d'informations),

IDEF1 est un **modèle informationnel** basé directement sur le modèle entité-relation de représentation des données,

IDEF2 est un **modèle dynamique** de représentation du comportement et des interactions temporelles entre les composants du système.




*le modèle dynamique IDEF2*

La méthode est largement basée sur le dialogue entre concepteur et les personnes impliquées dans l'esprit de SADT (validation par cycle auteur-lecteur).

Les travaux de Bravoco et Yadav s'inscrivent dans la démarche du programme ICAM. Utilisant le modèle fonctionnel (IDEF0) de la méthode IDEF (ICAM Definition), les auteurs présente le modèle générique baptisé "Composite View" synthétisant les traits

communs aux différentes organisations existantes "Factory Views". Ces organisations peuvent être décomposées en "Subsystems Views" qu'il s'agit de rendre génériques. En bref l'approche consiste à proposer un modèle général représenté sous forme SADT. La généralité du modèle est cependant limitée par le domaine d'étude : l'industrie manufacturière aéronautique.

MANUFACTURE products
- PLAN for manufacture
- MAKE and ADMINISTER schedules and BUDGET
- PLAN production
- PROVIDE resources
- OBTAIN materials
- PRODUCE product
  - CONTROL orders
  - CONTROL items & tools
  - PERFORM physical production
    - FABRICATE parts
    - ASSEMBLE component parts
    - PERFORM major assembly
  - TEST ,CHECK OUT
  - DELIVER product

*arborescence des diagrammes SADT proposée par Bravoco-Yadav*

### 2.1.2.2. le système de production modulaire de NOYES

références: [Noy 87]

Axés sur l'amélioration de la sûreté de fonctionnement (fiabilité et disponibilité) d'un système de production, les travaux de Noyes visent une structuration renforçant la modularité. Les fonctionnalités sont classées en considérant que les attributs produits,outils et programmes doivent être rassemblés, activés et mis en replis. Ces activités sont représentées à l'aide du formalisme SADT en quatre niveaux de décomposition. Ce modèle a la particularité de ne pas distinguer explicitement les parties opératives et les parties commandes.

ASSURER la production
- GERER le système
- SERVIR en pièces
  - GERER le service
  - PREPARER les pièces * — COMMUNIQUER, PILOTER, EXECUTER
  - STOCKER *
  - SERVIR une pièce *
- SERVIR en programmes
  - GERER le service
  - PREPARER le programme *
  - STOCKER *
  - SERVIR le programme *
- SERVIR en outils
  - GERER le service
  - PREPARER les outils *
  - STOCKER *
  - MESURER les outils *
  - SERVIR un outil *
- TRAITER une pièce
  - GERER le traitement
  - PREPARER le chargement
  - DECRIRE le programme
  - EXECUTER le traitement

*arborescence des diagrammes SADT proposée par Noyes*

* activité pouvant être décomposée en COMMUNIQUER, PILOTER, EXECUTER.

### 2.1.2.3. le Modèle du CRAN

références: [Lho 85]

La thèse de Lhoste présentée au Centre de Recherche en Automatique de Nancy comprend une contribution *"vers une méthodologie de conception de la commande d'unités automatisées de fabrication"*. Cette méthode ne traite pas directement la conception de l'architecture mais elle est conçue pour répondre aux besoins suivants: "*nécessité de continuité entre chacune des phases de création d'un système automatisé, nécessité d'un outil permettant une vision structurelle globale des besoins et suffisamment simple pour faciliter le dialogue entre concepteurs et exploitants*".

Les 4 étapes composant cette méthode sont:

1. Décomposition du processus en sous-processus: aucune règle fondamentale de décomposition n'est donnée mais des critères de structuration sont suggérés: géographique et fonctionnel. L'idée est d'associer à chaque niveau du processus de décomposition une fonction contrôle. Le formalisme est celui de SADT dont les notations sont conservées, mais les règles d'utilisation sont modifiées pour représenter les relations requête/ compte-rendu entre l'activité contrôle et les processus.

2. Etude des activités d'exploitation: la représentation de l'exploitation est basée sur un modèle d'état/transition et non un modèle SADT.

3. Etude de la fonction contrôle: la décomposition de la fonction contrôle en activités est basée sur une représentation à base de grilles supportant les spécifications des activités et des flux d'informations selon la hiérarchie:

contrôler:
- piloter
  - suivre
  - optimiser
  - régler
- conduire
  - percevoir
  - réfléchir
  - agir
- coordonner
  - surveiller
  - réagir
  - commander

*arborescence des diagrammes SADT proposée par Lhoste*

4.Description des fonctions: la méthode suggère l'utilisation du Grafcet ou de Réseau de Petri.

## 2.2. Synthèse des concepts retenus

Ce paragraphe effectue la synthèse des principaux concepts représentés dans ces modèles d'architecture avant d'aborder la formalisation détaillée dans le chapitre suivant.

Remarquons dès à présent l'absence de concept de modélisation des structures de données dans tous les modèles examinés sauf dans les travaux du projet CIM-OSA. Ce projet reprend d'ailleurs intégralement le modèle ERA sans lui apporter de spécificité.

### 2.2.1. Fonctions

Les modèles examinés sont surtout focalisés sur les fonctions conduite et planification du SAP. Certains modèles élargissent cependant ce champ en proposant des listes plus complètes de fonctionnalités. Généralement cette décomposition en fonctions se traduit par l'explicitation d'une dimension horizontale du modèle fonctionnel, c'est-à-dire orthogonale à la hiérarchie de contrôle. En effet il n'est pas possible de définir de relations hiérarchiques entre ces fonctions que l'on doit donc situer au même niveau.

| CAM-I | GRAI * | EDF | Composite View | SP Modulaire |
|---|---|---|---|---|
| design<br>decide | concevoir | | | |
| plan | planifier | | plan for manufacture<br>plan production | |
| acquire | approvisionner | | obtain materials<br>provide resources | servir |
| make | fabriquer | conduire | produce products | traiter |
| verify | contrôle qualité<br>maintenir | maintenir | | |
| deliver | livrer<br>gestion économique | gérer | administer schedule/budget | gérer |
| | gestion personnel | | | |

*synthèse des fonctionnalités mentionnées dans les modèles d'architecture examinés*

* en fait, compilation des variantes des grilles GRAI trouvées dans les différentes publications.

Il parait difficile de fixer une liste type de fonctions standards au vue de la diversité de ces modèles, diversité directement issue de la diversité des organisations des entreprises. La comparaison de ces différents modèles montre qu'il n'est pas possible d'établir de correspondance directe et rigoureuse entre les fonctionnalités listées. Ceci est essentiellement dû à la diversité des points de vue et à la différence des choix de structuration hiérarchisée. De plus les différents modèles ne couvrent pas la même étendue, allant du SAP et de sa partie commande à la représentation de toute l'entreprise intégrée "CIM".

D'une façon générale, il paraît utopique voire prétentieux de viser l'élaboration d'un modèle idéal figé pour tous les cas de figure. Comme le constate [McL 87], il nous paraît plus intéressant de fournir aux concepteurs des "briques" de base permettant de bâtir une architecture répondant à l'ensemble des besoins exprimés par le client plutôt que d'imposer une organisation rigide préfabriquée des fonctionnalités: "*Il est peut-être temps de repenser notre approche de l'intégration. D'une certaine façon, nos tentatives de définir une architecture de référence pour les usines automatisées revient un peu à essayer de définir le bâtiment parfait. Je crois que les architectes pourraient démontrer qu'un tel modèle de référence est inutile.*" Cette approche est d'ailleurs beaucoup plus cohérente avec la définition de l'architecture donnée au premier chapitre. Il ne s'agit pas de proposer un modèle de référence magique (l'architecture idéale), mais d'offrir le moyen de formaliser des concepts abstraits (éléments et relations) et des règles de construction permettant d'élaborer une structure répondant aux besoins de chaque application.

### 2.2.2. Hiérarchie

On constate que le concept de hiérarchie est omniprésent dans tous ces modèles comme dans les architectures matérielles proposées par les constructeurs. Nous nous intéressons ici essentiellement à l'aspect organisationnel de la hiérarchie en tant que concept de structuration. Il s'exprime par une liaison bidirectionnelle de transmission de commande ou de définition d'objectifs et de retour de compte-rendu d'exécution (positif ou négatif). Cette liaison définit une relation orientée entre les entités fonctionnelles et permet donc d'ordonner ces entités en niveaux. La plupart des modèles proposent de définir le nombre et les fonctionnalités de ces niveaux. Cette définition est basée :

- soit sur la **décomposition hiérarchisée du processus physique de production** (NBS, MPCS),
- soit sur l'examen des **rôles fonctionnels** affectés à chaque niveau (Williams)
- soit sur les **caractéristiques temporelles** des niveaux (GRAI, Esprit 932).

Mais comme pour la liste des fonctions (cf paragraphe précédent), il ne semble pas pertinent de dégager un **nombre de niveaux** standard, la décomposition dépendant là encore de la variété des organisations et surtout de la variété des points de vue. Il semble donc plus intéressant de fournir les mécanismes de spécification des relations hiérarchisées et de laisser le concepteur libre de définir le nombre et les fonctions de chaque niveau le plus adapté aux particularités du problème à résoudre.

### 2.2.3. Structure du processus de production

La structure du processus apparaît déterminante dans la structuration de l'architecture informatique du SAP. Les niveaux hiérarchisés de l'architecture sont souvent définis suivant la décomposition du processus (NBS, MPCS). Cette constatation peut sembler évidente dans la mesure où le système informatique-automatique est "tourné" vers le processus, c'est-à-dire doit planifier, conduire ou suivre ce processus. Contrairement aux modèles des méthodes de conception des systèmes d'information, les modèles d'architecture représentent explicitement les éléments du processus. Il est important pour le concepteur de connaître la structure du processus et de pouvoir spécifier pour chaque composant fonctionnel la partie du processus dont il est responsable, aussi bien directement que par l'intermédiaire de composants de niveaux inférieurs.

### 2.2.4. Notion de service

Plusieurs auteurs expliquent leur motivation d'élaborer un modèle de référence par analogie avec le modèle de référence OSI-ISO des architectures de communication. Dans l'esprit des travaux de Biemans, il nous semble intéressant de ne pas se contenter de remarquer cette analogie de structuration en niveaux, mais d'utiliser la notion de service pour faciliter la représentation et la spécification des architectures informatisées de SAP.

### 2.2.5. Communications

Les principaux type de liaisons que l'on peut distinguer dans ces modèles entre les composants d'une architecture sont :

1 - les liaisons "verticales" hiérarchiques de service (ou contrôle) bidirectionnelles à base de commande et de compte-rendu,

2 - les liaisons "horizontales" d'échange ou de partage de données unidirectionnelles entre entités de même niveau,

3 - les liaisons d'activation-synchronisation également unidirectionnelles.

Le premier type de liaison présent dans tous les modèles reflète l'importance accordée au concept de hiérarchie.

Le second type de liaison n'est pas toujours explicite dans tous les modèles. Il apparaît surtout dans les modèles comprenant une dimension fonctionnelle "horizontale" complémentaire de la dimension hiérarchique "verticale".

La distinction du troisième type de liaison est rarement introduite dans tous ces modèles. Ceci est du au fait que la spécification dynamique du contrôle constitue une étape ultérieure et un niveau de détail plus important par rapport aux modèles proposés.

Nous préciserons les différences sémantiques de ces communications dans le chapitre 6 paragraphe 1.2. concernant la modélisation des communications.

# 3. CONCEPTS UTILISES EN INGENIERIE DES ARCHITECTURES DE SAP

Les paragraphes précédants ont permis d'analyser les concepts des Méthodes de Conception des Systèmes d'Information et des Modèles d'Architecture de SAP. Ces concepts sont en partie utilisés par les spécialistes en conception. Nous avons vu quelles étaient les méthodes les plus utilisées et quelles étaient les adaptations apportées par les concepteurs. Cependant d'autres concepts utilisés comme **critères de structuration** sont peu ou mal formalisés. Ce chapitre examine donc ces concepts, propres au métier de l'ingénierie des architectures de SAP.

L'exemple choisi pour illustrer ces concepts est une des deux études menée de façon approfondie dans le cadre de ce travail de recherche. Il s'agit d'une application de gestion de mouvements de produits en raffinerie dont l'étude préalable a été effectuée par la société ESIA: "Tank Farm Automation Project: Preliminary System Design Study". Les indications fournies resteront partielles et générales pour préserver la confidentialité de cette étude industrielle.

## 3.1. Présentation de l'étude

Cette application concerne l'automatisation des mouvement de produits pétroliers dans une raffinerie. Elle comprend essentiellement l'automatisation des transferts entre réservoirs (tanks), l'automatisation des mélanges et la gestion du parc de réservoirs. Ce type d'application a pour particularité d'être relativement peu automatisée, contrairement au contrôle des processus de raffinage [ESIA: "The ESIA approach to product transfer automation, tank farm management and blending automation computerized systems." Novembre 1988]: *"L'automatisation des tâches de mouvements de produits, apparaissant moins sophistiquée que la conduite et l'optimisation du raffinage, plus difficile à justifier économiquement, nécessitant une démarche globale et une refonte de l'organisation, n'a jusqu'à présent pas fait l'objet de nombreux investissements."*

**Transfert de produits:**

Le transfert de produits représente l'activité principale du parc de réservoirs. Les produits sont reçus depuis des pipelines et des réservoirs, des produits finis sont expédiés par camions, trains, tankers ou pipelines, et pendant les opérations de mélange les produits semi-finis sont transférés de réservoirs vers d'autres réservoirs. Tous ces transferts utilisent différents réservoirs ou pipelines qui constituent l'origine et la destination des produits, ainsi que des pompes, vannes et conduites permettant le transfert proprement dit.

Ces éléments nécessitent:

- la préparation (configuration) et la commande des équipements (pompes, vannes) pour assurer le déroulement correct de l'opération,
- la vérification que seuls les transferts autorisés peuvent être effectués, et que les quantités nécessaires sont disponibles,
- les contrôles pour prévenir les fuites et autres incidents,
- le contrôle de la qualité des produits par rapport aux spécifications,
- le suivi des transferts conformément au planning.

**Mélange de produits:**

Le mélange permet d'obtenir des produits finis à partir de produits intermédiaires. Cela nécessite:

- le contrôle des ratios de mélange à partir d'un modèle de programmation linéaire nécessitant des données en provenance du laboratoire sur les caractéristiques des produits;
- la conduite des séquences d'opérations de mélange, de contrôle des débits, de vérification des analyseurs.

**Gestion du parc de réservoirs:**

Le gestion du parc assure la collecte et la gestion de toutes les données relatives à:

- l'inventaire pour connaître les quantités et caractéristiques des produits contenus dans chaque réservoir compte-tenu de sa géométrie (cylindrique vertical ou horizontal, sphérique), de la température, et des propriétés du produit;
- la supervision des réservoirs pour éviter l'assèchement ou le débordement (par détection de seuil mais aussi par anticipation en fonction des ordonnancements).

L'intérêt de cette application réside dans sa "globalité" et son étendue (budget global du système d'information de plus de 20 MF, 3500 heures de développement plannifiées):

- il ne s'agit pas d'un îlot automatisé mais d'une application complète,
- différentes fonctionnalités autres que la conduite sont nécessaires: maintenance, suivi de produit,
- la complexité de l'application donne sa signification à une méthode de conception,
- l'étendue géographique (4 kms$^2$) et la répartition du procédé de transport (synchronisation des sources, destinations, et équipements intermédiaires) implique des solutions distribuées.

L'application représente quelques 200 réservoirs, 300 vannes, 500 pompes, 800 transferts et mélanges différents.

## 3.2. Critères de structuration

Les critères de structuration mis en oeuvre par les concepteurs apparaissent au niveau du travail de structuration fonctionnelle et dans les documents de présentation de cette structure fonctionnelle. Insistons une nouvelle fois sur le fait que de tels documents sont présents essentiellement dans les études préalables, mais rarement dans les réponses à appel d'offre surtout basés sur la présentation de solutions. Cette structure fonctionnelle n'est pas seulement le résultat d'une analyse: la structure proposée est souvent révélatrice de **choix d'organisation** des composants fonctionnels les uns par rapport aux autres. Ceci sans qu'une solution opérationnelle à base de dispositifs de traitement ou de communication soit spécifiée. Ces choix d'organisation sont conformes aux "schémas intellectuels" d'architecture sur lesquels le concepteur élabore son raisonnement. La structuration fonctionnelle ne se contente pas de reformuler les besoins de l'utilisateur-client. La structuration fonctionnelle est au système d'information du SAP ce que le plan de l'architecte est au bâtiment: les matériaux ne sont pas encore choisis, mais la structure globale est détaillée.

### 3.2.1. Structuration basée le processus

La structuration du SI d'un SAP est largement basée sur la structure du processus de production. La connaissance de la structure de ce processus est donc un préalable indispensable. Différents niveaux de décomposition sont utilisés en fonction de :

- la complexité et l'étendue de l'application,
- le niveau de détail nécessité par l'étude.

#### 3.2.1.1. structuration en entités processus

La notion "**d'entité processus**" est basée sur un composant "principal" du processus effectuant une ou plusieurs opérations sur un flux de produit. Le terme entité processus est parfois remplacé par d'autres synonymes: "opérateur, cellule, unité, machine". Des exemples d'entités processus sont des réservoirs, des pompes, des vannes, ou une machine de découpe, d'usinage, etc... Une entité processus est une agrégation d'équipements: outils, actionneurs, capteurs. Cette notion est présentée par les concepteurs comme correspondant à un *"objet fonctionnel géré par le système d'information"*. Une entité processus peut être décrite de façon générique: on ne décrit pas un élément particulier (le réservoir n°70 456 situé à un endroit précis) mais l'objet abstrait ("réservoir") dont il s'agit de décrire les propriétés générales communes. Les principales variantes doivent être précisées. Cette approche "objet" permet de décrire les fonctions associées du SI de façon générique. Chaque entité processus correspond à un type d'opération particulier sur le produit: stockage, transport, mesure ou contrôle,

transformation. Cet aspect sera détaillé dans le chapitre sur la modélisation du processus de production. La **liste des entités** fournit donc une indication sur la nature et le niveau auquel a été effectuée la décomposition.

| | |
|---|---|
| contrôle | analyseur<br>ligne de mesure |
| manutention | ligne<br>pipe<br>pompe |
| stockage | réservoir |
| transformation | mélangeur |

*liste des entités processus utilisés pour la gestion des mouvements de produits*

Remarque: L'énumération des entités "réelles" (les instances des entités génériques) permettra d'évaluer quantitativement l'étendue et la complexité de l'application.

Pour chaque entité générique, seront décrits:

1 - le **rôle**: la description orientée procédé des fonctions de l'entité fonctionnelle;
2 - les **variantes**: la liste des principales variantes de cette même entité;
3 - les **équipements**: la liste des dispositifs physiques (type et nombre) qui constituent l'entité;
4 - les **informations**: la description des informations manipulées par le système d'information correspondant à cette entité;
5 - les **procédures**: la description des opérations effectuées par le SI de cette entité selon les différents modes d'exploitation prévus;
6 - les **interfaces homme-machine**: la liste des informations échangées sur les postes entités associées aux entités (visualisations, consignes);
7 - les **données échangées**: la liste des informations échangées avec les autres composants du SI ou avec d'autres systèmes externes.

Les points 4 et 5 concernent la description du système d'information selon deux points de vue différents et complémentaires. Le point 4 décrit le système d'information du **point de vue des données** (rôle passif par rapport au comportement du processus) et le point 5 décrit le système d'information du **point de vue des fonctions de la conduite du processus** (rôle actif par rapport au comportement du processus). Détaillons ces deux points qui conditionnent directement la définition du système d'information du SAP:

**les informations**

Différentes catégories d'informations sont traitées. Elles concernent :

- les **paramètres de référence** (constantes),
- les **informations collectées** (variables),
- les **informations calculées** (variables).

Ces informations sont elles-même relatives à trois domaines distincts :

- le **produit**,
- le **procédé** (le déroulement des opérations de production),
- le **processus** (les équipements physiques de production).

Ces informations peuvent donc être classifiées suivant une matrice:

| | produit | procédé | processus |
|---|---|---|---|
| paramètres de référence | | | |
| informations collectées | | | |
| informations calculées | | | |

A travers les différentes études examinées, nous avons pu remarquer que pour les premiers niveaux de la structure fonctionnelle, la structuration à partir de la décomposition du processus de production était prépondérante sur d'autres critères tels que la nature des fonctions. Ceci se traduit par la non spécification de la finalité de l'utilisation de chaque information. En d'autres termes si le détail de toutes les informations manipulées à ce niveau est généralement bien précisé, ces informations ne sont pas classées en fonction de l'usage fonctionnel qui en sera fait. La description est faite comme si ce niveau était considéré comme le producteur de toutes les données de base relatives au processus de production sans considération de l'utilisation de ces données par les fonctions du système. Nous verrons au paragraphe "structuration basée sur les fonctionnalités" que l'aspect fonctionnel devient en revanche prépondérant au niveau système.

**les modes et procédures d'exploitation**

La diversité des différents modes et procédures d'exploitation est une des particularités principales d'un SAP. Bien que très largement utilisée, la notion de mode d'exploitation est particulièrement mal définie: aucune définition générique satisfaisante n'est habituellement donnée. Elle diffère généralement pour chaque application en fonction:

- des conditions d'exploitation propres à chaque métier,
- des conventions de vocabulaire utilisées par le client ou le concepteur,
- du niveau hiérarchique considéré.

Deux modes d'exploitation principaux apparaissent cependant :

- **mode automatique**: ce mode est caractérisé par le fait que les ordres sont fournis par le système d'information, c'est-à-dire par des composants fonctionnels de niveau supérieur. L'action de l'opérateur est limitée au démarrage et à l'arrêt des installations, des validations locales et la supervision locale de l'exécution des séquences.
- **mode manuel** (ou local): dans ce mode l'opérateur peut donner des commandes directes en séquences.

Des commandes de changement de mode d'exploitation permettent de sortir du mode automatique pour reprendre la main en local en cas de défaut.

Remarque: Entre ces deux modes de base sont parfois imaginées de nombreuses variantes. La tendance actuelle est cependant à la **limitation du nombre de ces modes** afin:

- de limiter la complexité liée à la gestion de ces modes d'exploitation,
- de ne pas inciter à se contenter de l'utilisation de modes dégradés au lieu de résoudre les éventuels problèmes survenant en mode automatique.

exemple:

description de l'entité **réservoir**

(cette description a été simplifiée et les données banalisées)

- **rôle**:

Le rôle de l'entité réservoir est de collecter et de traiter toutes les données concernant l'état et les opérations du réservoir:

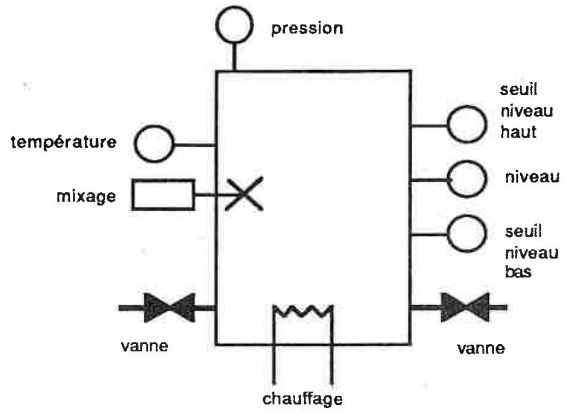
- vérifier la disponibilité du réservoir,
- effectuer les calculs de volume et de poids,
- exécuter les procédures spécifiques de chaque réservoir.

- les **variantes**:

.réservoir à plafond fixe,
.réservoir à plafond flottant interne ou externe,
.réservoir sphérique,
.réservoir cylindrique horizontal.

- les **équipements**:

.mélangeur,
.capteur de pression relative (pour les réservoirs de gaz liquéfié),
.capteur de température,
.vanne motorisée,
.vannes manuelles,
.jauge de niveau,
.détection de niveau haut,
.chauffage (pour les produits lourds).




*les équipements d'un réservoir*

- les informations :

.informations relatives au réservoir:

paramètres de référence :

numéro du réservoir,
nom du réservoir,
seuils de niveau (alarmes matérielles et logicielles),
caractéristiques du réservoir (volume nominal, hauteur, équations de calibration pour le calcul du volume, température maximum,...),

informations collectées:

informations processus:

état mixeur (en service/hors service),
état système de chauffage,

état du réservoir:

disponible,
en cours de remplissage,
en cours de vidage,
remplissage/vidage simultané,
en défaut (équipements ou instruments défaillants),
hors service (maintenance).

informations relatives au produit:

code et nom produit,
température moyenne,
densité à 15 degrés (laboratoire),
résultat d'analyse.

.informations calculées:

poids, volume du produit,
densité et qualité du produit,
débit instantané en cours de transfert,
durée de transfert,
taux de disponibilité.

| | produit | procédé | processus |
|---|---|---|---|
| **paramètres de référence** | référence | seuils,paliers | numéro réservoir |
| **informations collectées** | température | niveau | état |
| **informations calculées** | poids produit | débit instantané | taux de disponibilité |

- les **procédures**:

procédure de vérification du réservoir:

.état: .....procédure assurée en permanence,

.disponibilité:.procédure assurée quand un réservoir est
sélectionné et pendant la durée d'un transfert,
procédure de mixage:...la commande (démarrage ou arrêt) des mixeurs
est assurée depuis la salle de contrôle ou localement,
procédure de chauffage: le système doit contrôler les vannes de chauffage
selon des paliers prédéterminés. Le système assure
une balance thermique simplifiée pour chaque réservoir
et calcule la vitesse de variation de la température.

- les **interfaces homme-machine**:
  affichages: pour chaque réservoir,
  pour tout le site.
  rapports: les rapports sont générés sur demande de l'opérateur ou périodiquement.
  datation de tous les événements.

- les **données échangées**:
  données en provenance de la base de donnée process:
  débits produits,
  résultats d'analyse de qualité de produit.
  données envoyées vers la base de données process:
  niveau du réservoir,
  allocation du réservoir,
  température, densité et qualité du produit.

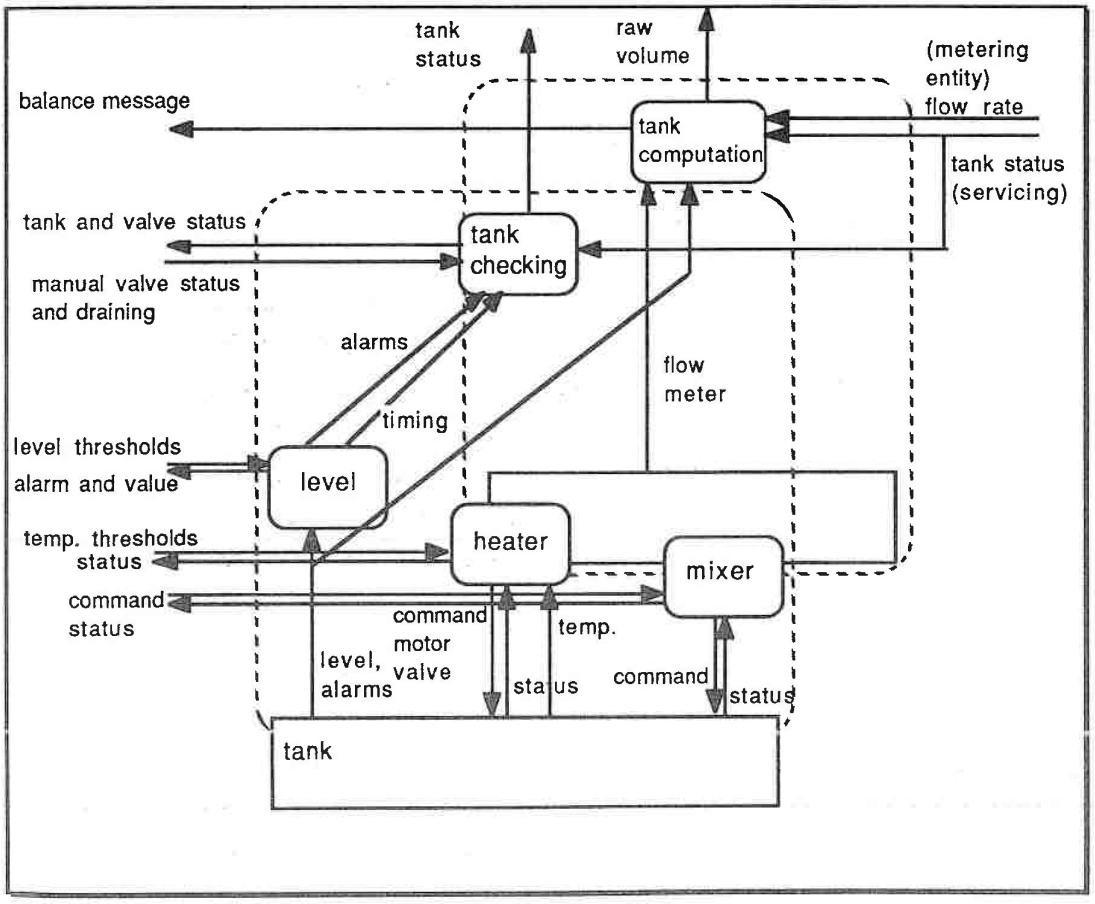



*schéma fonctionnel de l'entité reservoir*

### 3.2.1.2. structuration en équipements

A un niveau de décomposition plus fin, il nécessaire de considérer une entité comme une agrégation d'équipements réalisant chacun une opération élémentaire.

Par exemple si le réservoir effectue globalement une opération de stockage des produits, il se compose :

- du réservoir proprement dit avec ses différents capteurs de niveau, de pression et débit-mètres,
- d'un équipement de chauffage avec ses capteurs de température et ses vannes motorisées,
- d'un mélangeur (mixeur) pour maintenir une densité homogène du produit.

Ces différents équipements doivent être conduits et suivis pour garantir le comportement global de l'entité. Cela définit donc des fonctions particulières du système d'information habituellement situées au niveau des automatismes réflexes, séquentiels ou régulation.

Pour chaque équipement est donc défini de façon analogue aux entités:

1 - le **rôle**: la description orientée procédé des fonctions de l'équipement ;
2 - les **variantes**: la liste des principales variantes de ce même équipement;
3 - l' **instrumentation**: la liste des dispositifs physiques capteurs et actionneurs (type et nombre) qui constituent l'équipement ;
4 - les **informations**: la description des informations manipulées par le système d'information correspondant à cet équipement ;
5 - les **procédures**: la description des opérations effectuées par le SI de cet équipement selon les différents modes d'exploitation prévus;
6 - les **interfaces homme-machine**: la liste des informations échangées sur les postes opérateurs associés aux équipements (visualisations, consignes);
7 - les **données échangées**: la liste des informations échangées avec les autres composants du SI ou avec d'autres systèmes externes.

### 3.2.1.3. structuration en domaines

La notion de "domaine" constitue un autre critère de structuration selon un point de vue différent. Cette notion est associée à un produit ou une famille de produit et à un type d'opération globale particulière à ce(s) produit(s). Dans le cas où le processus de production est constitué de plusieurs flux de produits, il apparaît en effet pertinent de subdiviser ce processus selon la nature des flux. Le transfert et le mélange des produits pétroliers vers les réservoirs dédiés aux différents types de produits constituent ainsi deux domaines principaux. Le traitement des eaux usées, la gestion des flux de vapeur, d'air comprimé ou d'énergie sont aussi des exemples de la notion de domaine (voir aussi [Ver

89]). Le nombre de domaines est très particulier au procédé considéré. Comme pour la notion d'entité processus, on trouve différentes appellations pour désigner ce type de structuration: "ligne, unité, sous-système, etc..."

On peut rapprocher cette notion de domaine avec celle utilisée dans la méthode MERISE:

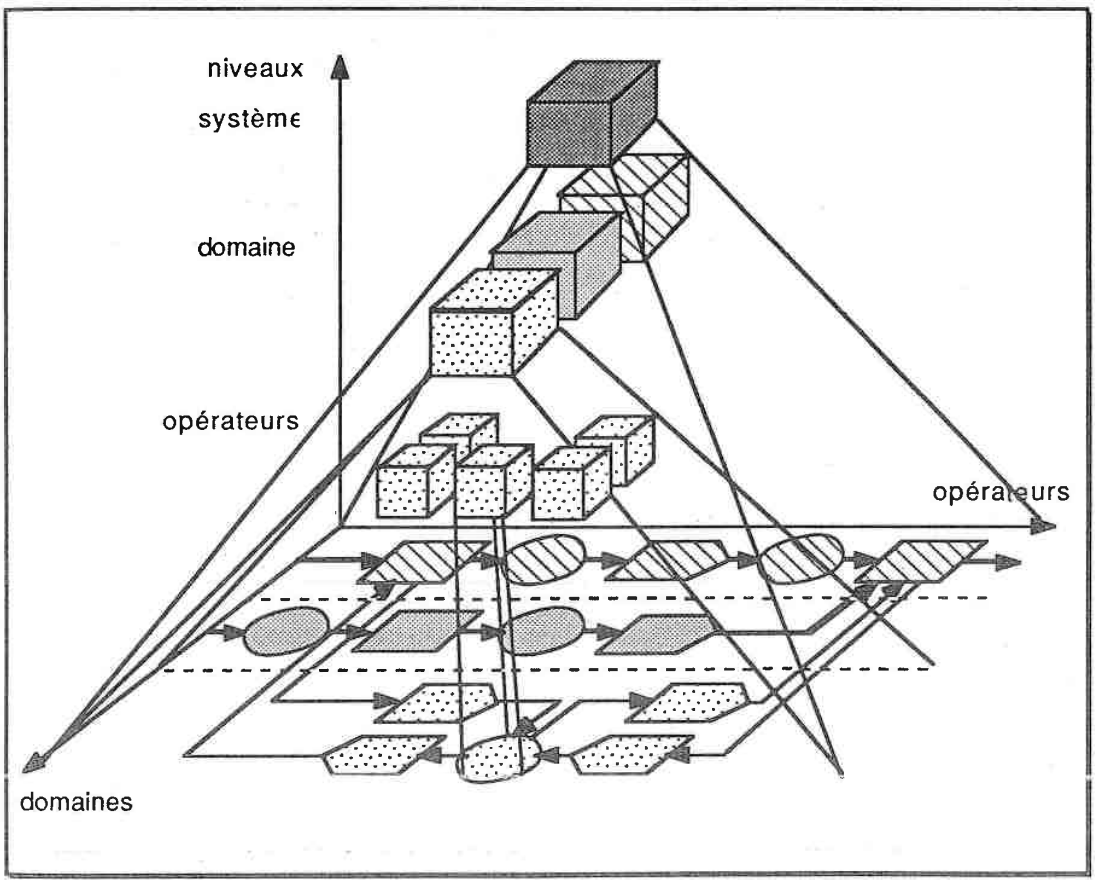
*"grande fonction de l'entreprise:*

*- relevant d'un centre de décision unique ou homogène,*

*- composé de grands modules d'activité fortement liés, qui constituent un ensemble quasi-autonome,*

*- gérant un ensemble de données spécifiques (création et mise à jour)."*

On peut donc dire que notre acceptation du domaine est au SAP ce qu'elle est aux procédures de gestion dans Merise.




*les différents niveaux de structuration du SAP*

Pour chaque domaine, on précisera:

1 - le **rôle**: la description orientée procédé du domaine;

2 - les **entités** : la liste des entités (type et nombre) qui constituent le domaine ;
3 - les **informations**: la description des informations manipulées par le système d'information correspondant à ce domaine;
4 - les **procédures**: la description des opérations effectuées par le SI de ce domaine selon les différents modes d'exploitation prévus;
5 - les **interfaces homme-machine**: la liste des informations échangées sur les postes entités associés aux domaines (visualisations, consignes);
6 - les **données échangées**: la liste des informations échangées avec les autres composants du SI ou avec d'autres systèmes externes.

### 3.2.2. Structuration basée sur les produits

Chaque domaine peut également être subdivisé en fonction de la classification des produits et sous-produits manipulés. Cette notion n'est pas toujours présente mais intervient généralement dans les systèmes complexes de transport. Ainsi chaque transfert élémentaire autorisé est référencé (produit, origine, destination). Un premier regroupement de ces transferts élémentaires en "sous-systèmes" est basé sur une même origine ou une même destination. Un deuxième regroupement est basé sur la nature des produits (une même famille) constituant un "système". Pour fixer les idées, notre application comprend:

- 800 cas de transfert et mélanges,
- 16 systèmes correspondant à autant de sous-produits de base différents.

La **liste des systèmes** fournit donc une indication sur la nature et le niveau auquel a été effectuée la décomposition.

A la différence des domaines, les procédés mis en oeuvre sont de même type. Le système d'information supportant la conduite et les autres fonctions est alors identique et souvent commun. Ce regroupement est du même type que les classifications mises en oeuvre dans la Technologie de Groupe afin de faciliter la gestion des gammes de fabrication. L'objectif est plutôt d'organiser clairement la production pour en faciliter la gestion. Les différents systèmes et sous-systèmes se traduisent essentiellement en terme de base de données référençant les différents transferts ou gammes.

exemple de systèmes associés à différents types de produits:
transfert brut,
transfert naphta,
transfert kérosène,
mélange gasoline,
transfert eaux usées.

Pour chaque système, seront décrites:

- le **produit**; le nom et la référence du produit;
- les **opérations**: la description littérale des opérations associées à chaque sous-système;
- les **entités**: la liste des entités associées à chaque transfert élémentaire.

exemple:

description du domaine "réception brut" (cette description a été simplifiée)

- le **produit**; "brut"

- les **opérations**: la réception du brut depuis les centres de réception dans les différents réservoirs est assurée selon des séquences batch (lot) dépendant du volume de chaque lot donné par la gestion du pipe X;

sous-système 1:

ligne de réception depuis le site n°1 vers les réservoirs de brut

| n° cas | sous-produit | origine | destination |
|---|---|---|---|
| 4 | B443 | site n°1 | réservoir B46 |
| 5 | B457 | site n°1 | réservoir B37 |
| ... | ... | | |

### 3.2.3. Décomposition hiérarchisée

A partir des niveaux de détail de description des composants définis dans les paragraphes précédents, il est possible de définir des points de repères simples pour faciliter la décomposition. D'un point de vue opérationnel, la décomposition en niveaux hiérarchisés effectuée par les concepteurs est basée sur des critères correspondant à des niveaux de détail de la décomposition du processus:

| **niveau courantes** | **contexte** | **principaux repères** | **appellations** |
|---|---|---|---|
| **système** | processus complet | programme de fabrication des produits finis pour une date donnée | gestion de production, |
| **domaine** | flux principal d'une famille de produits | ordonnancement et affectation des principales opérations | ordonnancement |
| **cellule/ligne** | fabrication d'un produit intermédiaire | produits connus par leur nom et leur position par rapport aux différentes entités, valeur ajoutée perceptible | lancement, supervision coordination |
| **entité** | opérations physiques de base sur un produit, | vue logique des produits et outils, et des caractéristiques modifiées, | coordination, automatismes |
| **équipement** | réalisation des opérations, | connaissance des variables physiques modifiées (coordonnées de position et dimensions géométrique des produits et des outils ou des caractéristiques principales des produits (température, pression,...), | automatismes séquentiels, régulations, |
| **variables** | capteur/ actionneur | correspond à l'instrumentation (transformation état physique-élémentaires.) | asservissements |

Ces critères vont permettre dans la tâche de structuration fonctionnelle, de situer les différentes fonctions en niveaux dans la structure globale du système.

### 3.2.4. Structuration basée sur les fonctionnalités

La nature des fonctionnalités supportées par le système d'information est également un critère important de structuration. Malgré la diversité des appellations utilisées, quatre fonctionnalités principales reviennent de façon invariante. Nous avons déjà vu au chapitre sur les concepts issus des modèle d'architecture de SAP qu'il était difficile d'établir une liste définitive des fonctions d'un SAP. Sans prétendre établir un modèle de référence, nous présentons ici quelques fonctions dites de "base" étant donnée leur généralité pour bien mettre en évidence la nature non-hiérarchique des relations entre ces fonctions. Une telle classification est donc donnée à titre indicatif. En fonction des besoins exprimés par le client, d'autres fonctions n'entrant pas dans ce cadre ou des sous-fonctions devront être explicitées afin d'être étudiées en détail.

#### fonction conduite

La fonctionnalité de base demandée (quand elle n'est pas déjà implantée) est généralement ce que l'on peut regrouper sous le terme "**conduite**" de processus.

Cette fonctionnalité permet de déterminer le comportement futur et immédiat du processus de production selon un procédé bien déterminé pour atteindre les objectifs de production en termes de productivité et de qualité. Les données concernant le procédé de fabrication (gammes d'usinage, programmes pièces pour MOCN par exemple) peuvent, dans certains métiers, provenir d'une fonction appelée "méthodes" (utilisant des techniques de Fabrication Assistée par Ordinateur) externe au SAP dans la mesure où elle est située hors contrainte temps-réel. Suivant les différents niveaux considérés, la conduite recouvre les automatismes, asservissements et régulation, la coordination inter-entités ou inter-domaine, l'ordonnancement temps-réel (tenant compte de l'état des équipements de production). Cette définition englobe la conduite automatique, mais aussi le pilotage par des opérateurs de conduite sur des postes centralisés ou locaux.

**conduite**: fonction de base du système d'information d'un SAP destinée à contrôler le comportement futur et immédiat du processus de production selon un procédé donné pour atteindre les objectifs de production exprimés en termes de qualité et de productivité.

Selon cette définition le contrôle-qualité, du moins son sous-ensemble "en-ligne" fait partie prenante de la conduite (donc permettant d'identifier, de corriger ou d'éliminer les défauts). En revanche le contrôle statistique de la qualité des produits en fin de ligne de fabrication ne rentre pas dans le cadre de cette définition. L'évolution de l'organisation des SAP tend d'ailleurs à rapprocher et à intégrer le contrôle qualité à la conduite afin de permettre une réaction plus rapide en cas d'incident. Il peut cependant être intéressant de

définir la fonction contrôle qualité de façon distincte afin d'en faciliter l'analyse et de mieux contrôler la cohérence globale de cette fonction. Cette approche n'est pas contradictoire avec une réalisation complètement intégrée. En premier lieu, il convient de préciser l'organisation globale du contrôle qualité:

- centre de contrôle en fin de ligne ou intégré à différents stades de la production;
- interaction directe ou non avec l'exploitation de la production.

La gestion des ressources nécessaires à la production est aussi partie intégrante de la conduite: gestion des outils, palettes, etc... pouvant être considérés comme des flux de produits internes.

### fonction suivi

Une deuxième fonctionnalité dont l'importance est aujourd'hui croissante est le "**suivi**" de production. Le suivi concerne la collecte et la synthèse des informations en provenance du processus relatives au comportement présent et passé du procédé ainsi qu'à l'état et la position des produits. Ces informations sont destinés aux opérateurs d'exploitation et aux fonctions externes de gestion du SAP, comme le calcul des coûts de production, la traçabilité de la production, le suivi des encours, etc... Contrairement aux informations collectées en retour dans la fonction conduite, les informations de suivi ne sont destinées:

- ni à vérifier l'adéquation entre le comportement du processus de production et les requêtes de conduite,
- ni à déterminer le comportement futur du processus de production selon des procédures déterministes.

**suivi**: fonction de base du système d'information d'un SAP destinée à recueillir et à synthétiser les informations relatives à l'état et au comportement présent et passé du procédé ainsi qu'à l'état et la position des produits.

### fonction maintenance

Une troisième fonctionnalité appelée "**maintenance**" permet de garantir la disponibilité du processus de production. Cette fonction est plus ou moins développée, et plus ou moins intégrée aux deux précédentes. Elle peut comprendre:

- la planification des interventions (maintenance préventive et entretien périodique), en se basant ou non sur les durées d'utilisation des équipements,
- assurer la classique maintenance curative en cas de défaillance avec dans certains cas l'aspect diagnostic,

- et parfois de la maintenance prédictive basée sur le suivi des dérives, vibrations et autres signes de dysfonctionnement,
- la gestion des ressources humaines (équipes de maintenance) et pièces détachées.

**maintenance**: fonction de base du système d'information d'un SAP destinée à assurer la disponibilité du processus de production.

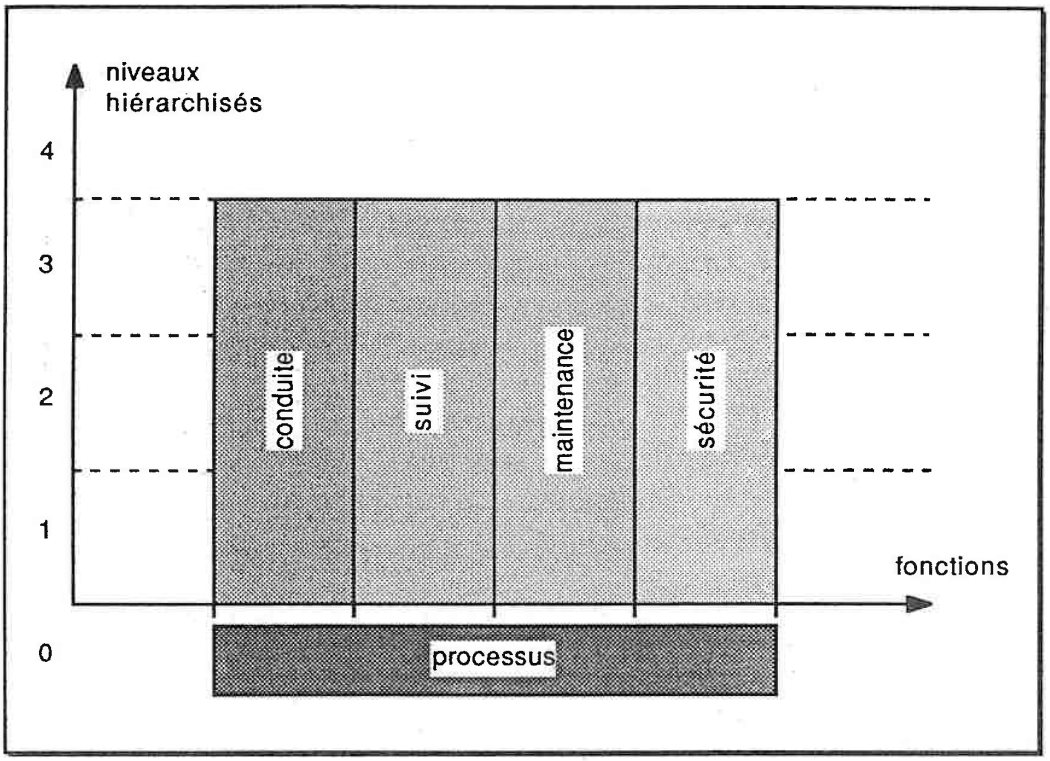
### fonction sécurité

Une dernière fonction d'un SAP est la fonction "**sécurité**". Cette fonction n'est pas explicitement présente dans tous les processus de production, mais parfois disséminée au sein des différents automatismes. Elle est par nature associée aux processus dangereux pour les personnes, les installations ou l'environnement. La sécurité est essentiellement basée sur la supervision d'alarmes spécifiques à la sécurité ou communes à la conduite, et le déclenchement de procédures automatiques ou manuelles de sécurité.

**sécurité**: fonction de base du système d'information d'un SAP destinée à assurer la non-occurence de **défaillances catastrophiques** qui pourraient mettre en danger les installations du système automatisé de production, les produits, le personnel d'exploitation ou l'environnement. .

Ces fonctions de base exploitent et élaborent des données relatives au processus de production, partagent des données mais ne sont pas en relation **hiérarchisées deux à deux**. Les relations de partage de données entre ces fonctions de base non-hiérarchisées entre elles définissent une dimension orthogonale à la dimension hiérarchique "verticale" traditionnellement représentée dans les modèles d'architecture de SAP.

Cette structuration "horizontale" apparaît aujourd'hui dans les solutions d'architectures opérationnelles grâce à l'apport des Réseaux Locaux et des Bases de Données Réparties permettant à des postes distincts d'exploiter en vue de différentes fonctionnalités les mêmes données de base issues du processus. Si aujourd'hui une telle structuration opérationnelle n'est possible qu'aux niveaux supérieurs, l'avénement des réseaux de terrain "fieldbus" offrant des fonctionnalités de BdeD répartie et temps réel, devrait permettre d'étendre l'adéquation de l'architecture opérationnelle à la structure fonctionnelle au niveau des automatismes réflexes.




*la dimension des fonctionnalités par rapport à la dimension de la hiérarchie*

Toutes ces fonctions de base peuvent cependant être elles-même structurées de façon hiérarchique: elles comportent chacune différents niveaux hiérarchisés en fonction du sous-ensemble du processus considéré et des horizons temporels perçus.

Ces fonctionnalités sont surtout employées en tant que critère de structuration aux niveaux supérieurs. Au niveau des automatismes associés au processus, ces fonctions de base sont beaucoup moins explicitées, les automatismes étant généralement vus comme les producteurs/consommateurs des données processus, indépendamment de la fonction de ces données.

## Conclusion

Cet examen des critères de structuration employés par les concepteurs d'architecture souligne l'importance accordée:

- à la structure du processus de production (entités de production et produits),
- à la structuration hiérarchisée en niveaux,
- à la dimension "multi-fonctionnelle".

Ces éléments sont particulièrement peu ou pas explicités dans les méthodes traditionnelles de conception des systèmes d'information. Ils sont en revanche présents dans les différents modèles d'architectures mais leur mise en oeuvre dans une perspective de conception n'est alors généralement pas précisée.

Notre contribution propose de baser la démarche méthodologique de conception d'architecture de SAP sur ces concepts tant au niveau des modèles utilisés que des procédures d'élaboration et de transformation de ces modèles.

# chapitre 6:

# FORMALISATION DES CONCEPTS

## INTRODUCTION

L'étude des connaissances en conception d'architectures du chapitre 5 montre que l'on peut enrichir les concepts des méthodes traditionnelles de conception de systèmes informatiques par des concepts spécifiques aux architectures de SAP et aux représentations utilisées par les ingénieristes, ou réciproquement, qu'il est possible de compléter les modèles d'architecture par des concepts et procédures méthodiques de conception.

Ce chapitre présente notre contribution à la formalisation de ces connaissances sous la forme d'un modèle générique de structuration fonctionnelle des architectures de SAP. Ce modèle, basé sur la notion de service, a pour but essentiel de faciliter l'analyse et la description des interactions entre les différents composants fonctionnels du SAP.

## 1. CONCEPTS DE HIERARCHIE ET DE COMMUNICATION

Deux grands types d'interactions ont été identifiés entre les éléments fonctionnels d'une architecture: les relations hiérarchiques et les échanges d'informations.

Le mot hiérarchie a été employé à plusieurs reprises dans ce document pour qualifier des relations structurelles tant dans les modèles méthodologiques d'analyse et de conception informatique que dans les modèles d'architecture informatisée de SAP. Ce concept essentiel doit être introduit explicitement dans le modèle de structuration fonctionnelle. Le premier paragraphe a pour but d'analyser les différents aspects que recouvre ce concept en se basant sur différentes études fondamentales portant sur ce sujet, et de le formaliser pour l'employer à bon escient.

Dans l'étude d'une architecture, la place des communications joue un rôle essentiel dans la structuration de l'application. Les différents types d'échanges d'informations doivent donc également être modélisés. Ceci fait l'objet du second paragraphe.

### 1.1. Différents concepts de hiérarchie

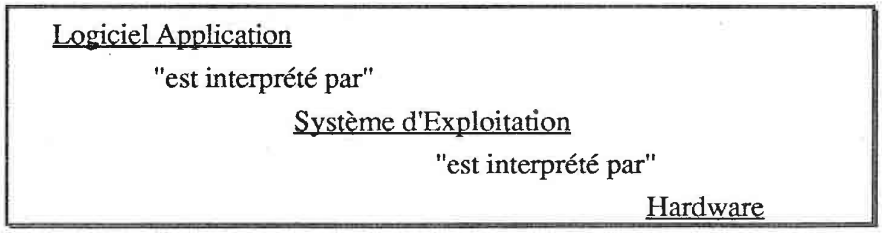
Le concept de hiérarchie, plus précisément de structure hiérarchisée multi-niveaux, a été étudié formellement par [Mes 70]. Les auteurs soulignent à propos de la définition de ce concept qu'il ne peut pas être défini de façon concise et générale sans tomber dans une énumération de toutes les alternatives possibles. Ils proposent de répondre à la question

en soulignant les différents points de vue de la notion de hiérarchie. Mesarovic détaille 3 notions de hiérarchie distinctes mais interdépendantes.

### 1.1.1. Hiérarchie de description ou de décomposition

La notion de "**strata**" est relative au **processus de description hiérarchisée** des systèmes complexes selon différents **niveaux de détail ou d'abstraction**: "*les systèmes complexes échappent par définition à une description complète et détaillée. Le dilemme réside entre la simplicité de la description nécessaire à la compréhension, et le besoin de prendre en compte les divers aspects du comportement d'un système complexe.*" La description hiérarchisée permet de préciser à chaque niveau les différentes caractéristiques ou variables pertinentes quant à la structure et le comportement des composants du système. On peut comme [Lap 88], définir plus précisément la nature de cette relation de nature ensembliste: "**est composée de / compose**". Cette relation est à la source de la plupart des méthodes d'analyse et de conception informatique déjà citée précédemment et elle justifie largement l'intérêt que portent les analystes-concepteurs à l'emploi des notations DFD ou SADT. Soulignons que cette relation permet de désigner les relations d'appartenance entre les composants d'un système mais pas leurs interactions directes, c'est-à-dire les échanges d'information.

Une autre relation, assimilée par Mesarovic au concept de strata, est explicitée par Randell [Ran 80] et Laprie à travers la relation "**est interprétée par**" qui permet de décrire les relations entre les composants opérationnels (donc à un même stade de développement final) d'un système:




*la relation hiérarchisée "d'interprétation" de Randell*

Cette relation peut être rapprochée de la liaison "mécanisme" de SADT.

### 1.1.2. Hiérarchie de décision

La 2 ème notion détaillée par Mesarovic est celle de "**layer**" dans le contexte de processus complexe de **prise de décision**. *"La structuration en niveaux de décision permet de prendre en compte le besoin d'agir sans délai et celui de devoir mieux comprendre la situation, grâce à une décomposition de la prise de décision en décisions*

*plus simples"*. Ce point de vue correspond donc à une division fonctionnelle "verticale" du système [Kal 85] . Cette notion est à la base de la méthode GRAI où les aspects **temporels d'horizon et de période** sont utilisés en tant que critères de décomposition des niveaux de décision [Dou 85]. La relation associée à cette notion est basée sur la définition par un niveau de décision supérieur des **objectifs** ou "decision frame" du niveau de décision inférieur qui lui-même doit prendre des décisions respectant ce cadre.

### 1.1.3. Hiérarchie de coordination

La 3 ème notion de hiérarchie, celle "d'**echelon**", est relative à l'aspect organisation de la structure au sens **coordination** des sous-systèmes. Si le point de vue décision précédemment décrit (layer) met surtout en avant la dimension "verticale", ce concept d'echelon s'intéresse plus à la dimension "horizontale" d'un système dans la mesure où un élément de niveau supérieur ne s'intéresse pas seulement globalement au niveau inférieur mais au contraire distingue et coordonne les différents éléments de ce niveau dont il a la responsabilité d'assurer un comportement global cohérent. Cette relation hiérarchisée est formalisée par une liaison bidirectionnelle "contrôle / feed-back" selon la classique boucle de contrôle chère aux automaticiens. De nombreux travaux d'automatique théorique [Tit 75], [Bin 84] ont étudié les problèmes de coordination dans de telles structures hiérarchisées où un bloc de contrôle doit se baser sur les interactions ou couplages horizontaux des composants de niveau inférieur. On retrouve cette notion de coordination dans la plupart des modèles d'architecture évoqués dans le chapitre précédent.

### 1.1.4. Définition

Mesarovic choisir de définir la hiérarchie indirectement par les caractéristiques qu'elle présente:

"*arrangement vertical de sous-systèmes constituant le système global, caractérisé par la priorité d'action ou le droit d'intervention des sous-systèmes de haut niveau et la dépendance des performances des niveaux supérieurs vis à vis des niveaux inférieurs.*"

Nous emploierons dans ce document le terme hiérarchie sous le sens suivant:

| |
|---|
| **hiérarchie:** relation structurelle orientée entre les composants de l'application. |
| Cette relation définit un graphe orienté sans boucle dont les sommets sont les composants du système et les arcs sont la représentation de ces relation entre les composants. |

Cette définition permet d'établir une relation d'ordre partiel entre les composants d'un système. Soient deux composants A et B d'un système S: un composant A est dit d'ordre supérieur par rapport à un autre composant B pour une relation hiérarchique donnée si il existe une relation hiérarchique de A vers B. Réciproquement B sera dit d'ordre inférieur. Cette relation "est d'ordre supérieur à" est non-réflexive, non-symétrique et transitive:

Soient A, B et C trois composants d'un système S,
si A "est d'ordre supérieur à" C
et C "est d'ordre supérieur à" B
alors A "est d'ordre supérieur à" B.

Cela signifie que cette appellation sera étendue si il existe un ensemble E fini de composants "intermédiaires" ordonnés selon cette relation permettant d'établir par transitivité que A d'ordre supérieur par rapport à B.

#### 1.1.5. hiérarchisation et centralisation

La confusion entre hiérarchisation et centralisation est fréquente. La troisième notion de hiérarchie présentée dans le paragraphe précédant indique que la coordination entre deux composants fonctionnels suppose un niveau **fonctionnel** supérieur. La hiérarchie de coordination ne fait qu'expliciter la nécessité de l'existence d'une forme abstraite de coordination entre plusieurs composants. En d'autres termes, c'est une vue de l'esprit signifiant qu'il y a toujours une forme de coordination. Cela ne signifie pas que cette coordination doive être implantée dans un dispositif centralisé. "Centralisation" est à opposer à "répartition" ou "distribution" et non pas à "hiérarchisation". En d'autres termes, il est possible d'implanter la coordination de façon répartie selon un mécanisme de coopération. Cette différence est peut-être plus facile à saisir si on considère l'aspect abstraction de la hiérarchie. Par exemple des données peuvent être hiérarchisées, c'est-à-dire représenter des niveaux d'abstraction différents et en même temps être soit centralisées, soit réparties.

### 1.2. Communications

#### 1.2.1. Echanges d'informations: niveau application et niveau communication

On peut distinguer deux niveaux de spécification des interactions entre les entités fonctionnelles d'une application.

- Un premier niveau fonctionnel correspond à la description des relations hiérarchiques entre les fonctions de l'application: relation "utilise le service de".

- Un deuxième niveau correspond à la description détaillée des échanges d'informations.

Le niveau de spécification des échanges d'informations peut être considéré comme un niveau de spécification plus détaillé **explicitant** la nature des relations hiérarchiques précédemment décrites.

Nous différencions liaison et relation: la liaison suppose un échange (un flux), pas la relation.

**liaison**: échange de message entre les composants d'un système.

**relation**: association entre les composants d'un système

La définition d'une liaison suppose la définition du message échangé et des composants concernés par l'échange en tant qu'émetteur ou en tant que récepteur.

**message**: ensemble d'informations correspondant à un échange indivisible entre les composants d'un système.

La relation de décomposition hiérarchisée n'est donc pas une liaison selon cette définition puisqu'elle ne correspond à aucun flux. En revanche la relation hiérarchique "echelon" ou "utilise le service de" est une relation abstraite qui devient dans un modèle fonctionnel représentant les flux d'informations, une liaison bidirectionnelle que nous appelerons une "**liaison de service**" et supportant les messages de "contrôle / feed-back".

En plus de ces relations hiérarchiques, il est nécessaire d'adjoindre des **liaisons d'échanges de données** et des **liaisons de signaux de synchronisation** qui traduisent des relations d'interaction fonctionnelle différentes de la relation de service.

A ce niveau de spécification, on ne considère les échanges d'informations qu'au **niveau application**, c'est-à-dire du point de vue des fonctions applications. L'aspect communication au sens protocoles de communication n'est donc pas détaillé. Concrètement cela signifie par exemple que l'on ne doit spécifier que les compte-rendus de niveau application et non pas les compte-rendus d'acquittement liés aux protocoles de communication. Ces messages seront d'ailleurs mis en œuvre par les fonctions techniques de communication et non pas par les fonctions de service de l'application. Or les fonctions

techniques ne seront spécifiées que plus tard dans la phase de conception et non pas dans l'étude fonctionnelle de l'application.

### 1.2.2. Différents types de communication

Différents flux d'informations possédant, du point de vue des systèmes de communication, des caractéristiques différentes doivent être représentés :

- les liaisons verticales hiérarchiques bidirectionnelles à base de messages de requête et de compte-rendu,
- les liaisons horizontales d'échange de données entre entités de même niveau ou avec des services extérieurs,
- les liaisons supportant des signaux de déclenchement-synchronisation.

Nous avons vu qu'il est important de pouvoir représenter les relations entre le personnel utilisateur (les opérateurs) et le système. Ces échanges d'informations ne sont en réalité pas spécifiques. A un certain niveau d'abstraction, un opérateur peut être considéré comme une entité de traitement de l'information selon des procédures déterministes ou non de prise de décision. Dans cette perspective, il suffit de classer les échanges d'informations entre le système et les opérateurs dans l'une des 3 catégories précédemment citées.

#### 1.2.2.1. Liaisons de service

La liaison de service constitue le support de la structuration fonctionnelle hiérarchisée. Cette liaison bidirectionnelle se compose d'une requête (RQ) et d'un compte-rendu (CR) comprenant les différents paramètres qui définissent la nature de la relation de service. Elle permet de réaliser la relation "utilise" entre deux entités fonctionnelles respectivement utilsateur et fournisseur d'un service. Ces liaisons sont comme nous l'avons vu bien identifiées dans la plupart des modèles d'architecture de SAP. Elles n'apparaissent cependant pas dans les modèles issus de méthodes de conception de systèmes d'information qui ne représentent que des échanges d'informations unidirectionnels n'explicitant pas cette notion "d'utilisation".

Le service lui-même doit également être défini en terme de fonctionnalité par rapport à son contexte, c'est-à-dire un sous-système du processus physique de production. (Par exemple découper une pièce particulière, suivre la fabrication). Cette définition du service est une définition du "quoi" par rapport au processus de production. La définition des paramètres RQ et CR précise "comment", c'est-à-dire avec quelles informations sera réalisée cette fonction de service. (Par exemple les paramètres de dimension de la pièce à découper (RQ) et les paramètres de bonne ou mauvaise exécution de la découpe (CR)).

Le nombre et la diversité des paramètres fournis en requête est un indicateur de la **flexibilité** de l'entité fonctionnelle et du sous-ensemble du processus de production associé. Dans le cas où un opérateur effectue toujours la même opération de transformation sur le même type de produit, la RQ peut être implicite. Seul un signal d'activation-synchronisation doit être donné afin de déclencher l'opération au moment voulu. En revanche une machine de peinture par exemple, capable de peindre en différentes couleurs différents modèles de voiture, devra recevoir l'indication du type de modèle pour récupérer les données de trajectoire du bras de robot et l'indication de la référence de la couleur de la peinture.

#### 1.2.2.2. Liaisons de partage de données

Le partage de données se distingue de la relation de service par son aspect unidirectionnel. Ceci est dû au fait que le producteur de la donnée n'est pas concerné par l'utilisation qui est faite de cette donnée par le ou les consommateurs. Le producteur ne doit donc pas recevoir de compte-rendu de niveau application (à ne pas confondre avec un acquittement de niveau communication). Ces relations correspondent à des liaisons intra-niveaux entre des entités adjacentes ou entre des entités du système et des entités externes au système.

exemple: *le transfert de programme pièce entre un service de FAO et une machine outil à commande numérique correspond typiquement à ce type de relation: la FAO ne connait pas quels sont les objectifs de production, combien de produits seront fabriqués grâce au programme.*

Ce type de relation permet de supporter :

- les relations de coopération [Kal 85] entre centres de décision d'un même niveau,
- les relations client-serveur de lecture-écriture de données entre une fonction de traitement et une fonction de gestion de données.

#### 1.2.2.3. Liaisons de synchronisation

Les relations précédentes décrivent la structure de traitement fonctionnel de l'application. Un niveau de spécification plus avancé passe par la description de la structure de contrôle. Un autre type de relation est nécessaire pour représenter cette structure dans la mesure où le contrôle n'est pas forcément associé aux relations de service de définition des objectifs. En effet une entité fonctionnelle $EF_{i,j}$ de niveau j peut recevoir une requête d'une autre entité fonctionnelle $EF_{k,j+1}$ et être déclenchée:

- soit par cette même entité fonctionnelle $EF_{k,j+1}$ mais à un instant différent et indépendant de la transmission de la requête,

- soit par un signal-événement issu du niveau inférieur,
- soit par un signal envoyé par un opérateur,
- soit se déclencher elle même suite à un événement interne (horloge interne par exemple).

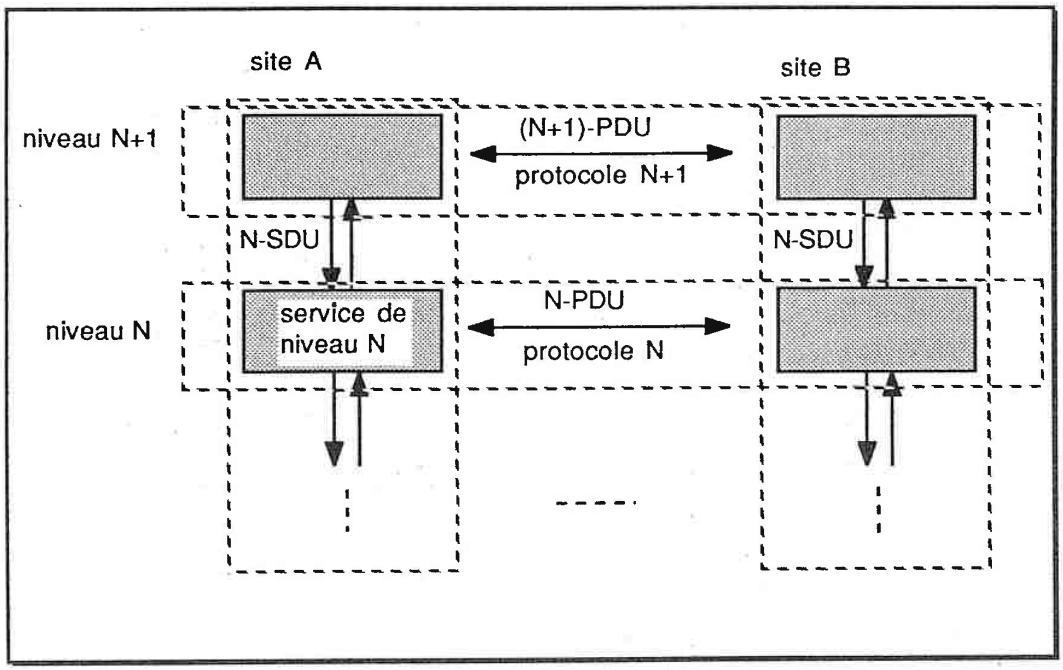
exemple:

*Dans une application de mouvement de produits en raffinerie, la phase de configuration du trajet pour un transfert donné (choix des tanks, pipes, pompes, etc...) peut être assimilée à la définition des paramètres de service pour chaque entité, et le contrôle du transfert peut être assimilé au déclenchement des opérations.*

## 1.3. Formalisation par le concept de service

### 1.3.1. Définition de la notion de service

La notion de service a été popularisée par son utilisation dans la constitution du modèle de référence OSI (Open Systems Interconnection) de l'ISO (International Standards Organisation). [référence IS 7498, AFNOR Z70.001]. Le service au sens OSI permet de spécifier les relations entre deux "couches" adjacentes: *"le service rendu par une couche N à une couche N+1"*. Un service est spécifié par la définition des données échangées: les N-SDU, Service Data Units de niveau N. Cette spécification externe fait abstraction de la réalisation du service, c'est-à-dire dans le cas des communications, fait abstraction du détail des mécanismes de réalisation du "protocole" utilisé. Plus précisément le service se manifeste par l'échange de données et de paramètres au moyen d'une "requête" émise par l'utilisateur et une "confirmation" émise en retour par le fournisseur de ce service.




*services et protocoles de communication dans le modèle OSI de l'ISO*

D'une façon plus générale, mais toujours dans le même esprit, le service peut être défini [Lap 85] comme:

| **service**: comportement d'un système tel qu'il est perçu par son ou ses utilisateurs en réponse à ses requêtes. |
|---|

Cette définition souligne la nature abstraite du service qui ne spécifie pas **comment**, c'est-à-dire par quel mécanisme interne, ce comportement est réalisé. Cette notion de comportement est intéressante dans la mesure où elle permet de ne pas simplement considérer un sous-système en tant que transformation d'information, mais comme un contrôleur d'activité à un niveau d'abstraction donné.

Le service permet de définir une relation hiérarchisée "**utilise le service de**" entre l'utilisateur consommateur et le système producteur de ce service. Cette relation est **structurellement stable** dans la mesure où les rôles assurés respectivement par le producteur et l'utilisateur ne peuvent être échangés.

### 1.3.2. Modélisation des architectures de SAP par la notion de service

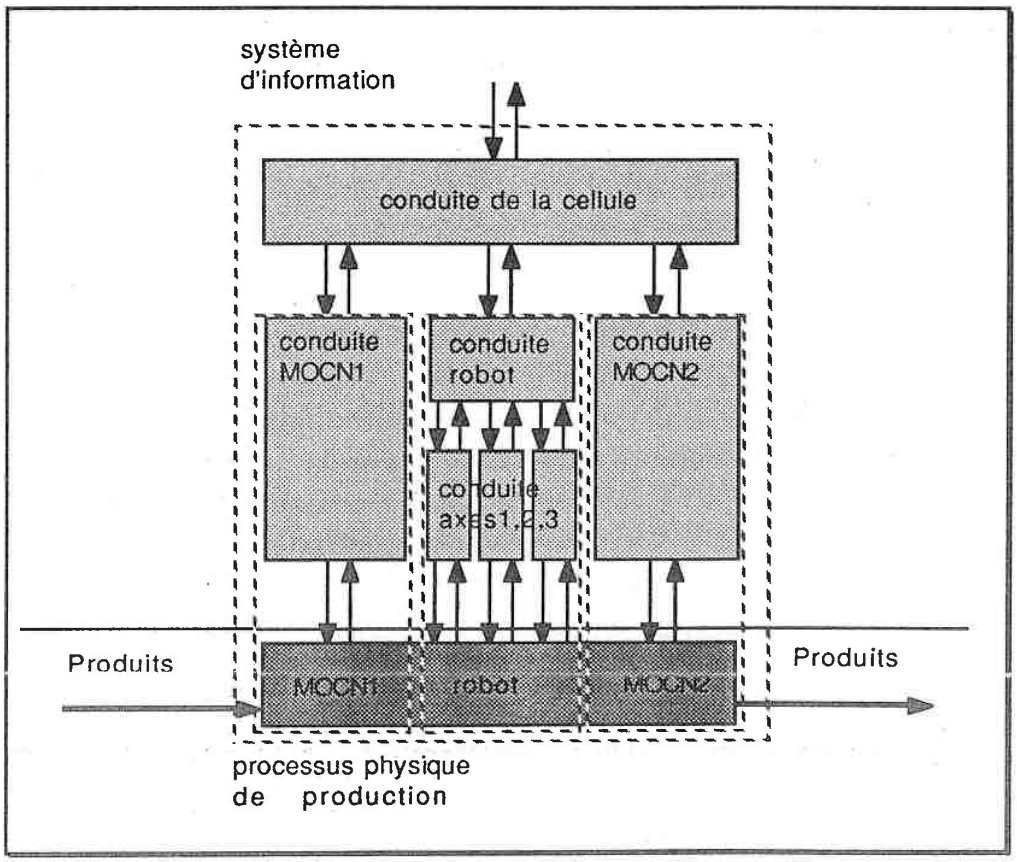
Comment utiliser cette notion de service pour modéliser les architectures de SAP ?

Un service va permettre de spécifier la fonctionnalité externe d'un composant fonctionnel (une entité) sans avoir à décrire les mécanismes nécessaires à cette réalisation. Chaque entité est décrite en terme de comportement attendu par son utilisateur.

Pour cela un service de SAP sera défini par:

- le **type de fonctionnalité** (conduite, suivi, maintenance, sécurité),
- le **sous-ensemble du processus de production** concerné,
- l'**horizon** et la **période** de ce service (au sens GRAI, cf chapitre 5),
- les **paramètres** transmis en **requête** et les paramètres attendus en **compte-rendus** par l'utilisateur du service.

Pour illustrer cette approche, reprenons l'exemple de la cellule d'usinage présenté dans [Tho 86] pour introduire la notion de service et de contexte:




*exemple de modélisation avec le concept de service d'une cellule de fabrication*

**Un service intègre à la fois la modélisation du processus informatisé et la fonction attendue du système d'information exprimé en terme de finalité.** De ce point de vue la notion de service permet donc d'intégrer l'approche fonctionnelle cartésienne et l'approche systémique: on ne définit pas séparément la fonction avec ses entrées / sorties de la modélisation du système à informatiser (le processus de production) , mais on définit la fonction par le service, lui-même défini par rapport au processus et par ses paramètres en E/S: les Requêtes et les Compte-Rendus.

### 1.3.3. service et hiérarchie

La formalisation de la notion de service repose sur la relation "use" permettant de définir une relation directionnelle (orientée) de hiérarchie entre le consommateur et le producteur de ce service, si les rôles ne peuvent être échangés [Lap 88]. Cette relation définit donc une relation d'ordre partiel entre les composants de l'application. Les composants peuvent suivant cette relation être classés en niveaux. Nous adopterons la convention de numérotation la plus courante dans l'ordre croissant en partant des entités en contact direct avec le processus de production, le processus étant symboliquement noté niveau 0. Nous désignerons donc par **$EF_{i,j}$ , l'entité fonctionnelle numéro i de niveau j.**

#### service et dimension de la hiérarchie de coordination

Le concept de service est directement lié à la structuration hiérarchisée (echelon) d'un système. Il est à rapprocher du concept d'échelon dans la mesure où un service est spécifié par une relation définie sous forme de primitives requête / compte-rendu correspondant à la notion contrôle / feed-back (ou command/report). Globalement on peut donc assimiler la relation hiérarchisée "use" à la relation de la hiérarchie de coordination "echelon". Une légère différence conceptuelle entre ces notions provient du fait que le service concerne plutôt la spécification du niveau inférieur vu par le niveau supérieur (le service rendu par un élément de niveau i-1 à un élément de niveau i), sans considération directe de la notion de coordination, c'est-à-dire les relations avec les niveaux inférieurs. Dans la mesure où à ce niveau d'étude, nous ne cherchons pas à déterminer le "comment" (les algorithmes) de la coordination, l'approche service semble la plus adaptée.

#### service et dimension de la hiérarchie de décision

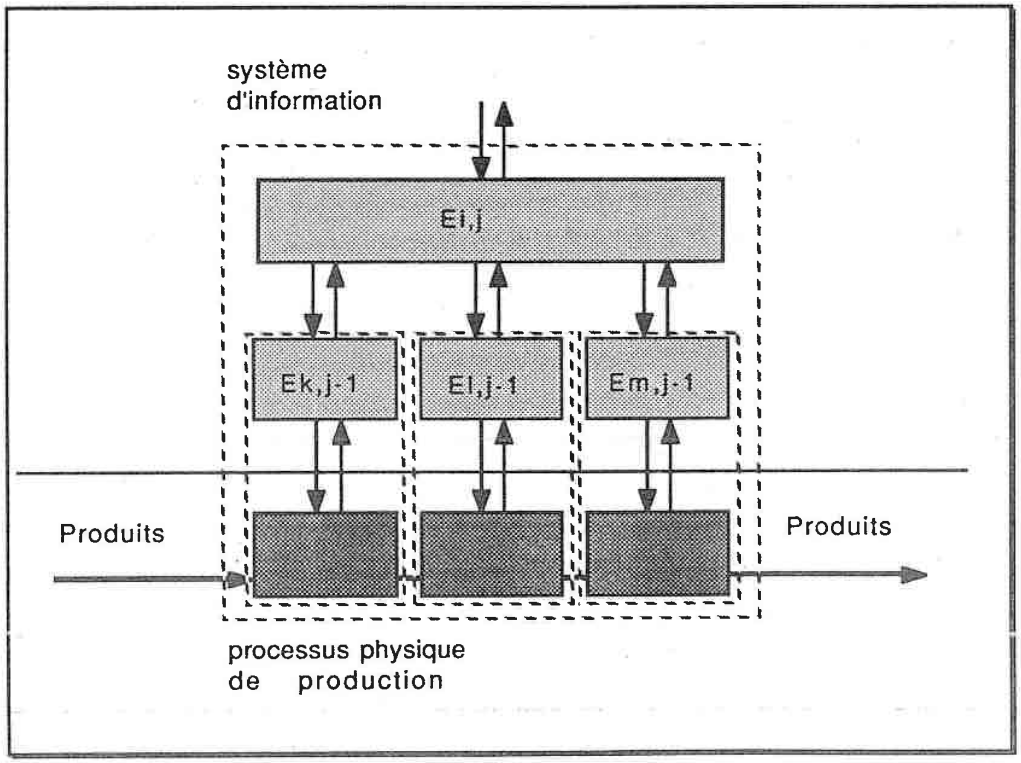
Le concept de service permet de représenter la hiérarchie de décision (layer). La précision de l'horizon et de la période temporels permet de bien identifier le niveau de décision de chaque niveau. L'emploi de modèle générique de l'entité fonctionnelle assurant une fonction de service n'implique d'ailleurs pas que cette fonction joue un rôle

de coordination: une entité peut ne définir les objectifs d'une entité unique de niveau inférieur.

**service et dimension de la topologie du processus**

Etant défini par rapport au contexte du processus, le service permet d'intégrer la dimension horizontale de décomposition / structuration du processus de production à la dimension verticale de la hiérarchie de coordination. En effet un niveau de description donné du processus est systématiquement associé une entité de service fonctionnel. A un niveau de description plus détaillé seront de même associées des entités de service de niveau hiérarchique inférieur. Le sous-ensemble du processus de production associé au service fonctionnel est le "champ d'action" ou le "champ de vision", il sera appelé le contexte de ce service.

**contexte**: le contexte d'une entité fonctionnelle est le sous-ensemble du processus de production dont cette entité a la responsabilité d'assurer le service.




*intégration des dimensions processus de production et hiérarchie de coordination par le concept de service*

**service et dimension de la hiérarchie de description**

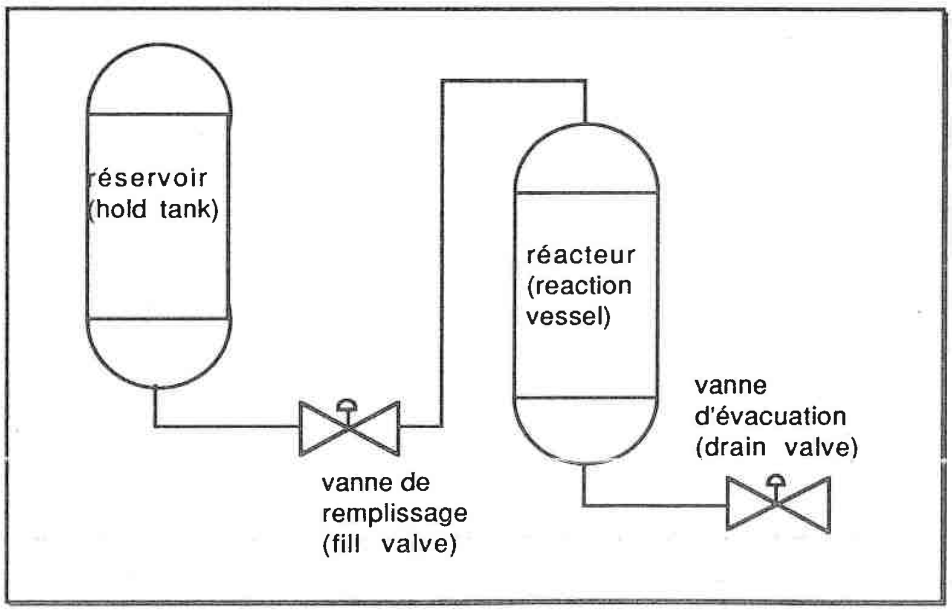
Un avantage supplémentaire de l'approche service visible sur ce schéma est d'associer proprement la hiérarchie de description ou décomposition (strata) à la hiérarchie "use" (layer). En effet la décomposition d'un service en sous-systèmes ou composants conduit systématiquement à définir un élément de contrôle et des services de niveaux inférieurs qui pourront à leur tour être décomposés. La décomposition peut être menée à différents niveaux de détail suivant les besoins de l'étude, le niveau le plus fin étant constitué par le niveau capteur/actionneur.

## 1.4. Exemple

L'exemple du système de contrôle du réservoir présenté "Control System Requirements" par [Ward 88] permet de montrer que l'on peut distinguer deux niveaux de spécification de la structure fonctionnelle du système.

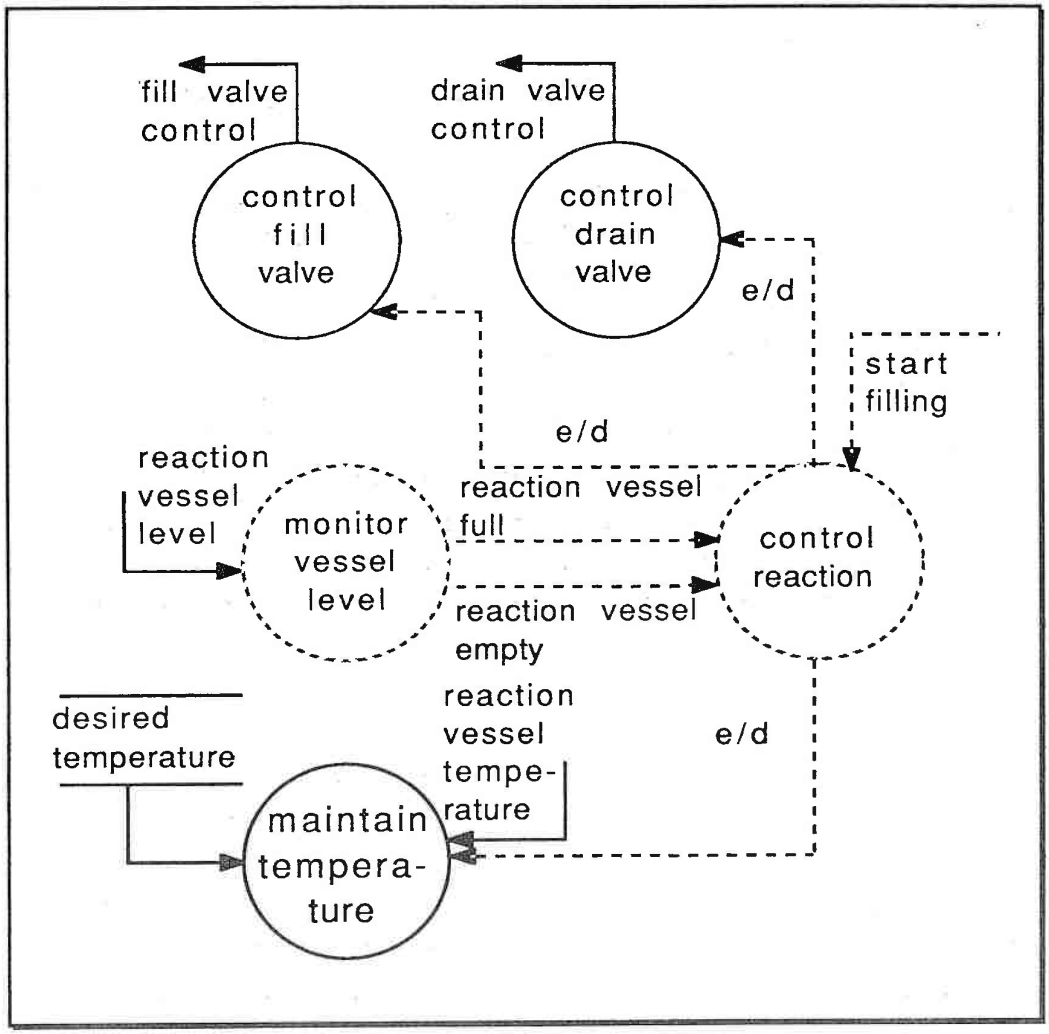
Les fonctions du système sont:

- laisser entrer le flux liquide depuis le réservoir de stockage vers le réacteur quand le signal de commande "start filling" est donné;
- stopper le flux liquide quand le niveau dans le réacteur atteind une certaine valeur;
- maintenir la température dans le réacteur à une valeur constante pendant une période de temps donnée;
- drainer le liquide depuis le réacteur après la fin de la période de temps.




*exemple du système de contrôle d'un réacteur [War 88]*

La représentation fonctionnelle proposée par Ward dans le formalisme Ward-Mellor est la suivante:

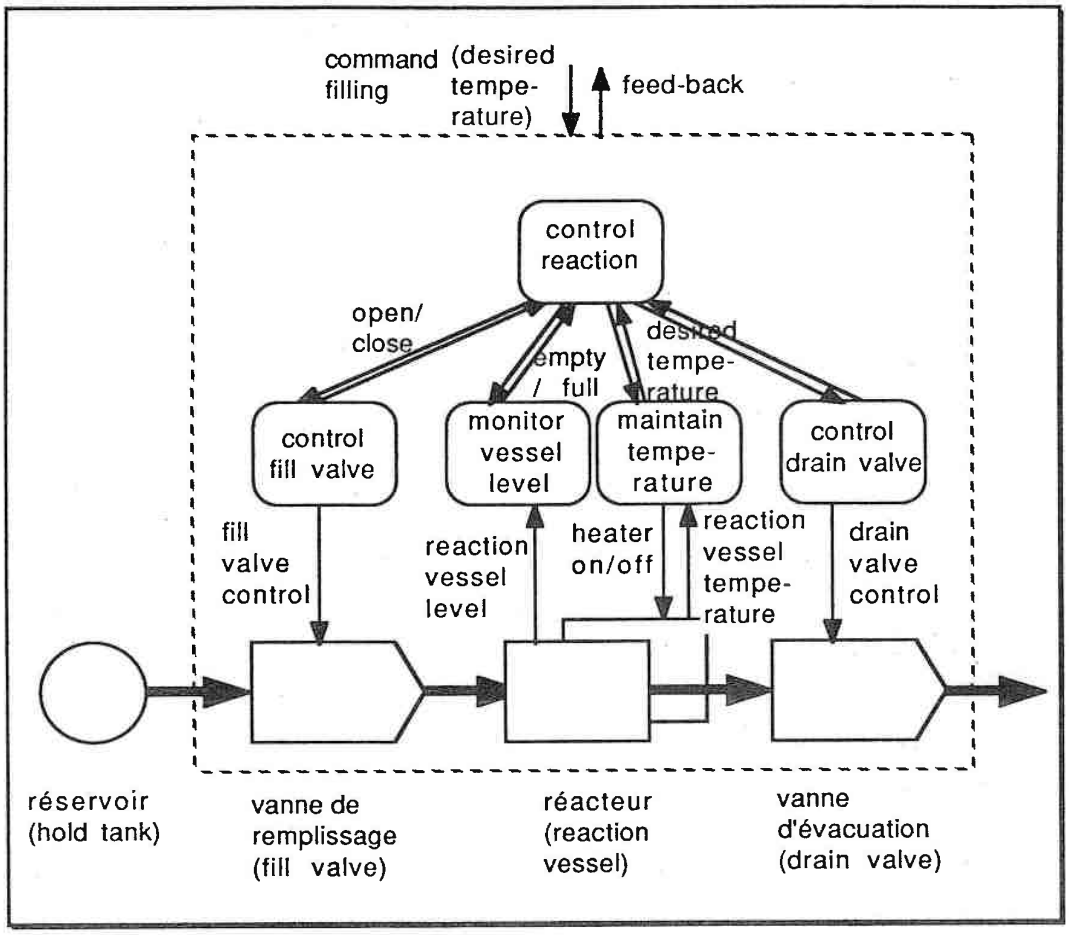



*spécification fonctionnelle proposée par Ward selon le formalisme Ward-Mellor [War 88]*

Avec le modèle de Structuration Fonctionnelle proposé, on peut considérer deux niveaux de représentation de la spécification fonctionnelle:

- un premier niveau très abstrait et systématique, où l'analyste-concepteur ne fait pas de choix en particulier sur la définition des signaux de synchronisation entre les entités, mais fait apparaître toutes les relations fonctionnelles de service pour bien définir le rôle de chaque fonction,

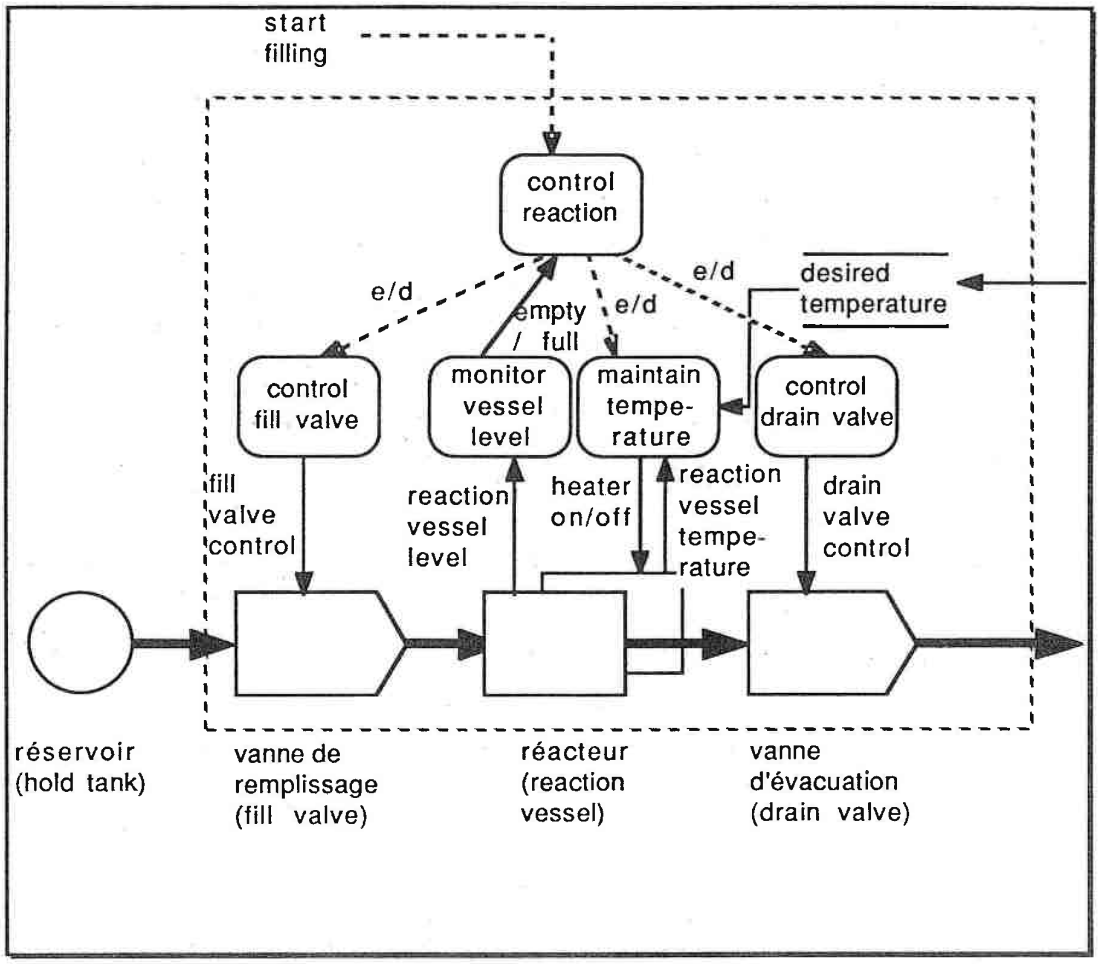
- un deuxième niveau plus "organique", où les relations fonctionnelles sont supportées par des échanges de messages, des données partagées et des signaux de synchronisation, et où l'on fait apparaître les structures de données.




*Structuration Fonctionnelle de 1er niveau de l'exemple du système de contrôle d'un réacteur*

La première spécification peut être considérée comme la spécification fonctionnelle (formelle) des besoins exprimés par le demandeur. Elle se situe au niveau d'une spécification de type SADT.

La deuxième spécification correspond à une spécification de la solution fonctionnelle imaginée par le concepteur en termes de fonctions de base d'un système d'information de traitement, mémorisation et échanges d'informations, solution indépendante des solutions techniques qui seront choisies dans les étapes suivantes.




*Structuration Fonctionnelle de 2ème niveau de l'exemple du système de contrôle d'un réacteur*

Ce modèle est en fait très proche du modèle de Ward-Mellor.

La première différence visible sur cet exemple simple est la lisibilité de la hiérarchisation des entités fonctionnelles en liaison avec la structure du processus de production. Cette lisibilité, absente des modèles d'analyse fonctionnelle de logiciel, est conforme avec celle des différents modèles d'architectures de SAP déjà présentés.

On constate également que notre formalisme incite à s'interroger sur la présence explicite ou implicite des relations de service (Requête et Compte-Rendu) entre les entités fonctionnelles des deux niveaux hiérarchiques adjacents: ici "control reaction" au niveau supérieur et "control fill valve", "monitor vessel level", "maintain temperature" et "control drain valve" pour le niveau inférieur.

Après cette présentation qualitative du modèle générique de structuration fonctionnelle, le paragraphe suivant est destiné à la formalisation de ces connaissances.

## 2. REPRESENTATION DES CONNAISSANCES

Avant de formaliser le modèle générique de structuration fonctionnelle d'architecture de SAP que nous venons de proposer, ce paragraphe présente le formalisme de représentation des connaissances retenu. Les formalismes de représentation issus des travaux de recherche sur l'intelligence artificielle permettent une représentation à la fois formelle et intuitive, contrairement aux formalismes disponibles jusqu'alors: mathématiques, graphiques et langage naturel [Gon 87]. La représentation des connaissances peut cependant être elle-même basée sur des formalismes très divers. En plus de l'intérêt que représente en soi une formalisation, le formalisme doit faciliter le travail de spécification dans la perspective d'une éventuelle implantation de ce modèle au sein d'un environnement informatique d'aide à la conception d'architectures de SAP. Une implantation partielle de ces concepts à l'aide d'un environnement générateur de système expert est présentée en annexe de ce document.

Après une présentation des objectifs et un rappel sur les différents formalismes de représentation des connaissances (seulement ceux proposés dans les modèles conceptuels ou cognitifs de conception de SBC et non pas dans les logiciels SBC eux-mêmes), ce paragraphe présente les choix retenus pour se doter d'un outil de conceptualisation formelle des connaissances utilisées en conception des architectures de SAP.

### 2.1. Caractéristiques attendues du formalisme

Les caractéristiques attendues d'une bonne représentation formelle sont [Win 84] de:

- rendre les éléments importants explicites,
- exposer les contraintes naturelles facilitant certaines classes de calculs,
- être complète, c'est-à-dire tout ce qui doit être dit,
- être concise, dire les choses efficacement,
- être claire, compréhensible,
- faciliter le traitement, le stockage et la récupération d'informations,
- supprimer les détails rarement utilisés tout en permettant de les visualiser si nécessaire.

### 2.2. Classification des connaissances

Les différents éléments de la connaissance peuvent être classifiés en deux grandes catégories, quel que soit le domaine considéré. La classification proposée dans le modèle

dit "sous-jacent" (c'est-à-dire conceptuel) par Abel semble représentative des différentes classifications publiées, souvent très proches au vocabulaire près:

- les **connaissances statiques** traduisant le "savoir" de l'expert :
  - le **domaine observé**: les concepts et relations structurelles ,
  - les connaissances abstraites liées à la **tâche de l'expert**: les concepts et relations de base, règles de synthèse ou d'abstraction.

  Ces connaissances correspondant pour nous aux modèles formels ou informels utilisés dans la conception d'architecture
- les **connaissances dynamiques** traduisant le "savoir-faire" de l'expert, en d'autres termes les procédures d'expertise correspondant aux différentes tâches de conception:
  - aspect **procédural**: tâches ou procédures d'expertise mettant en oeuvre les
  - **mécanismes** de raisonnement: règles génériques opérants sur les concepts et formalisant l'expertise de l'expert ou l'exploitation des graphes de causalité ou de dépendance entre les concepts.

### 2.2.1. Connaissances statiques

Les connaissances statiques peuvent être modélisées à partir de deux structures fondamentales d'informations [Pun 84] :

- les structures **hiérarchisées** où les concepts sont formalisés sous forme "d'objets" ou de "frames" permettant une **structuration arborescente** hiérarchisée en classes (relation "is-a") et offrant un mécanisme d'héritage des attributs ;
- les structures **parallèles connectées** (non-hiérarchisées) composées d'objets appartenant à des classes a priori indépendantes mais nécessitant pour modéliser un propos donné certaines connexions. Ces liaisons, que nous pourrions qualifier de **maillées** par opposition à la structure arborescente précédente, symbolisent la notion de **relation** linguistique propositionnelle, correspondant à celles formalisées dans les modèles conceptuels de données (Merise, Chen, Niam) ou dans les réseaux sémantiques: propriétés de structure [Gal 88]: par exemple "est le père de / est le fils de".

### 2.2.2. Connaissances dynamiques

L'approche des systèmes à base de connaissance consiste à énoncer les connaissances dynamiques sous forme de règles. L'intérêt des règles est leur simplicité d'expression et leur aspect déclaratif (au sens non-procédural). L'expérience montre qu'un seul niveau de règles ne permet pas une structuration correcte, et induit des difficultés dans la gestion de cette base et dans le contrôle des performances. De plus

certaines connaissances ne s'appliquent pas directement aux connaissances statiques mais à la logique d'application des règles elles-même. La notion de "méta-règles", règles s'appliquant aux règles permet la représentation de telles connaissances. Associée à la possibilité de structurer les règles en "catégories" ou "paquets", cela permet une gestion plus efficace de la base de règle.

## 2.3. Comparaison de différents modèles conceptuels de représentation de connaissances

Les **modèles de définition des besoins** (requirements definition) sont généralement basés sur les concepts d'objet, de réseau sémantique ou de modèle relationnel. Les capacités d'abstraction, de généricité et le côté intuitif de l'approche objet sont des apports décisifs pour la formalisation des connaissances statiques. La représentation d'entités réelles ou abstraites est très naturelle avec l'aide du formalisme objet grâce aux possibilités offertes par les notions de classe et d'héritage (structure relationnelle "hiérarchiques"). Cependant la représentation dans un modèle orienté objet des relations au sens des modèles Entité-Relation (-Attribut) ou Modèles Conceptuels des Données (structures relationnelles "parallèles") n'est pas directe. Un exemple type de l'utilité de cette notion de relation dans notre étude est la relation définissant l'implantation d'une entité fonctionnelle sur un ou plusieurs dispositifs de traitement. Or si d'un côté la conceptualisation des relations est particulièrement détaillée dans les approches de type entité-relation, l'approche entité-relation n'intègre pas la notion d'héritage.

Plusieurs démarches montrent l'intérêt d'intégrer ces deux aspects "objet" et "relation" au niveau d'un modèle conceptuel. Le tableau suivant compare les différents concepts présents dans un certain nombre de formalismes de représentation orientée **objet** et **relation** des connaissances au niveau conceptuel:

- le Modèle Sous-Jacent (MSJ) d'Abel [Abe 88],
- la Représentation des Connaissances centrée Objet (RCO) de Marcenac [Mar 88],
- le "framework for requirements models" (RMF) de Greenspan [Gre 82].

Nous avons également inclus:

- l'extension du modèle Entité-Association utilisée dans le domaine de la conception des automatismes, retenue par le projet BASE-PTA par Frachet [Fra 88],
- et le Modèle Conceptuel des Données de Merise [Tar 83] [Tar 85] à titre de comparaison.

Remarque:

Il est important de distinguer le problème de l'implantation des concepts de relation dans une structure ou un langage objet, de notre objectif d'adaptation d'un modèle objet pour faciliter la modélisation de ces concepts de relation. L'implantation des relations avec un langage orienté objet si elle n'est pas directe, peut être effectuée de différentes manières en fonction:

- des possibilités du langage,
- des propriétés de la relation particulière à implanter (cardinalités, contraintes, attributs de la relation).

La correspondance entre entités-propriétés et objets-attributs est bien sûr directe. Sans rentrer dans le détail, l'implantation des relations peut être effectuée:

- soit par création d'attributs,
- soit par création d'objets et d'attributs.

| | MSJ | RCO | RMF | BASE-PTA | MCD |
|---|---|---|---|---|---|
| **Aspect Statique** | | | | | |
| **objets** | concepts | objets (type 1) | objets | entités | entités-type |
| attributs | attributs | propriétés (type 4) | propriétés | attributs | propriétés-type |
| valeurs | valeurs | état (type 5) | | | |
| classe | | catégorie (type 2) | classe | classe générale | |
| **héritage** | | "is-a" | classification généralisation | spécialisation / généralisation | |
| instanciation | | | | occurence | |
| **agrégation** | | "is-a-part-of" | agrégation | agrégation | |
| ensemble | | "is-a-set-of" | | | |
| **relation** | relations | relation (type 3) | assertions | association | relations-type |
| cardinalité | | | | cardinalité | cardinalité |
| contraintes | | | | contraintes | contraintes |
| **Aspect Dynamique** | | | | | |
| tâche ou | tâche, | | | processus | traitement |
| fonction | | | activités | | |
| règles | règles, | règles et actions | | | |
| catégories | | | | | |

*tableau comparatif de différents modèles conceptuels de représentation des connaissances*

## 2.4. Choix retenu

Le formalisme conceptuel retenu a donc pour but:
- d'intégrer les deux structures relationnelles "hiérarchiques" et "parallèles" évoquées dans le paragraphe précédent,
- de se baser sur une approche déclarative.

Il associe pour cela:
- le formalisme **objet** (tel qu'il est utilisé dans la plupart des systèmes à base de connaissance c'est-à-dire restreint au triplet objet-attribut-valeur, donc sans la notion de méthodes / messages des langages orientés objets (LOO)),
- étendu avec la notion de **relation** entre les objets pour la représentation des connaissances statiques,
- et le formalisme des **règles** de production pour la représentation des connaissances dynamiques.

Ce choix objets-règles est justifié par le souhait de conserver une unité de représentation déclarative, et non procédurale comme les LOO [Mar 88]. Dans les systèmes à base de connaissances, la séparation explicite entre l'aspect statique et l'aspect dynamique est pour cela généralement préférée à l'intégration de "méthodes" ("attachement procédural") retenue dans les LOO.

Pour la partie statique, ce choix est analogue avec les choix du projet BASE-PTA [Fra 88] pour un modèle entité-relation étendu avec des concepts objets, ou les choix pour un "Extended Entity Relationship: EER" retenus dans la méthode M* par [Di L 87].

Quatre éléments de base sont donc utilisés de façon distincte:
- l'**objet** proprement dit,
- la **connexion** permettant de représenter tout type de raccordement ou de flux matériel ou logique entre objets,
- la **relation**,
- la **règle**.

Ce choix est justifié par l'intérêt conceptuel de ces éléments mais également par le fait que les connaissances représentées avec ce formalisme peuvent être implantées sans grandes difficultés d'adaptation à l'aide de techniques logicielle de type générateur de système à base de connaissance. Une implantation partielle de ces concepts à l'aide de l'environnement générateur de système expert G2 est présentée en annexe de ce document.

Les objets, connexions et relations peuvent posséder des attributs (parfois appelées propriétés comme dans les modèles entité-relation, propriétés "terminales" dans les réseaux sémantiques, ou "slots" dans certaines structures objets). Les valeurs des attributs associés aux objets (ou connexions) peuvent être des constantes, des variables booléennes, quantitatives, symboliques, ou textuelles (chaînes de caractères). Ces variables peuvent être définies par défaut, calculées par des formules spécifiques ou génériques, ou bien sûr par des règles.

La notion de connexion doit être bien distinguée de la notion de relation. Une connexion permet de représenter un flux entre des objets "communicants" (quelque soit la nature du flux: produits, information, énergie). Une relation ne correspond pas à un flux mais à une association sémantique statique entre des objets (exemple classique: la relation "est marié avec"). Une connexion à la même structure qu'un objet proprement dit, c'est-à-dire un certain nombre d'attributs contenant des valeurs. Ils diffèrent par leur processus d'instanciation puisque l'instanciation d'une connexion ou d'une relation n'est possible qu'à partir des instances d'objets à relier.

Nous utiliserons par la suite le terme général **élément** ou élément de base pour désigner ces objets, connexions, relations, règles.

## 2.5. Notations

Les tables suivantes correspondent aux définitions des classes d'objet, de relations et de connexions. Les notations utilisées sont inspirées du générateur de systèmes-experts G2 (Gensym Corp.) utilisé pour les développements présentés dans la troisième partie de cette thèse. Chaque concept sera défini par un certain nombre d'attributs, propres à chaque élément de base retenu (objet, connexion, relation, règle).

Les objets possèdent des **attributs, relations et connexions spécifiques** que l'on doit définir. Pour chaque classe d'objet, on spécifie la **classe de niveau supérieur** dont les attributs, relations et connexions sont directement hérités. Ils ne nécessitent donc pas une nouvelle saisie, mais sont rappelés à l'utilisateur. L'**instanciation** permet de préciser si les classes peuvent être instanciées ou si elles sont intermédiaires, c'est-à-dire non-terminales dans la hiérarchie arborescente des classes. La **définition graphique** d'un objet est un "icône" (dessin) correspondant à une représentation commune des instances de la classe (à ne pas confondre avec une éventuelle représentation graphique des classes et de leurs relations).

Afin d'alléger les notations, il est utile de définir une fois pour toutes la relation particulière de composition / décomposition permettant de représenter la hiérarchie entre un objet et les objets qui le composent. Seuls les objets instanciables seront décomposables et non les objets intermédiaires. Cette relation doit être associée à des mécanismes de vérification de cohérence entre les connexions établies avec l'objet de niveau supérieur et ceux de niveau inférieur. En ce qui concerne la représentation graphique, il doit de plus être possible de passer directement à partir de la représentation graphique de l'instance de l'objet au sous-espace hiérarchique contenant ses objets composants. Cette relation sera donc disponible, c'est-à-dire instanciable, pour toute instance de tout objet sans avoir à déclarer la relation au niveau de chaque classe. Ceci revient à déclarer cette relation au niveau de la classe de base "objet", ce qui permet à toutes les classes d'en hériter. On précise donc au niveau de chaque classe $C_i$ :

- si la classe est décomposable et / ou élémentaire,
- dans le cas où elle est décomposable, quelles sont les classes $\{C_j\}$ dont peuvent se composer les instances de la classe $C_i$.

Les connexions et les relations ne seront pas décomposables.

| une définition d'**objet** | |
|---|---|
| classe | <nom de classe> |
| classe supérieure | <toute classe d'objet> |
| instanciation | <instanciable ou intermédiaire> |
| décomposition | <décomposable en <classes d'objets autorisées>>, ou élémentaire> |
| attributs spécifiques | <nom d'attribut, type, valeur par défaut>, |
| attributs hérités | |
| relations spécifiques | <toute classe de relation>, |
| relations héritées | |
| connexions spécifiques | <toute classe de connexion>, |
| connexions héritées | |
| description graphique | |

Les connexions possèdent des attributs comme tous les objets, mais elles ne sont pas connectées entre elles. La définition graphique précise la forme du trait de l'instanciation d'une connexion. Une connexion ne peut être instanciée qu'à partir d'une instance d'objet.

| une définition de **connexion** | |
|---|---|
| classe | <nom de classe> |
| classe supérieure | <toute classe de connexion> |
| instanciabilité | <instanciable ou intermédiaire> |
| attributs spécifiques | <nom d'attribut, type, valeur par défaut>, |
| attributs hérités | |
| relations spécifiques | <toute classe de relation>, |
| relations héritées | |
| description graphique | |

On considère que les relations ne sont ni connectées, ni reliées entre elles. Les relations entre les instances ne possèdent pas de représentation graphique mais sont associées à une table de correspondance (ce qui n'empêche pas de représenter graphiquement les relations entre classes comme dans un modèle conceptuel de données). Les instances de relations contiennent les tuples désignant les instances d'objets en relation. La consultation et la mise à jour de ces tuples doit être possible:

- globalement au niveau d'un tableau de tous les tuples correspondant à la relation,

- localement au niveau de chaque instance d'objet avec la désignation des instances en relation avec l'instance locale (ceci correspond à un attribut de type liste qui serait mis à jour automatiquement à chaque modification de l'attribut dans chaque objet en relation).

| une définition de **relation** | |
|---|---|
| classe | <nom de classe> |
| classe supérieure | <toute classe de relation> |
| instanciabilité | <instanciable ou intermédiaire> |
| attributs spécifiques | <nom d'attribut, type, valeur par défaut>, |
| attributs hérités | |
| classes reliées et cardinalités | <toute classe d'objet (minimum, maximum)>, |

### Tables d'instanciation

| un **<objet>** | |
|---|---|
| nom | <nom de l'objet> *identifiant de l'instance* |
| <nom d'attribut> | <valeur par défaut> |
| ... | ... |
| <nom de relation> | <instance de l'objet en relation> |
| ... | ... |

| une **<connexion>** | |
|---|---|
| nom | <nom de la connexion> *identifiant de l'instance* |
| <nom d'attribut> | <valeur par défaut> |
| ... | ... |
| <nom de relation> | <instance de l'objet en relation> |
| ... | ... |

| table **<relation>** | | | | |
|---|---|---|---|---|
| nom | <nom de la relation> *identifiant de l'instance* | | | |
| <nom d'attribut> | <valeur par défaut> | | | |
| ... | ... | | | |
| <nom de classe><br><nom de classe> | <instance> | <instance> | <instance> | ... |
| <instance> | | | | |
| <instance> | | | | |
| ... | | | | |

*(les commentaires sont en italique)*

# 3. FORMALISATION DES MODELES

Ce paragraphe présente de façon détaillée les modèles génériques de structuration fonctionnelle et d'architecture opérationnelle selon le formalisme retenu dans le paragraphe 2 précédent. Cette présentation concerne uniquement les connaissances déclaratives. Les connaissances procédurales liées à la démarche méthodique de mise en oeuvre de ces modèles fera l'objet du chapitre 7.

Il est important de ne pas considérer les modèles proposés comme des modèles figés définitivement. Le choix du formalisme déclaratif orienté objet a précisément pour but de permetrre un enrichissement progressif de ce modèle générique sans remettre en cause sa structure de base grâce aux possibilités d'héritage. En fonction des particularités du domaine étudié ou des procédures de conception particulières à l'entreprise, le modèle proposé pourra être enrichi pour capitaliser l'expérience en conception accumulée.

## 3.1. Elements de base

### référence

Afin de référencer tous les éléments (objets, connexions et relations) intervenant dans le modèle d'une architecture, des attributs sont destinés à recevoir la référence au document correspondant. Tous les éléments utilisés hériterons donc de ces éléments "référencés". On pourra de plus utiliser un attribut destiné à recevoir des commentaires explicatifs.

| une définition d'**objet** | |
|---|---|
| classe | **objet-référencé** |
| classe supérieure | objet |
| instanciation | intermédiaire |
| décomposition | indéfinie (car intermédiaire |
| attributs spécifiques | référence est donnée par une variable textuelle,<br>*(référence au document fourni par le client)*<br>commentaires est donnée par une variable textuelle,<br>*(commentaires explicatifs)* |

| une définition de **connexion** | |
|---|---|
| classe | **connexion-référencée** |
| classe supérieure | connexion |
| instanciabilité | intermédiaire |
| attributs spécifiques | référence est donnée par une variable textuelle,<br>*(référence au document fourni par le client),*<br>commentaires est donnée par une variable textuelle,<br>*(commentaires explicatifs)* |

| une définition de **variable** | |
|---|---|
| classe | **variable-référencée** |
| classe supérieure | variable |
| instanciabilité | intermédiaire |
| attributs spécifiques | référence est donnée par une variable textuelle,<br>*(référence au document fourni par le client),*<br>commentaires est donnée par une variable textuelle,<br>*(commentaires explicatifs)* |

| une définition de **relation** | |
|---|---|
| classe | **relation-référencée** |
| classe supérieure | relation |
| instanciabilité | intermédiaire |
| attributs spécifiques | référence est donnée par une variable textuelle,<br>*(référence au document fourni par le client),*<br>commentaires est donnée par une variable textuelle,<br>*(commentaires explicatifs)* |

## 3.2. Système Automatisé de Production

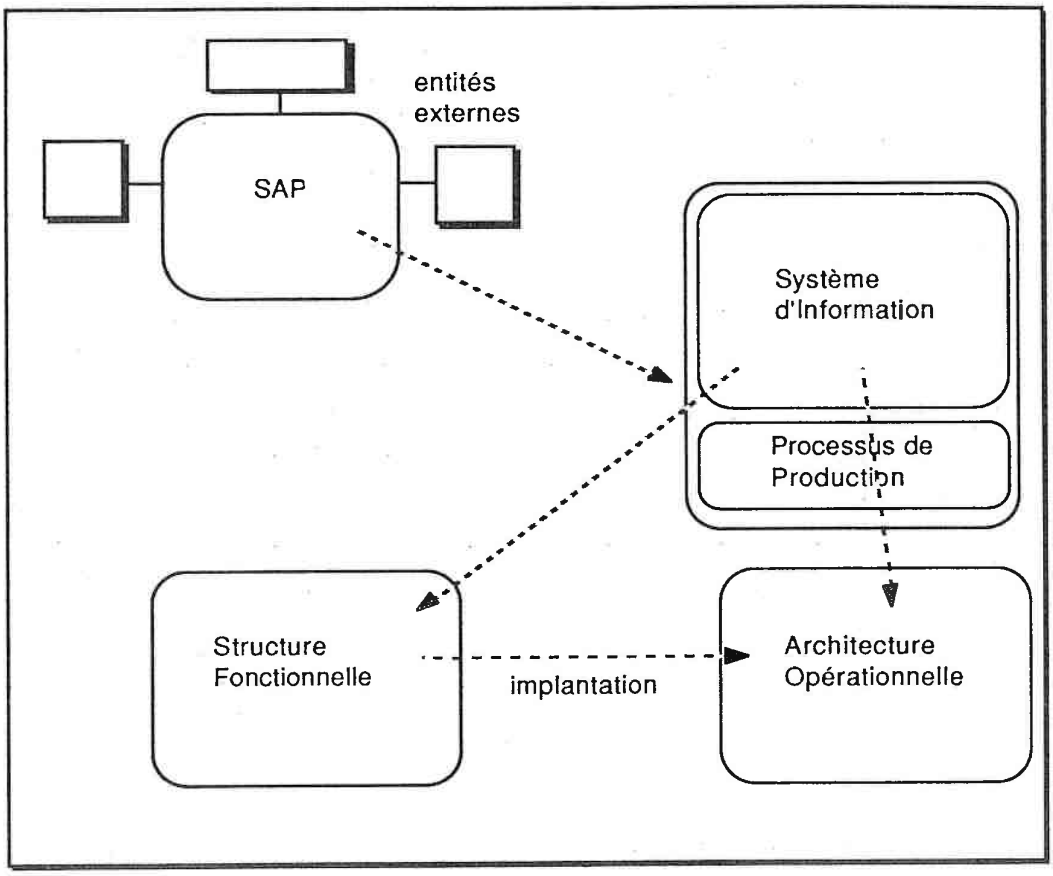
Le premier élément appréhendé par le concepteur est le système global appelé **système automatisé de production**. Il se compose du système d'information, du processus-de-production et de liaisons-fonctionnelles. Le SAP doit être situé dans son environnement: il faut donc identifier les relations existant entre le SAP et les "entités externes". Le diagramme de ce premier niveau de représentation est appelé un diagramme de contexte de façon analogue avec la méthode SA/RT.

| une définition d'**objet** | |
|---|---|
| classe | **système-automatisé-de-production** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (système-d'information, processus-de-production) |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | liaisons-fonctionnelles |
| connexions héritées | |
| description graphique |  |

**interface externe**

Le SAP est connecté par des liaisons-fonctionnelles à des interfaces-externes: l'**interface externe** permet de représenter les éléments externes à la prestation avec lesquels des communications doivent être prévues.

| une définition d'**objet** | |
|---|---|
| classe | **interface-externe** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | liaisons-fonctionnelles |
| connexions héritées | |
| description graphique |  |




*les relations entre les différents concepts de structuration du SAP*

### 3.3. Système d'Information

Le **système d'information** correspond à la définition donnée dans le premier chapitre. Il est constitué d'un modèle de structure fonctionnelle et d'un modèle d'architecture opérationnelle.

| une définition d'**objet** | |
|---|---|
| classe | **système-d'information** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (structure-fonctionnelle, architecture-opérationnelle) |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **contexte-SAP** relie à processus-de-production<br>*(un système-d'information est dédiée à un processus)* |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique |  |

#### contexte

La relation **contexte-SAP** associe le système d'information dédié au processus-de-production.

| une définition de **relation** | |
|---|---|
| classe | **contexte-SAP** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | système-d'information (1,1), processus-de-production (1,1)<br>*(tout système-d'information a pour contexte un processus-de-production et tout processus-de-production est le contexte d'un système-d'information )* |

### 3.3.1. Structure Fonctionnelle

La structure fonctionnelle du système-d'information est composée d'entités-fonctionnelles, de liaisons-fonctionnelles et d'utilisateurs. La structure fonctionnelle est connectée aux opérateurs du processus-de-production et aux entités-externes par des connexions de type liaisons-fonctionnelles.

| une définition d'**objet** | |
|---|---|
| classe | **structure-fonctionnelle** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (entités-fonctionnelles , liaisons-fonctionnelles , utilisateurs) |
| attributs spécifiques | |
| attributs hérités | référence is given by a text variable,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **contexte-SF** relie à modèle-processus<br>*(une structure-fonctionnelle est dédiée à un modèle-processus)*<br>**implantation-SF** relie à architecture-opérationnelle<br>*(une structure-fonctionnelle est répartie sur une ou plusieurs architectures-opérationnelles et une architecture-opérationnelle supporte une structure-fonctionnelle )* |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique |  |

**contexte**

La relation **contexte-SF** (structure fonctionnelle) associe une structure-fonctionnelle dédiée à un modèle-processus.

| une définition de **relation** | |
|---|---|
| classe | **contexte** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | structure-fonctionnelle (1,1), modèle-processus (1,1)<br>*(toute structure-fonctionnelle a pour contexte un modèle-processus et tout modèle-processus est le contexte d'une structure-fonctionnelle )* |

**implantation**

La relation **implantation-SF** associe à la structure-fonctionnelle, une ou plusieurs solutions d'architecture-opérationnelle sur laquelle elle est implantée.

| une définition de **relation** | |
|---|---|
| classe | **implantation-SF** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | structure-fonctionnelle (1,n), architecture-opérationnelle (1,1)<br>*(toute structure-fonctionnelle a pour implantation un ou plusieurs architecture-opérationnelle et tout architecture-opérationnelle est l'implantation d'une structure-fonctionnelle )* |

**entités fonctionnelles**

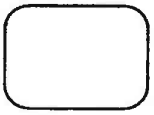
Les objets représentant les concepts fonctionnels de base du système d'information sont les **entités fonctionnelles** et les **liaisons fonctionnelles**. Une entité fonctionnelle peut

- soit se décomposer de façon itérative en entités-fonctionnelles connectées par des liaisons-fonctionnelles,
- soit être élémentaire si un niveau de décomposition supplémentaire n'est pas nécessaire.

Une entité fonctionnelle a pour attributs sa **fonctionnalité** (conduite, suivi, maintenance ou sécurité), le **niveau hiérarchique** où elle se situe dans l'architecture, son appartenance à des installations **existantes** ou non, sa **criticité** estimée vis à vis de la disponibilité du système d'information.

Une entité fonctionnelle a pour **contexte** le sous-ensemble des opérateurs du processus auquel elle est dédiée, et pour **implantation** le dispositif de traitement qui la supporte.

| une définition d'**objet** | |
|---|---|
| classe | **entité-fonctionnelle** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (entités-fonctionnelles , liaisons-fonctionnelles ) ou élémentaire |
| attributs spécifiques | **fonctionnalité** est donnée par une variable symbolique et a pour valeurs (conduite, suivi, maintenance), *(indique de quel type de fonction il s'agit)* **niveau** est donnée par une variable quantitative, *(numérotation du niveau hiérarchique dans l'ordre croissant à partir du processus physique de production),* **existant** est donné par un variable booléenne *(indique si l'entité fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place),* **criticité** est donnée par une variable quantitative, *(pondération du niveau de criticité de l'entité)* **horizon** est donnée par une variable quantitative, *(intervalle de validité des objectifs de l'entité)* **période** est donnée par une variable quantitative, *(intervalle de révision des objectifs de l'entité)* |

| | |
|---|---|
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **contexte** relie à opérateurs<br>*(une entité fonctionnelle est dédiée à la conduite, au suivi ou à la maintenance d'un sous-ensemble du processus)*<br>**implantation** relie à dispositif-de-traitement<br>*(une entité fonctionnelle peut-être répartie sur plusieurs dispositifs de traitement et chaque dispositif de traitement supporte plusieurs entités fonctionnelles)* |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique |  |

**contexte**

La relation **contexte** associe aux opérateurs du processus les entités fonctionnelles dédiées.

| une définition de **relation** | |
|---|---|
| classe | **contexte** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | entité-fonctionnelle (1,n), opérateurs (1,n)<br>*(toute entité-fonctionnelle a pour contexte un ou plusieurs opérateurs et tout opérateur est le contexte d'une ou plusieurs entité-fonctionnelle)* |

**implantation**

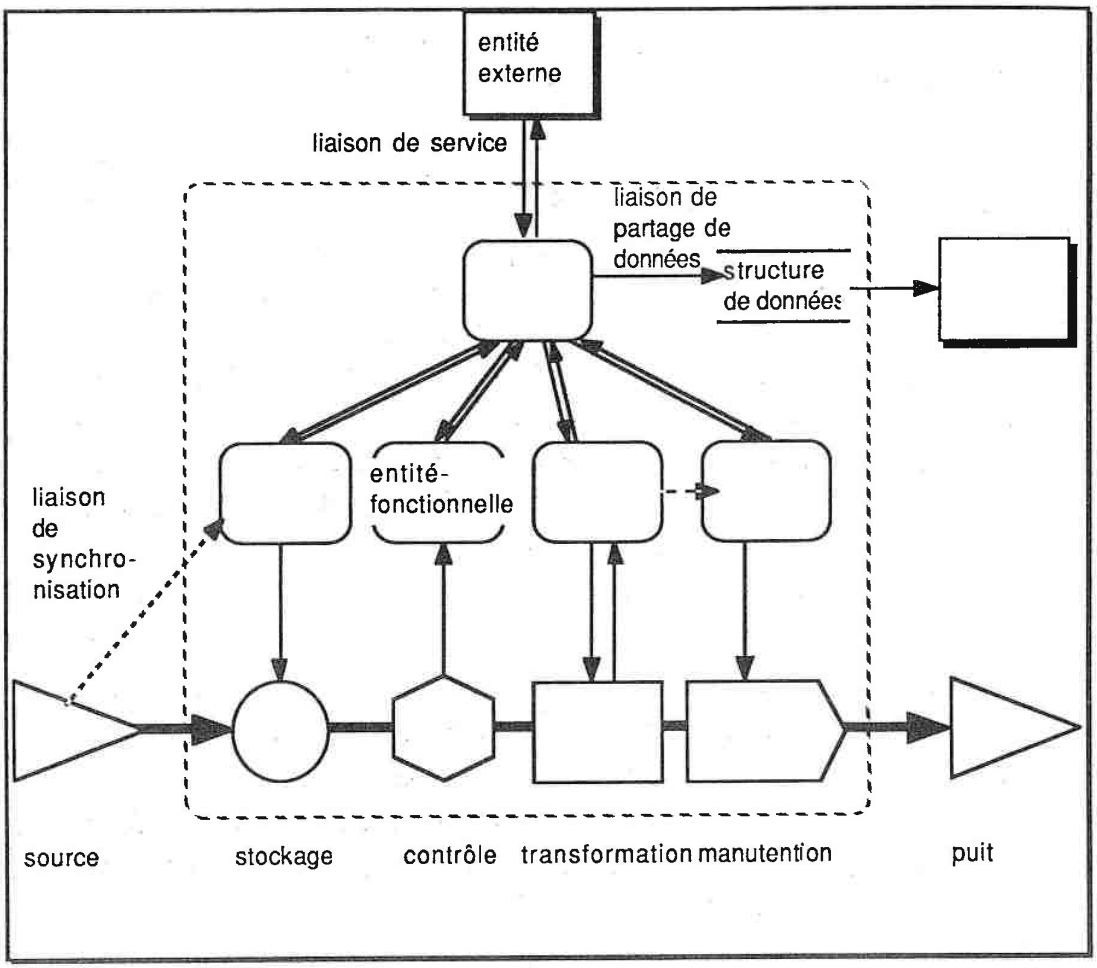
La relation **implantation** associe aux entités fonctionnelles les dispositifs de traitement sur lesquelles elles sont implantées.

| une définition de **relation** | |
|---|---|
| classe | **implantation** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | entité-fonctionnelle (1,n), dispositif-de-traitement (1,n)<br>*(toute entité-fonctionnelle a pour implantation un ou plusieurs dispositifs-de-traitement et tout dispositif-de-traitement est l'implantation d'une ou plusieurs entité-fonctionnelle)* |

### structure-de-données

La structure de données permet de représenter la fonction mémorisation des données accédées en lecture et écriture par les entités fonctionnelles.

| une définition d'**objet** | |
|---|---|
| classe | **structure-de-données** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | **niveau** est donnée par une variable quantitative,<br>*(numérotation du niveau hiérarchique dans l'ordre croissant à partir du processus physique de production),*<br>**existant** est donné par un variable booléenne<br>*(indique si l'entité fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place),* |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **implantation** relie à dispositif-de-traitement<br>*(une entité fonctionnelle peut-être répartie sur plusieurs dispositifs de traitement et chaque dispositif de traitement supporte plusieurs entités fonctionnelles)* |
| relations héritées | |
| connexions spécifiques | données-partagées |
| connexions héritées | |
| description graphique | ----------<br>---------- |




*les éléments du modèle générique de structuration fonctionnelle du SAP*

### liaison-fonctionnelle

Une **liaison-fonctionnelle** permet de connecter des entités-fonctionnelles entre elles et a pour but de supporter un message.

| une définition de **connexion** | |
|---|---|
| classe | **liaison-fonctionnelle** |
| classe supérieure | connexion-référencée |
| instanciabilité | intermédiaire |
| attributs spécifiques | **existant** est donné par un variable booléenne<br>*(indique si la liaison-fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place),* |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **support** relie à message<br>*(une liaison-fonctionnelle supporte un message)* |
| relations héritées | |
| description graphique | non-défini |

### support

La relation **support** indique quelle liaison fonctionnelle supporte un message donné.

| une définition de **relation** | |
|---|---|
| classe | **support** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | message (1,n), liaison-fonctionnelle (1,1)<br>*(tout message a pour support une ou plusieurs liaison-fonctionnelle (cas de la diffusion d'un même message à plusieurs destinataires) et toute liaison-fonctionnelle est le support d'un seul message)* |

**message**

Un **message** est un ensemble de données, caractérisé par sa **taille**, sa **durée de validité**, et son **temps de transfert.**

| une définition de **variable** | |
|---|---|
| classe | **message** |
| classe supérieure | variable-référencée |
| instanciabilité | instanciable |
| décomposition | décomposable en (données) |
| attributs spécifiques | **taille** est donnée par une variable quantitative et calculée par la formule:<br>**taille de message = somme de toutes les tailles des données composant le message**<br>*(taille du message en octets),*<br>**durée-de-validité** est donnée par une variable quantitative et calculée par la formule:<br>**durée-de-validité de message = minimum de toutes les durée-de-validité des données composant le message**<br>*(délai à partir de sa création au-delà duquel la valeur de la donnée n'est plus significative),*<br>**temps-de-transfert** est donnée par une variable quantitative et calculée par la formule:<br>**temps-de-transfert de message = minimum de toutes les temps-de-transfert des données composant le message**<br>*(délai maximum du temps de transmission entre le producteur et le consommateur de la donnée)* |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **support** relie à liaison-fonctionnelle<br>*(un message est supporté par une liaison-fonctionnelle )* |

**donnée**

Une **donnée** est une information élémentaire composant un message et est échangée par l'intermédiaire d'une liaison fonctionnelle. Une donnée est caractérisé par sa **taille**, sa **durée de validité**, et son **temps de transfert**.

| une définition de **variable** | |
|---|---|
| classe | **donnée** |
| classe supérieure | variable-référencée |
| instanciabilité | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | **taille** est donnée par une variable quantitative,<br>*(taille du message en octets),*<br>**durée-de-validité** est donnée par une variable quantitative,<br>*(délai à partir de sa création au-delà duquel la valeur de la donnée n'est plus significative)*<br>**temps-de-transfert** est donnée par une variable quantitative,<br>*(délai maximum du temps de transmission entre le producteur et le consommateur de la donnée)* |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |

**liaison de service**

La **liaison de service** est une liaison fonctionnelle bidirectionnelle correspondant à deux messages appelés Requête et Compte-Rendu.

| une définition de **connexion** | |
|---|---|
| classe | **liaison-service** |
| classe supérieure | liaison-fonctionnelle |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>existant est donné par un variable booléenne<br>*(indique si la liaison-fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place),* |
| relations spécifiques | **requête** relie à message<br>*(une liaison-fonctionnelle supporte une requête )*<br>**compte-rendu** relie à message<br>*(une liaison-fonctionnelle supporte un compte-rendu )* |
| relations héritées | |
| description graphique |  |

**requête**

La **requête** est le rôle particulier joué par un message dans la relation de service pour constituer les paramètres du service offert par le destinataire de la liaison de service à l'utilisateur origine de cette même relation.

| une définition de **relation** | |
|---|---|
| classe | **requête** |
| classe supérieure | support |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | message (0,n), liaison-service (1,1)<br>*(tout message peut être une requête pour une ou plusieurs liaison-service et toute liaison-service a pour requête un seul message)* |

### compte-rendu

Le **compte-rendu** est le rôle particulier joué par un message dans la relation de service pour constituer les paramètres en retour du service offert par le destinataire de la liaison de service à l'utilisateur origine de cette même relation.

| une définition de **relation** | |
|---|---|
| classe | **compte-rendu** |
| classe supérieure | support |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | message (0,n), liaison-service (1,1)<br>*(tout message peut être un compte-rendu pour une ou plusieurs liaison-service et toute liaison-service a pour compte-rendu un seul message)* |

### liaison de donnée-partagée

La **liaison de donnée-partagée** est une liaison fonctionnelle unidirectionnelle orientée du producteur du message vers son consommateur.

| une définition de **connexion** | |
|---|---|
| classe | **donnée-partagée** |
| classe supérieure | liaison-fonctionnelle |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>existant est donné par un variable booléenne<br>*(indique si la liaison-fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place).* |
| relations spécifiques | |
| relations héritées | **support** relie à message<br>*(une liaison-fonctionnelle supporte un message)* |
| description graphique |  |

**liaison de synchronisation**

La **liaison de synchronisation** est une liaison fonctionnelle unidirectionnelle orientée du producteur du message vers son consommateur.

| une définition de **connexion** | |
|---|---|
| classe | **synchronisation** |
| classe supérieure | liaison-fonctionnelle |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>existant est donné par un variable booléenne<br>*(indique si la liaison-fonctionnelle fait partie de la fourniture ou est déjà existante dans le système en place)*,relations spécifiques |
| relations héritées | **support** relie à message<br>*(une liaison-fonctionnelle supporte un message)* |
| description graphique |  |

**utilisateur**

L'**utilisateur** désigne un opérateur dialoguant avec une entité fonctionnelle.

| une définition d'**objet** | |
|---|---|
| classe | **utilisateur** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique |  |

### 3.3.2. Architecture Opérationnelle

L'**architecture opérationnelle** est composée de dispositif-de-traitement, de système-de-communication, et d'utilisateurs. Le système d'information est connecté avec le processus-de-production et les entités-externes par des connexions de type liaisons-fonctionnelles.

| une définition d'**objet** | |
|---|---|
| classe | **architecture-opérationnelle** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (dispositif-de-traitement, système-de-communication, utilisateurs) |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **contexte-AO** relie à processus-physique<br>*(une architecture-opérationnelle est dédiée à la conduite, au suivi ou à la maintenance du processus-physique)*<br>**implantation** relie à architecture-opérationnelle<br>*(une architecture-opérationnelle est l'implantation d'une structure-fonctionnelle et une structure-fonctionnelle a pour implantation une ou plusieurs architecture-opérationnelle )* |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique | |

**contexte**

La relation **contexte-AO** (Architecture Opérationnelle) associe une architecture-opérationnelle dédiée à la conduite, au suivi ou à la maintenance du processus-physique.

| une définition de **relation** | |
|---|---|
| classe | **contexte-AO** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | architecture-opérationnelle (1,1), processus-physique (1,1)<br>*(toute architecture-opérationnelle a pour contexte un processus-physique et tout processus-physique est le contexte d'une architecture-opérationnelle)* |

**Dispositifs de traitement:** supportent les entités fonctionnelles

| une définition d'**objet** | |
|---|---|
| classe | **dispositif-de-traitement** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **contexte-AO** relie à processus-physique<br>*(une architecture-opérationnelle est dédiée à la conduite, au suivi ou à la maintenance du processus-physique)* |
| relations héritées | |
| connexions spécifiques | liaison-opérationnelle, liaison-de-jonction |
| connexions héritées | |
| description graphique |  |

**systèmes de communication**

Les systèmes de communication en réseaux seront modélisés sous la forme d'un bus virtuel et d'éléments de jonction représentant le système de communication dans son ensemble, ceci quelque soit la topologie réelle du futur produit réseau matériel. Les communications supportées par des dispositifs point à point seront représentés par des connexions opérationnelles.

Un élément-de-jonction permet de relier un dispositif de traitement à un réseau-opérationnel par l'intermédiaire d'une liaison de jonction.

| une définition d'**objet** | |
|---|---|
| classe | **élément-de-jonction** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | réseau-opérationnel, liaison-de-jonction, liaison-opérationnelle |
| connexions héritées | |
| description graphique | |

Les connexions **réseau-opérationnel** relient les différents éléments de jonction.

| une définition de **connexion** | |
|---|---|
| classe | **réseau-opérationnel** |
| classe supérieure | connexion-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| description graphique | |

Les liaisons de jonction relient les dispositifs de traitement aux éléments-de-jonction.

| une définition de **connexion** | |
|---|---|
| classe | **liaison-de-jonction** |
| classe supérieure | connexion-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| description graphique | |

Les liaisons-opérationnelles relient les dispositifs de traitement en point à point.

| une définition de **connexion** | |
|---|---|
| classe | **liaison-opérationnelle** |
| classe supérieure | connexion-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| description graphique | |

## 3.4. Processus de Production

Le processus physique de production constitue le cœur, la raison d'être du SAP. Une bonne compréhension de ce processus est indispensable à l'élaboration d'un système d'automatisation et d'informatisation structuré, cohérent et efficace. "*Tout système de contrôle de process implique un modèle du système à contrôler*" [Ove 88]. La modularité et la cohésion fonctionnelle du SI nécessitent l'identification de la structure du processus pour adapter celle du SI selon un principe bien connu en programmation, cité par Stevens [Ste 74] "*calquer la structure de la conception sur la structure du problème, autrement dit la forme doit suivre la fonction.*" La difficulté consiste à déterminer le degré de détail de la description suffisant pour la compréhension. Ce chapitre a pour but de montrer quels sont les niveaux de détail pertinents pour la modélisation du processus de production. Un formalisme simple supporté par un outil graphique doit faciliter cette modélisation. Nous nous appuierons sur les travaux portant sur la conception et la simulation des processus.

### 3.4.1. Flux de Production

Tout processus de production est destiné à effectuer des **actions sur des flux de produits**. Ces actions, mettant en œuvre des lois physiques, chimiques ou biologiques, sont destinées à augmenter directement ou indirectement la valeur ajoutée de ces produits. La notion de produit est à prendre au sens large: elle recouvre à la fois des matières ou de l'énergie dans différents états et couvre les différents stades d'élaboration du cycle de production: matières premières, produits semi-finis et finis. Nous faisons abstraction de la distinction entre processus continus, semi-continus (batch ou par lot) et manufacturiers (ou discrets).

La première analyse du processus consiste à **identifier et classifier les différents flux**. Ces flux définissent différents sous-système du processus de production parfois appelés "domaines" (cf paragraphe sur les concepts de structuration de l'ingénierie). Ces domaines peuvent par exemple correspondre à des sous-systèmes de production de pièces détachées destinées ensuite à être assemblées. Il est donc nécessaire de préciser les entrées / sorties des flux de chaque domaine pour assurer leur intégration. Un autre exemple de domaine est celui des flux d'énergie ou "**communs**": air comprimé, vapeur, eau, distribués dans les différents domaines relatifs aux flux de produits. La Gestion Technique des Communs (GTC) est une fonction en tant que telle et doit être appréhendée séparément toujours dans un soucis de définition cohérente et complète. Un autre exemple consiste à considérer les domaines relatifs aux flux de ressources telles que des outils ou des pallettes de transport.

Une définition séparée des différents domaines doit conduire à mieux expliciter toutes les **interactions** et les **besoins de coordination** entre ces domaines. La description de ces interactions est établie à partir de la définition des entités constituants les différents domaines.

### 3.4.2. Classification des composants d'un processus de production

La description d'un processus de production par une séquence d'événements correspondant au passage des produits par les différentes étapes de la fabrication est possible en théorie, mais cela nécessite de prendre en compte tous les états des équipements et leurs interactions. Cette approche est trop complexe et fastidieuse pour être opérationnelle en pratique à ce stade de la conception [Ove 88]. Une approche structurelle interprétant l'usine comme un ensemble de processus physiques parallèles en interaction est en phase d'avant-projet suffisante au concepteur du système d'information pour :

- comprendre le fonctionnement et l'organisation globale du SAP,
- dégager la structure du processus pour ensuite adapter au mieux celle du système d'information.

**Entités d'un processus**

Un domaine du processus de production est donc un système composé d'un ensemble fini d'entités agissant sur ces produits. Chaque entité est considérée comme une "boîte-noire" recevant des produits en entrée et générant des produits en sortie après exécution d'actions ou opérations.

Selon cette approche un processus est donc caractérisé par:

- la nature des entités (les composants du système) et
- leur agencement en termes d'entrées-sorties, (c'est à dire la structure de ce système).

Toujours selon une approche systèmique, les produits peuvent être positionnés par rapport à un référentiel comprenant les trois axes [Le M 77]:

- forme ou état,
- espace,
- temps.

La forme ou l'état d'un produit est l'ensemble de ses caractéristiques physiques modifiables ou mesurables. De ce point de vue il est donc possible de classifier les entités ("processeurs") agissant sur les produits ("processés") par leur type d'action suivant les axes de ce repère. Nous définirons ainsi en fonction de la finalité principale de l'opération, des entités de **transformation, manutention, stockage et contrôle.**

Ces quatre catégories sont généralement reprises à quelques variantes près dans le cadre des approches de simulation, en particulier dans les approches "orientées objet". Nous citerons seulement quelques exemples significatifs:

- la Méthode de Description de Fabrication ("Manufacturing Description Method") de l'approche IBM IMMS [Eng 85] ("Integrated Manufacturing Modeling System"):
  *MAKE, MOVE, REST, VERIFY,*
- les "macro-primitives" du "macro-langage développé à l'ENSAM par [Amm 85] : *circulation, stockage, postes de travail, plus générateur de flux d'entrée et de sortie,*
- les objets standards disponibles dans OASYS [Bel 8x]:
  *STATIONS ET MACHINES, STOCKS, TRANSPORT,* plus *famille de produit et ressources,*
- la classification évoquée par [Hol 87] :
  *STOCKAGE, MANUTENTION , USINAGE,* ainsi que les *interfaces d'entrée et de sortie.*
- la classification par effet de Allen et Smith "DCLASS" citée par [Ray 87]:
  *TRANSPORT,* et différents types de transformation
  *MATERIAL REMOVAL, PROPERTY CHANGE, GEOMETRY CHANGE, ASSEMBLY.*

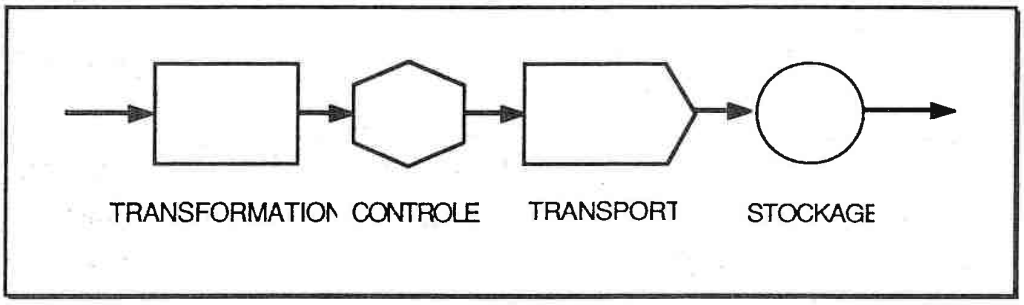
L'utilisation croissante de l'approche objet en simulation s'explique par sa forte capacité de modélisation des objets réels et de leurs propriétés structurelles et comportementales [Hol 87], [Bez 84] .

| | **manufacturier** | **process continu** |
|---|---|---|
| **entités de transformation:** | découpe<br>usinage | chauffage<br>électrolyse |
| **de manutention:** | convoyeur<br>chariot filoguidé | pompe<br>conduite |
| **de stockage:** | magasin<br>pile | réservoir<br>accumulateur |
| **de contrôle:** | machine à mesurer | analyseur |

*exemples de différents types d'entités du processus de production*

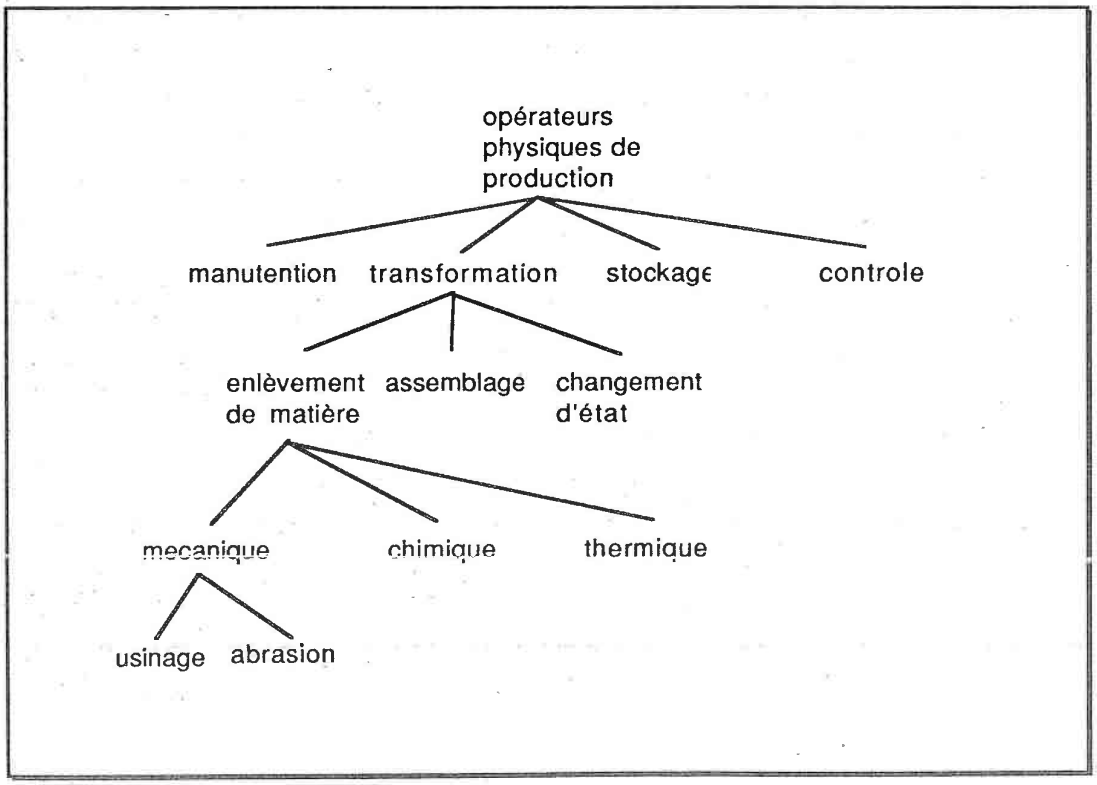
Différentes conventions de notation sont proposées dans ces méthodes ou sont normalisées par différents organismes (par exemple "Procedure charting symbols"de

l'American Society of Mechanical Engineers ASME citée dans [Bin 78]). Nous avons retenu les notations suivantes:




*représentation graphique des différents entités du processus de production*

Une telle classification peut être considérée comme un premier niveau abstrait dans une classification des éléments de processus de fabrication. Chaque élément peut ensuite être particularisé de façon hiérarchique. Les niveaux intermédiaires de l'arbre décrivant cette hiérarchie sont constitués des différents types de processus de plus en plus précis (cf. schéma), le dernier niveau (terminal) [Hol 87] définissant une machine opérationnelle proposée par un constructeur dont on quantifie ou qualifie les attributs.




*classification hiérarchisée des entités du processus de production*

Une telle taxonomie permet une modélisation en termes d'objets dont les attributs peuvent être hérités ou particularisés. L'approche objet est de plus en plus utilisée en simulation car elle permet de constituer des catalogues d'objets réutilisables pour modéliser très rapidement une application particulière, et ceci sans compétence approfondie des langages de simulation. Nous retiendrons cette approche très intuitive pour modéliser le processus au cours de l'analyse du contexte.

### Opérations

Une des difficultés posée par ce type de classification est de classer des machines flexibles telles que les robots effectuants des tâches successives de manutention et d'assemblage par exemple. Il convient en fait de distinguer entités et opérations, une entité pouvant exécuter plusieurs opérations. Cette notion d'opération correspond à la notion d'activité que l'on retrouve dans les réseaux GRAI, les actigrammes de SADT ou les process des Process Interaction Diagrams de [Ove 88]. Les opérations supportées par une entité peuvent être classées en opération principale ou en opérations secondaires en terme de valeur ajoutée. La qualification d'une entité dans une des catégories précédentes correspond à l'opération principale. Bien que l'affectation d'une opération à l'une des catégories ne pose généralement pas de problème dans la pratique, nous proposons les définitions suivantes:

- **transformation**: opération modifiant les caractéristiques ou **attributs physiques** (forme ou état) d'un produit,
- **manutention** (ou transport): opération modifiant la **position** d'un produit dans l'espace sans modification de ses caractéristiques physiques,
- **stockage**: opération maintenant un produit dans sa position **sans modification** de ses caractéristiques physiques pendant un **temps donné**.

Rq: *Selon ces définitions,un stockage peut donc être actif pour maintenir le produit dans son état initial: maintien en température, brassage, etc... ou passif pour un stock ne nécessitant pas d'action physique contrôlée: stock intermédiaire de produits manufacturiers stables et non périssables. Ainsi une fermentation n'est pas un stockage mais une transformation éventuellement passive si elle n'est pas contrôlée. Selon cette définition, un stockage peut être une "non-activité" bien que des ressources soient utilisées pendant un temps donné.*

A ces opérations nous ajoutons:

- **contrôle** (ou mesure): opération déterminant les caractéristiques physiques ou la position d'un produit à un instant donné **sans modification** de ses caractéristiques physiques ou de sa position.

Rq: *ce type d'opération est intéressant à identifier explicitement dans la mesure où il nécessite des fonctions d'automatisme spécifiques pour contrôler la mesure: déplacement d'un palpeur de mesure ou automatisme de prise d'échantillon par exemple.*

Pour être tout à fait complet dans la modélisation systémique du processus, il est nécessaire d'utiliser des entités particulières d'entrée et de sortie:

- **source** et **puits**: entités fictives respectivement produisant et consommant des flux de produits.

### Entités élémentaires d'un processus

Une entité est dite **élémentaire** dans la mesure où elle ne peut être décomposée en entités répondant aux définitions précédentes. Une telle décomposition est possible chaque fois que l'on peut définir une relation séquentielle de sortie --> entrée de produit entre deux composants d'un processus. Quand plusieurs opérations doivent nécessairement se dérouler en parallèle (simultanément), le type d'entité est celui de l'opération principale.

### Composition d'entités

A partir de ces entités, il est bien entendu possible de composer ou réaliser une agrégation de composants en "macro-entités" que l'on pourra classifier dans une des quatres catégories précédentes en fonction de la finalité principale des opérations. Une telle approche ascendante d'assemblage d'entités simples en entités plus sophistiquées correspond à la relation "compose". La relation réciproque "est composé de" permet de décrire de façon itérative et descendante un composant quelconque d'un processus de production. Le nombre de niveaux de décomposition est a priori quelconque et éventuellement arbitraire, c'est-à-dire reflétant la vision de l'analyste-concepteur sur l'organisation du processus. Dans la pratique, de tels "macro-composants" constituent par exemple des cellules ou des lignes de production, des ateliers de fabrication ou des unités de production. (cf. schéma)

### Entités génériques

Dans les applications complexes où le processus de production est composé de nombreux entités, il est préférable d'identifier les différents types d'entités similaires présentant les mêmes propriétés. Dans une application de mouvement de produits en

raffinerie on peut ainsi recenser un nombre d'entités de base assez restreint: réservoirs, conduites, vannes, pompes, analyseurs, alors que le nombre total de ces entités est de l'ordre de quelques centaines. Il n'est donc pas question d'étudier pour chaque entité particulière son système d'information de façon spécifique. On définit donc les entités de base pouvant comporter un certain nombre de variantes et l'étude du système d'information est également fait de façon générique. Le recensement du nombre exact de ces entités permettra d'effectuer le dimensionnement de l'application.

#### Niveau capteur/actionneur

La décomposition d'un processus physique de production est donc un processus itératif jusqu'à un niveau terminal. Afin de permettre la structuration des niveaux bas des automatismes, il est nécessaire de décomposer plus finement les entités élémentaires définis précédemment. Le niveau de détail des entités automatisées le plus fin intéressant l'automaticien est constitué par la description des dispositifs de conversion information-action ou état physique, c'est-à-dire concrêtement les actionneurs permettant de déclencher les transformations physiques et les capteurs permettant de connaître l'état physique d'un actionneur ou d'un produit. Dans le cadre de la conception d'architecture, ce niveau de détail peut ne pas être nécessaire pour les grands projets avant les études détaillées. Seule une estimation en nombre de points d'instrumentation est alors nécessaire.

### 3.4.3. Structure du processus de production

Cette structure est définie par la description de l'ensemble des interactions entre les composants. Ces interactions peuvent être représentées soit sous la forme d'un graphe orienté (diagramme de flux) reliant les entités élémentaires, soit sous la forme d'une matrice d'entrées-sorties. La forme graphique est la plus conviviale à manipuler pour les analystes-concepteurs, sachant que les deux représentations sont équivalentes sémantiquement et que des transformations automatiques sont possibles. Les **noeuds** du graphe sont donc les **entités** (ou process) **actives**, les **arcs** ou **connexions** représentent les **interactions** constituées soit par les **échanges de produits** passifs (continus ou discrets) [Ove 88] (et non pas les entités de transport). Chacun de ces objets possèdent des attributs permettant de spécifier des caractéristiques quantitatives de débit. Afin de supporter plusieurs niveaux de description, il est nécessaire de pouvoir formaliser la relation "est composé de" entre une entité et des entités constituants.

### 3.4.4. Formalisation du processus de production

Ce paragraphe présente la formalisation du processus de production effectuée selon le même formalisme objet que celui employé pour les modèles de Structure Fonctionnelle et d'Architecture Opérationnelle.

Le **processus-de-production** est composé d'entités (internes) et d'entités-externes (représentant les interfaces produit amont et aval) connectés par des flux-produits. Le processus-de-production est connecté au système-d'information par des liaisons-fonctionnelles.

| une définition d'**objet** | |
|---|---|
| classe | **processus-de-production** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (domaines, liaisons-fonctionnelles, flux-produits, opérateurs-externes) attributs spécifiques |
| attributs hérités | référence est donné par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle |
| connexions héritées | |
| description graphique | |

Le **domaine** permet de structurer le processus de production suivant les principaux flux de produits.

| une définition d'**objet** | |
|---|---|
| classe | **domaine** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | décomposable en (entités, liaisons-fonctionnelles, flux-produits) |
| attributs spécifiques | |
| attributs hérités | référence est donné par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | liaison-fonctionnelle , flux-produits |
| connexions héritées | |
| description graphique | |

Les **entités** sont les principaux constituants du processus destinés, comme nous l'avons vu dans les paragraphes précédents, à manipuler des flux de produits. Les entités sont sous le contrôle du système d'information par l'intermédiaire des liaisons de service.

| une définition d'**objet** | |
|---|---|
| classe | **entité** |
| classe supérieure | objet-référencé |
| instanciation | intermédiaire |
| décomposition | intermédiaire<br>*(un objet non-instanciable ne peut être décomposé)* |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | service, flux-produits, |
| connexions héritées | |
| description graphique | |

Le **flux de produit** permet de spécifier les relations structurelles entre les entités du processus de production.

| une définition de **connexion** | |
|---|---|
| classe | **flux-produit** |
| classe supérieure | connexion-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | débit-produit est donné par une variable quantitative |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| description graphique |  |

Les entités du processus de production sont des **entités internes** de **transformation, manutention, stockage, contrôle** ou **externes source** ou **puit.**

Une **entité interne** est une entité du processus de production faisant partie du système étudié. Une entité interne est caractérisé par le débit de produit en entrée et en sortie, ainsi que le nombre d'opérations de base effectuée par unité de temps.

| une définition d'**objet** | |
|---|---|
| classe | **entité-interne** |
| classe supérieure | entité |
| instanciation | intermédiaire |
| décomposition | intermédiaire<br>(un objet non-instanciable ne peut être décomposé) |
| attributs spécifiques | **débit-entrée** est donné par une variable quantitative,<br>*(capacité de traitement de l'entité),*<br>**débit-sortie** est donné par une variable quantitative,<br>*(capacité de traitement de l'entité),*<br>**nombre-d'opérations** est donné par une variable quantitative, |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique | |

Une entité de **transformation** effectue une transformation sur les flux de produits par modification de leurs caractéristiques.

| une définition d'**objet** | |
|---|---|
| classe | **transformation** |
| classe supérieure | entité-interne |
| instanciation | intermédiaire ou instanciable |
| décomposition | décomposable en (entité-interne) ou élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-entrée est donné par une variable quantitative,<br>débit-sortie est donné par une variable quantitative,<br>nombre-d'opérations est donné par une variable quantitative, |
| relations spécifiques | **consommation** relie à produit<br>*(une entité "consomme" un produit sur lequel elle opère)*<br>**production** relie à produit<br>*(une entité "produit" un produit sur lequel elle opère)* |
| relations héritées | |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique |  |

La transformation s'effectue sur les produits qu'elle "consomme" en entrée.

| une définition de **relation** | |
|---|---|
| classe | **consommation** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | transformation (1,n), produit (0,n)<br>*(toute transformation effectue la transformation à partir d'un ou plusieurs produits et tout produit peut être objet d'une ou plusieurs transformations)* |

La transformation fabrique les produits qu'elle "produit" en sortie.

| une définition de **relation** | |
|---|---|
| classe | **production** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | transformation (1,n), produit (0,n)<br>*(toute transformation effectue la production d'un ou plusieurs produits et tout produit peut être produit par une ou plusieurs transformations)* |

Une **entité neutre** n'effectue aucune modification des caractéristiques des produits: c'est le cas des entités de contrôle, de stockage et de manutention.

| une définition d'**objet** | |
|---|---|
| classe | **entité-neutre** |
| classe supérieure | entité-interne |
| instanciation | intermédiaire |
| décomposition | |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-entrée est donné par une variable quantitative,<br>débit-sortie est donné par une variable quantitative,<br>nombre-d'opérations est donné par une variable quantitative, |
| relations spécifiques | **traitement** relie à produit<br>*(un entité "traite" un produit sur lequel elle opère)* |
| relations héritées | |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique | |

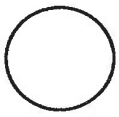
La relation traitement indique quel produit est manipulé par les entités neutres.

| une définition de **relation** | |
|---|---|
| classe | **traitement** |
| classe supérieure | relation-référencée |
| instanciabilité | instanciable |
| attributs spécifiques | aucun |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| classes reliées et cardinalités | entité-neutre(1,n), produit (0,n)<br>*(toute entité-neutre effectue le traitement d'un ou plusieurs produits et tout produit peut être l'objet d'un traitement par une ou plusieurs entités-neutres)* |

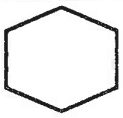
Une entité de **manutention** effectue une modification de la position spatiale des produits.

| une définition d'**objet** | |
|---|---|
| classe | **manutention** |
| classe supérieure | entité-neutre |
| instanciabilité | instanciable |
| décomposition | décomposable en (entité-interne) ou élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-entrée est donné par une variable quantitative,<br>débit-sortie est donné par une variable quantitative,<br>nombre-d'opérations est donné par une variable quantitative, |
| relations spécifiques | |
| relations héritées | traitement relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique |  |

Une entité de **stockage** maintient les produits dans un état et une position donnés pendant un certain délai.

| une définition d'**objet** | |
|---|---|
| classe | **stockage** |
| classe supérieure | entité-neutre |
| instanciabilité | instanciable |
| décomposition | décomposable en (entité-interne) ou élémentaire |
| attributs spécifiques | **capacité** est donné par une variable quantitative,<br>*(quantité totale de produit que peut stocker l'entité)* |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-entrée est donné par une variable quantitative,<br>débit-sortie est donné par une variable quantitative,<br>nombre-d'opérations est donné par une variable quantitative, |
| relations spécifiques | |
| relations héritées | traitement relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique |  |

Une entité de **contrôle** détermine des caractéristiques sur les produits sans modification de ces caractéristiques.

| une définition d'**objet** | |
|---|---|
| classe | **contrôle** |
| classe supérieure | entité-neutre |
| instanciabilité | instanciable |
| décomposition | décomposable en (entité-interne) ou élémentaire |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-entrée est donné par une variable quantitative,<br>débit-sortie est donné par une variable quantitative,<br>nombre-d'opérations est donné par une variable quantitative, |
| relations spécifiques | |
| relations héritées | traitement relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| connexions spécifiques | |
| connexions héritées | service, flux-produits, |
| description graphique |  |

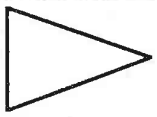
Les **entités externes** constituent les entrées-sorties des flux de produits vis à vis du système étudié.

| une définition d'**objet** | |
|---|---|
| classe | **entité-externe** |
| classe supérieure | entité |
| instanciation | intermédiaire |
| décomposition | (un objet non-instanciable ne peut être décomposé) |
| attributs spécifiques | |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | **traitement** relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| relations héritées | |
| connexions spécifiques | flux-produits, |
| connexions héritées | |
| description graphique | |

La **source** constitue une entrée des flux de produits vis à vis du système étudié.

| une définition d'**objet** | |
|---|---|
| classe | **source** |
| classe supérieure | entité-externe |
| instanciabilité | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | **débit-sortie** est donné par une variable quantitative, |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | traitement relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| connexions spécifiques | |
| connexions héritées | flux-produits, |
| description graphique |  |

La **puit** constitue une sortie des flux de produits vis à vis du système étudié.

| une définition d'**objet** | |
|---|---|
| classe | **puit** |
| classe supérieure | entité-externe |
| instanciabilité | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | **débit-entrée** est donné par une variable quantitative, |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle,<br>débit-produit est donné par une variable quantitative |
| relations spécifiques | |
| relations héritées | traitement relie à produit<br>*(une entité "traite" un produit sur lequel elle opère)* |
| connexions spécifiques | |
| connexions héritées | flux-produits, |
| description graphique |  |

Chaque **produit** est associé aux entités qui le manipule.

| une définition d'**objet** | |
|---|---|
| classe | **produit** |
| classe supérieure | variable-référencée |
| instanciation | instanciable |
| décomposition | décomposable en (produits) ou élémentaire |
| attributs spécifiques | position est donnée par une variable position, |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | traitement relie à produit<br>*(un produit est"traité" par une entité)* |
| relations héritées | |
| connexions spécifiques | |
| connexions héritées | flux-produits, |
| description graphique | |

| une définition d'**objet** | |
|---|---|
| classe | **actionneur** |
| classe supérieure | objet-référencé |
| instanciation | instanciable |
| décomposition | élémentaire |
| attributs spécifiques | nombre-d'opérations est donné par une variable quantitative, |
| attributs hérités | référence est donnée par une variable textuelle,<br>commentaires est donnée par une variable textuelle, |
| relations spécifiques | |
| relations héritées | |
| connexions spécifiques | |
| connexions héritées | service, |
| description graphique | |

## Conclusion

Nous avons proposé dans ce chapitre 6 une formalisation des concepts génériques retenus au chapitre 5. Le formalisme utilisé constitue la base d'une implantation en vue d'un outil d'aide à la conception d'architecture. La formalisation proposée permet d'envisager une traduction simple dans le format d'un environnement avancé de développement de type atelier logiciel [Xer 88] ou de type générateur de base de connaissances [G2 88] comme nous le présentons en annexe. Le choix d'un formalisme de type système à base de connaissances a également pour but de faciliter l'enrichissement progressif du modèle, ceci afin de capitaliser l'expérience accumulée par les concepteurs.

L'utilisation d'un tel modèle générique suppose la définition des procédures méthodiques parcourues tout au long de la phase de conception d'architecture. Le chapitre suivant a pour but de présenter ces différentes tâches d'analyse et de conception.

# chapitre 7:

# TACHES D'ANALYSE ET DE CONCEPTION

## INTRODUCTION

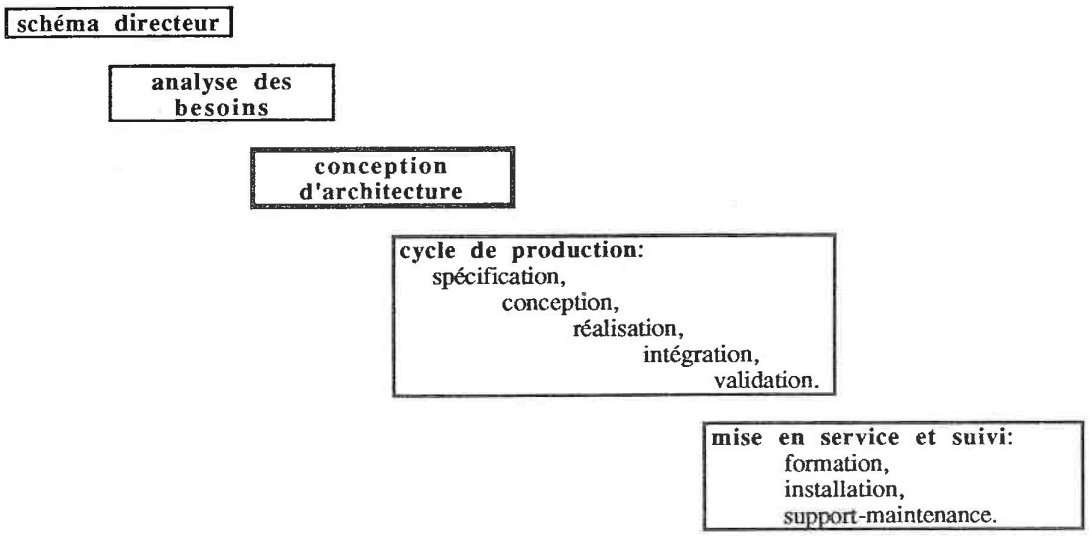
Après avoir mis en évidence et modélisé les objets ou concepts utilisés par les concepteurs d'architectures, nous allons aborder les procédures constituant les différentes tâches de la conception. Comme dans tous les problèmes de conception, il est possible de découper le processus global de conception en tâches distinctes. Ce découpage s'apparente à l'organisation traditionnelle des études en génie logiciel selon un cycle de développement ou "cycle de vie". Comme dans ce domaine, la décomposition ne doit pas préjuger un déroulement purement séquentiel de ces tâches (c'est d'ailleurs pourquoi nous avons retenu le terme tâche et non pas le terme étape ou phase). De nombreux auteurs ont montré la nécessité de procéder à des itérations permettant notamment d'améliorer la communication entre concepteur et utilisateur: *"La linéarité du schéma constitue une simplification pour en faciliter la lecture. En réalité, il existe des bouclages associés aux différents points de contrôle."* [Bod 83]. Le propos de cette décomposition est avant tout d'expliciter les axes essentiels d'investigation d'une telle étude, de dégager les différents documents et informations nécessaires en entrée, les documents produits en sortie, et de déterminer les différents points de contrôle de l'étude.

Nous rappelons dans un premier paragraphe "Avant-projet de conception d'architecture" où se situent la phase de conception d'architecture par rapport aux autres phases du cycle de vie d'un projet et quels sont les objectifs que doit viser le découpage en tâches. Le deuxième paragraphe "Etude comparative de quelques méthodes de spécification des besoins, d'analyse et de conception de systèmes" étudie les étapes proposées dans différentes méthodes existantes dans des domaines voisins. Le troisième paragraphe "Présentation des tâches retenues" introduit le découpage retenu en tâches principales.

# 1. DECOUPAGE EN TACHES DES ETUDES D'AVANT-PROJET DE CONCEPTION D'ARCHITECTURE

## 1.1. Position de l'avant-projet de Conception d'Architecture dans le cycle de vie d'un projet

Comme nous l'avons déjà abordé dans la première partie consacrée à la "Problématique", les études de Conception d'Architecture réalisées en avant-projet ou en réponse à des appels d'offres se situent avant une décision d'investissement du prescripteur, donc avant la phase de réalisation du projet. Les études de conception d'architecture interviennent après les études plus générales de type Schéma Directeur qui fixent les orientations stratégiques et technologiques sans aller jusqu'au détail des configurations de chaque projet. Le Schéma Directeur fournit à la Conception d'Architecture des informations dont il faudra tenir compte pour la définition des composants et de l'organisation de l'architecture. Le schéma suivant résume la position de la phase de Conception d'Architecture par rapport aux différentes étapes du cycle de vie d'un projet.




*la position de la phase de Conception d'Architecture par rapport aux étapes du cycle de vie d'un projet*

## 1.2. Objectifs du découpage en tâche

La qualité des études et développements de systèmes informatiques fait l'objet de nombreuses études. La définition de cette notion:

**qualité**: aptitude d'un produit ou service à satisfaire les besoins des utilisateurs.

montre d'une façon évidente que la qualité de la solution repose sur la qualité des études de conception (de l'analyse du besoin à l'élaboration de la solution). Au niveau d'une étude d'avant-projet, la partie qualité de la réalisation n'est bien-sûr pas prise en compte directement même si certaines exigences peuvent déjà être explicitées.

L'amélioration de la **qualité** et de la **productivité** des études d'avant-projet (cf. chapitre Contexte Ingénierie) peuvent s'avérer contradictoires. Nous déduisons les objectifs principaux du découpage en tâches:

**1 - Séparer explicitement les aspects analyse et conception**, c'est-à-dire faciliter la distinction entre le "quoi" (les fonctions, ce que doit réaliser le système) et le "comment" (les solutions). Nous avons vu que faute de modèle adapté, les concepteurs raisonnent trop spontanément en termes de solutions opérationnelles. La définition de tâches distinctes orientées d'une part vers l'analyse et la structuration fonctionnelle et d'autre part vers la conception de solutions permet un découplage clair de ces activités. Retarder les choix de solutions tant que les différents critères de décisions n'ont pas été suffisamment explicités, permet de préserver la marge de manoeuvre du concepteur et donc d'augmenter les chances de se rapprocher de la solution la plus appropriée au problème posé. Cette séparation bien connue des informaticiens est à la base des méthodes de génie logiciel ou d'analyse de la valeur (voir le paragraphe suivant), mais n'est pas suffisamment appliquée dans le cadre des études d'architecture.

**2 - Faciliter le travail de recueil des informations**. Ce travail de préparation est une condition nécessaire au bon déroulement de l'étude puisqu'il alimente en données de base toutes les tâches suivantes. La diversité des applications rend difficile une formalisation des données de base nécessaires au concepteur. L'identification de ces données de base repose essentiellement sur l'expérience des concepteurs. Pour ne pas négliger cet aspect, il convient de bien expliciter cette activité préliminaire et de fournir un support méthodique adapté permettant de capitaliser l'expérience acquise à travers les différents projets.

**3 - Préciser les différents documents produits**. L'intérêt premier d'une méthode au sein d'une entreprise est d'être appliquée rigoureusement et systématiquement. Une telle application suppose la définition précise des documents produits. Le contenu des documents devra être présenté selon un plan de référence. Dans le cadre de ce travail de recherche nous nous contenterons de définir quels sont les documents produits par chaque tâche.

**4 - Ne pas introduire de rupture entre les différentes tâches**. Une séparation claire ne doit pas pour autant introduire de rupture entre les tâches. Une certaine continuité doit être présente tout au long du cycle de conception pour garantir

l'efficacité de la procédure. Pour cela les résultats d'une tâche doivent être exprimés sous une forme directement utilisable par la tâche suivante. Ceci suppose une cohérence globale au niveau des concepts de chaque tâche et des formalismes choisis.

**5 - Permettre la "décompilation" des connaissances** mises en oeuvre par les ingénieristes concepteurs. La démarche d'un expert consiste en effet souvent à appliquer des procédures de raisonnement empirique constituant des raccourcis difficiles à expliciter. Un découpage des différentes tâches accomplies par l'expert doit faciliter l'identification des règles de savoir-faire.

## 1.3. Etude comparative de quelques méthodes de spécification des besoins, d'analyse et de conception de systèmes

Ce paragraphe étudie comment sont structurées quelques méthodes d'analyse et / ou de conception de systèmes d'information ou de systèmes en général. Le lecteur désireux en première lecture de ne pas rentrer dans le détail de ces modèles pourra se reporter directement à la "présentation des tâches retenues" au paragraphe 1.4.

### 1.3.1. Etude préalable dans MERISE

référence: [Tar 83] , [Tar 85]

Parmi la couverture remarquablement complète que fait la méthode MERISE du cycle de vie d'un projet informatique, l'étude préalable correspond aux études d'avant-projet. Elle est située après une réflexion de type schéma directeur et avant le lancement d'une étude détaillée des besoins spécifiques. Les objectifs d'une telle étude sont résumés par *"savoir où l'on va avant d'aller trop loin "*. En effet *"la décision de faire n'est pas acquise au moment du lancement d'une telle étude, le domaine est à préciser et l'enveloppe budgétaire n'est qu'un ordre de grandeur"*.

Trois principes essentiels président le développement de l'étude préalable:
- obtenir un **niveau de détail suffisant** pour que les résultats de l'étude préalable soient **démonstratifs** et objectifs,
- mener l'investigation de manière telle qu'il soit possible d'obtenir les résultats escomptés dans un **délai réduit**,
- préserver une **vision globale** des phénomènes sur l'étendue du domaine.

Les étapes de l'étude préalables sont:

1- **recueil**:
- **observation globale**: connaître les acteurs, structurer et préciser le champ d'étude, dégager le SER,

- **diagnostic**: approfondir ou vérifier la connaissance des astreintes ou dysfonctionnements de l'existant, des besoins à satisfaire dans le cadre des politiques de l'entreprise, en faire une projection sur le futur et affiner la perception des motifs du projet;

2- **conception**:

- **conception générale**: concevoir des modèles de fonctionnement futur sous le double aspect données-traitements, (les modèles élaborés plus que décrire les solutions, précisent plutôt les "besoins" ("requirements") en termes d'organisation et de pilotage que devra supporter le futur système)
- **études des options opérationnelles**: élaborer les solutions en explorant les grands axes d'investigation: progiciel ou spécifique, architecture répartie ou centralisée;

3- **organisation**:

- **conception des solutions types**:
  - approfondir la répartition des tâches attribuées à chaque poste de travail compte tenu d'une architecture retenue,
  - définir les principaux messages reçus, manipulés ou transmis,
  - évaluer les charges de travail, les actions de formation à prévoir, l'acceptabilité des solutions proposées,
  - esquisser la structure des données (base unique ou partitionnée),
  - préciser la nature, le nombre et la charge des équipements et périphériques,
  - évaluer la charge de développement de logiciels spécifiques.

4- **appréciation**:

- **planification**: préciser les scénarios de mise en oeuvre pressentis et évaluation de l'adéquation aux objectifs de l'entreprise,
- **bilan**: synthèse économique et qualitative des solutions et planning retenus.

Ces principes conduisent à sélectionner un Sous-Ensemble Représentatif (SER) rejoignant la notion de pertinence que nous avons introduite dans le chapitre Contexte Ingénierie. La méthode préconise en effet d'utiliser les même outils que pour l'étude détaillée. Il est donc nécessaire pour respecter les délais réduits évoqués ci-dessus, de limiter le champ de l'étude préalable en choisissant certains sous-ensembles. Ce choix est différent de celui que nous avons retenu puisque nous proposons de prendre en compte toute l'application mais à un niveau de détail (de "profondeur") limité avec une démarche spécifique adaptée.

### 1.3.2. Analyse d'opportunité dans IDA

référence: [Bod 83]

La méthode IDA (Interactive Design Approach) développée à l'institut d'Informatique de Namur est assez proche de la méthode Merise quant au domaine d'application (la gestion), aux modèles utilisés et au découpage en étapes retenu. L'étude d'architecture est ici appelée "Analyse d'Opportunité". Sa particularité est d'insister sur "l'étude critique du système existant" en la mettant en parallèle avec la "définition du projet cadre". Les différentes tâches proposées sont:

1 - **identification du projet** :
  - spécification des causes profondes d'insatisfaction,
  - localisation des déficiences,
  - définition de la frontière du projet.

2 a - **définition du projet cadre: spécification des besoins**:
  - objectifs del'organisation,

- objectifs informationnels (qualité, comportement),
- activités concernées,
- critères d'efficacité (informationnelle, organisationnelle, économique et de réalisation).

2 b - **étude critique du SI existant**:
- diagramme des flux d'informations,
- critique du SI existant: fonctionnelle et structurelle.

3 - **élaboration de solutions**:
- définition fonctionnelle,
- technologie,
- personnes,
- structures d'organisation.

4 - **choix d'une solution**:
- évaluation des critères d'efficacité,
- calcul de rentabilité.

L'intérêt de l'approche proposée est l'importance accordée à la formulation des objectifs replacés dans le cadre organisationnel du projet. La première étape concerne essentiellement le prescripteur du système. Si l'ingénieriste peut intervenir à ce niveau en tant que conseil, nous avons pris en compte cet aspect dans notre étude en récupérant les informations produites par cette tâche.

Contrairement à l'étude préalable de Merise, les auteurs préconisent l'étude de l'ensemble de l'application à un niveau de détail limité avec l'aide d'un outil adapté (Diagramme de Flux d'Informations).

### 1.3.3. Requirements Definition de SADT

référence: [Ros 77]

Ross présente une méthodologie systématique de spécification des besoins "Requirements Definition" (RD), s'arrêtant avant la conception, et basée sur le langage SADT. Nous ne reviendrons pas sur les notations graphiques utilisées sur lesquelles nous avons déjà fait de nombreux commentaires dans le chapitre consacré aux "Concepts de modélisation d'une Architecture". Les RD sont définis comme*"une évaluation attentive des besoins qu'un système doit satisfaire" ("a careful assessment of the needs that a system is to fulfill")*. Ils recouvrent tout ce qui précède la partie conception, c'est-à-dire:

- "l'analyse du contexte":

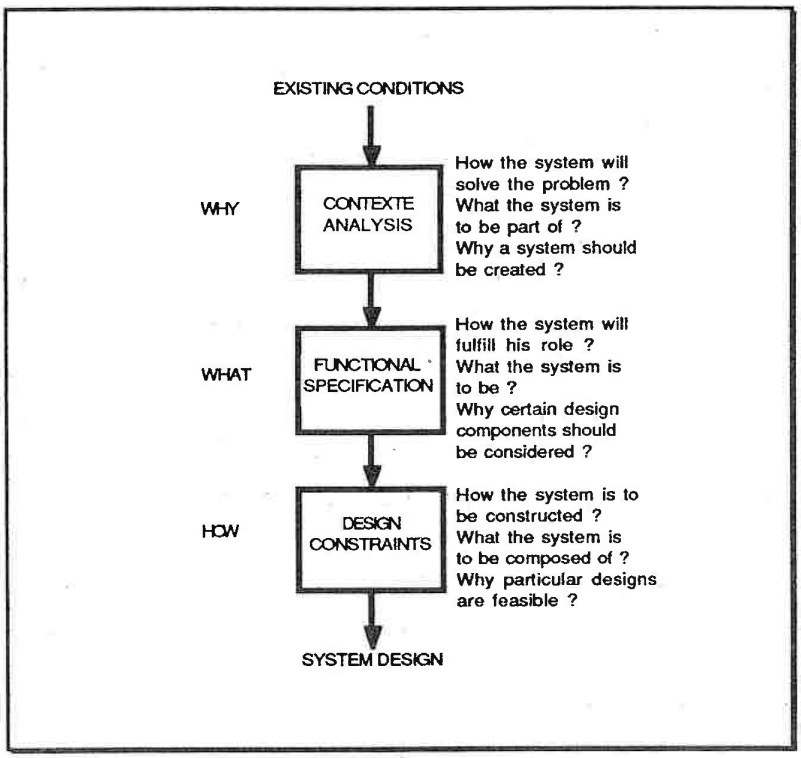
  "**why**": les raisons pour lesquelles le système doit être créé et pourquoi des données techniques, opérationnelles ou économiques constituent des conditions limites pour le système.

- la "spécification fonctionnelle":

  "**what**": une description de ce que le système doit être, en termes de fonctions qu'il doit accomplir.

- les "contraintes de conception":

  "**how**": un résumé des conditions spécifiant comment le système doit être construit et implanté ?




*les étapes de spécification des besoins proposées dans SADT*

L'idée, pour proposer une approche "faisable" d'un point de vue technique, opérationnel et économique, est de se baser sur la définition d'une *"architecture fonctionnelle"*. Cette architecture fonctionnelle est définie comme *"une représentation (layout) rigoureuse des activités assurées par le système et des choses avec lesquelles ces activités interagissent"*. Un processus de conception doit permettre de définir ensuite *"l'architecture système"* supportant les fonctions de cette architecture fonctionnelle. Une des idées mise en oeuvre est de répondre de façon itérative à des questions par des questions, c'est-à-dire selon une approche de décomposition descendante *"pourquoi, quoi, comment?"*

### 1.3.4. Information Systems Design par ZACHMAN

référence: [Zac 87]

L'auteur propose de redéfinir de façon cohérente les différentes tâches de la conception d'un système d'information. L'intérêt de la présentation est de faire ressortir clairement les différents points de vue (interlocuteurs, concepts modélisés). Le cadre d'étude proposé se compose d'étapes définies par analogie à d'autres domaines de conception (architecture immobiliaire, constructions aéronautiques):

- "**scope description**": description "grossière" de l'étendue du système, des objectifs décrivant ce que le client a en tête;
- "**model of the business**": retranscription structurée par le concepteur de ce qu'il a compris de l'application selon le point de vue du prescriteur ("owner's view");
- "**model of the information system**": transformation des besoins en produits "engineering design" ("designer's view");
- "**technology model**": liste de fournitures ("bill of materials") représentant le point de vue du développeur ("builder's view").

Ces trois représentations principales (les modèles) peuvent être examinés selon:
- "***what the thing is made of***", c'est-à-dire la structure, représentée par le modèle des **données**;
- "***how the thing works***", c'est-à-dire les transformations, représentés par le modèle des **traitements**;
- "***where the flows exist***", donc les flux représentés par le modèle du **réseau**.

Ces différents points de vue conduisent à considérer la conception selon une grille à deux dimensions. Chaque case de la grille correspond à une modélisation de l'application utilisant un modèle adapté.

| | **données** | **traitements** | **réseau** |
|---|---|---|---|
| **scope description** | | | |
| **model of the business** | | | |
| **model of the information system** | | | |
| **technology model** | | | |

*cadre des différentes tâches de conception d'un système d'information proposé par Zachman*

L'auteur évoque aussi trois autres aspects qu'il estime secondaires:
- "*who is doing what*", c'est-à-dire les responsabilités, représentées par le diagramme d'organisation;
- "*when the events take place*", c'est-à-dire la dynamique, représentée par le diagramme de séquencement;

- "*why choices are made*", c'est-à-dire les motivations représentées par les objectifs.

### 1.3.5. Analyse de la valeur (AV)

référence: [Jou 85], [AFN X50], [DGA 86]

Cette démarche est intéressante à examiner dans la mesure où elle a très tôt et dès sa naissance (l'après-guerre aux Etats-Unis: "Value Analysis" par L.D.Miles) mis en évidence l'importance de l'expression fonctionnelle des besoins dans la conception de produits quels qu'ils soient. La vocation de l'AV est de favoriser la **remise en cause des choix techniques afin d'optimiser les coûts de réalisation d'un produit** (ou système). Cette remise en cause est basée sur l'identification des besoins des utilisateurs par une analyse fonctionnelle détaillée et le chiffrage des fonctions. La démarche ne consiste donc pas simplement à débarrasser le produit de toutes les fonctions "inutiles". Elle propose au contraire de bien expliciter les fonctions qu'il doit assurer, de définir des critères quantitatifs d'évaluation (donc de qualité) et de mettre un coût en face de chacune de ces fonctions (et non pas chacun de ses composants).

Les caractéristiques pratiques de l'AV sont:

- le travail de groupe renforçant la "**participation**" des personnes concernées par un projet et favorisant la **synergie** entre des **équipes multi-disciplinaires**,
- l'importance accordée à la **recherche d'informations** sur le problème posé mais aussi sur les solutions existantes et éventuellement concurrentes,
- l'utilisation de **techniques de créativité**.

Une étude d'Analyse de la Valeur est constituée des différentes phases suivantes:

- **orientation**, définition de l'étude et de ses objectifs,
- **information**, rassemblement de toutes les données se rapportant à l'étude,
- **analyse** des fonctions et des coûts,
- **recherche** d'idées et des voies de solutions,
- **étude** et évaluation des **solutions**,
- **bilan** prévisionnel, proposition de choix et décision,
- **suivi** de la réalisation, bilan réel.

Ces différentes phases parcourues selon une démarche itérative, s'appuient sur un certain nombre de méthodes et de moyens.

L'**Analyse Fonctionnelle** a pour but de traduire l'expression des besoins de l'utilisateur en "fonctions de service" (*"action attendue d'un produit pour répondre au*

*besoin d'un utilisateur donné"*), destinées précisément à satisfaire ces besoins. Les "fonctions techniques" sont les fonctions complémentaires parfois nécessaires à la réalisation des "fonctions de service" mais dont il s'agit de bien évaluer la pertinence technico-économique. Les méthodes ou techniques utilisées pour l'analyse fonctionnelle sont empiriques, très peu ou pas formalisées et surtout très orientées vers l'analyse de produit. Parmi ces techniques, la méthode FAST, la "rosace des fonctions" et surtout le Cahier des Charges Fonctionnels sont les plus souvent citées.

La méthode **FAST** pour "Functional Analysis System (au sens systématique) Technique" a pour but d'ordonner sous la forme d'un diagramme les fonctions essentielles du système étudié. Cette méthode représente graphiquement la décomposition des fonctions selon la relation orientée (de la gauche vers la droite) "comment" (réciproque: "pourquoi") liant une fonction à des sous-fonctions de niveau inférieur. On parvient donc jusqu'aux fonctions "techniques" ou fonctions de "conception" permettant de réaliser les fonctions de service: à la différence des fonctions de service, ces fonctions techniques ne sont pas perceptibles à l'extérieur du système et dépendent des choix de conception. Des relations verticales "quand" permettent de traduire l'aspect temporel.

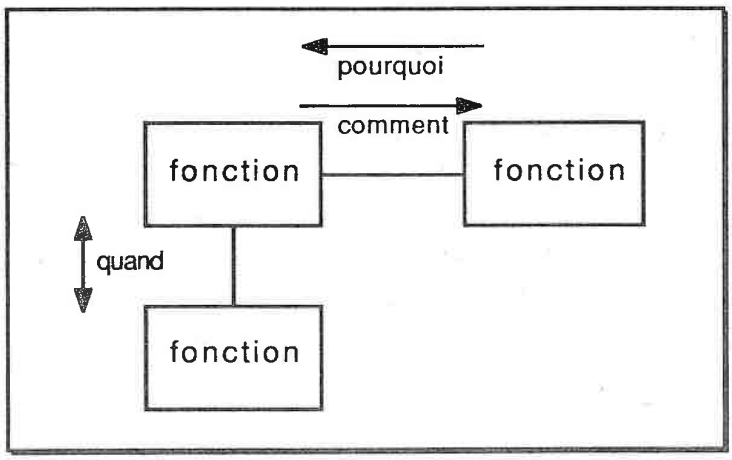
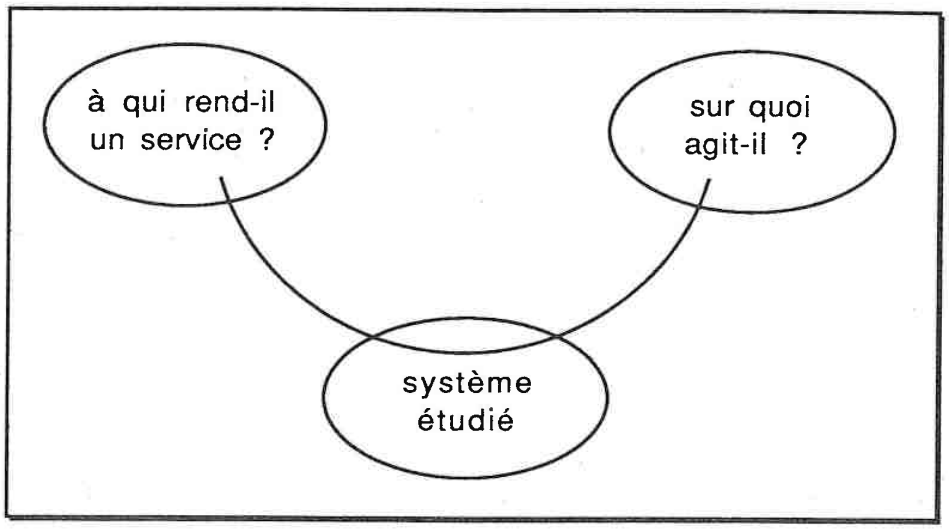



*schéma fonctionnel de la méthode FAST*

Une autre méthode d'analyse des fonctions, parfois appelée la "**rosace des fonctions**", est basée sur la représentation graphique des relations entre les utilisateurs et éléments extérieurs du système étudié. Ces relations "traversant" le système correspondent aux fonctions du système et sont appelées "*chemins fonctionnels*". Les fonctions sont considérées comme des actions ou réactions du système sur des éléments de son environnement.




*l'approche de la "rosace des fonctions" en Analyse de la Valeur*

Le **Cahier des Charges Fonctionnel** (CdCF) [AFN 84] dont la normalisation est en cours (référence) a pour vocation essentielle de définir les besoins en préservant le maximum de latitude au concepteur. Le CdCF a pour but:

- de clarifier les responsabilités entre le demandeur et le concepteur,
- d'exprimer les besoins des utilisateurs,
- de provoquer chez le concepteur la conception et la réalisation du produit (ou système) le plus efficient,
- de favoriser le dialogue entre le demandeur et le concepteur.

Ces différents objectifs correspondent donc tout à fait à ceux que nous poursuivons. Cette approche est basée sur une expression purement fonctionnelle des besoins, sans introduire d'éléments de solutions, et par une limitation de l'expression des contraintes sous forme de critères d'acceptabilité, de seuil de flexibilité et de taux d'échange. Le CdCF n'est pas une technique de représentation, contrairement aux deux "méthodes" cités ci-dessus. Mais il propose une structure méthodique d'étude: étapes, points à préciser, etc..., cette étude pouvant faire appel à une ou plusieurs techniques de représentation. Le plan proposé comprend:

- la présentation générale du problème: produit, marché, contexte, objectifs, besoins, environnement,
- l'expression fonctionnelle du besoin: fonctions principales et contraintes, critères d'appréciation, niveaux de ces critères,
- l'appel à variantes (s'il y a lieu),
- le cadre de réponse (pour faciliter le dépouillement).

Ces représentations, conçues pour analyser des produits, sont relativement pauvres "conceptuellement" et "formellement" parlant pour décrire des systèmes complexes tels que les systèmes d'information. Nous retiendrons cependant l'importance accordée aux relations entre le système et ses utilisateurs et à l'assimilation de ces relations à la notion de chemin fonctionnel. D'une façon plus générale, l'apport de l'AV à la Conception d'Architecture se situe plus dans l'importance accordée à l'évaluation technico-économique des fonctions (et non des composants matériels). La démarche participative est également intéressante mais difficile et surtout coûteuse pour des applications non-reproductibles.

## 1.4. Présentation des tâches retenues

Malgré les différences de vocabulaire et les nuances dans les choix du découpage, on retrouve typiquement dans les méthodes examinées 5 tâches essentielles:

| | MERISE | IDA | SADT | Zachman | AV |
|---|---|---|---|---|---|
| **Analyse du Contexte** | recueil: (observation diagnostic) | identification du projet, étude critique | analyse du contexte | scope description | orientation / information |
| **Structuration Fonctionnelle** | conception | définition du projet cadre | spécification fonctionnelle | model of the business | analyse fonctionnelle |
| **Conception d'Architecture Opérationnelle** | organisation (options opérationnelles) | élaboration de solutions | conception | model of the information system | recherche idées et voies de solutions |
| **Elaboration de Solutions** | conception (solutions types) | élaboration de solutions | Ø | technology model | étude de solutions |
| **Evaluation technico-économique** | appréciation | choix de solutions | Ø | Ø | bilan / choix |

*tableau comparatif des tâches proposées dans différentes méthodes*

A partir des différentes décompositions en tâches étudiées dans les paragraphes précédents, nous avons retenu une décomposition simple basée sur les quatre premières tâches. Nous avons tenté de rapprocher cette décomposition de celle couramment employée dans le développement du logiciel. Ce rapprochement a été retenu afin de ne pas "dépayser" les ingénieurs-concepteurs dont la formation est traditionnellement informaticienne. La décomposition proposée est également appuyée sur le découpage ou plutôt les découpages utilisés par les concepteurs rencontrés. Notre apport a surtout consisté à bien expliciter ces tâches souvent mal dissociées, en particulier la distinction entre l'aspect analyse fonctionnelle et spécification de la solution. Les termes retenus pour la dénomination de ces tâches ont été choisis pour expliciter le plus clairement possible leur contenu et pour éviter les confusions avec les étapes de développement de logiciel. Ce choix de vocabulaire comporte une part d'arbitraire et n'a bien entendu aucune prétention sur la normalisation des appellations. Pour une mise en oeuvre opérationnelle, le plus important sera de retenir des appellations qui réunissent le consensus le plus large de la

part des utilisateurs de la méthode. Le choix devra donc tenir compte des usages au sein de l'entreprise et du vécu des analystes-concepteurs.

Des objectifs et des procédures examinées auprès des ingénieristes, nous proposons la décomposition suivante en précisant le contenu résultant de chaque tâche:

1 - **tâche d'Analyse du Contexte,**
les objectifs, besoins et contraintes tels qu'ils sont exprimés par l'utilisateur;

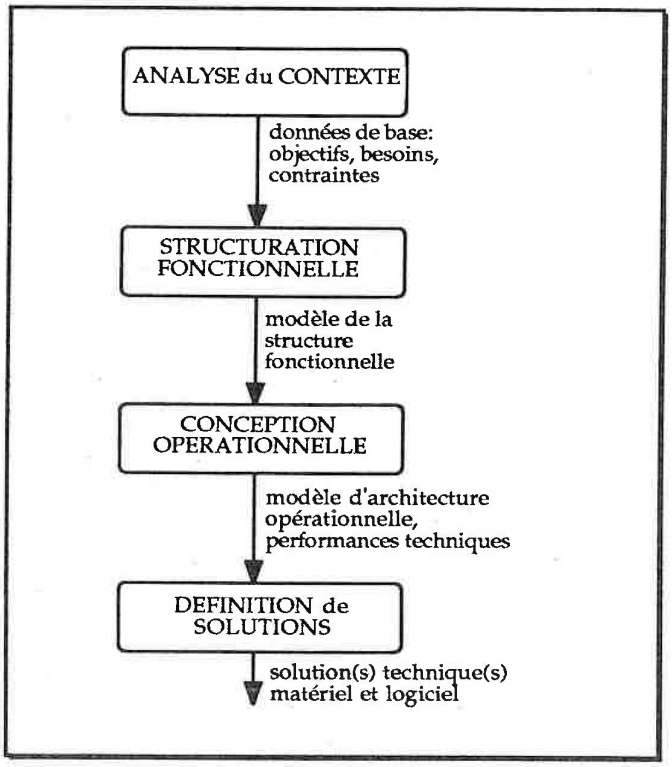
2 - **tâche de Structuration Fonctionnelle,**
les besoins tels qu'ils sont compris par le concepteur;

3 - **tâche de Conception d'Architecture Opérationnelle,**
l'architecture telle qu'elle est élaborée par le concepteur;

4 - **tâche de Définition de Solutions,**
les fournitures et développements prévus pour réaliser le projet;

Les deux dernières tâches du tableau comparatif: Elaboration de Solutions et Evaluation technico-économique, difficilement dissociables dans la pratique, ont été regroupées en une tâche appelée: **Définition de Solutions**. En effet, l'Elaboration de Solutions, déterminant les fournitures et développements prévus pour réaliser le projet ne sont pas faits indépendamment des considérations économiques. L'Evaluation technico-économique établissant le bilan de la rentabilité du projet pour la solution retenue, ne peut être effectuée par les concepteurs que s'ils disposent des éléments économiques nécessaires, ce qui n'est pas toujours le cas. Cet aspect explique le fait que deux des quatre méthodes présentées ne couvrent pas ce point, mais ne doit pas faire négliger son importance.

Précisons dès maintenant que notre contribution porte sur l'analyse du contexte jusqu'à la conception de l'architecture opérationnelle, mais que nous n'avons pas approfondi le choix des solutions matérielles et logicielles. Pour cette dernière partie du problème, nous ne ferons qu'en indiquer les principales directions .




*les différentes tâches de l'étude préalable d'architecture de SAP*

### 1.4.1. Analyse du Contexte

Le but de la tâche d'**Analyse du Contexte** est de préparer l'étude et de fournir l'essentiel des données de base de l'application. Le résultat est un document préparatoire comprenant principalement la description brute du projet par le demandeur. Cette expression sous une forme littérale ni structurée, ni formalisée comprend les objectifs, les fonctions attendues et les contraintes de l'application.

La communication, sous toutes ses formes, entre les prescripteurs-utilisateurs et le concepteur est donc à la base de ce travail. Pour le concepteur, l'objectif est de se familiariser, de s'imprégner du contenu de l'application, des objectifs, des besoins et des contraintes. Une connaissance préalable du métier des interlocuteurs par l'ingénieriste constitue bien sûr un plus très important pour la communication et la compréhension des besoins et contraintes.

#### Informations en entrée

Les informations nécessaires en entrée sont recueillies dans tous les documents pertinents vis à vis de l'étude:

- **documents techniques** d'expression des besoins relatifs:
  - au processus de production,
  - aux conditions d'exploitation et aux postes utilisateurs ou opérateurs,

- aux règles d'organisation et de gestion de la production;
- **schéma directeur** de l'entreprise s'il existe,
- **règlements, conventions** professionnelles, **standards** ou **normes** applicables.

Documents auxquels il faut adjoindre les compte-rendus d'entretiens et de réunions avec les différents partenaires du projet.

**Résultats produits**

Le dossier résultant de cette tâche préparatoire fournit :

- la **définition** du projet,
- les **objectifs** de l'application,
- les **limites** du système étudié,
- la liste des **besoins fonctionnels**,
- les différentes **contraintes** liées à l'application.

### 1.4.2. Structuration Fonctionnelle

Le but de la tâche de **Structuration Fonctionnelle** est de permettre une vérification de la complétude (complétude relative c'est-à-dire vérifier qu'aucune donnée essentielle à la conception d'architecture n'a été négligée) et de la cohérence de l'expression des besoins. Pour cela l'expression brute des besoins est traduite en une représentation fonctionnelle structurée et formalisée, reflétant la compréhension du concepteur sur ce que doit réaliser le futur système. Cette traduction des besoins en fonctions est indépendante de tout choix technique de solution. Le cas échéant, si cette modélisation fait apparaître des ambiguïtés ou des incohérences, des précisions supplémentaires seront demandées aux interlocuteurs concernés.

**Informations en entrée**

Le dossier produit par l'Analyse du Contexte contenant toutes les informations collectées sur l'application.

**Résultats produits**

Un dossier de structuration fonctionnelle cohérente et structurée à la fois synthétique et suffisamment détaillé pour fournir les informations nécessaires à la conception.

### 1.4.3. Conception d'Architecture Opérationnelle

Le but de la tâche de **Conception d'Architecture Opérationnelle** est d'élaborer une architecture capable de supporter les fonctions attendues par les utilisateurs. Le modèle d'Architecture Opérationnelle représente les dispositifs "logiques"

du futur système informatique. Les fournisseurs et leurs différents matériels ou logiciels ne sont pas encore fixés, mais la structure des dispositifs de traitement, de gestion de données et de communication est modélisée. La définition de cette Architecture Opérationnelle doit permettre d'établir les spécifications des performances opérationnelles des futurs composants techniques des solutions choisis dans l'étape suivante. A ce niveau sont introduites les différentes contraintes liées:

- à la répartition des fonctions,
- aux conditions d'exploitation et
- à la sûreté de fonctionnement.

Des choix sont donc effectués en fonction de différents critères issus de l'analyse du contexte.

**Informations en entrée**

Le dossier produit par l'Analyse du Contexte (comprenant en particulier les objectifs exprimés en terme de performances à satisfaire, ainsi que les contraintes liées au choix de conception) et le dossier de Structuration Fonctionnelle.

**Résultats produits**

Le dossier de présentation du modèle de l'Architecture Opérationnelle et des spécifications de performances des futurs composants matériels et logiciels. Ce dossier explicite les décisions prises pour supporter les fonctions en satisfaisant les différentes contraintes imposées.

### 1.4.4. Définition de Solutions

Le but de la tâche de **Définition de Solutions** est d'obtenir une solution technique d'Architecture matérielle et logicielle. A partir de la recherche de produits offerts sur le marché, cette solution est le résultat d'un compromis technique et économique satisfaisant différentes contraintes limitant le choix de fournisseurs, certains critères d'évaluation fixés par le prescripteur, et minimisant la part de développements spécifiques.

**Informations en entrée**

Le dossier d'Architecture Opérationnelle précédemment défini et le dossier d'Analyse du Contexte comprenant en particulier les contraintes liées au choix de solutions.

**Résultats produits**

Le dossier de justification comprenant:

- la description pour chaque solution:
  - description de chaque sous-système: matériel et logiciel, et fonctions supportées,
  - description de chaque système de communication et des échanges supportés
- le chiffrage détaillé des coûts,
- les arguments de justification.

## Conclusion

La phase d'Etude d'Architecture d'un projet d'ingénierie de SAP se situe donc entre les étude de type Schéma Directeur et le développement de l'application. Nous avons proposé un découpage en 4 tâches: Analyse du Contexte, Structuration Fonctionnelle, Conception d'Architecture Opérationnelle, Définition de Solutions. Le but de ce découpage est d'abord d'expliciter clairement les axes d'investigation et les procédures mises en oeuvre dans une étude de conception, de dégager les différents documents et informations nécessaires en entrée et les documents produits en sortie. Dans les chapitres suivants, le détail des activités, procédures et moyens associés aux différentes tâches est analysé.

# 2. ANALYSE DU CONTEXTE

La première tâche d'Analyse du Contexte est primordiale dans le processus de conception puisqu'elle permet d'organiser l'étude d'architecture et d'identifier le problème à résoudre. Ce chapitre met l'accent sur un point très important: la collecte des données de base à partir desquels le concepteur va travailler.

Le premier paragraphe "Description des activités" dresse la liste des points à traiter au cours de cette première tâche. Le second paragraphe "Collecte des données de base" présente une classification détaillée des données à collecter. Le troisième paragraphe évoque quelques "Techniques d'identification et de formulation des besoins" pouvant être mises en oeuvre dans cette tâche.

## 2.1. Description des activités

La première tâche d'Analyse du Contexte a pour but de préciser les objectifs, besoins et contraintes tels qu'ils sont exprimés par l'utilisateur. Cette tâche préliminaire comporte deux activités essentielles:

- la définition et l'organisation de l'étude,
- la collecte des données de base.

### 2.1.1. Définition et organisation de l'étude

La **définition de l'étude** et de ses objectifs (à ne pas confondre avec les objectifs de l'application étudiée) doit définir dès le départ en accord avec le client:

- l'**objet** et l'**étendue de l'étude** doivent être situées par rapport au projet de l'entreprise;
- le **budget** (forfaitaire en général) et les **délais** alloués à l'étude sont fixés en fonction d'un planning global ou d'autres impératifs stratégiques;
- le **niveau d'étude et de remise en cause**, en particulier s'il s'agit d'améliorer une solution existante.

L'**organisation de l'étude** permet:

- de définir le **plan de travail** et le **planning**;
- d'identifier les **interlocuteurs** à interroger, de définir leur fonction et responsabilités dans l'entreprise, ainsi que leur rôle dans le cadre de l'étude.

### 2.1.2. Collecte des données de base

La **collecte des données de base** constitue l'essentiel de l'Analyse du Contexte. Elle est effectuée à travers les documents pertinents vis à vis de l'étude et les réunions ou entretiens avec les différents interlocuteurs. Nous avons déjà évoqué la difficulté d'identifier les données pertinentes dans le processus de conception d'architecture. Afin de faciliter la collecte des informations, un moyen simple de synthétiser l'expérience des concepteurs est d'établir une liste ("check-list") de tous les points potentiellement importants. Cette liste constitue à la fois un guide d'entretien pour le concepteur, et un plan d'étude (ainsi que le plan du document) d'analyse du contexte. La limite d'une telle approche est la difficulté d'être exhaustif. L'obligation de conserver à ce stade une bonne efficacité et de travailler sur un support informel (le langage naturel) incitent à retenir cette solution.

## 2.2. Classification des données de base

Ce paragraphe est le fruit de réflexions entamées au sein d'ESIA sur les études d'avant-projet [ESI 88] et poursuivies avec les industriels "utilisateurs" du groupe SAP de l'EXERA depuis 1987 [SAP 88]. La **classification** suivante recense les éléments qui interviennent sur la définition d'une architecture.

### 2.2.1. Nature des données de base

Une des difficultés de la Conception d'Architecture consiste à identifier et bien sûr prendre en compte parmi la masse de documents et d'informations relatives à une affaire, les données pertinentes ayant une réelle influence sur les choix d'architecture. Le respect des délais généralement très courts affectés à cette phase de conception passe par le repérage de ces données que l'on peut qualifier de "justes nécessaires".

Une seconde difficulté consiste à déterminer précisément à quel niveau ces données doivent intervenir au long du processus de conception. Cette question est particulièrement difficile à résoudre. La classification s'efforce cependant de se baser sur la signification de ces données du point de leur prise en compte. Ainsi nous avons distingué:

- les données liées aux **objectifs** du projet,
- les données liées aux **besoins fonctionnels**,
- les données liées aux **contraintes**.

Ces notions sont utilisées ici sous le sens suivant:

Un projet est mis en oeuvre pour répondre à des "objectifs", qui expriment donc le "pourquoi" de ce projet:

**objectif**: expression du désir des utilisateurs traduisible en termes de critères d'appréciation.

Il est en effet important non-seulement d'exprimer des besoins, mais aussi de se doter des moyens d'apprécier dans quelle mesure ils sont pris en compte. Nous reprendrons la définition du critère d'appréciation proposée dans les normes sur l'Analyse de la Valeur:

**critère d'appréciation**: critère retenu par le demandeur pour apprécier la manière dont une fonction est remplie ou une contrainte respectée.

Pour atteindre ces objectifs, le projet est exprimé sous la forme de "besoins" indépendants des solutions, et qui expriment donc ce que les utilisateurs attendent du projet, le "quoi". Ces besoins seront spécifiés (et formalisés) en termes de "fonctions de service":

**besoin fonctionnel**: expression du désir des utilisateurs traduisible en terme de fonction de service et indépendante des choix de solutions techniques. [AFN 84]

Pour parvenir à une solution, les concepteurs devront intégrer différentes "contraintes":

**contrainte**: limite à la liberté du concepteur dans ses choix de solutions. [AFN 84]

D'un point de vue plus détaillé, de quoi sont constituées les données de base ?

La vocation d'un SAP est de **produire** et donc d'atteindre un certain niveau de productivité et de qualité. Il faut donc rassembler des informations liées :

- au **produit**,
- au **procédé** de fabrication: connaissance des procédures ou gammes de fabrication,
- au modèle de **processus** de fabrication: connaissance des moyens (partie opérative + partie commande) et de leur comportement par rapport au produit).

L'objectif de l'entreprise étant de réaliser des profits, il est nécessaire de **gérer** la production pour assurer un niveau de rentabilité. Il faut donc rassembler des informations liées:

- aux **procédures d'exploitation**: connaissance des règles de pilotage, d'ordonnancement et de gestion de production.

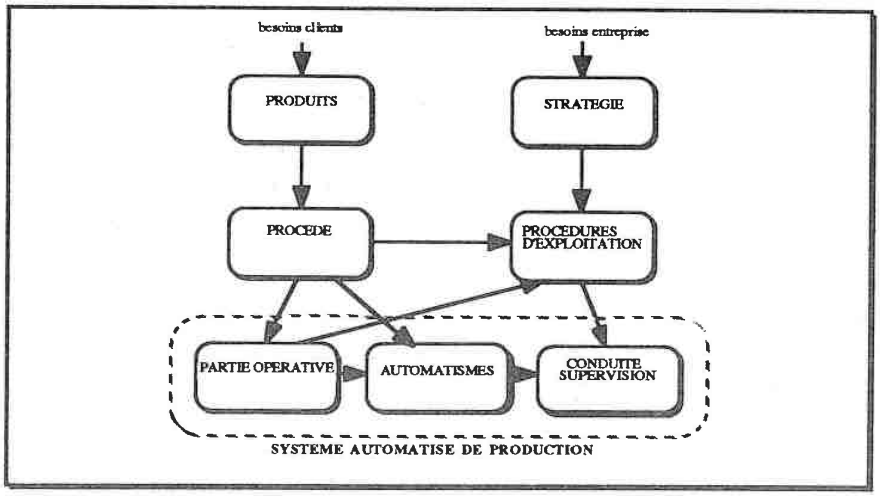
Ces éléments **directement liés au projet** déterminent les objectifs, les fonctions que devra supporter l'application et les contraintes spécifiques qui devront être prises en compte.

D'autres éléments relatifs au **contexte général de l'entreprise** et à l'**application** vont influencer les choix parmi les différentes solutions possibles:

- les **objectifs** et **stratégies**: connaissance des besoins et contraintes de l'entreprise concernant l'application,
- l'**existant** dans tous les cas où le projet doit intégrer tout ou partie d'une installation déjà en place,
- le **métier** de l'automaticien: état de l'art de la technologie informatique et automatique, connaissance en conception et en réalisation d'applications.

Les concepteurs interrogés insistent sur la nécessité d'une approche globale de l'ensemble de ces facteurs pour aboutir à une solution satisfaisante.

Il faut noter que les éléments concernant l'exploitation ainsi que les objectifs et stratégies sont particulièrement mal identifiés et surtout mal formalisés dans le cadre d'études de conception d'architectures.




*les éléments déterminants dans la définition de l'Architecture d'un SAP.*

Nous distinguerons donc les éléments particuliers qui concernent directement les objectifs, ressources et contraintes du projet étudié, des éléments plus généraux relatifs au contexte et à la stratégie de l'entreprise, ou à des contraintes générales de développement de l'application.

### 2.2.2. Classification

#### 2.2.2.1. Objectifs directement liés au projet

Les objectifs de base représentent les **motivations** et les **exigences** concernant le projet. Ils ciblent quelques points particuliers découlant de la stratégie de l'entreprise. Ces objectifs peuvent d'une manière générale être exprimés sous la triple forme de :

**QUALITE, COUTS, DELAIS.**

**Motivations**

A la différence de la stratégie générale, les motivations sont des objectifs traduisant le "pourquoi" ou l'opportunité du projet et exprimés de façon plus précise et mesurables. En fait ces objectifs sont rarement quantifiés et sont plutôt *"généreux"* que précis [Tar 85], car le chiffrage est certes difficile. Il est important d'identifier dès le début d'un projet, et dans la mesure du possible de quantifier ces objectifs. Sans dresser ici une liste exhaustive de tous les objectifs possibles, nous proposons à titre indicatif la liste suivante:

- amélioration de la rentabilité,
- augmentation du taux d'engagement des machines,
- réduction des délais,
- optimisation du coût d'investissement,
- augmentation de la sûreté de fonctionnement *,
- diminution des coûts de main d'oeuvre,
- réduction des stocks et encours,
- optimisation de la consommation matière ou énergie,
- augmentation de la qualité des produits,
- intégration au système de gestion de production,
- amélioration de la qualité de la gestion,
- facilitation de la maintenance,
- augmentation de la flexibilité: possibilité d'effectuer des changements rapides de fabrication,

La plupart de ces objectifs ont été cités par les entreprises utilisatrices de SAP dans l'enquête de l'EXERA [Lej 88].

*Remarque:

La notion de sûreté de fonctionnement sera détaillée dans la troisième partie mais précisons tout de suite qu'elle recouvre celles de fiabilité, de disponibilité, de sécurité et d'intégrité:

**Exigences de qualité**

Il est également important d'expliciter les exigences de qualité qui sont des objectifs qualitatifs traduisant, non pas le "pourquoi" mais plutôt le "comment", c'est-à-dire les caractéristiques de qualité attendues du projet. Ces objectifs sont plus difficilement formalisables ou quantifiables. Citons à titre indicatif (voir sur ce sujet les normes sur la qualité, dont par exemple [GAM 87] ):

- **évolutivité**:
  - possibilité de modifier et d'améliorer le système de contrôle-commande (augmentation du degré d'automatisation par exemple),
  - possibilité d'étendre et adjoindre de nouvelles fonctionnalités,
  - possibilité de mettre en oeuvre la production de produits nouveaux sur une installation existante,
  - possibilité de transférer le logiciel sur d'autres environnements: portabilité;

- **souplesse de mise en oeuvre**:
  - prise en compte et intégration de l'existant,
  - mise au point modulaire et démarrage par tranches,
  - possibilités de tests: testabilité;

- **souplesse d'exploitation**:
  - modes d'exploitation,
  - facilité de maintenance: maintenabilité,
  - documentation,
  - traçabilité
  - etc...

Une fois identifiés, ces objectifs doivent être pondérés en fonction de la priorité accordée par le prescripteur, afin de déterminer les objectifs les plus critiques.

**Ressources affectées au projet**

Les ressources affectées au projet sont bien entendu déterminantes dans le choix de la solution opérationnelle. Cela recouvre l'ensemble des **moyens humains et financiers** affectés au projet.

Si les ressources ne constituent pas un objectif, nous avons choisi de les placer à ce niveau dans la mesure où, au final, toute réalisation d'un objectif sera évaluée en fonction des moyens affectés. Le fait de savoir si les ressources constituent des objectifs ou des contraintes n'apporte pas beaucoup à la résolution du problème.

Le **planning** d'affectation de ces moyens, les délais de mise en oeuvre constituent aussi des contraintes que le concepteur doit connaître.

#### 2.2.2.2. Contexte et contraintes

Dans cette catégorie interviennent d'une part des contraintes générales liées au contexte de l'entreprise où se situera le futur système automatisé, et d'autre part des contraintes spécifiques directement liées au projet.

**Politique stratégique de l'entreprise**

Différentes contraintes ou objectifs non-techniques, du moins directement, peuvent largement influencer les choix d'architecture. Ces informations sont en partie contenues dans le Schéma Directeur de l'entreprise quand il existe.

- volonté d'indépendance vis à vis des fournisseurs, ou au contraire,
- accords privilégiés avec certains fournisseurs,
- homogénéisation du parc (pour faciliter la maintenance, la formation, ...)
- contraintes d'exportation,
- disponibilité du service après-vente fournisseur,
- image de marque par rapport aux concurrents, aux clients ou au personnel (motivation),
- stratégie technologique innovatrice ou conservatrice,
- etc...

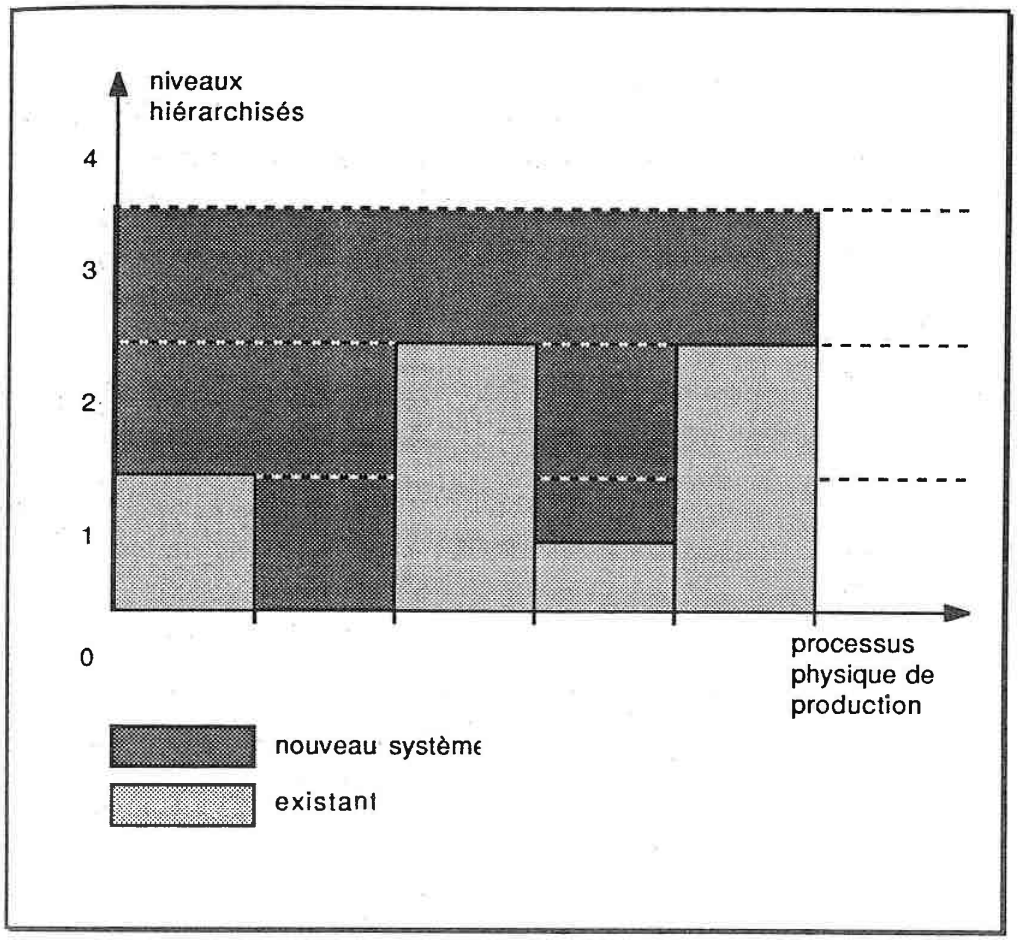
**Contraintes de l'existant**

La prise en compte et l'intégration de l'**existant** (c'est-à-dire les installations d'automatisation existantes) doit être appréhendée dès le début de l'étude. Il constitue généralement une des principales sources de contraintes, sauf pour les rares applications complètement nouvelles. D'un point de vue architecture, il est primordial de bien identifier:

- les niveaux de services fonctionnels assurés par les équipements existants,
- les flux d'informations à assurer entre les équipements existants et les futurs équipements,
- les protocoles utilisés par ces équipements pour garantir la compatibilité des futurs équipements.

Si on reprend le schéma hiérarchisé traditionnel, l'existant est représenté par la "surface fonctionnelle" correspondant aux équipements des sous-systèmes du processus

déjà automatisés et le niveau de service fourni par les fonctions des équipements d'automatisation:




*la place de l'existant dans l'architecture du SAP*

On peut considérer que la conception d'un système automatisé de production à partir d'installations et d'équipements existants consiste à homogénéiser les niveaux de service.

### Contraintes réglementaires

Cette rubrique recense différentes contraintes générales, dont le cadre dépasse celui du projet, mais qu'il est nécessaire de prendre en compte:

- règlements internes,
- réglementation et conventions professionnelles,
- normes de sécurité: la sécurité concerne les personnes, les installations et l'environnement,
- etc...

**Compétences au sein de l'entreprise**

L'état de ces compétences au sein de l'entreprise ainsi que la politique de formation du personnel influencent notamment le degré de technicité des solutions technologiques retenues.

Les compétences de l'entreprise peuvent être classifiées suivant deux axes:

| par DOMAINE: | par PHASE: | | | | |
|---|---|---|---|---|---|
| | conception | réalisation | mise en service | maintenance | exploitation |
| procédé | | | | | |
| processus | | | | | |
| exploitation | | | | | |
| automatismes | | | | | |
| informatique | | | | | |

**Contraintes spécifiques diverses**

Les **contraintes d'environnement**:

- poussière,
- humidité,
- températures élevées,
- projections d'huiles ou de copeaux,
- parasites électro-magnétiques,
- etc...,

peuvent conduire au choix de matériels renforcés ou à l'emploi de dispositifs de protection.

La **sécurité du réseau** au sens de:

- la confidentialité (accès aux données),
- la protection (sauvegarde),
- la responsabilité de la validité des données.

### 2.2.2.3. Besoins fonctionnels

De nombreuses informations pertinentes dans la conception de l'architecture sont issus des domaines suivants. Elles vont permettre de définir plus précisément les fonctions attendues du SAP.

**Produit et Procédé de fabrication**

La compréhension globale du procédé de fabrication au sens recette ou gamme de fabrication d'un produit permet d'identifier:

- les caractéristiques dynamiques du système de production,
- les relations complexes de couplage entre les principales variables du procédé,
- une partie des informations à recueillir ou à fournir.

Les **interlocuteurs**: concepteur procédé (en fonction du métier chimiste, thermicien , etc...), peuvent fournir les informations suivantes:

- **conditions de fonctionnement**:

description des opérations et transformations élémentaires effectuées sur le produit,
enchaînements des opérations,
paramètres, limites (seuils) de tolérance,
informations en retour demandées par les concepteurs procédé pour l'amélioration de la connaissance du procédé.

- **caractéristiques des produits**:

les mesures effectuées sur les produits,
la description des défauts à éviter,
les limites d'âge (produits périssables).

Ces mesures peuvent être nécessaire au contrôle-commande, mais aussi pour d'autres fonctions comme le contrôle-qualité ou le tri en différentes catégories (ou "1$^{er}$ ou 2$^{ème}$ choix").

Rq: pour l'automaticien, le fournisseur des machines et le concepteur des produits peuvent aussi fournir des informations liées au procédé.

**Processus de production**

Le processus de fabrication, défini comme l'ensemble des machines ou constituants de partie opérative réalisant les opérations de transformation des produits, est également un aspect primordial en ce qui concerne la structuration de l'application.

Les mécaniciens (concepteurs ou responsables machines) fournissent les informations nécessaires à l'automatisation du processus:

- comportement physique de la **partie opérative** (inertie mécanique par exemple),
- caractéristique techniques et implantation des **capteurs et actionneurs,**
- **lois** de commande, asservissements et régulation,
- **contraintes de temps**, temps de cycle et cadences de fabrication
- protections et **sécurités,**
- **modes de défaillance,**
- **modularité** des équipements (découpage fonctionnel et répartition des automatismes par rapport aux équipements),
- problèmes d'**implantation**: distances, environnement.

A ce niveau, le processus de production sera modélisé selon le formalisme présenté dans le chapitre sur la "formalisation des connaissances" par l'analyste-concepteur. Cette modélisation des flux de production et des différents "opérateurs du processus" permet en effet d'obtenir rapidement une bonne connaissance de la structure et de l'organisation du processus. Nous avons vu au chapitre consacré aux "Concepts utilisés en ingénierie" que cette structure du processus avait une grande influence sur la structure du système d'information.

### Conditions d'Exploitation

Les conditions d'exploitation sont définies de façon très spécifique à chaque métier. La notion de mode d'exploitation est donc d'une façon générale mal définie. Un travail important reste à faire sur ce sujet. On trouvera une première contribution à la formalisation de ce concept dans [Ver 89]. Les paragraphes suivants présentent un aperçu des modes utilisés dans différents domaines manufacturier ou continu.

### Modes de transition: Mise en service, démarrage et arrêt

Les modes de **mise en service**, de **démarrage**, et d'**arrêt** concernent les différentes possibilités de faire passer le processus de production de l'état arrêt à un état de production donné ou réciproquement.

Les informations précisant ces transitions sont par exemple:

- périodicité de mise sous-tension / hors tension liées aux habitudes de l'usine,
- opérations préparatoires au démarrage (par exemple approvisionnement de composants, mise à température) et mode de lancement de ces opérations (automatique ou manuel),
- mise en service centralisée ou modulaire,
- démarrage en automatique centralisé ou modulaire,

- modes de configuration des moyens automatique ou manuelle, générale ou progressive par poste,
- procédures de validation automatiques ou manuelles.

**Modes de production**

Les modes de **production** concernent les différentes possibilités de faire fonctionner et de piloter le processus de production en fonction de son état pour fabriquer les produits de façon conforme aux objectifs de qualité ou de productivité.

Les modes sont généralement divisés en modes "normaux" et en modes "dégradés":
- modes **normaux**:
  - automatique continu, par cycle (par produit ou série),
  - vidage automatique ou commandé des machines,
- modes **dégradés**:
  - retouche,
  - "passe travers" ou "marche forcée" (manutention seule),
  - sans chargement / déchargement automatique,
  - sans contrôle automatique des produits,
  - etc...

**Modes d'intervention**

Les modes d'**intervention** concernent les différentes possibilités d'intervenir sur le processus de production pour modifier son état: interventions de maintenance ou d'entretien. Donc dans ces modes, le processus n'est pas utilisé pour fabriquer les produits de façon conforme aux objectifs de qualité ou de productivité.

On trouve donc à ce niveau des modes tels que:
- modes **manuels** ou modes opérateurs,
- modes de **réglage** des machines,

Il est important de préciser si ces modes manuels ou semi-automatiques comportent ou non des sécurités.

**Dialogue homme / machine**

Les dialogues entre les hommes et le système informatique sont d'abord définis de façon globale par la **définition des différents postes** des opérateurs.

Au niveau fonctionnel, il s'agit de spécifier le **contenu du dialogue**, c'est-à-dire quelles sont les informations échangées.

La relation de ces postes avec le système d'information doit également être précisée en indiquant:

- le sous-ensemble du processus concerné (sous la "responsabilité" de ce poste),
- la ou les fonctionnalités correspondantes.

Ces informations doivent permettre de situer le poste par rapport à l'architecture globale du système.

La localisation de ces postes doit être précisée géographiquement par rapport aux installations:

- postes placés à proximité des équipements de production,
- postes de conduite centralisée,
- ou tout autre disposition.

A un niveau plus détaillé de définition, le **type** de poste et son ergonomie sont parfois précisés. Par exemple:

- pupitre à voyants et poussoirs,
- terminal,
- micro-ordinateur,
- etc...

Ces précisions sont d'ordre opérationnel, c'est-à-dire conduisent à définir le type de solution retenue pour supporter le dialogue homme-machine. Ceci constitue une contrainte à la liberté du concepteur souvent imposée pour standardiser les postes de conduite au sein de l'entreprise.

### Fonctions d'exploitation

Si la fonction essentielle du système d'information d'un SAP est la planification et la conduite de la production, de nombreuses fonctions complémentaires sont nécessaires ou permettent d'améliorer les performances. Généralement la présentation des fonctions attendues par le client est informelle. Pendant l'analyse de du contexte le concepteur doit s'assurer qu'aucune fonction importante n'a pas été omise. Le cadre des quatre fonctions présentées et définies au chapitre des concepts utilisés en ingénierie: CONDUITE, SUIVI, MAINTENANCE, SECURITE fournit quatre axes d'investigation pour vérifier la complétude fonctionnelle globale du SAP. La structuration de ces fonctionnalités de base sera effectuée au cours de la tâche suivante de structuration fonctionnelle.

## 2.3. Techniques d'identification et de formulation des besoins

Dans certains cas la définition des besoins par le client peut ne pas être suffisamment avancée. L'ingénieriste doit alors participer à cette définition en jouant en quelque sorte le rôle de révélateur de ces spécifications (l'expérience montre que cela ne se fait pas toujours sans douleur). Le paragraphe suivant indique quelques éléments bibliographiques sur les techniques d'identification et de formulations des besoins.

Différentes approches pouvant être suivies à ce stade ont été étudiées par Davis [Dav 82] dont nous résumons ici les éléments essentiels. Ces travaux portent sur la définition d'une méthode de **choix de stratégie** (approche pour atteindre un objectif) de détermination des "information requirements" en fonction du **degré d'incertitude** de l'application. La difficulté d'obtenir des spécifications correctes et complètes est due à 3 facteurs essentiels: les limites humaines, la variété et la complexité des systèmes, et la difficulté des relations utilisateurs-analystes.

Les limitations humaines et les comportements (ajustement par rapport à un point de référence, influence de l'existant, et des événements récents, mauvais sens des statistiques) montrent que l'interview des utilisateurs ne suffit pas à définir les spécifications. Une méthodologie au sens ensemble de méthodes (procédure ordonnée ou systématique) et de techniques doit donc permettre de :

- faciliter la structuration et l'analyse du "problem space" (la représentation du problème par l'homme par opposition au problème lui-même: "task environment"),
- pallier les limitations humaines et prouver la correction et la complétude.

L'auteur note que les structures conceptuelles étudiées dans les différentes méthodes sont beaucoup plus détaillées que les procédures et la documentation.

Quatre **stratégies** de spécification sont détaillées:

1: enquête utilisateurs,

2: analogie (dérivation de systèmes existants),

3: synthèse des caractéristiques du système considéré,

4: prototypage.

Le choix parmi ces stratégies passe par la détermination des caractéristiques affectant le degré d'incertitude aux niveaux organisation globale et application détaillée:

- stabilité et déterminisme des activités et décisions du système à informatiser,
- simplicité,
- innovation du système informatique,
- nombre et expérience des utilisateurs dans l'utilisation et la planification de systèmes et d'applications informatiques,

- formation et expérience des analystes.

Le degré d'incertitude est donc évalué par rapport à l'existence et à la stabilité d'un ensemble de spécifications, la capacité des utilisateurs à spécifier leurs besoins, et la capacité des analystes à expliciter les spécifications et à évaluer la correction et la complétude.

Un certain nombre de techniques de collecte d'informations ("facts gathering techniques") sont connues et pratiquées depuis de nombreuses années [Bin 78]. Parmi ces techniques on retiendra principalement:

- entretien ou interview,
- étude de documents,
- observation,
- questionnaires.

Nous invitons le lecteur intéressé à approfondir les références citées.

## CONCLUSION

La tâche d'Analyse du Contexte comporte deux activités essentielles d'une part de définition et d'organisation de l'étude, et d'autre part de collecte des données de base. Cette collecte doit permettre au concepteur de se familiariser avec l'application dans son contexte et de récupérer toutes les informations nécessaires à l'étude d'architecture. Ces données exprimées par le client n'étant traditionnellement pas formalisées, nous avons choisi de proposer comme support de la collecte des données de base, une classification sous forme de "check-list". L'établissement de cette liste est le fruit du recueil d'expériences auprès de concepteurs.

# 3. STRUCTURATION FONCTIONNELLE

La tâche de Structuration Fonctionnelle constitue un apport important par rapport à l'approche empirique couramment pratiquée. Si l'introduction d'une nouvelle étape ne pose pas de problème sur le principe, cela soulève des difficultés pratiques de mise en oeuvre pour préserver la productivité des concepteurs. Nous avons déjà souligné dans la première partie l'importance de ne pas surcharger la tâche des concepteurs. Ce point doit toujours être gardé à l'esprit dans ce chapitre.

La Structure Fonctionnelle matérialise le résultat de la communication entre le client et le concepteur. *"La qualité de la communication entre deux individus repose le plus souvent sur la compréhension réciproque du modèle interne de l'autre"* [Boy 88]. Cela signifie pouvoir vérifier la bonne compréhension du problème posé par le client avant d'aborder la conception proprement dite et la bonne délimitation précise du contenu du projet.

Le détail de la procédure méthodique et de l'utilisation du modèle fonctionnel retenu pour la structuration est présenté dans le premier paragraphe. Le second paragraphe "Validation de la structure fonctionnelle" analyse la difficulté de s'assurer de la bonne compréhension du problème par le concepteur. Ce paragraphe évoque les deux approches pouvant être suivies pour passer d'une expression "brute" informelle à une expression structurée formelle: une approche "manuelle" ou une approche "automatisée" ou du moins systématique.

## 3.1. Description des activités

Ce paragraphe détaille les différentes activités de spécification de la structure fonctionnelle permettant de définir progressivement tous les éléments jugés pertinents du modèle de l'application.

La Structuration Fonctionnelle consiste à :

- identifier les fonctions du système,
- organiser la structure de ces fonctions,
- spécifier le détail des interactions fonctionnelles à travers les données manipulées et les échanges d'informations,
- spécifier les interfaces avec les utilisateurs,
- quantifier les échanges d'informations (débits et contraintes de temps).

Cette tâche est en pratique réalisée par l'instanciation du modèle générique de Structure Fonctionnelle en un modèle particulier ou spécifique.

### 3.1.1. Définition des interfaces externes

Afin de définir clairement les frontières de l'application, les interfaces externes du système étudié avec les autres systèmes de l'entreprise doivent être spécifiés. On différencie les différentes liaisons définissant les services des différentes fonctionnalités de base du SAP des liaisons de partage de données.

### 3.1.2. Définition des fonctions de base

A partir des fonctions identifiées dans l'Analyse du Contexte, il s'agit de définir les fonctions principales de l'application. A ce niveau les fonctions doivent être abordées dans leur globalité et non dans le détail. Ces grands "blocs fonctionnels" sont définis de façon indépendante et ne sont pas en relation hiérarchisée les uns par rapport aux autres. Ceci permet de définir dans une plus grande cohérence chacune des fonctions car le concepteur sera obligé de s'interroger sur ce que recouvre chaque fonction à tous les niveaux hiérarchiques et pour chaque sous-ensemble du processus de production.

A titre indicatif sont proposées les quatre fonctions de base générique présentées au paragraphe 3 du chapitre 5 sur les "concepts utilisés en ingénierie".

**conduite,**
**suivi,**
**maintenance,**
**sécurité.**

Il conviendra de les définir de façon précise en fonction des choix d'organisation du SAP particulier étudié. Ces fonctions étant précisées, elles sont ensuite décomposées en différents niveaux hiérarchisés. Pour chaque fonction, le service global attendu est défini en conformité avec les liaisons externes définies précédemment.

### 3.1.3. Définition des services des entités fonctionnelles

Les fonctions de base définies précédemment sont décomposées selon une procédure descendante itérative en services et sous-services. Le service d'une entité fonctionnelle est défini à chaque niveau par:

- la fonction principale considérée,
- son contexte ou "champ d'application" ("scope") définissant les sous-ensembles du processus concernés par ce service,
- son horizon et sa période temporelle (au sens GRAI: intervalle de validité et intervalle de révision),
- une description littérale de ce qui est attendu par le client,
- les services de niveau inférieur utilisés.

exemples:

| | |
|---|---|
| fonction | conduite |
| contexte | chaîne d'assemblage |
| horizon | journée |
| période | heure |
| description | .... |
| services utilisés | conduite manutention pièces détachées et conduite machines assemblage |

| | |
|---|---|
| fonction | maintenance |
| contexte | ligne numéro 3 |
| horizon | immédiate |
| période | immédiate |
| description | maintenance curative par détection d'incidents |
| services utilisés | .... |

Les services sont définis aux différents niveaux correspondant à la décomposition du processus de production dont notamment :

- le niveau **système** global comprenant l'ensemble du processus,
- le niveau **domaine** pour chaque flux principal de ce processus,
- le niveau **cellule** ou ligne marquant les principales opérations de transformation du produit,
- le niveau **opérateur** pour les opérations de base sur les flux de produits,
- le niveau **équipement** permettant de réaliser ces opérations,
- le niveau des **capteur / actionneur**.

(pour plus de précisions, voir le chapitre sur les concepts utilisés en ingénierie)

Ces niveaux sont donnés à titre de repère indicatif, des niveaux intermédiaires peuvent être définis pour les besoins de chaque étude. La décomposition des services est poursuivie jusqu'à ce que le concepteur estime qu'elle est suffisante à la définition de l'architecture. En pratique cela correspond au niveau de détail à partir duquel les entités fonctionnelles ainsi définies ne sont plus réparties et / ou peuvent être chiffrées.

### 3.1.4. Procédure de définition des services

L'approche suivie pour définir chacun de ces niveaux pourra être mixte, ascendante et descendante. Si une approche descendante est plus adaptée à une démarche systémique globale, la nécessité pour le concepteur de se familiariser avec les particularités du processus de production et ses automatismes peut suffire à justifier une approche ascendante commençant par la définition des entités fonctionnelles de bas niveau. Cette approche devra bien entendu être consolidée par une analyse globale regroupant toutes les entités en un modèle global, permettant donc de vérifier la cohérence de tous les composants définis séparément.

Comme nous l'avons déjà évoqué dans le chapitre sur les concepts utilisés en ingénierie et dans la présentation du modèle fonctionnel, il est possible de distinguer deux niveaux de spécification fonctionnelle. Un premier niveau permet de proposer une structure théorique basée sur la structure du processus de production et la décomposition des fonctions principales de l'application en entités hiérarchisées. Cette première représentation (proche d'une représentation de type SADT) ne modélise que les entités fonctionnelles de chaque fonction principale, les liaisons de service et de partage de données (seulement entre les fonctions principales).

C'est seulement dans la modélisation fonctionnelle de deuxième niveau (proche d'une représentation de type SA/RT), que cette structure est traduite en structure fonctionnelle de type informatique explicitant tous les différents échanges d'informations et les structures de stockage de données. Cette modélisation de deuxième niveau fait apparaître la structure de contrôle réelle de la future architecture.

Pour des concepteurs familiarisés avec l'application (présence d'installations existantes ou connaissances de réalisations antérieures), la structure fonctionnelle analysée est en général directement effectuée au niveau 2.

### 3.1.5. Définition des échanges d'information

Les entités fonctionnelles du système étant définies, il faut préciser pour chacune d'elle ses différents échanges d'information. A chaque service correspond la définition d'un couple requête-compte-rendu. On définit ensuite les liaisons de données partagées et de synchronisation. Ces différents échanges d'information sont également quantifiés en vue de l'étude du dimensionnement des systèmes de communication de l'architecture opérationnelle. (Cet aspect dimensionnement des systèmes de communication est détaillé dans le chapitre suivant sur la Conception de l'Architecture Opérationnelle.)

### Définition des requêtes et compte-rendus

Les services définis précédemment doivent être détaillés pour préciser quels paramètres sont échangés en requêtes (RQ) et compte-rendus (CR). La requête fournit les paramètres définissant le service attendu de l'entité destinataire. Le compte-rendu fournit les paramètres indiquant la bonne ou mauvaise exécution du service attendu.

Les paramètres de requête et compte-rendu doivent être spécifiés par :

- le nom du ou des messages,
- les données contenues dans ces messages,
- l'origine du message,
- le destinataire.

### Définition des structures de stockage de données et des liaisons de données partagées

Chaque entité fonctionnelle est capable de partager, c'est-à-dire mettre à disposition des autres entités, différentes données mémorisées, collectées ou des données calculées à partir des données collectées. Inversement chaque entité fonctionnelle peut nécessiter un accès à certaines informations disponibles chez d'autres entités. Toute entité fonctionnelle est donc productrice et consommatrice de données qu'il est nécessaire de spécifier. Les liaisons de partage de données correspondent donc à des primitives d'écriture ou de lecture vis à vis de structure de stockage de données. Comme nous l'avons déjà évoqué, ces liaisons de données partagées seront facilement différenciées des liaisons de service en examinant:

- si ces données définissent le service attendu de la fonction par son utilisateur,
- si le producteur de la donnée est intéressé par l'utilisation qui est faite de la donnée pour nécessiter le retour d'un compte-rendu.

Une entité fonctionnelle ne peut partager des données qu'avec des entités de même niveau appartenant à une fonction de base différente. Cette limitation permet de tirer le meilleur parti de l'organisation hiérarchisée en niveaux en évitant d'aboutir à une structure "maillée" où la disponibilité devient difficile à assurer.

A ce niveau de l'étude, la définition des besoins en partage de données de chaque entité fonctionnelle peut être présentée sous la forme d'accès en écriture ou lecture à une base de donnée virtuelle unique et hiérarchisée. En effet pour chaque entité fonctionnelle, il est possible de spécifier:

- quelles sont les données consommées sans préciser leur origine,
- quelles sont les données produites sans préciser leur destination.

Une validation de la cohérence peut ensuite être effectuée au niveau global de cette base de données virtuelle (BdDV) afin d'assurer que toutes les données consommées possèdent un seul producteur et que toutes les données produites possèdent au moins un consommateur. Cette BdDV est hiérarchisée: à chaque niveau d'entité fonctionnelle correspond un niveau dans la BdDV.

**Définition des synchronisations**

Dans de nombreux cas, il est nécessaire de différencier la définition des paramètres de requête de service et la définition du signal de synchronisation déclenchant l'exécution de la fonction de traitement de l'information. De tels signaux déclenchant une entité fonctionnelle donnée peuvent provenir de l'entité utilisatrice émettant la requête, des entités de niveau inférieur d'entités de même niveau. Ces signaux présentent typiquement des contraintes de temps élevées. Leur spécification est donc importante pour le repérage des couplages inter-entités fonctionnelles et la détermination des performances en temps de transfert requises des systèmes de communication.

## 3.2. Validation de la structure fonctionnelle

Comme nous l'avons vu dans le chapitre précédent, nous partons de l'hypothèse où le client a exprimé ses besoins en langage naturel, avec toutes les ambiguïtés que cela peut comporter. La solution à terme est d'obtenir directement et automatiquement une expression formelle de ces besoins. Cependant le but de ce travail étant d'apporter une contribution à la situation actuelle de l'ingénierie, explique le choix de cette hypothèse.

Deux approches principales peuvent être suivies pour passer d'une expression "brute" informelle à une expression structurée formelle. La première approche, traditionnelle, consiste pour le concepteur à traduire les besoins en un modèle dont les éléments permettent de formaliser les concepts du domaine étudié. Une deuxième approche, faisant actuellement l'objet de recherches, consiste à automatiser cette traduction en passant le plus rapidement possible par des modèles ou langages formels intermédiaires.

Nous retiendrons les définitions proposées dans [Aye 88] :

**validité** d'un modèle: définie comme la plus ou moins grande adéquation entre le modèle et la réalité modélisée.

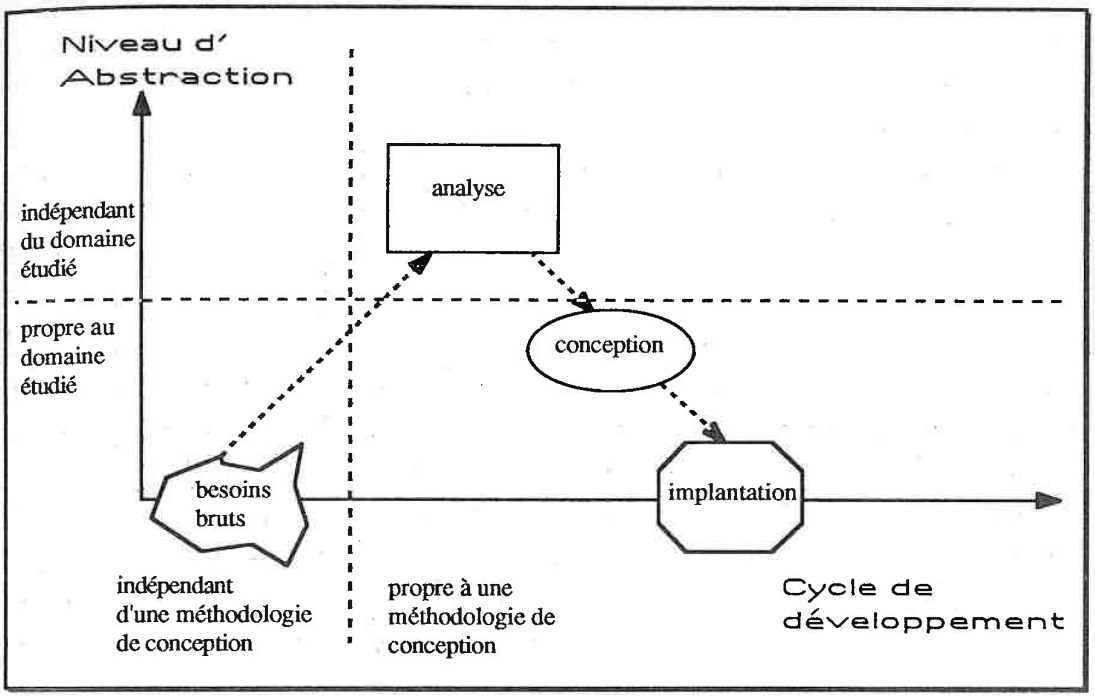
**cohérence** d'un modèle: définie comme la plus ou moins grande adéquation entre le modèle et des contraintes, ou des règles de fonctionnement, elles mêmes formalisées.

Remarque: *La cohérence d'un programme classique est la satisfaction des spécifications alors que la validité est la correspondance entre les spécifications et la réalité.*

### 3.2.1. Démarche traditionnelle

La plupart des méthodes d'analyse fonctionnelle reposent sur l'utilisation d'un formalisme donné pour décrire un (ou plusieurs) point(s) de vue particulier(s). Un certain nombre de concepts correspondant à ces points de vue sont à la disposition de l'analyste. L'analyste effectue donc lui-même la traduction entre l'expression en langage naturel des besoins et la représentation en langage formel des fonctions de l'application. Dans les méthodes SA ou SADT par exemple, l'expression littérale des besoins est convertie par l'analyste en une représentation fonctionnelle utilisant respectivement les notations de Diagrammes de Flux de Données ou des Actigrammes.

La difficulté est de s'assurer de la validité et de la cohérence de cette représentation. Il y a en effet un fossé important entre ces deux formes d'expression:




*la démarche courante d'analyse fonctionnelle*

Dans la mesure où cette traduction est "manuelle" et peu systématisée, la validité peut difficilement être contrôlée. La qualité de cette traduction repose essentiellement sur le savoir-faire de l'analyste. Ce constat étant jugé insatisfaisant, les méthodes font reposer la validité sur des règles de cohérence [Aye 88]:

en se basant sur le fait que:

'si le modèle est valide, alors il est cohérent',

on utilise la contraposée:

'tant que le modèle n'est pas cohérent, il ne sera pas valide'.

Cette première approche est donc intrinsèquement limitée, mais apporte malgré tout un progrès d'autant plus important que la vérification de la cohérence peut être assistée par un outil informatique. Remarquons que la validité est d'autant plus difficile à établir que le rôle déonthologique de l'architecte est non seulement de convaincre que les besoins sont pris en compte, mais aussi de les compléter par toutes les fonctions nécessaires mais non-exprimées car "évidentes" ou "oubliées".

Pour améliorer cette situation, les programmes d'assurance qualité proposent d'introduire des jalons de validation (ou points de contrôle) contractuels entre le

demandeur et le concepteur-réalisateur. Une procédure simple consiste à faire valider la représentation formelle par le client. Cette approche suppose que ce même client soit en mesure de comprendre et d'interpréter le formalisme utilisé afin de se rendre compte si ses besoins ont été compris. Sans qu'il soit bien entendu question de douter des capacités intellectuelles de ces interlocuteurs, ces difficultés sont réelles et aussi compréhensibles. Cette situation correspond à la position d'un particulier devant le plan que lui propose un architecte pour réaliser la maison de ses rêves. Certaines subtilités peuvent échapper au client car les notations ne lui sont pas forcément familières. Le concepteur aura alors beaucoup de mal à faire valider sa proposition sans un travail de "formation" important.

Notre expérience nous a montré que certains concepts évidents pour les spécialistes étaient très abstraits pour des personnes n'ayant pas été confrontées directement à un problème de conception. Il est donc très délicat de constituer un modèle de représentation qui soit à la fois suffisament riche pour être utile au concepteur (c'est-à-dire comportant des concepts de base propres à son métier de concepteur) et suffisament simple pour être un outil de communication avec son client (ces concepts de base risquant d'être trop abstraits).

### 3.2.2. Spécification systématique

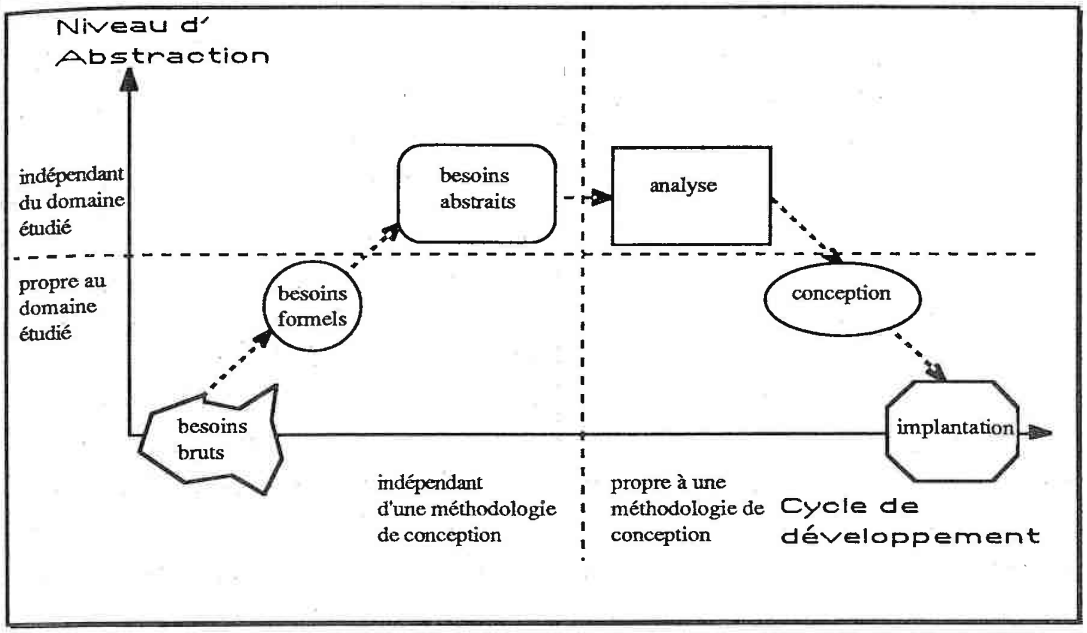
Une autre approche consiste à combler ce "fossé" entre l'expression brute des besoins par le client et la représentation formelle des fonctions du système par des représentations intermédiaires. Chaque terme présent dans l'expression brute littérale des besoins est repris dans un formalisme de spécification des besoins. Si la traduction automatique du langage naturel en spécifications formelles n'est pas encore envisageable, cette démarche consiste à introduire un ou plusieurs niveaux d'expression intermédiaires:

1 - un niveau plus proche du langage naturel reproduisant les concepts du discours de l'utilisateur propres au domaine étudié,

2 - un niveau indépendant du domaine étudié utilisant des concepts très généraux,

3 - un niveau basé sur les concepts d'une technique de modélisation propre à une méthodologie particulière.

L'introduction de ces niveaux intermédiaires permet d'effectuer des contrôles de cohérence "à la source" de l'expression des besoins, ou du moins de s'en rapprocher.




*la démarche de spécification assistée*

De nombreux travaux portant sur la définition de modèles conceptuels ou de formalismes de représentation des connaissances correspondent à cette approche. Un nombre restreint propose cependant l'utilisation du premier niveau permettant de modéliser le point de vue du client. Les travaux de Scott récemment publiés illustrent bien cette approche partant de la traduction directe du langage naturel. D'autres travaux sont plutôt orientés vers l'adaptation de modèles spécifiques en modèles plus généraux (c'est-à-dire au niveau 2, basés sur des concepts très généraux, indépendants du domaine étudié et de la méthode de conception) pour l'analyse de système: Shemer propose ainsi l'utilisation d'un réseau sémantique particulier.

#### 3.2.2.1. Analyse des besoins assistée par modélisation logique

référence: [Sco 88]

Scott propose l'utilisation d'un langage de programmation logique pour la transformation systématique des documents de spécification des besoins (exprimés en langage naturel) en une spécification basée sur le modèle d'une méthode donnée. L'application présentée concerne le domaine des systèmes de communication militaires. L'approche proposée comporte 4 étapes:

Une traduction manuelle ("**symbolize**") des phrases en clauses Prolog, utilise des prédicats prédéfinis, pour symboliser les principaux concepts spécifiques du domaine considéré: *"système, interface, console, fonction, réception de données ou de messages,*

*contenu de message, transmission de données ou de messages"*. Le résultat est appelé "canonical requirements facts".

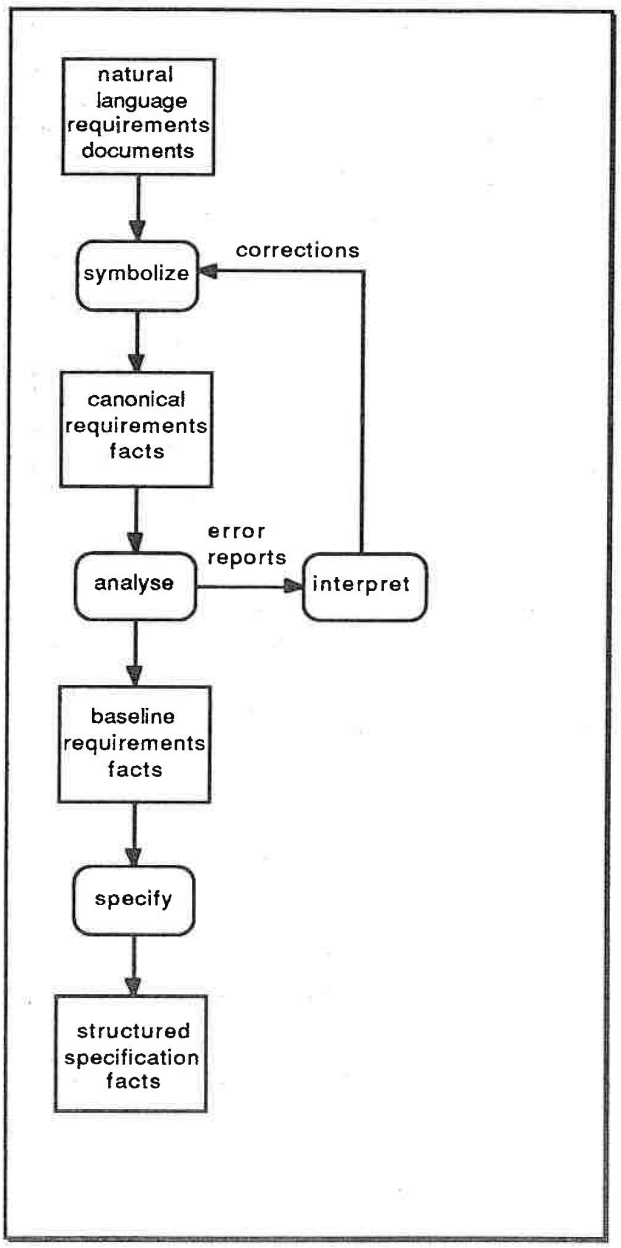
La deuxième étape ("**analyse**") analyse les concepts formels du domaine précédemment définis pour:

- d'une part, découvrir ("**interprete**") les cas d'incohérence (ex: fonction appartenant à une entité qui n'est ni le système, ni une autre fonction) et d'incomplétude (ex: fonction ne recevant pas de données), et générer des rapports d'erreur correspondant,
- et d'autre part, convertir les concepts spécifiques du domaine en concepts généraux indépendants du domaine: *"entrée, sortie, niveau, groupe"*.

Ces concepts généraux sont de plus indépendants du modèle de la méthode choisie.

Une dernière étape ("**specify**") transforme ces concepts généraux en concepts particuliers propres à une méthodologie particulière, en l'occurrence ceux d'un modèle de flux de données (Yourdon-De Marco): *"diagramme, membre, externe, process, dataflow"*. (La transformation du textuel au graphique n'est pas encore réalisée automatiquement.)

La "traçabilité" est assurée par la consignation systématique des références au document de base dans toutes les clauses.




*les étapes principales de la méthode d'analyse des besoins de Scott*

exemple de traduction d'une phrase de spécification:

expression en langage naturel:
§3.7.2.1.The communications functions shall handle the reception of messages containing information on events from sensors....

faits canoniques correspondants:
interface (sensor, environment, Ø)
receive-messages (communications, sensor, §3.7.2.1.a)
message-content (sensor, events, §3.7.2.1.b)

besoins de base:
in (communications, sensor, §3.7.2.1.a)

spécification structurée:
external (sensor, diagram context, reference environment)
dataflow (events, sensor, communications, §3.7.2.1.a)

### 3.2.2.2. Système d'aide à l'analyse de systèmes

référence: [She 87]

L'auteur étudie la phase importante de "requirements specification" du cycle de vie du logiciel en soulignant que si de nombreuses études sont parues sur le sujet, elles portent essentiellement sur le formalisme des documents, plutôt que sur le processus d'élaboration de ces documents. Il est en effet important de faire la différence entre le contenu du document et la façon d'aboutir à sa formulation. L'auteur estime que la distinction entre la compréhension des besoins et la spécification des "requirements" est insuffisante. Il attribut cet état de fait au manque de modèle ou cadre supportant l'analyse de système (A.S.).

Au concept trop restrictif de "requirements specification", l'auteur préfère celui de "systems analysis" recouvrant les 3 aspects:

- pourquoi le système doit être développé (problems, needs),
- ce qu'il doit accomplir (requirements),
- comment il doit être construit (conceptual design).

Shemer donne une liste de définitions de l'analyse de système en notant que cette activité est plus perçue comme un art que comme une discipline structurée. Si on considère le développement d'un logiciel comme une suite de transformation de modèles, l'AS correspond aux 3 premières transformations: pbs et besoins, requirements, conceptual design. Explicitant l'activité de l'analyste, Shemer la compare à la "reconstitution d'un puzzle dont l'image définitive n'apparait pas et les fragments d'informations sont obtenus en désordre". L'objectif est d'étudier si une expertise d'analyse peut être formalisée.

Le modèle proposé intègre à égalité process et événements, données, entités auxquels il convient d'ajouter le niveau de décision. Sur cette base est développé un réseau sémantique de 10 classes possédant des attributs: 3 éléments de base (entités, process et données) et 7 relations (représentant le contrôle, la synchronisation, la généralisation, l'aggrégation, l'équivalence ou la mesure des performances, l'entrée-sortie, le rôle). Ce modèle est supporté par un système expert d'aide à l'analyse: SYS_AIDE.

#### 3.2.2.3. Système d'audit en gestion de production

référence: [Tho 88]

SAGP, Système d'Audit en Gestion de Production, est un autre exemple de cette démarche, certes propre à la gestion de production, mais certainement reproductible dans un cadre plus général. Ce système illustre l'apport des techniques de L'Intelligence Artificielle à *l'acquisition de données relatives à l'environnement d'un système de gestion de production non prises en compte dans une représentation donnée*, en l'occurence celles de la méthode GRAI déjà citée, *mais néanmoins nécessaires pour mener à bien une analyse complète du système*. Il s'agit en fait de comparer les modèles GRAI de l'application étudiée avec des modèles de référence correspondant à des contextes particuliers que l'on doit donc être en mesure d'identifier. Le système est construit autour de l'environnement G-Base (base de données orientée objet) et G-Logis (programmation logique de type Prolog).

*"L'éditeur contextuel"* est destiné à la définition du contexte de production propre à l'entreprise étudiée. Ce prototype tient compte actuellement de données habituellement difficiles à formaliser (cf paragraphe analyse du contexte):

- l'environnement macro-économique de l'entreprise,
- le type de produits,
- les caractéristiques du marché,
- le type de sous-traitance,
- les caractéristiques du système physique.

D'autres caractéristiques peuvent venir enrichir le modèle basé sur une représentation objet évolutive. La simplicité d'utilisation de l'outil est renforcée par l'emploi d'une analyse syntaxique.

## Conclusion

L'utilisation d'un modèle de structuration fonctionnelle permet de s'assurrer d'un certain nombre de qualité de l'analyse: complétude et cohérence. Le modèle peut également faciliter le dialogue entre demandeur et concepteur si chaque partie consent à se former ou se familiariser avec les notations.

Une traduction inverse du modèle générique en spécification littérale en langage naturelle pourrait permettre de faciliter la communication quand les différents interlocuteurs ne sont pas familiarisés avec le modèle utilisé. La spécification systématique semble constituer un axe de recherche prometteur pour les prochaines années même si nous n'avons pas pu approfondir cet aspect dans le cadre de notre travail. L'utilisation industrielle passe par la disponibilité de produits-outils fiables et conviviaux. En se basant sur l'hypothèse que l'expression des besoins en langage naturel reflète bien les besoins réels, cette approche ne dispense pas des procédures d'assurance-qualité délimitant notamment les responsabilités respectives de chacune des parties.

# 4. CONCEPTION D'ARCHITECTURE OPERATIONNELLE

La conception d'architecture opérationnelle est peut-être la tâche la plus complexe à modéliser dans la mesure où elle fait intervenir des éléments exprimant des points de vue différents. De plus les décisions prises aux différents niveaux de l'étude ne sont pas indépendantes sans qu'il soit toutefois possible d'en expliciter les corrélations. Nous avons cependant pu identifier trois activités essentielles. Le premier paragraphe "Description des activités" présente ces différentes activités de Conception d'Architecture Opérationnelle.

## 4.1. Description des activités

La conception d'architecture opérationnelle aboutit à la **spécification** d'une **solution d'architecture** que nous qualifierons "d'**opérationnelle**" dans la mesure où elle définit les **caractéristiques techniques** et les **performances opérationnelles** que devront satisfaire les composants matériels et logiciels. A la différence de la tâche de Structuration Fonctionnelle, le travail de conception est basé sur l'organisation des fonctions de service du système en différents composants de la solution et non pas sur la structuration des besoins en fonctions. Ce travail de "répartition des fonctions" de l'application sur les différents dispositifs de l'architecture opérationelle est l'objet principal des décisions de conception. Le travail du concepteur consiste ensuite d'une part à "analyser la robustesse aux défaillances" de la (ou les) solution(s) en prenant certaines décisions pour garantir un niveau de sûreté de fonctionnement conforme aux objectifs, et d'autre part à "spécifier les performances" que devront offrir les matériels et logiciels de base de la future architecture.

La conception de l'architecture opérationnelle est donc basée sur:

- la **répartition des fonctions** sur des dispositifs de traitement, qualifiables de virtuels ou logiques, c'est-à-dire abstraits de tout choix matériel parmi les catalogues fournisseurs;
- la spécification des **fonctions techniques** nécessaires à la réalisation des fonctions de service de la Structure Fonctionnelle;
- l'**analyse de la robustesse aux défaillances** de l'architecture, permettant d'assurer un comportement satisfaisant de l'application en présence de défaillances;
- la **spécification des performances opérationnelles**, limitée dans le cadre de notre contribution aux systèmes de communication par l'étude des flux d'informations.

### 4.1.1. Répartition des fonctions

La répartition des fonctions se rapproche du problème de la conception du logiciel. Un certain nombre de critères particuliers liés à la distribution sur des sites distants doivent de plus être intégrés. Dans la mesure où les dispositifs de traitement et de communication peuvent présenter des taux de disponibilité insuffisants par rapport aux besoins de l'application, on cherche traditionnellement à renforcer l'autonomie et la cohésion des sous-systèmes en limitant le couplage entre ces sous-systèmes. Pour la plupart issus du domaine du développement informatique, des critères de structuration et des principes de modularité ont été étudiés pour une structuration efficace en "modules" et en "tâches". Généralement, dans les exemples traités, l'étendue de l'application est relativement restreinte dans la mesure où elle peut facilement être implantée sur un seul dispositif de traitement, calculateur ou automate, même si la notion de module est assimilée à la notion de "site virtuel" implantable physiquement sur n'importe quel site physique réparti: le contrôleur de robot de [Gom 84] [Gom 86], la fameuse pompe de Kramer étudiée dans [El F 85] [Zak 84] [Lon 87], le contrôle d'une cellule d'assemblage repris dans [Lon 87]. Les études de critères de couplage et de cohésion sont aussi orientés selon ce point de vue [You 78] [Sok 80] .

Deux approches essentielles permettent de mieux cerner les caractéristiques de la modularité :

- une approche **empirique** issue de l'expérience acquise en programmation et basée sur la définition de critères, règles ou principes permettant d'évaluer la qualité de la modularité d'un découpage fonctionnel,
- une approche **formelle** basée sur les propriétés des communications à base de règles de structuration dont l'application peut être systématisée.

Les paragraphes suivants font la synthèse de ces deux approches.

#### 4.1.1.1. Facteurs de cohésion et de couplage du logiciel

[You 78] et [Sok 80] ont approfondi l'étude des différents facteurs affectant la modularité du logiciel. Afin d'appréhender la notion de modularité, 2 concepts de base sont utilisés, correspondant aux points de vue externe ou interne de l'élément considéré:

| |
|---|
| **couplage**: degré d'interdépendance ou quantité de relations entre un élément et les composants de son environnement externe. |

**cohésion**: degré de dépendance entre les composants internes d'un élément considéré.

L'indépendance totale entre deux éléments est définie comme la possibilité de fonctionner sans la présence de l'autre.

**Couplage:**

Les 3 premiers facteurs cités par [You 78] ayant une influence sur le couplage intéressent la distribution sur différents sites:

- le **type de connexion**,
- la **complexité de l'interface** (vue comme la diversité des données ou objets échangés et non pas la quantité d'informations),
- le **type d'information** associé (donnée ou contrôle).

Nous ne retiendrons pas le quatrième concernant le couplage purement logiciel:

- "l'instant de liaison" (exécution, chargement, édition de lien, compilation ou codage).

Différents types de couplage des modules logiciels sont également étudiés par [Sok 80] :

- couplage par le contenu,
- par une zone commune,
- par des références externes,
- par le contrôle (arguments) ou
- par les données.

**Cohésion:**

Sept niveaux croissants de cohésion sensiblement équivalents sont proposés par [Ste 74] [You 78] et [Sok 80] à quelques nuances près:

1- **coïncidentale**, (issu seulement de choix, d'optimisation ou de contraintes de découpage du programme)
2- **logique**, (rassemblement d'éléments de même type)
3- **temporelle** ou **classique**, (éléments devant être exécutés dans un même intervalle de temps)
4- **procédurale**, (éléments associés par un même algorithme ou procédure (itération, décision ou enchainement))
5- **communicationnelle**, (opérant sur les mêmes entrées ou les mêmes sorties)
6- **séquentielle** ou **informationnelle**, (opérant en séquences ou sur une structure de données communes)

7- **fonctionnelle** (tendant vers la réalisation d'une même fonction).

La cohésion fonctionnelle montre l'importance d'effectuer dès la tâche de structuration fonctionnelle un "bon découpage" des besoins fonctionnels. Une approche sémantique simple est proposé par [You 78] pour identifier les différentes fonctions. Elle est basée sur la description complète en langage naturel de la fonction du module. A titre d'exemple citons:

- Si plusieurs verbes sont employés alors la cohésion n'est pas fonctionnelle.

Remarque: *L'approche SADT [Ros 77] pour laquelle chaque activité doit être décrite par un verbe repose sur la même constatation.*

- L'emploi de termes relatifs au temps dénote une cohésion temporelle, procédurale ou parfois séquentielle.
- Si le complément d'objet de la phrase est collectif, la cohésion est probablement logique. .

Des termes comme initialiser, nettoyer, effacer, indiquent une cohésion temporelle.

On se rend compte à travers ce dernier critère de cohésion fonctionnelle de l'importance de la phase précédente de Structuration Fonctionnelle dans la mesure où le passage de besoins non structurés à leur représentation structurée et hiérarchisée n'est pas neutre dans les choix de répartition qui seront fait pendant la Conception de l'Architecture Opérationnelle. Il semble d'ailleurs délicat d'établir une distinction claire entre critères de structuration liés à la tâche analytique et critères liés à celle de conception. C'est pourquoi dans le contexte des études d'architecture, les critères employés pour prendre des décisions sur la répartition des fonctions sont ceux de couplage liés aux échanges de données.

#### 4.1.1.2. Adaptation aux systèmes temps-réel répartis

Dans le domaine des applications temps-réel de contrôle-commande de procédés industriels, on trouve deux approches: l'une empirique plutôt opérationnelle (c'est-à-dire basée sur l'aspect performances des composants de l'architecture opérationnelle) et issue de celle décrite dans le paragraphe précédent, l'autre plus formelle basée sur la communication.

**Démarche "opérationnelle":**

[Gom 84] adapte les critères de cohésion de Yourdon aux tâches temps-réel, donc selon une approche basée sur les caractéristiques des fonctions ou tâches, mais plus orientée sur les performances opérationnelles de l'architecture:

- dépendance des E/S,

- fonctions temps-critiques,
- besoins en puissance de calcul,
- exécution périodique,
- temporel et
- fonctionnel.

Cette approche reste très empirique et difficile à formaliser.

**Démarche "communications":**

Les études menées au CRIN abordent la cohésion et le couplage du point de vue des communications. Cette approche propose de se baser sur la décomposition d'un système à partir des propriétés de dépendance entre les **entrées / sorties** pour aboutir à une structure en modules communicants minimisant les couplages et garantissant la cohésion des modules. La prise en compte du parallélisme de l'environnement, c'est-à-dire du processus de production, est à la base de cette démarche. Ce parallélisme est *"capturé en termes de simultanéités possible ou d'exclusions temporelles entre les E/S"*.

Les modules sont donc eux-même identifiés par leur cohésion et non selon une approche fonctionnelle comme une analyse type flux de données définissant les différentes sous-fonctions composant le système. Seul un premier niveau de décomposition en "unités fonctionnelles" est directement tiré du cahier des charges.

**Cohésion:**

[El F 85] donne ainsi une liste de critères de cohésion basés sur les relations:

- des entrées concourent ou contribuent au déclenchement d'une même sortie,
- des sorties concourent à la réalisation d'un même objectif,
- sont en relation de succession, d'exclusion mutuelle ou peuvent être déclenchées par une même sortie,
- une entrée contribue au déclenchement d'une sortie.

Cette vue externe de la cohésion par les E/S permet de ne pas nécessiter la connaissance d'un niveau de détail supplémentaire de l'élément étudié.

**Couplage:**

Le couplage n'est matérialisé que par les échanges d'informations. Une méthode de décomposition [El F 85] basée sur 19 règles de découpage par affectation des E/S permet d'aboutir jusqu'au niveau tâche. La méthode débute par un recensement des variables et actions permettant de lister les entrées, les sorties et leurs relations de dépendance.

Un premier "macro-découpage" décompose le système en unités-fonctionnelles correspondant aux "macro-actions" définies à partir des indications fournies dans le CdC. Un premier découplage est effectué en créant une unité-fonctionnelle (U-F) artificielle

regroupant les sorties et les entrées communes aux U-F de base. Cette opération revient en fait à tenir compte de la structuration du processus de production quand plusieurs "actions" concernent le même élément du processus: exemple les actions chauffer et refroidir du système de régulation thermique par capteur solaire commandent la même pompe.

La deuxième étape consiste à structurer les U-F en modules. Un module est crée pour chaque sortie de l'ensemble des sorties indépendantes entre elles (aucune entrée commune, pas de relation de précédence temporelle, de synchronisation ou d'exclusion mutuelle) regroupant le plus d'entrées. Les entrées spécifiques à chacune de ces sorties sont affectées au module correspondant. Les sorties restantes sont affectées au module possédant toutes ou le maximum des entrées et, en cas d'égalité, avec une préférence au module auquel l'affectation ne crée pas un nouveau couplage inter-module. Les sorties consécutives ou en exclusion sont regroupées dans le même module.

La dernière étape concerne le découpage des modules en tâches.

Lonchamp [Lon 87] propose la formalisation et la systématisation de la "structuration logique" à partir des relations de dépendance.

La spécification de ces relations est formulée sous la forme d'un graphe non-orienté dont:

- les **noeuds** sont les **E/S** et
- les **arcs** sont les **relations de dépendance** entre les E/S.

La pondération des arcs reflète le degré de dépendance:

- poids 1 pour l'exclusion mutuelle et la causalité,
- poids infini pour les E/S d'un même port bidirectionnel.

Le graphe est partitionné en recherchant le minimum de composantes complètes telles que le poids des arcs intercomposantes soit minimum. A chaque composante est associé un module avec ses E/S. L'exploitation des relations de dépendance entre les E/S a l'avantage d'être systématique et rigoureuse. Elle ne constitue cependant qu'un élément de la répartition. Dans le contexte avant-projet, le recensement exhaustif des E/S et des relations peut s'avérer fastidieux sur de grands projets, bien qu'un outil puisse faciliter cette tâche. Or nous avons vu que toutes les E/S ne sont pas équivalentes en distinguant différents types de liaisons. Cette distinction est un premier moyen indirect de tenir compte de ces relations de dépendance.

#### 4.1.1.3. Autres critères de répartition: contraintes opérationnelles

D'autres critères de répartition directement liés aux contraintes opérationnelles de l'application sont exprimées par le prescripteur. Elles concernent:

- la **topologie**:
  - Les systèmes complexes d'automatisation de production sont caractérisées par la **distribution géographique** des processus de production. Cela entraine des contraintes de **distance** pouvant imposer la répartition des fonctions sur différents sites.
  - De plus le processus de production est habituellement découpé en unités modulaires en fonction de la structure même des machines de production ou en fonction de l'organisation en unité de production. Chacune de ces unités doit alors posséder son propre système informatisé.

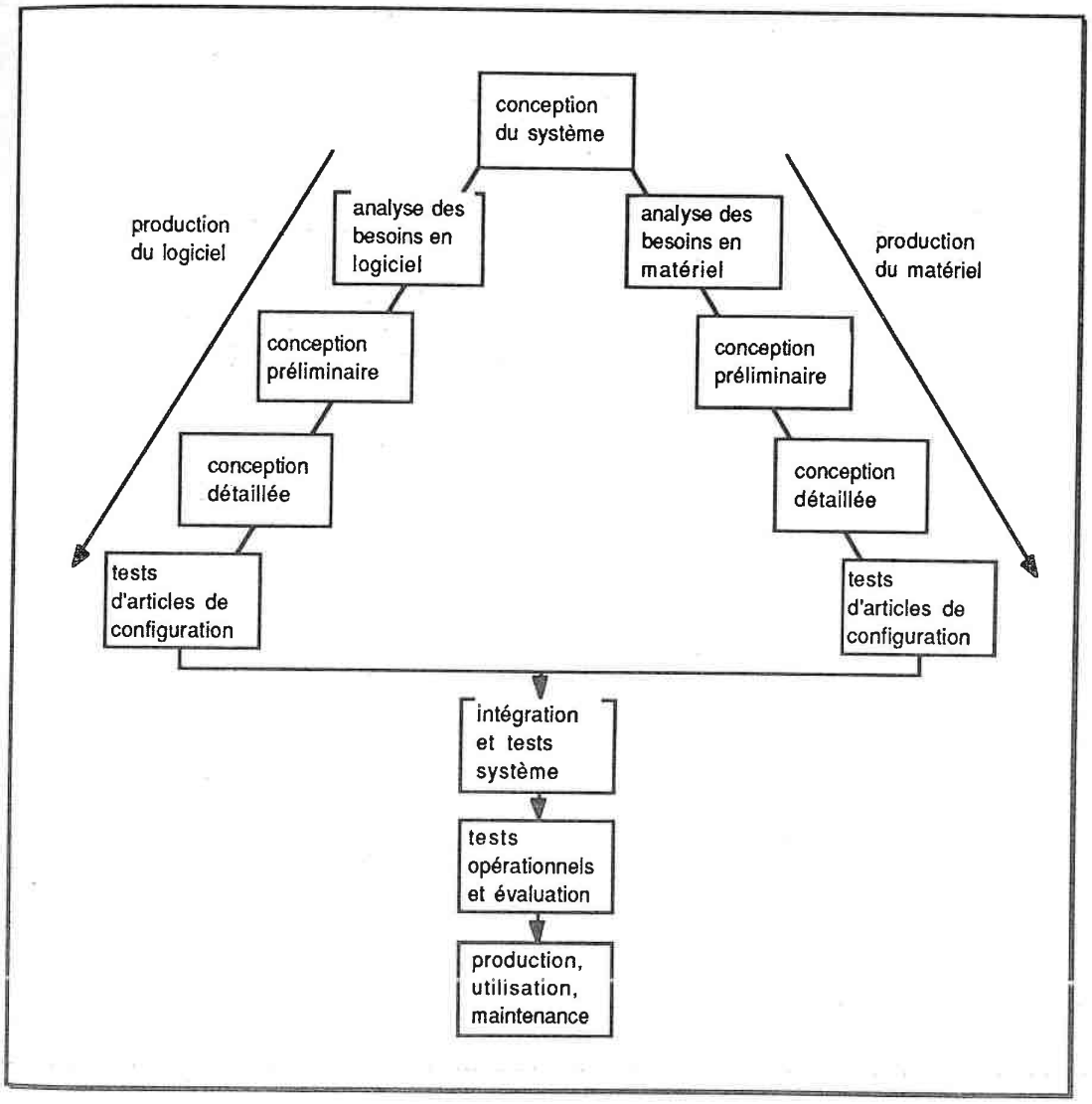
- la **modularité**:
  - La mise en route par phase de larges systèmes peut être favorisée par un découpage modulaire des systèmes informatiques de commande.
  - La commercialisation de sous-systèmes de commande et la réutilisation de sous-ensembles en provenance de réalisations antérieures est également facilité par un développement modulaire sur des machines distinctes.

- les **conditions d'exploitation**:
  - L'organisation des postes opérateurs de conduite/suivi des installations suivant une structure centralisée ou répartie localement influe directement sur la répartition opérationnelle.
  - La définition de modes d'exploitation (mode normal ou dégradé, mode de démarrage, mode de test) peuvent favoriser tel ou tel type d'architecture.

Il est important de noter que ces contraintes sont souvent prépondérantes au point de conditionner largement la définition de l'architecture.

### 4.1.2. Spécification des fonctions techniques

Rappelons que nous nous intéressons aux applications où la partie matérielle (hardware) de l'architecture ne fait pas l'objet de développements spécifiques mais est réalisée par intégration de produits matériels disponibles sur le marché. Il n'y aura donc pas à considérer de développement "parallèle" au développement du logiciel comme c'est le cas pour certains systèmes ou produits temps-réels spécifiques (les systèmes temps-réels embarqués par exemple).

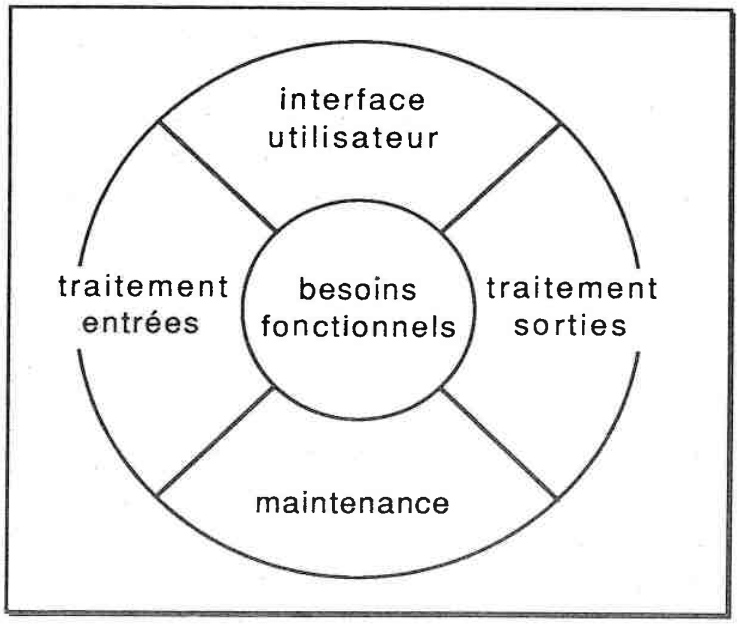



*le cycle de vie des systèmes comprenant un développement matériel*

Le passage du modèle des besoins représenté par la Structure Fonctionnelle à l'élaboration de la solution représentée par l'Architecture Opérationnelle du système

suppose la spécification des "fonctions techniques" ou "fonctions de conception" [AFN 84]. On peut considérer cette spécification (méthode SA/RT, [IGL 88]) comme l'adjonction d'une "couche technologique" autour des besoins fonctionnels nécessaire à la réalisation des "fonctions de service" (qui elles répondent directement aux besoins). Les fonctions techniques proposées dans cette méthode supportent les interactions avec l'environnement:

- l'interface utilisateur,
- les fonctions de communications:
  -le traitement (en fait le prétraitement: codage, conversions, etc...) des entrées,
  - le traitement des sorties,
- la maintenance du système d'information.




*la couche technologique dans l'architecture opérationnelle [IGL 88]*

L'intérêt d'une telle séparation est d'accroître l'évolutivité en augmentant la modularité de l'application, de simplifier l'architecture par une délimitation précise des fonctionnalités.

### 4.1.3. Analyse de la robustesse aux défaillances

Cette partie fait l'objet des développements de la troisième partie de ce document.

### 4.1.4. Spécification des performances opérationnelles

La spécification des performances d'une architecture de SAP suppose la définition :
- du nombre d'entrées-sorties (TOR ou analogiques) en provenance ou à destination du processus de production (en l'attente de réseaux de terrain),
- de la charge de traitement des différentes unités centrales (CPU), des capacités mémoires des matériels et systèmes d'exploitation calculateurs, automates, etc..., pour satisfaire les temps de réponse des contraintes temps-réel,
- des bilans de flux que doivent supporter les systèmes de communication type réseaux locaux en termes de débit et de temps de transfert de message.

Nous avons limité notre réflexion au troisième point relatif aux communications.

L'analyse quantitative des flux d'information est entreprise au cours de la structuration fonctionnelle du système étudié. On suppose donc:
- les fonctions de l'application identifiées et
- les différents échanges d'information (échanges internes entre ces fonctions au sein du système, et échanges externes entre les fonctions du système et l'environnement (opérateurs, machines du processus de production et autres calculateurs externes au système considéré)) qualifiées selon le type de liaison (cf. le modèle de structuration fonctionnelle).

Nous montrerons pourquoi il est important d'effectuer cette analyse à partir de la structure fonctionnelle et non pas seulement sur l'architecture opérationnelle.

#### 4.1.4.1. Rôle de l'analyse des flux

L'analyse quantitative des flux d'information a un rôle important dans le processus de conception d'une architecture en fournissant les données nécessaires pour **choisir** et **dimensionner** les dispositifs de communication.

Cette analyse effectuée à partir de la structuration fonctionnelle débouche donc sur la spécification des performances des dispositifs de communication de l'architecture opérationnelle. La première tâche de l'analyse:

**1- la quantification des échanges de données élémentaires** ("volumétrie" des informations), est effectuée pendant la structuration fonctionnelle.

La deuxième:

**2- le calcul des bilans quantitatifs pour le dimensionnement des systèmes de communication**, est effectuée au cours de l'élaboration de l'architecture opérationnelle puisque cela nécessite la connaissance de la répartition des fonctions sur les différents dispositifs.

La détermination quantitative des flux ne pose pas de problème théorique important, mais plutôt des problèmes pratiques dans la mise en place d'un outil d'assistance. La difficulté consiste à fournir un outil qui allège véritablement la tâche du concepteur. En effet le paradoxe posé par un outil d'assistance à l'analyse quantitative des flux est d'accroître la charge de travail du concepteur en nécessitant une saisie exhaustive donc **fastidieuse** des données de base. Une approche limitée au calcul des bilans de flux risque de se résumer à l'intégration de fonctionnalités d'une calculette à un outil de dessin [Sar 87].

Dans la pratique une telle revue systématique et détaillée n'est que rarement effectuée en contexte avant-projet et cela pour deux raisons essentielles:

- le concepteur n'a pas toujours accès à toute l'information nécessaire à cette quantification,
- l'objectif est plus souvent d'évaluer des ordres de grandeur permettant de valider des options de choix d'architectures pour ne se lancer dans une étude détaillée plus rigoureuse que dans les cas où un problème de dimensionnement est perçu.

L'expérience montre que la plupart des applications sont en effet pour l'instant peu gourmandes en terme de charge du réseau sauf dans certains cas particuliers qu'il convient de détecter pour une analyse plus fine.

De ce constat nous tirons les principaux objectifs de cette étude:

- **identifier les procédures** d'évaluation quantitative,
- **identifier le niveau de précision juste nécessaire** en rapport avec les résultats attendus,
- **alléger au maximum la saisie des données de base** par des mécanismes simples d'utilisation.

#### 4.1.4.2. Quantification des échanges élémentaires de données

Le résultat attendu de cette tâche est de fournir des données chiffrées permettant d'effectuer les bilans de flux dans la tâche suivante. Cette détermination suppose que nous définissions :

- les caractéristiques devant être quantifiées (débits, temps de transfert, ...)

- les données de base fournies par le client permettant cette quantification,
- la précision de la quantification (les ordres de grandeur),
- les procédures d'évaluation,
- les mécanismes facilitant la saisie des données.

*Remarque*: Les besoins généraux ont été étudiés en fonction des différents secteurs professionnels (continu, batch, manufacturier) [Lej 86] [Gal 86]. Cette expression statistique des besoins permet surtout aux constructeurs ou projets de normalisation de définir les caractéristiques des futurs RLI. Mais elle n'est pas suffisante pour définir des besoins génériques facilitant la définition des besoins particuliers d'une application donnée.

**Caractéristiques quantitatives des échanges de données**

Les différents paramètres de quantification des échanges de données permettant une évaluation des besoins en communication sont bien connus [Pri 81] :

- longueur ou taille des données,
- temps de transfert,
- période ou fréquence d'émission,
- débit = longueur * fréquence.

a. La **taille des données** correspond à la dimension des données échangées par les applications (exprimée généralement en octets). Cette taille ne correspond pas a priori à la dimension des trames échangées sur le support, ceci du fait de la segmentation et les overheads des protocoles de communication.

b. Le **temps de transfert** correspond au temps total de transmission d'application à application. Du point de vue du système de communication, cela comprend donc les temps passés dans les files d'attente avant émission, les temps de traitement par les différents protocoles et le temps de transmission proprement dit.

temps de transmission = longueur de la trame / vitesse de transmission.
(s) (o) (o/s)

Le temps de propagation est généralement négligeable devant le temps de transmission.

c. La **période** ou la **fréquence d'émission** est également considérée "en sortie" d'application, c'est-à-dire vue par les producteurs de données. Cette caractéristique est bien sûr plus difficile à déterminer pour les flux apériodiques de type événement que pour

les flux d'états périodiques. Il faut surtout préciser si on considère la période moyenne ou la période maximale ("average or peak value"). Cette dernière valeur permet d'effectuer un dimensionnement correct des systèmes de communication pouvant supporter la charge maximale.

d. Le **débit** d'un échange de données correspond au rapport de la taille des données par la période d'émission (ou au produit de la taille par la fréquence). La donnée d'un débit n'est pas toujours significative en soi pour définir les besoins de communication: la transmission d'un fichier d'1 Mo toutes les 10 mn n'est pas équivalente en terme de charge de réseau à la transmission de 100 octets tous les 60 mS.

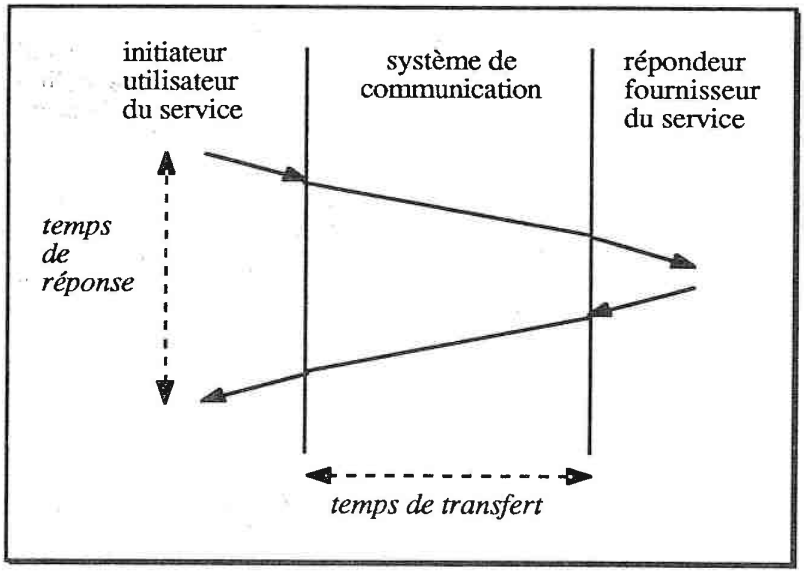
**Données de base permettant la quantification**

Les paramètres présentés dans le paragraphe précédent correspondent à la spécification des besoins du point de vue du concepteur du système de communication de l'application. Ces paramètres ne correspondent pas directement aux données exprimées par l'utilisateur de l'application. Il faut donc examiner quelles sont les données de base disponibles à partir desquelles les paramètres quantitatifs des échanges de données peuvent être évalués.

a. La **taille** des informations peut être donnée explicitement dans les documents fournis par le client. Cependant, tous les flux n'étant généralement pas décrits, le dimensionnement d'une partie non-négligeable des données est laissée à l'initiative du concepteur. Son expérience lui permet de déterminer des ordres de grandeur en fonction du type d'information et de liaison.

b. Les temps de transfert peuvent être donnés, soit directement (c'est généralement le cas pour les messages de contrôle-synchronisation), soit indirectement par la notion de temps de réponse. Le **temps de réponse**, exprime le besoin de l'utilisateur vis à vis des relations de service hiérarchisées: il correspond au temps compris entre l'émission d'un message (une requête) par une application (initiateur) et la réception d'une réponse (un compte-rendu) en provenance de l'application correspondante (répondeur) [Pri 81]. Le temps de réponse comprend donc le temps de traitement du répondeur. Connaissant ce temps de réponse, on en déduit le temps de transfert que doit permettre le système de communication. Le temps de mise à disposition des informations pour les utilisateurs est également utilisé pour exprimer les besoins au niveau des interfaces. Les contraintes de temps de réponse sont souvent spécifiées par les automaticiens en collaboration avec les concepteurs de la partie opérative ou par les responsables de l'exploitation.

La durée de validité d'une donnée peut également être significative et permettre l'évaluation des contraintes de temps de transfert: en effet les traitements effectués sur les données issues du processus doivent être effectués avant la fin de validité de ces données. La définition des durées de validité est issue des renseignements fournis par les responsables process.




*temps de réponse au niveau application et temps de transferts réseau*

c. La détermination de la **fréquence** des échanges est basée, en fonction du type d'information considéré:

- sur la fréquence des opérations process,
- des flux de produits,
- des changements de fabrication et
- de mode d'exploitation,
- sur le nombre de transactions ou d'interrogations opérateur, ou
- sur un rapport de proportionnalité avec d'autres flux d'information etc...

En effet les flux d'information ne font que représenter des activités ou des états physiques du processus de production, ainsi que des conditions de son exploitation [Ber 84]. Seules ces données sont explicitées par le client.

D'où l'importance d'une analyse préliminaire du processus, de son exploitation et de la structuration explicite des fonctionnalités de l'application, indispensables à la bonne compréhension des besoins que devra supporter le système de communication. Un raisonnement effectué directement sur une architecture opérationnelle ne permet pas une

telle compréhension du fait de la confusion entre les éléments relatifs aux besoins et les autres éléments liés aux solutions.

**Précision de la quantification**

Si une précision rigoureuse est souhaitable pour un dimensionnement précis des systèmes de communications, nous avons vu que les données disponibles ne permettaient souvent que de proposer des estimations pour les paramètres quantitatifs. Sauf quand une étude plus fine doit être effectuée, une précision de l'ordre de 10 % permet d'établir l'ordre de grandeur des besoins vis à vis des systèmes de communication.

**Formalisation des procédures de quantification**

Il s'agit d'associer aux liaisons fonctionnelles (les échanges de données) de la représentation fonctionnelle de l'application, les paramètres quantitatifs précédemment définis. Donnons quelques définitions préalables:

Un **message** est un ensemble de données destinées à être transmises en une seule transaction (l'atomicité de la transaction correspond au point de vue des processus applications et non pas du point de vue des systèmes de communication: un message peut être transmis après segmentation transparante pour l'application en plusieurs trames). Un message peut donc être constitué par un fichier, un programme, un signal de synchronisation, une donnée ou une liste de données, etc... Un message sera ainsi caractérisé par un type d'information.

Une **liaison** (fonctionnelle) est définie par un message, son émetteur et ses destinataires. Une liaison peut être considérée comme un objet possédant des attributs dont le concepteur doit définir les valeurs pour chaque instance. Une liaison est instanciée par la connexion de deux ou plusieurs instances d'entité fonctionnelle. Une connexion correspond à l'établissement d'une relation orientée entre des instances d'entité fonctionnelle: un émetteur et un ou plusieurs récepteurs.

| **liaison:** | **exemple:** |
|---|---|
| **attributs:** | |
| nom du message associé: | longueur à découper |
| références: (n° paragraphe) | 4.3.1., 6.2.2. |
| commentaire: (texte libre) | |
| émetteur: | contrôleur |
| destinataires: | machine de découpe |
| type de message: | requête |
| taille du message: | 1 octet |
| période minimale: | 1 seconde |
| temps de transfert: | 100 ms |
| débit:= taille / période | 1 o/s |

Si le débit est calculé automatiquement, restent à saisir trois informations quantitatives.

**taille:**

Nous avons vu que ce paramètre est très dépendant de l'application étudiée. Généralement fournie dans les documents clients, la valeur est spécifiée et l'on associe à cette valeur la référence au document correspondant. Dans le cas (fréquent) où aucune valeur n'est précisée, le concepteur doit formuler une hypothèse. Pour cette variable il est donc intéressant de fournir des valeurs par défaut en fonction des types de liaison et d'information.

**période:**

Nous disposons du schéma de structure fonctionnelle et du schéma du processus de production correspondant. Nous avons vu que la plupart des données de production que supporte le système de communication peuvent être dimensionnées à partir des données de dimensionnement du processus de production et des conditions d'exploitation, données fournies par le client. Si nous avons déjà modélisé le processus, il est possible d'établir directement les formules d'évaluation des périodes des différents messages.

*exemple de formule de quantification de la période d'émission d'un message en fonction de données process:*
longueur à découper
période minimale:= (vitesse maxi d'avancement du verre / taille mini des volumes )

La saisie par le concepteur d'une formule d'évaluation a pour avantages:
- de **faire apparaître explicitement les hypothèses de la quantification** et
- de **prendre en compte directement les modifications** des paramètres de dimensionnement du processus.

Au paragraphe "formules d'évaluation" est décrit l'implantation de ce mécanisme.

**temps de transfert:**

Les temps de transfert sont plus délicats à déterminer car les contraintes de temps de réponse ne sont spécifiées par le client que pour les cas critiques. Il faut donc pouvoir formuler des hypothèses par défaut pour tous les autres cas. L'expérience montre qu'il est possible statistiquement d'associer des ordres de grandeur aux différents types de messages dans un contexte industriel donné.

*exemple:*

données de contrôle-synchronisation de manipulateurs en robotique : de l'ordre de 500 µS [Bor 86]

temps de rafraîchissement de données process en process continu: 500 mS à 1 S

**Mécanismes d'aide à la saisie**

L'apport de l'outil est donc de faciliter par des mécanismes d'aide à la saisie, le paramétrage des valeurs par défaut pour chaque étude particulière et l'aide à la quantification des flux à partir des données de base du processus et de son exploitation.

**typage des échanges d'information et valeurs par défaut**

L'examen des procédures de quantification souligne l'intérêt de pouvoir associer par défaut des paramètres quantitatifs aux différents types de liaisons et aux différents types d'informations. La définition d'une hiérarchie de types de base et la possibilité pour le concepteur de définir des types spécifiques à un domaine ou même à une application particulière accroit la souplesse d'utilisation d'un outil d'analyse des flux. Une telle classification des échanges d'informations allant du plus générique au plus spécifique permet de capitaliser l'expérience de quantification en constituant progressivement un catalogue enrichi au cours des études réalisées. Une approche objet avec mécanisme d'héritage est tout à fait adaptée à la capitatlisation de l'expérience, tout en préservant la cohérence du modèle de base.

A chaque type d'échange seront associées des valeurs types représentant l'ordre de grandeur habituellement considéré par le concepteur. Cet ordre de grandeur sera d'autant plus précis que le type d'échange sera particulier à l'application considérée.

**formules d'évaluation**

Nous avons vu que la période d'émission des données pouvait dans de nombreux cas être estimée à partir des données dimensionnantes du process ou des conditions d'exploitation. La partie process en particulier peut être facilement modélisée de façon systémique (éléments de transformation, transport, stockage et flux de produits) où les différentes opérations et les flux de produits seront quantifiés. Ces données quantitatives devront pouvoir être utilisées par des formules permettant d'en déduire la quantification des périodes d'émission. Ces formules doivent pouvoir exprimer des relations de proportionnalité (simple ou inverse) par rapport à un ou plusieurs coefficients. Les formules standard proposées dans le système G2 (cf annexe et référence [G2 88]) permettent une telle réalisation.

#### 4.1.4.3. Bilans de dimensionnement

Le calcul des bilans de flux suppose une première répartition des fonctions et l'affectation des échanges d'informations à des systèmes de communication opérationnels. Ces bilans permettent d'évaluer les performances que devront supporter les produits réseaux de communication. Avant la recherche de produits, le calcul des bilans permet de vérifier si les ordres de grandeur sont compatibles avec l'état de l'art de la technologie et les investissements envisagés. Par exemple, un débit de plus de 10 Mbits/s risque de nécessiter un saut technologique important. Des contraintes de temps de transfert trop serrées conduiront aussi à une remise en cause de la répartition des fonctions. Quand ces performances ne peuvent être satisfaites par des solutions technologiques compatibles avec le budget objectif, il est nécessaire de modifier les choix de répartition de l'Architecture Opérationnelle.

## 5. DEFINITION DE SOLUTIONS

Cette tâche porte sur le **choix de composants** parmi les systèmes et produits matériels et logiciels disponibles sur le marché, ainsi que le choix de **techniques et outils de développements**.

Le choix de solutions satisfaisant les spécifications précédemment obtenues est basé sur:

- la **recherche de sous-systèmes** de traitement, communication et stockage,
- le **chiffrage** de chacune des solutions,
- un premier **choix technico-économique** parmi les différentes solutions possibles,
- un travail de **synthèse** portant sur la rédaction d'un document présentant les résultats de la conception avec l'argumentaire justificatif.

Les deux premières tâches sont souvent effectuées simultanément et font généralement appel:

- à la connaissance qu'a le concepteur des solutions disponibles sur le marché *,
- à l'étude de la documentation fournisseur,
- à un dialogue avec les différents fournisseurs pour définir le détail des configurations et le chiffrage correspondant.

* Une part importante de l'expérience du concepteur réside dans cette connaissance des solutions. Cette expérience est d'abord obtenue à travers les réalisations antérieures ayant permis l'acquisition de connaissances précises des possibilités réelles des différentes solutions techniques. La mise à jour de ces connaissances est réalisée par un travail important de veille technologique, de test ou évaluation de produits, de développements de maquette et d'études de faisabilité.

La recherche de sous-systèmes est basé sur la satisfaction "locale" des fonctionnalités et performances techniques, mais aussi sur les possibilités globales d'intégration des sous-systèmes entre-eux: compatibilité, possibilités de communication. Des contraintes particulières peuvent également être imposées par le client concernant directement ou indirectement le choix de certains fournisseurs ou certains matériels:

- contraintes de maintenance (homogénéisation des parcs de pièces de rechange, formation des équipes de maintenance),
- contraintes de formation des opérateurs,
- schéma directeur de l'entreprise privilégiant tel ou tel type de technologie,

- volonté d'indépendance vis à vis des fournisseurs (supposant une diversification volontaire des sources d'approvisionnement), ou au contraire
- accords privilégiés avec certains fournisseurs,
- image de marque par rapport aux concurrents, aux clients ou au personnel (motivation), pouvant influencer le degré de technicité et d'innovation d'une solution,
- disponibilité du service après-vente fournisseur,
- contraintes de disponibilité du service après-vente,
- contraintes d'exportation vers certains pays.

## CONCLUSION

Nous avons examiné dans ce chapitre 7 quelles étaient les activités effectuées au cours de la conception pour aboutir à une ou plusieurs solutions d'architecture opérationnelle de SAP. Ces activités ont été regroupées en quatre tâches principales pour lesquelles il est possible d'identifier les données nécessaires en entrée et les documents produits en sortie de tâche. Une telle décomposition, si elle ne fait pas apparaître très clairement les itérations inévitables et nécessaires mais difficilement formalisables, permet de bien expliciter les différents axes que doit investiger le concepteur.

## CONCLUSION de la 2ème Partie

La deuxième partie de ce document constitue donc l'apport essentiel de notre contribution à l'enrichissement méthodique de la conception d'architectures. Nous avons d'abord examiné les différentes sources de connaissances méthodologiques théoriques existantes ou connaissances pratiques utilisées par les experts en conception. De cette analyse nous avons pu isoler quels étaient les concepts génériques de modélisation les plus adaptés à l'analyse et la conception d'architectures, en particulier le modèle générique de Structuration Fonctionnelle d'architecture de SAP. Les concepts rassemblés ont été formalisés selon un formalisme adapté à la formalisation des connaissances. Les activités effectuées au cours de la conception ont été regroupées en tâches principales d'Analyse du Contexte, de Structuration Fonctionnelle, de Conception d'Architectures Opérationnelles et de Définition de Solution.

Si nous n'avons pu approfondir tous les axes d'investigation correspondant à ces différentes tâches, nous avons souhaité découvrir un peu mieux l'impact de l'aspect sûreté de fonctionnement sur le processus de conception d'architectures. Cet axe abordé au cours de la tâche de "Conception d'Architectures Opérationnelles" constitue donc l'objet de la troisième partie.

## 3ème partie:

# ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES D'UNE ARCHITECTURE

## INTRODUCTION

Par rapport aux différentes tâches de la Conception d'Architectures, l'Analyse de la Robustesse aux Défaillances (ARD) intervient au cours de la Conception de l'Architecture Opérationnelle. Elle a pour but de garantir une qualité satisfaisante de la solution d'architecture en terme de disponibilité. Elle intègre pour cela dans le processus de conception, l'étude qualitative de la sûreté de fonctionnement. Notre démarche a consisté à :

- étudier les méthodes (et outils) disponibles dans cette discipline,
- analyser l'applicabilité de ces méthodes à notre problème de conception,
- examiner et s'inspirer de la démarche suivie par les concepteurs.

La première difficulté d'évaluer la sûreté de fonctionnement d'un système complexe est de devoir définir ce qui constitue la panne du système. Un système automatisé de production est d'une telle complexité qu'il ne peut être question de définir directement ce qu'est la panne pour tout le système. L'architecture informatique est-elle en panne si une seule information est manquante ? Toute architecture de SAP est conçue pour être "robuste" à des défaillances élémentaires de ce type. Notre objectif est d'affiner la connaissance des interactions entre la conception et la nécessité de garantir un certain niveau de sûreté de fonctionnement. Nous avons vu au cours de la première partie que les problèmes de disponibilité constituaient un symptôme type d'une architecture inadaptée.

Le premier paragraphe donne la présentation synthétique des "Motivations et objectifs" de cette étude. Nous précisons au paragraphe "Définitions de base" les éléments et définitions bibliographiques sur lesquels nous nous appuyons. Le paragraphe "Analyse des méthodes et outils de sûreté de fonctionnement " examine l'adaptation des méthodes existant dans cette discipline au contexte étudié. Le lecteur familiarisé avec les définitions et les méthodes de sûreté de fonctionnement pourra se reporter directement au chapitre suivant: "Procédures d'Analyse et Mesures Correctives". Afin de s'assurer de la faisabilité d'un outil supportant cette démarche, une maquette a été développée à l'aide d'un environnement générateur de système à base de connaissances. Ce chapitre présente le contenu de la démarche proposée au concepteur d'architectures. Nous détaillerons dans le paragraphe "Environnement de développement" comment G2 autorise une approche de type prototypage.

# chapitre 8:

# Objectifs, Définitions et Analyse des Méthodes de Sûreté de Fonctionnement

## 1. MOTIVATIONS et OBJECTIFS.

Ce paragraphe synthétise quels ont été les motivations et objectifs de cette étude consacrée à la prise en compte de la sûreté de fonctionnement dans la conception des architecture de SAP.

### 1.1. Les motivations

Nous avons vu dans la première partie de ce document que différents facteurs conduisent l'ingénieriste à être de plus en plus sollicité pour concevoir des **architectures réparties**, (essor rapide des techniques informatiques, besoins de partage d'information et d'intégration). A priori la répartition des fonctions sur les sous-systèmes permet par rapport aux solutions centralisées de limiter la portée de la défaillance d'un élément d'une architecture au sous-système considéré. Mais ceci n'est vrai que si les inévitables échanges d'informations (messages, commandes, partages de données, ...) ne créent pas un complexe **réseau maillé de dépendances** liées à la disponibilité de ces informations.

Or la sûreté de fonctionnement d'un SAP est primordiale: les objectifs essentiels étant exprimés avant tout en termes de productivité (donc de disponibilité), qualité et sécurité (donc de fiabilité). La complexité croissante des installations automatisées de production renforce la nécessité d'étudier explicitement l'architecture du SAP du point de vue de la fiabilité et de la disponibilité.

Pour atteindre un niveau de sûreté de fonctionnement satisfaisant, cet aspect doit être intégré à la conception de l'architecture. Une évaluation a posteriori ne permet pas d'aboutir efficacement à une solution satisfaisante car les choix de conception effectués préalablement peuvent difficilement être remis en cause. Le choix de systèmes redondants ou tolérants aux fautes, l'étude de mode de fonctionnement en présence de défaillances, parfois appelés modes dégradés, sont non-seulement familiers aux concepteurs, mais sont généralement aussi importants que l'étude du fonctionnement en conditions normales. La pertinence du choix d'une solution redondante est souvent difficile à justifier du point de vue technico-économique. La multiplication de "modes de marche dégradés" peut

entraîner des coûts de développement importants, et induire des effets pervers d'une utilisation abusive de ces modes n'incitant pas à rechercher les véritables causes de défaillances.

Si ces considérations sont familières aux concepteurs, les études de sûreté de fonctionnement d'une architecture d'un SAP ne sont généralement pas basées sur les méthodes traditionnelles de cette discipline sauf dans les cas où la sécurité joue un rôle primordial (sécurité des personnes ou de l'environnement, dans le domaine nucléaire par exemple). Et en général l'étude de sûreté de fonctionnement porte essentiellement sur le processus de production et les automatismes de sécurité. Si les automaticiens emploient quotidiennement la notion de mode de marche, seul le GEMMA synthétise une expérience empirique de la spécification (et non de la conception) limitée au niveau des automatismes séquentiels. Pour une architecture de SAP, une démarche beaucoup plus globale intégrant l'ensemble du système d'information est nécessaire.

La difficulté d'évaluer la robustesse aux défaillances de tels systèmes provient aussi du fait qu'ils sont souvent spécifiques et rarement reproductibles. Le manque de temps et de ressources qu'il est possible de consacrer aux études préalables dans un contexte avant-projet conduit l'ingénieriste "productique" à ne pas mettre en oeuvre les méthodes et outils classiques du fiabiliste tels que ceux utilisés pour l'étude d'une centrale nucléaire, de systèmes militaires embarqués [Gri 88] ou d'un satellite de télécommunication. De plus on ne dispose généralement pas de toutes les données élémentaires nécessaires pour effectuer des analyses quantitatives .

Nous avons donc souhaité apporter notre contribution à cet aspect sûreté de fonctionnement dans la mesure où:

- un choix sélectif en ce qui concerne les développements était nécessaire car nous ne pouvions approfondir tous les aspects de la conception d'architecture de SAP,
- cet aspect semblait au départ le plus mal maîtrisé,
- nous disposions d'un environnement de développement a priori adapté à ce type d'étude.

## 1.2. Les objectifs.

Les objectifs fixés ont donc été:

**élaborer une démarche et étudier la faisabilité d'un outil facilitant l'analyse qualitative de la sensibilité d'une architecture à des défaillances opérationnelles, analyse permettant la définition de solutions fonctionnelles, opérationnelles ou matérielles pour pallier ces défaillances.**

Cette démarche nécessairement itérative doit aboutir à une architecture robuste, c'est-à-dire sûre de fonctionnement. Insistons sur le fait que les défaillances des éléments d'une architecture ne sont pas obligatoirement supprimées, mais leurs effets sont minimisés pour satisfaire le comportement global de l'application. Précisons que dans le cadre de cette étude seules les défaillances des éléments du système d'information (calculateurs, réseaux) sont prises en compte et non ceux du processus de production bien qu'une approche globale de tout le SAP soit souhaitable à terme.

Ce projet ayant pour but de formaliser et valider les concepts identifiés au cours de ce travail de recherche, nous n'avons pas visé le développement d'un produit opérationnel, mais la définition d'une maquette pour valider la faisabilité de cette approche.

## 2. DEFINITIONS DE BASE

Ce chapitre rappelle tout d'abord quelques définitions terminologiques de base utilisées en sûreté de fonctionnement.

### 2.1. Définition générale

Les définitions utilisées sont largement tirées des travaux du LAAS et en particulier ceux de Laprie sur la définition des notions de base de la sûreté de fonctionnement [Lap 84], [Lap 85], [Lap1 88]. La clarté et la précision des résultats de ce travail de définition dans sa dernière contribution [Lap2 88] est à souligner .

Le concept général de la sûreté de fonctionnement est le concept fédérateur :

**sûreté de fonctionnement**: aptitude d'un système à fournir le service que ses utilisateurs en attendent.

En utilisant la définition du service déjà donnée:

**service**: comportement du système tel qu'il est perçu par ses utilisateurs.

Le service est donc une abstraction de son comportement puisqu'il dépend du point de vue de l'utilisateur au sens le plus large.

**utilisateur**: autre système (humain ou physique) qui interagit avec le système considéré.

On perçoit déjà dans cette définition la difficulté d'évaluer ce qui constitue "la sûreté de fonctionnement d'une architecture de SAP" dans la mesure où l'architecture, en tant que système complexe:

- fournit non pas un seul mais de nombreux services,
- interagit avec de nombreux utilisateurs.

Les paragraphes suivants rappellent les définitions des trois aspects constitutifs essentiels de la sûreté de fonctionnement :

- les entraves, circonstances indésirables causes ou résultats de la non-sûreté de fonctionnement;
- les mesures de la sûreté de fonctionnement;

- les moyens, approches, méthodes et outils permettant d'obtenir la sûreté de fonctionnement.

## 2.2. Entraves à la sûreté de fonctionnement

Les **entraves** à la sûreté de fonctionnement sont les défaillances, les erreurs et les fautes.

### 2.2.1. Notion de défaillance

La **spécification** d'un service étant la description agréée du service attendu, on définit :

**défaillance**: déviation du service par rapport aux spécifications.

On emploi parfois aussi les termes dysfonctionnement ou défaut.

On dira que le service est **approprié** s'il est conforme à la spécification.

**service approprié**: service conforme à la spécification.

Si le service est inapproprié, donc différent de la spécification, il y a alors défaillance. Pour un système informatique, la défaillance est observable par rapport aux informations produites par le système étudié (ses sorties); la défaillance est constituée par :

- des valeurs erronées,
- des informations n'arrivant pas en temps voulu.

Cette définition suppose que la notion de performance du service est précisée. En effet la transition entre service approprié et service inapproprié n'est pas toujours "tout ou rien" comme dans le cas simple d'un fusible par exemple. Il faut alors préciser quel est le seuil de performance en-dessous duquel le service est considéré comme inapproprié. Notons donc qu'une spécification fonctionnelle doit non-seulement préciser le comportement normal, mais aussi les comportements "anormaux", c'est-à-dire envisager les différents modes de défaillance.

La notion de mode de défaillance se rapporte à la manifestation perceptible de la défaillance.

**mode de défaillance**: effet par lequel une défaillance est observée sur un composant du système.

Cette notion est délicate à appréhender pour un système informatique [Hou 87], dans la mesure où on ne peut pas considérer individuellement toutes les combinaisons de valeurs erronées que peut prendre la fonction étudiée, ni considérer qu'un mode correspond à l'occurence d'une quelconque des sorties erronées. Il est donc difficile de présenter une liste exhaustive des mode de défaillance d'un système d'information.

Cette difficulté au niveau de l'analyse de la sûreté de fonctionnement est à rapprocher de la difficulté d'être exhaustif au niveau des tests d'un logiciel.

### 2.2.2. Notion d'erreur

L'activation d'une erreur latente entraîne une défaillance.

**erreur**: partie de l'état d'un système, différente de ce qu'elle devrait être.

### 2.2.3. Notion de faute

Une erreur provient d'une faute.

**faute**: incident dont l'occurrence crée une erreur.

exemple: un défaut de programmation, une instruction incorrecte, (bogue ou "bug") est une faute; cette faute peut entraîner une erreur dans les données calculées; et cette erreur peut affecter le service délivré, la consigne d'un actionneur par exemple, et ainsi provoquer une défaillance.

L'origine d'une faute est :
- un phénomène **physique** ou
- une imperfection **humaine**.

Suivant la phase de création de cette faute, on peut distinguer les fautes de **conception** des fautes **opérationnelles** relatives à l'exploitation du système. Ceci n'est qu'une vue de l'esprit puisque toute défaillance d'un système créé par l'homme peut être imputée à une faute de conception. (Une faute physique sur un calculateur étant par exemple dûe à une mauvaise connaissance de l'hostilité de l'environnement.)

- Une faute est dite **active** lorsqu'elle produit une erreur. Une faute est dite **dormante** dans le cas contraire.
- Une faute est dite **latente** tant qu'elle n'a pas été reconnue en tant que telle, sinon elle est dite **détectée**.

- La persistance temporelle d'une faute peut être **permanente** ou **temporaire**.
- Enfin une faute peut être **accidentelle** ou **intentionnelle**.

### 2.2.4. Notion de propagation et Propagation des défaillances dans une architecture de SAP

En résumé des définitions précédentes:
une *erreur est la manifestation d'une faute dans le système*,
et une *défaillance est l'effet d'une erreur sur le service délivré.*

La propagation d'une défaillance est relative aux relations de causalité entre les défaillances des composants d'un système:

| **propagation**: faute conséquente d'une défaillance d'un composant pour le système qui le contient ou le ou les composants avec qui il interagit. |
|---|

On peut résumer la chaîne de causalité par le schéma suivant :

... ----> faute ----> erreur ----> défaillance ----> faute ----> ...

*la propagation des défaillances des composants d'un système*

L'étude de la propagation est donc basée sur l'étude des relations ou interactions entre les composants d'un système. Ces relations sont généralement peu explicites dans la représentation du système matériel. En revanche, une représentation fonctionnelle a pour vocation de modéliser ces relations. Dans la modélisation fonctionnelle du SAP que nous avons retenue, ces relations correspondent aux :

- relations de **décomposition hiérarchisée** d'un système en sous-système ou composants,
- relations d'interaction liées aux différents types d'**échanges d'informations**.

## 2.3. Mesures de la sûreté de fonctionnement

Les différentes mesures de la sûreté de fonctionnement correspondent à différents points de vue.:

Le point de vue de la **continuité de service** est basé sur la mesure de la fiabilité et la disponibilité:

pour les systèmes non-réparables, on s'intéresse à la fiabilité:

**fiabilité**: mesure de la continuité de délivrance du service approprié (temps jusqu'à défaillance).

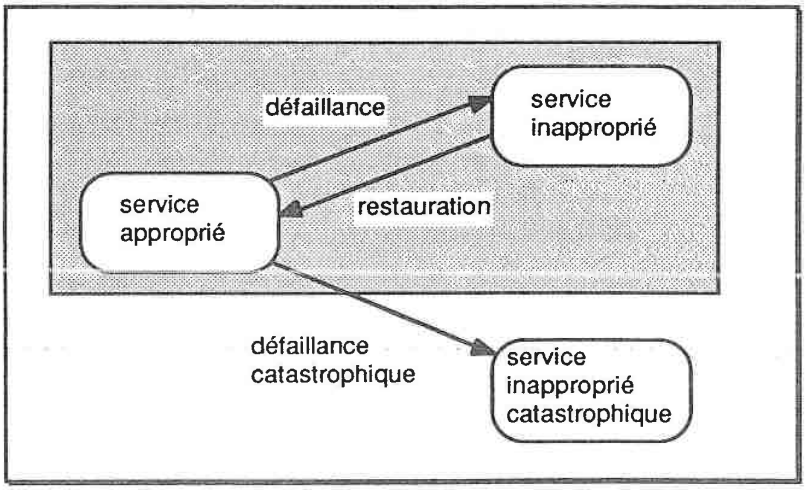
pour les systèmes réparables, on considère la disponibilité:

**disponibilité**: mesure de la délivrance du service approprié par rapport à l'alternance service approprié, service inapproprié.

et la maintenabilité:

**maintenabilité**: mesure du temps depuis la dernière défaillance survenue jusqu'à restauration du service approprié.

Un autre point de vue concerne la notion de sécurité au sens de la non-occurence de **défaillances catastrophiques**, et suppose la notion de danger des personnes, des installations (l'outil de production) ou des produits.




*les différents états d'un service du point de vue de la sûreté de fonctionnement [Lap 88]*

Souvent appréhendée pour son aspect qualitatif, la sécurité peut être quantifiée par extension de la notion de fiabilité:

**sécurité**: mesure du temps jusqu'à défaillance catastrophique.

Le terme sécurité est employé au sens "safety" et ne doit pas être confondu avec le point de vue suivant ou le terme sécurité est employé au sens "security".

Pour le point de vue de la **préservation de la confidentialité et de l'intégrité** des informations, on parle aussi de sécurité. Cette notion n'est habituellement pas quantifiée.

remarque: L'importance de la notion de service dans ces définitions souligne l'intérêt d'utiliser un modèle fonctionnel hiérarchisé précisément basé sur cette notion de service.

### 2.4. Moyens de la sûreté de fonctionnement

Les **approches** utilisées pour la conception et la réalisation d'un système sûr de fonctionnement sont [Lap 88]:

la **prévention**: empêcher l'occurence ou l'introduction de fautes par **construction**;

la **tolérance**: fournir un service conforme à la spécification en dépit des fautes par **redondance**;

l'**élimination**: réduire la présence de faute par **vérification**;

la **prévision**: estimer la présence, la création et les conséquences de fautes par **évaluation**;

la **maintenance**: préserver ou améliorer l'aptitude du système à délivrer un service conforme à la spécification par **élimination**. (curative ou préventive)

Ces approches sont combinées en fonction de l'objectif:

- pour tendre vers un système exempt de fautes par prévention et élimination: on parle d'**évitement**;
- pour fournir au système l'aptitude à délivrer un service conforme à la spécification par prévention et tolérance: on parle d'**obtention**;
- pour assurer la confiance dans l'aptitude du système par élimination et prévision: on parle de **validation.**

## 3.2. Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité (AMDEC)

références: [Hou 87], [CEI 85], [Coq 88], [Gui 86], [Lie 76], [Vog 86], [Beo 88]

### 3.2.1. Présentation générale

L'AMDE ou AMDEC (ou encore "Failure Modes and Effects Analysis, FMEA") permet de rechercher les défaillances pour chaque composant d'un système, d'en estimer les risques d'apparition, et la gravité des conséquences. Elle passe en revue l'ensemble des organes (approche **matérielle**) ou l'ensemble des fonctions (approche **fonctionnelle**) du système étudié, et parfois l'ensemble des phases d'un processus. A partir des causes possibles l'analyste-concepteur est amené à définir les mesures correctives destinées à limiter ou pallier les effets de ces défaillances. C'est donc une technique d'analyse des défaillances prévisionnelle, inductive (de la cause vers les effets), systématique et qualitative. La méthode a donc vocation d'être intégrée à la conception d'un système pour éviter d'aboutir à des modifications coûteuses.

Le support de la méthode est constitué d'une **grille** généralement composée de colonnes permettant de :

- décrire les **modes de défaillance** des composants (indiquant de quelle manière le composant risque de ne pas fonctionner),
- décrire les **causes initiales** possibles,
- décrire les **effets** ou conséquences sur le système (sur le produit, le process, le moyen de fabrication dans le cas des SAP),
- estimer la **criticité** évaluée à partir de la gravité, de la fréquence, et de la probabilité de non-détection,
- les **mesures correctives** à prendre pour réduire cette criticité.

Chaque grille correspond à l'analyse d'un composant ou d'un organe du système dont est précisée la désignation, l'identification du composant étudié et sa fonction. Ces grilles peuvent être adaptées aux besoins particuliers de l'étude ou du système étudié: on peut par exemple préciser les moyens de détection, la phase d'utilisation, les performances de la fonction.

| AMDEC | | | | Référence | | | Date |
|---|---|---|---|---|---|---|---|
| Organe | | | | Fonction | | | |
| Mode de défaillance | Causes | Conséquences locales / globales | | Criticité: Gravité | Fréquence | Détection | Mesures correctives |
| | | | | | | | |

*une grille AMDEC*

Remarque: un mode de défaillance d'une fonction peut survenir quand [Vog 86]:

- la fonction ne se réalise pas,
- la fonction cesse,
- la fonction se réalise mais avec des performances insuffisantes,
- la fonction s'effectue à un instant non désiré.

Des tableaux permettent de définir qualitativement les critères [Coq 88] de:

- **gravité**: importance des conséquences sur la production, les installations ou les utilisateurs,
- **fréquence**: probabilité d'apparition de la défaillance ou taux de défaillance,
- **détection**: probabilité de non-détection à temps de la défaillance.

| GRAVITE | production | incidence produit | incidence process | incidence moyen |
|---|---|---|---|---|
| mineure | micro-arrêt < 5 mn | aucune | aucune | aucune |
| faible | faible perte < 10 mn | aucune | légère reprise | aucune |
| importante | perte importante | nécessite une reprise importante | | faible |
| critique | perte très importante arrêt < 1 h | perte importante de produit non-récupérable | arrêt | détérioration |
| catastrophique | pertes très importantes et risques pour les utilisateurs arrêt > 1 h | | | détérioration complète |

*les critères de gravité d'une AMDEC*

| FREQUENCE | technologies connues | technologies nouvelles |
|---|---|---|
| inexistant | aucune défaillance n'a été observée | la technologie est maîtrisée |
| possible | une défaillance occasionnelle est apparue | la technologie est maîtrisée, mais toutes les conditions ne sont pas réunies pour garantir la non-défaillance |
| fréquent | il y a eu des problèmes dans le passé | rien ne permet de prévoir s'il n'y aura pas de défaillances |
| répétitif | il y a eu des défaillances répétitives | la technologie n'est pas maîtrisée |

*les critères de fréquence d'une AMDEC*

| DETECTION | manifestation | conséquences |
|---|---|---|
| 1 | signes avant-coureurs, action préventive possible | peuvent être évitées |
| 2 | imprévisible mais immédiatement identifiée | peuvent être évitées |
| 3 | rapidement identifiée | ne peuvent être évitées |
| 4 | pas décelable (manque de moyens de détection ou localisation difficile), action corrective complexe | ne peuvent être évitées |

*les critères de détection d'une AMDEC*

La criticité est alors obtenue par le produit des notations de ces sous-critères avec une éventuelle pondération.

| CRITICITE | définition |
|---|---|
| 1: négligeable | pas de mesures spéciales à prendre |
| 2: marginale | les conséquences tout en n'étant pas négligeables peuvent être admises |
| 3: critique | les conséquences peuvent être importantes |
| 4: catastrophique | les conséquences doivent être évitées à tout prix |

*les critères de criticité d'une AMDEC*

### 3.2.2. Application de l'AMDE au Logiciel

L'Analyse des Effets des Erreurs Logiciel (AEEL) [Thi 86] a pour objectif de caractériser la criticité des modules d'un logiciel vis à vis d'un thème d'étude tel que la sécurité ou la disponibilité. Cette méthode est donc une transposition directe de l'AMDE au domaine du logiciel. Les défauts (fautes de programmation) étant propagés au cours du cycle de développement du logiciel, il est bien entendu souhaitable d'initialiser cette méthode dès la fin de la phase d'analyse fonctionnelle. L'analyse consiste à supposer l'existence d'erreurs logicielles dans un module et à analyser les erreurs induites en cascade dans les différents modules jusqu'à une éventuelle action sur les variables externes au processus. Les causes originelles du défaut pouvant être multiples, elles ne sont donc pas définies en détail mais appréhendée par types ou catégories pour limiter l'ampleur de l'étude: erreur de calcul, d'algorithme, sur les données traitées, d'interfaçage entre les procédures, de transfert de données avec l'environnement, de synchronisation entre les tâches. L'évaluation de la criticité dépend de l'objectif de l'étude : sécurité, disponibilité. Une pondération de la criticité des modules en fonction du nombre d'erreurs répertoriées et de leur criticité permet de hiérarchiser les modules.

### 3.2.3. Utilisation des AMDE en conception

L'AMDEC, centrée sur les modes de défaillance, est en quelque sorte le pendant de l'analyse fonctionnelle, elle-même plutôt centrée sur les modes de fonctionnement normal, c'est-à-dire en l'absence de défaillances. Elle est parfois considérée comme préparatoire à l'utilisation d'autres techniques plus formelles et quantitatives (cf paragraphes suivants) car elle permet d'approfondir la connaissance qualitative du comportement du système en présence de défaillances. Quand les données quantitatives élémentaires ne sont pas disponibles ou d'une précision insuffisante (et cela constitue la majorité des cas en Production Automatisée), l'AMDE est une des seules techniques utilisables.

Les AMDE peuvent être appliquées en phase de conception avec pour objectif [Gad 86] d'identifier:

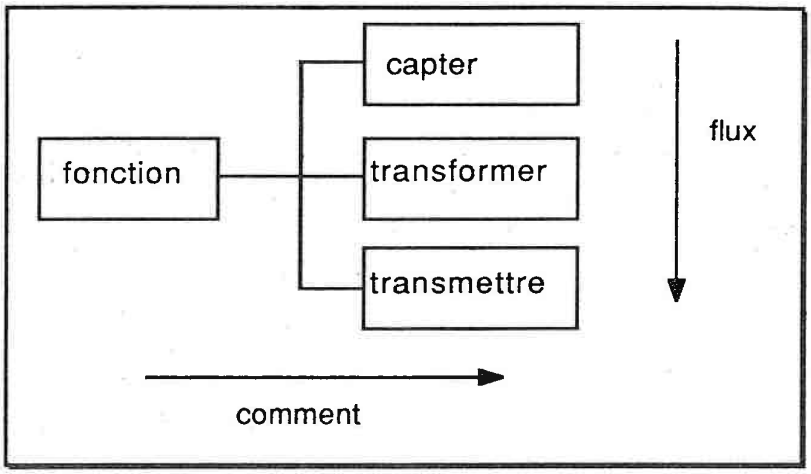
- les éléments fondamentaux,

- les points de panne unique,
- les points critiques,
- les modes communs annulant les redondances matérielles ou fonctionnelles,
- les risques,
- les possibilités de mission dégradées.

[Gad 86] propose de commencer les AMDE au niveau fonctionnel, donc à un niveau d'abstraction relativement élevé, avant de passer à des niveaux de détails plus fins pour parvenir à l'étude des constituants. Les méthodes RELIASEP du CEP [Vog 86] et l'Analyse Préliminaire Fonctionnelle des Risques [Bou ] sont basées sur l'approche fonctionnelle jugée préférable à l'approche composants matériels:

- *"une analyse partant du matériel est, pour être exhaustive,* ***disproportionnée par rapport à l'objectif*** *(analyse inutile de caractéristiques de fonctions mineures),*
- *l'analyse de fonctions effectuées par des* ***éléments géographiquement dispersés peut facilement être oubliée,***
- *les* ***relations fonctionnelles*** *précises ne sont pas facilement mises en évidence de façon lisible et il est alors difficile d'en trouver les relations de causes à effet,*
- *les justifications des différents paramètres de conception n'étant pas disponible sous une forme simple il est nécessaire, lors de toute* ***modification****, de refaire le raisonnement (perte de temps, risque d'oublis donc efficacité diminuée),*
- *la véritable redondance opérationnelle doit être appréhendée au niveau de la fonction et non du composant."*

Cette approche fonctionnelle est donc basée sur l'emploi d'un modèle fonctionnel représentant la structure du système étudié (fonctions composant le système, relations et flux inter-composants). On trouve par exemple des diagrammes SADT ou dans Reliasep un modèle simple appelé "arbre fonctionnel" permettant de représenter l'expression du besoin, selon des notations inspirées des diagrammes FAST et du Cahier des Charges Fonctionnel de l'Analyse de la Valeur et de la méthode HIPO en informatique (Hierarchical Input Process Output). Sont explicitées les fonctions principales, les performances, les phases, les contraintes. L'arbre permet de décomposer systématiquement les fonctions principales en définissant: capter le flux, le transformer et transmettre ce flux.




*la représentation d'un système par arbre fonctionnel*

Ces arguments avancés pour l'approche fonctionnelle s'appliquent directement aux études d'architecture de SAP si l'on considère que :

- les fonctions sont souvent **réparties** sur plusieurs dispositifs opérationnels,
- les relations fonctionnelles sont constituées par les **échanges d'informations** explicites dans le modèle,
- il est nécessaire de pouvoir tester **plusieurs solutions d'architecture** pour un même modèle fonctionnel.

Le **modèle fonctionnel structuré** que nous avons proposé pour la structuration fonctionnelle du système d'information du SAP semble donc en première approche adéquat. L'exploitation systématique et automatisable d'un schéma fonctionnel ne semble cependant pas faire l'objet d'études particulières.

### 3.2.4. Apports et Avantages

En résumé, les AMDE(C) permettent donc:

- d'identifier les risques importants de façon systématique (exhaustivité) pour tous les composants du système étudié,
- d'évaluer leurs effets,
- de faciliter la maîtrise de la réduction des risques en préconisant des mesures correctives.

Elles sont aisément applicables en phase de conception et en particulier dès les spécifications fonctionnelles.

### 3.2.5. Limites et Difficultés

Les principales limites et difficultés de cette méthode sont [Gad 86] des difficultés techniques et des difficultés de gestion.

La principale limite technique inhérente à cette méthode est son orientation sur l'analyse d'une défaillance élémentaire qui **ne prend pas en compte l'étude des combinaisons de défaillances**. Les informations disponibles n'étant pas toujours suffisantes au début de l'analyse, il est nécessaire de pouvoir procéder de façon itérative suivant les niveaux de détail auxquels il est possible de travailler et surtout la nécessité d'approfondir les sous-ensembles les plus critiques.

Une étude exhaustive des modes de défaillance devient rapidement difficile à gérer sans outil informatique support. Un certain nombre de travaux portent sur le développement de tels outils, souvent à usage interne [Gui 86], pour faciliter la saisie, la gestion et l'exploitation des grilles d'analyse bâties autour d'une base de données. Cette exhaustivité avec ou sans outil informatique est difficile à assurer et conduit à passer beaucoup de temps à recenser et étudier des défaillances dont le risque n'est pas très important par rapport aux défaillances critiques. Cet aspect est contradictoire avec notre objectif de ne pas accroître la charge de travail des concepteurs.

## 3.3. Analyse des Causes Insidieuses

références: [Beo 88]

### 3.3.1. Présentation

La méthode a pour but de rechercher les combinaisons de conditions pouvant entraîner un événement imprévu. Cette méthode d'analyse inductive cherche a mettre en évidence les causes de défaillance ayant pour origine les fautes de conception. A l'origine conçue pour l'analyse des circuits électriques, elle a été étendue au domaine du logiciel. Elle s'effectue donc à partir de schémas électriques ou du code source. La conséquence directe est qu'elle ne peut donc que s'appliquer au produit "fini" et non à la conception. La méthode consiste à appliquer des révélateurs basés sur des indices topologiques: par exemple en ce qui concerne le logiciel, utilisation de variables, branchement conditionnel, étiquette d'instruction, etc... En d'autres termes on recherche des motifs potentiellement problématiques pour aboutir aux conséquences sur le système. La méthode est donc largement basée sur l'expérience d'analyse liée à ces motifs.

### 3.3.2. Apports et Avantages

Cette méthode permet par vocation de détecter des incidents non-prévus. Elle est particulièrement utilisés pour la validation dans l'étude des système embarqués en complément avec des méthodes classiques.

### 3.3.3. Limites et Difficultés

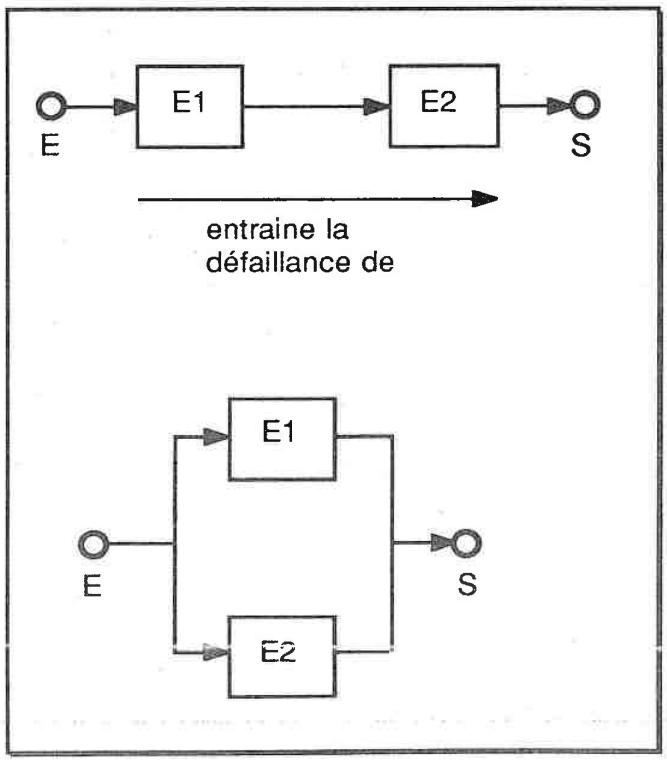
Cette méthode nécessite l'emploi de logiciels puissants et coûteux. Elle a été développée au sein de grandes entreprises et il est difficile d'y avoir accès, en particulier pour tout ce qui concerne les motifs. Cette méthode n'est pas applicable en phase de conception.

## 3.4. Diagrammes de Fiabilité

références: [Pag 80], [Ber 85]

### 3.4.1. Présentation

Le Diagramme de Fiabilité (DdF) est un outil de description simple du système étudié par l'ingénieur fiabiliste. Cette représentation est proche de la description fonctionnelle du système, mais est orientée vers l'expression du comportement du système vis à vis des défaillances de ses composants. La représentation graphique modélise donc les relations de conséquences des défaillances des composants sur le système. Les éléments dont la défaillance entraîne la défaillance du système sont disposés en **série**, ceux dont la défaillance ne provoque la défaillance du système qu'en combinaison sont disposés en **parallèle**. Le DdF est donc un graphe comportant une entrée et une sortie dont les sommets sont les composants du système et les arcs les relations entre ces composants. Le système possède donc un certain nombre de "chemins de succès" entre entrée et sortie traduisant les états de marche du système.

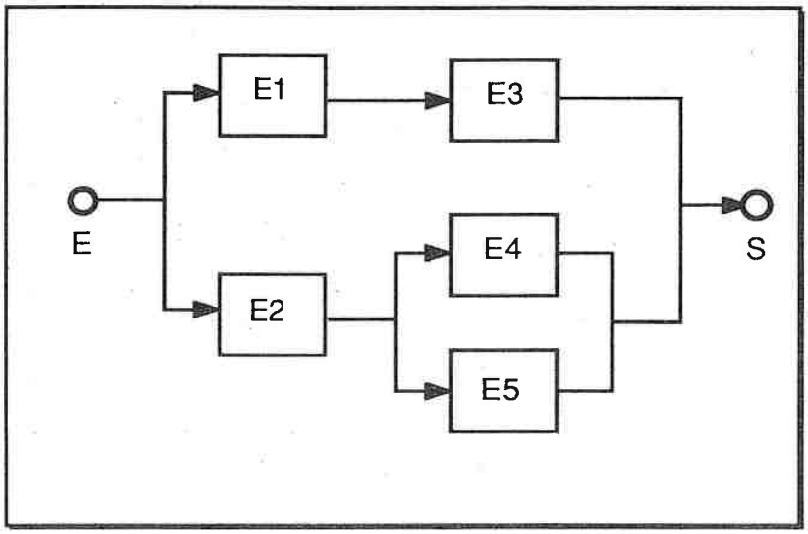



*les diagramme série et parallèle de fiabilité*

Des symboles particuliers représentant la redondance (r / n) ou la redondance passive, et le regroupement de composants en "blocs" permettent de simplifier les

schémas complexes. La définition d'éléments fictifs est nécessaire pour représenter des relations de dépendance complexes entre les éléments du système (mode commun par exemple) [Pag 80] .

Les diagrammes de fiabilité supposent l'indépendance des sous-systèmes, c'est-à-dire que la défaillance d'un sous-système n'entraîne pas la défaillance directe d'un autre sous-système. En d'autres termes la réparation du seul sous-système défaillant suffit à rendre de nouveau le système disponible.




*un exemple de diagramme de fiabilité*

Les diagrammes de fiabilité sont souvent utilisés en conception de systèmes pour étudier automatiquement les combinaisons de pannes entraînant la perte des diverses fonctions du système, et pour effectuer des calculs de fiabilité et de disponibilité [Ber 85].

### 3.4.2. Apports et Avantages

Cette méthode est applicable au niveau des spécifications fonctionnelles ou au niveau opérationnel. Le formalisme est simple et permet l'analyse qualitative du système par la recherche des coupes minimales et le calcul de mesures de fiabilité et disponibilité.

### 3.4.3. Limites et Difficultés

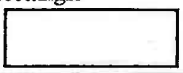
Cette méthode ne permet pas la représentation des interactions avec l'environnement ni la représentation des dépendances entre les défaillances des éléments, ni celle des stratégies de réparation. L'élaboration du Diagramme suppose la connaissance du système donc nécessite une représentation fonctionnelle "classique" du système étudié pour établir les relations de conséquence des défaillances.

## 3.5. Arbres de Défaillance

références: [Pag 80], [Hou 87],

### 3.5.1. Présentation

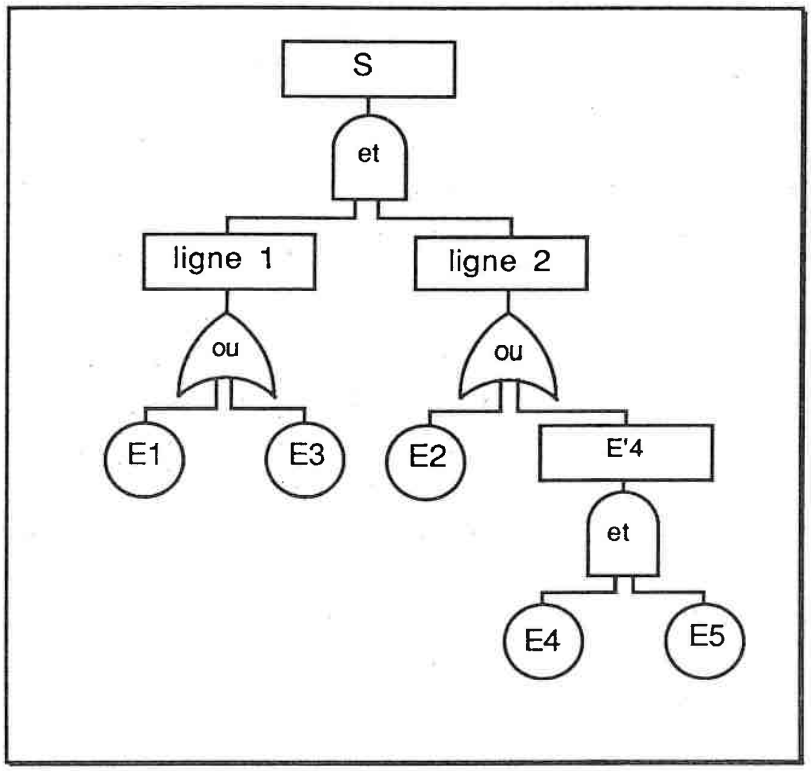
Les Arbres de Défaillance (AdD) (Fault Tree) sont constitués de graphes qui à partir d'un dysfonctionnement donné, indésirable et unique, explicitent les combinaisons de différentes défaillances qui peuvent y conduire. Ce processus déductif définit successivement les événements jusqu'à un niveau où l'analyste estime que la défaillance est élémentaire, c'est-à-dire ne doit plus être décomposée. A ce niveau la probabilité d'occurence des événements de base doit pouvoir être estimée quantitativement. Les AdD sont représentés à partir de notations symboliques graphiques constituant les noeuds du graphe.

| **concept** | **désignation /**<br>**symbole graphique** | **signification** |
|---|---|---|
| événements | rectangle  | événement résultant de la combinaison d'événements plus élémentaire |
| | cercle  | événement élémentaire |
| | losange  | événement ne pouvant être considéré comme élémentaire mais dont le développement des causes n'est pas jugé pertinent |
| | double losange  | événement dont le développement des causes sera effectué ultérieurement |
| portes logiques: | porte OU  | l'événement de sortie est généré si une ou plusieurs entrées sont présentes |
| | porte ET  | l'événement de sortie est généré si et seulement si toutes les entrées sont présentes |

*les principaux symboles utilisés dans les Arbres de Défaillance*

Des symboles particuliers non représentés dans ce tableau permettent de simplifier le dessin des schémas complexes.

L'événement indésirable (la sortie) peut donc s'exprimer sous forme d'une fonction booléenne des événements élémentaires en entrée. Cette fonction permet de rechercher les coupes minimales (cf paragraphe suivant).




*un exemple d'Arbre de Défaillance à partir du Diagramme de Fiabilité précédent*

### 3.5.2. Apports et Avantages

Cette approche permet de ne se focaliser que sur les événements indésirables contrairement aux AMDE. Elle peut être effectuée à partir des spécifications fonctionnelles d'un système ou sur le système opérationnel. Des sous-arbres permettent de simplifier la représentation des arbres complexes. Cette modélisation donne accès aux calculs des mesures de la sûreté de fonctionnement.

### 3.5.3. Limites et Difficultés

Le principal reproche fait à cette méthode est d'aboutir à des graphes rapidement très compliqués pour l'étude de problèmes non-triviaux. La mise en application pose aussi le problème du choix des événements indésirables qui doivent être:

- bien définis pour que l'arbre soit complet et
- pas trop généraux pour ne pas conduire à un arbre trop important qui rassemblerait toutes les défaillances élémentaires.

Par exemple partir de la défaillance globale du système d'information d'un SAP ne peux pas aboutir à un arbre exploitable.

## 3.6. Coupes minimales

références: [Pag 80], [Ber 85], [Hou 87]

La méthode des coupes minimales consiste à rechercher l'ensemble des "coupes" (cut set), c'est-à-dire des ensembles d'éléments dont la panne simultanée entraîne la panne du système. Une coupe est dite **minimale** quand cette coupe ne contient aucune autre coupe. Autrement dit la présence d'un élément est nécessaire pour entraîner la panne du système. L'intérêt de déterminer ces coupes minimales est donc fondamental puisqu'il définit des combinaisons significatives de pannes au niveau du système.

Une coupe peut être interprétée comme un ensemble d'élément "coupant" les chemins de succès d'un diagramme de fiabilité. Un diagramme série admet 2 coupes minimales {E1} et {E2}. Un diagramme parallèle admet 1 coupes minimales {E1, E2}. Les coupes minimales peuvent donc être calculées automatiquement à partir des diagramme de fiabilité [Ber 85] ou des arbres de défaillance [Pag 80], [Hou 87]. L'application des lois d'absorption de l'algèbre de Boole permet de réduire les expressions booléennes et de donner directement ces coupes minimales.

exemple: d'après l'exemple représenté par un diagramme de fiabilité et un arbre de défaillance, on obtient l'équation:

S = (E1 ou E3) et (E2 ou (E4 et E5)) = (E1 + E3) . (E2 + (E4 . E5))
= E1.E2 + E1.E4.E5 + E3.E2 + E3.E4.E5
= E1.E2 + E2.E3 + E1.E4.E5 + E3.E4.E5

On retrouve bien les coupes minimales visibles sur le diagramme de fiabilité, chaque coupe n'en contenant aucune autre :

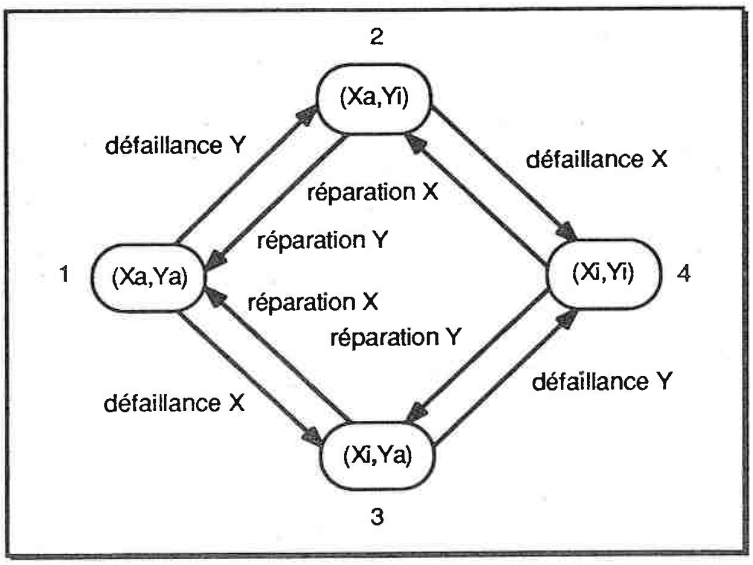
(E1, E2), (E2, E3), (E1, E4, E5), (E3, E4, E5).

## 3.7. Graphes de Markov

références: [Lap 88] , [Beo 88]

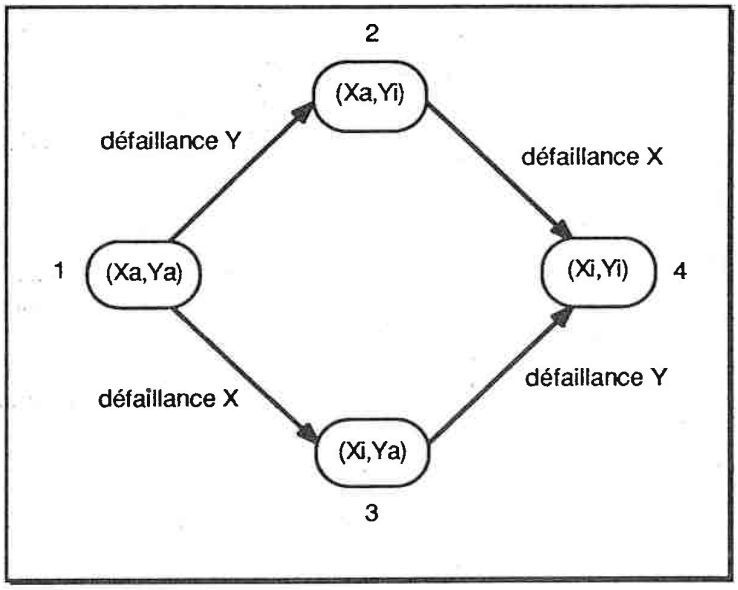
### 3.7.1. Présentation

Cette approche consiste à représenter le système étudié par un graphe d'état orienté dont les sommets sont les différents **états** du système et dont les arcs sont les **transitions** entre ces états (graphe d'état). Chaque arc est valué par le taux de transition entre les états reliés.




*le graphe de Markov d'un système réparable à deux composants ayant chacun un réparateur*

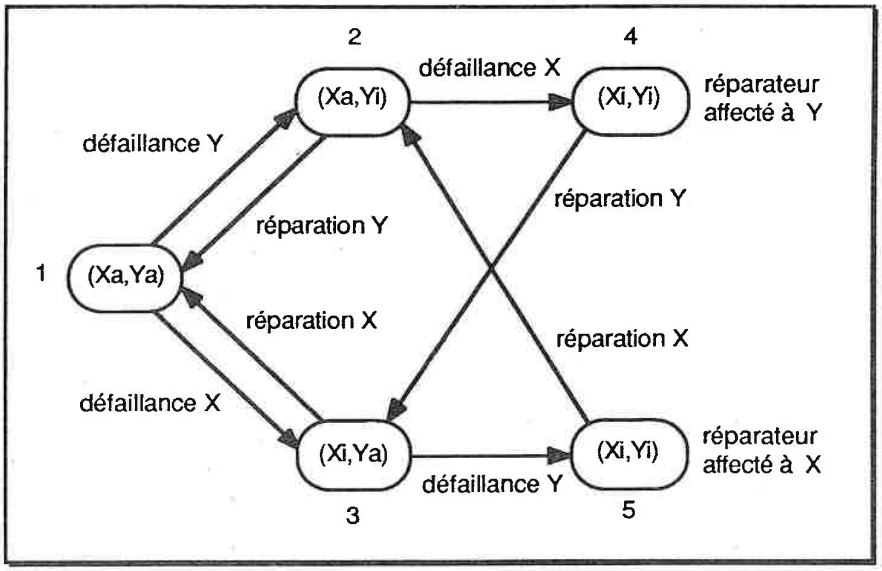
Le graphe d'un système non-réparable diffère de celui d'un système réparable par l'absence des transitions de réparation.




*le graphe de Markov d'un système non-réparable à deux composants*

Si la connaissance de l'état d'un système à un instant donné est suffisante pour déterminer son évolution ultérieure et si les taux de transition entre les états sont **constants**, le graphe est dit **markovien** et homogène (ou graphe de Markov). Des graphes dits semi-markovien et des graphes à états fictifs permettent d'étudier des taux non-constants.

On remarque que le graphe ne préjuge pas de la structure du système. Le même graphe représente ainsi un système à deux composants en série ou en parallèle (redondant).




*le graphe de Markov d'un système réparable à deux composants avec un seul réparateur géré en FIFO*

La représentation matricielle de ces graphes permet d'informatiser les calculs de fiabilité ou de disponibilité. De nombreux outils manipulant ces matrices sont d'ailleurs disponibles.

### 3.7.2. Apports et Avantages

Les avantages de cette approche souvent utilisée sont [Lap 88]:

- leur aptitude à tenir compte des dépendances entre les composants,
- la possibilité de calculer plusieurs mesures à partir d'un même modèle,
- l'aptitude à modéliser des taux de transition non-constants par la méthode dite des états fictifs que nous ne présenterons pas,
- la majorité des logiciels de sûreté de fonctionnement repose sur cette modélisation.

### 3.7.3. Limites et Difficultés

Cette représentation est hélas assez lourde puisque pour un système à n composants à 2 états, il est nécessaire de représenter $2^n$ états. Cette difficulté de conserver une qualité de lisibilité et d'exploitabilité suffisante conduit à l'utiliser plutôt pour l'analyse d'un sous-système dont les éléments sont fortement dépendants [Gon 80].

## 3.8. Réseaux de Petri Stochastiques

références: [Lap 88],[Beo 88]

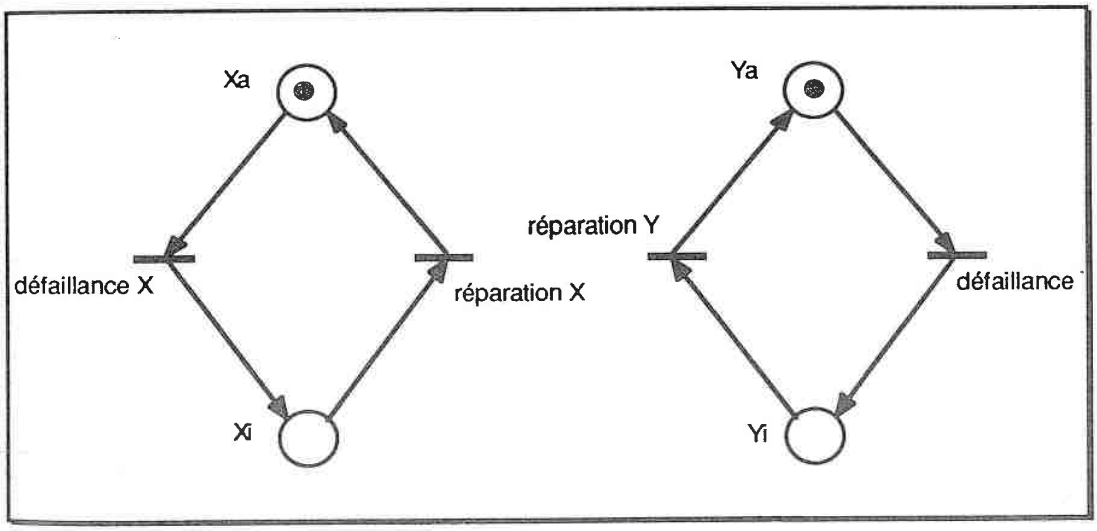
### 3.8.1. Présentation

Les Réseaux de Petri sont traditionnellement représentés sous la forme de graphes orientés bipartis dont les deux types de sommets sont:

- les places, représentées par des cercles, désignent les **états** des différents composants du système, et
- les transitions, représentées par des barres, désignent les événement relatifs aux **changements d'état** et sont associées aux **taux** de transition (taux équivalents à ceux des graphes de Markov,

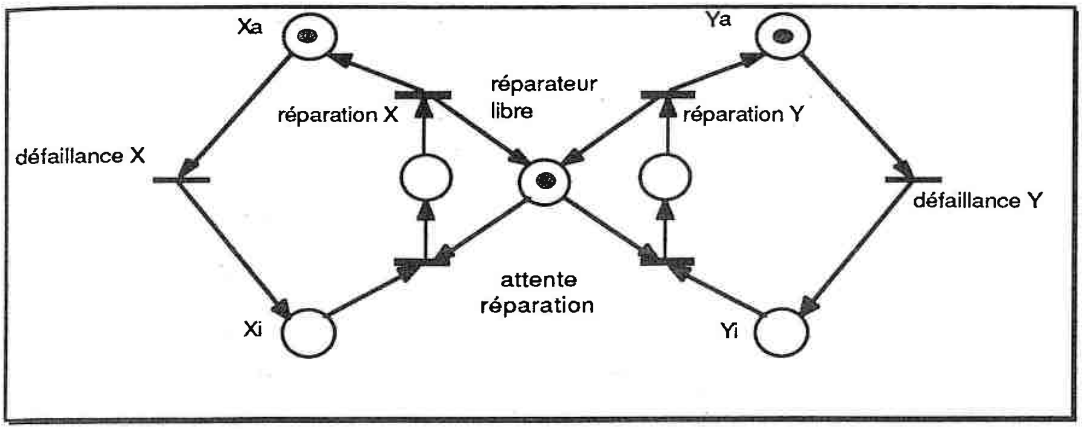
et les arcs mettent en relation les places vers les transitions et réciproquement.

A chaque place, on peut associer un marquage représenté par des jetons à l'intérieur des places. Le graphe des marquages du réseau de Petri est équivalent au modèle Markovien.




*le réseau de Petri stochastique d'un système à deux composants avec chacun un réparateur*

Le RdP permet de représenter facilement différentes politiques de réparations: [Lap 88]




*le réseau de Petri stochastique d'un système à deux composants avec un seul réparateur partagé FIFO*

### 3.8.2. Apports et Avantages

Les RdP sont applicables au niveau des spécifications. Ils permettent de prendre en compte les problèmes de parallélisme et de concurrence. Le traitement informatique est possible et il existe de nombreux logiciels.

### 3.8.3. Limites et Difficultés

La complexité des graphes des marquages limite leur utilisation à des sous-ensembles de systèmes complexes et impose l'utilisation d'outils de manipulation de graphes. Leur utilisation est donc généralement réservée à l'étude du comportement dynamique d'un système.

## CONCLUSION

De cet examen des méthodes d'analyse et d'évaluation de la sûreté de fonctionnement d'un système complexe, il ressort qu'une approche qualitative est plus adaptée pour les avant-projets de conception d'architecture. La principale raison de cette adaptation réside dans la finalité même de l'analyse de la robustesse aux défaillances d'une architecture: la prise en compte de la sûreté de fonctionnement n'a, dans le contexte avant-projet, pas pour but de calculer un taux de fiabilité ou de disponibilité précis, mais d'assurer la "qualité" de la conception. Cette "qualité" sera satisfaisante dans la mesure où l'analyse de la robustesse aux défaillances permet de s'assurer que, non seulement l'architecture supporte convenablement les fonctions liées au comportement "normal" de l'application (c'est-à-dire en l'absence de défaillances), mais aussi se comporte de façon satisfaisante en présence de défaillances. Dans cette perspective, l'analyse de la robustesse est dans le processus de conception le complément logique de l'analyse fonctionnelle.

La seconde raison est la difficulté d'obtenir des résultats significatifs si l'on ne dispose pas des données quantitatives élémentaires nécessaires au calcul des taux de fiabilité ou de disponibilité. Dans le cas où ces informations sont disponibles, l'approche quantitative étudiée par Lesage [Les 89] à l'aide de graphes de Markov donne des résultats intéressants. Une telle approche est plus adaptée à des études comparatives approfondies de différentes options fondamentales d'architecture, en particulier pour le développement de systèmes matériels ou dans des domaines où la sécurité joue un rôle primordial. Pour des études d'ingénierie plus ponctuelles dont l'objectif principal n'est pas l'étude de la sûreté de fonctionnement mais la conception d'une solution par intégration de systèmes existants, nous préconisons une démarche proche de l'Analyse des Modes de Défaillance et de leurs Effets, mais adaptée et assistée pour en éviter certains inconvénients. La présentation de cette démarche fait l'objet du chapitre suivant.

# chapitre 9:

# Méthode proposée: Procédures d'Analyse et Mesures Correctives

## 1. ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES D'UNE ARCHITECTURE DE SAP

Le paragraphe suivant synthétise le principe de la démarche retenue pour l'étude la robustesse aux défaillances d'une architecture de SAP avant d'aborder le détail des procédures d'analyse.

### 1.1. Le Principe de la Démarche Retenue

La démarche retenue repose sur deux tâches d'analyse et de spécification de solutions correctives parcourues selon un processus itératif. L'analyse est initialisée sur une première ébauche de l'architecture et aboutit, après une succession d'analyses et de corrections, à une solution présentant une qualité satisfaisante de robustesse aux défaillances.

Dans la majorité des cas où l'étude de la sûreté de fonctionnement d'une architecture de SAP ne nécessite pas une étude approfondie mettant en oeuvre des techniques classiques, la démarche des concepteurs consiste d'abord à suivre une démarche d'analyse:

**envisager successivement, sans chercher à dresser un inventaire exhaustif de tous les cas de pannes,**

- **le non-fonctionnement des dispositifs de traitement,**
- **le non-fonctionnement des dispositifs de communication,**

**et évaluer les conséquences de ces défaillances sur l'ensemble du système.** La démarche est donc inductive, partant des défaillances élémentaires des composants d'une première ébauche d'architecture opérationnelle, pour évaluer les effets sur l'ensemble du système. Or cette analyse, relativement lourde, est rarement effectuée car les relations de dépendance entre les composants de l'architecture opérationnelle d'un SAP sont nombreuses et mal explicitées. Notre contribution propose d'évaluer la faisabilité d'un outil supportant cette analyse. En effet cette analyse est délicate à effectuer sur l'architecture opérationnelle où n'apparaissent pas les relations de causalité ou de dépendance entre les composants. La démarche méthodique que nous avons proposée conduit à élaborer le modèle de structuration fonctionnelle du SI du SAP. Or ce modèle représente explicitement les interactions entre les entités fonctionnelles et doit donc

permettre la recherche "systématique" des effets de défaillances choisies par le concepteur. En d'autres termes, la recherche des effets des défaillances peut être effectuée par une analyse du graphe que constitue le modèle de structuration fonctionnelle. Ceci permet d'évaluer qualitativement l'importance relative d'un composant sur le bon fonctionnement de l'ensemble du système. Un tel outil nous apparait indispensable pour envisager cette analyse, qui effectuée "manuellement" s'avère fastidieuse pour les études d'architecture couramment pratiquées.

La démarche complémentaire à cette analyse porte sur l'**amélioration du comportement du système en présence de défaillances** (l'amélioration de sa "**robustesse**") **en renforçant la fiabilité ou l'autonomie des différents composants du système**. Ceci est en effet le moyen le plus simple d'éviter la "propagation" d'une défaillance élémentaire à l'ensemble du système. La robustesse des sous-systèmes peut être améliorée en prenant différentes décisions de conception:

- en optimisant les choix de **répartition** des entités fonctionnelles sur les différents équipements de l'architecture opérationnelle,
- en définissant des **taux de fiabilité** suffisants pour ces équipements, et en choisissant au besoin des **solutions redondantes,**
- en définissant les **modes de fonctionnement dégradés** de l'installation.

L'outil envisagé devra donc permettre la définition de ces différentes solutions correctives et leur prise en compte dans la procédure automatique d'analyse des effets des défaillances afin que le concepteur visualise les gains de robustesse procurés par ses différentes décisions.

### 1.1.1. approche opérationnelle et fonctionnelle

Deux niveaux de modélisation de l'architecture sont nécessaires. Du point de vue de l'analyste-concepteur, les modes de défaillances sont d'abord envisagés en terme de défaillance des équipements de l'architecture opérationnelle. Les concepteurs sont plus habitués à raisonner en termes de pannes des équipements qu'en termes de disponibilité de fonctions, ceci pour mieux percevoir la probabilité d'apparition des défaillances. Dans cette approche l'effet global devra pouvoir être interprété également au niveau de l'architecture opérationnelle. Etant données les contraintes de la conception d'une architecture de SAP identifiées dans la première partie de ce document (délais et informations disponibles), l'étude de la sûreté de fonctionnement limite le niveau de détail de l'analyse aux "macro-composants" de l'architecture opérationnelle: on ne précise pas la nature exacte des calculateurs ou automates, des systèmes de communication, et encore moins le détail des composants ou cartes électroniques internes des dispositifs informatiques [Dev 88].

Une telle analyse menée à partir des défaillances des composants opérationnels intègre de façon originale "l'approche élémentaire" menée au niveau des éléments matériels et "l'approche fonctionnelle" habituellement dissociées [Hou 87]. L'analyse de défaillances directement sur un modèle fonctionnel est pertinente dans une approche exhaustive. La possibilité de déclarer les défaillances au niveau des entités de la structure fonctionnelle permettra de suivre l'une ou l'autre de ces approches. Cette distinction n'est toutefois pas fondamentale dans le cadre notre approche si l'on considère que nos hypothèses permettent la transposition directe entre le niveau opérationnel et le niveau fonctionnel.

Le détail de l'analyse ne peut être effectué qu'à partir de la structure fonctionnelle. Cette structure est en effet indispensable pour identifier les relations de dépendance existant entre les différents composants de l'architecture. Raisonner uniquement sur l'architecture opérationnelle revient en fait à reconstituer mentalement la partie de la structure fonctionnelle correspondant à la chaîne de causalité du sous-système étudié. L'interprétation fine de cette chaîne de causalité et la décision des solutions correctives à apporter ne peuvent donc être effectuées correctement sans la connaissance de la structure fonctionnelle. Quelque soient les défaillances des équipements de l'architecture opérationnelle, **toute défaillance sera interprétée comme l'indisponibilité d'un certain nombre d'informations dont il s'agit d'évaluer les effets.** Nous considèrerons la "propagation" des défaillances du point de vue de la disponibilité des informations: les liaisons fonctionnelles d'échanges d'informations constituent les relations de causalité permettant d'induire les effets d'une défaillance. Le modèle de structuration fonctionnelle constitue donc un graphe de ces relations de dépendance: les sommets étant les entités fonctionnelles et les arcs étant les liaisons d'échange d'informations.

### 1.1.2. exhaustivité

Cette approche est comparable aux AMDE ou AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité) dans la mesure où l'on cherche à déterminer les effets locaux et globaux de défaillances élémentaires pour définir les mesures correctives appropriées. Mais quand une analyse exhaustive n'est pas nécessaire, l'étude des différents modes de défaillance sera laissée à l'initiative du concepteur. Autrement dit, si une démarche AMDE n'est pas indispensable, le choix des défaillances étudiées dépendra du concepteur et de son expérience à détecter les cas les plus sensibles ou les plus probables. Ce choix peut sembler peu satisfaisant sur le plan théorique, mais est justifié par l'objectif de préserver l'efficacité du concepteur dans les études où la sécurité et la fiabilité ne sont pas des objectifs de premier plan. Cette démarche "à

exhaustivité variable" n'est pas en contradiction, mais complémentaire avec une approche strictement exhaustive: la même procédure d'analyse inductive peut être, suivant les besoins, employée de façon sélective ou exhaustive. L'outil peut donc le cas échéant être utilisé comme un outil d'aide à l'élaboration d'AMDE. D'autre part, si les AMDE ont pour vocation d'étudier uniquement les défaillances élémentaires, l'outil peut être utilisé pour étudier les combinaisons de défaillances.

Une approche inductive (causes --> effets) a été préférée à une approche déductive (effets --> causes) de type Arbre de Défaillances. Ceci ne signifie pas du tout que cette approche ne présente pas d'intérêt, mais sa mise en oeuvre dans le contexte des études préalables d'architectures pose certaines difficultés. Le choix des événements indésirables est une première difficulté. De plus il n'existe pas à notre connaissance d'études sur la construction automatique des arbres à partir des descriptions fonctionnelles et topologiques de systèmes d'information. La transposition des travaux menés à l'EdF sur les systèmes thermo-hydrauliques et électriques [Anc 87] n'est pas évidente et constitue un sujet de recherche en soi. Les travaux menés au LAAS sur la conception de logiciels sûrs de fonctionnement [Hou 87] proposent une élaboration pour l'instant "manuelle" des arbres de défaillance à partir de descriptions fonctionnelles formalisées avec SADT. Cette approche peut être directement appliquée en remplaçant ce formalisme de description par notre modélisation de la structure fonctionnelle.

## 1.2. Les Données de Base

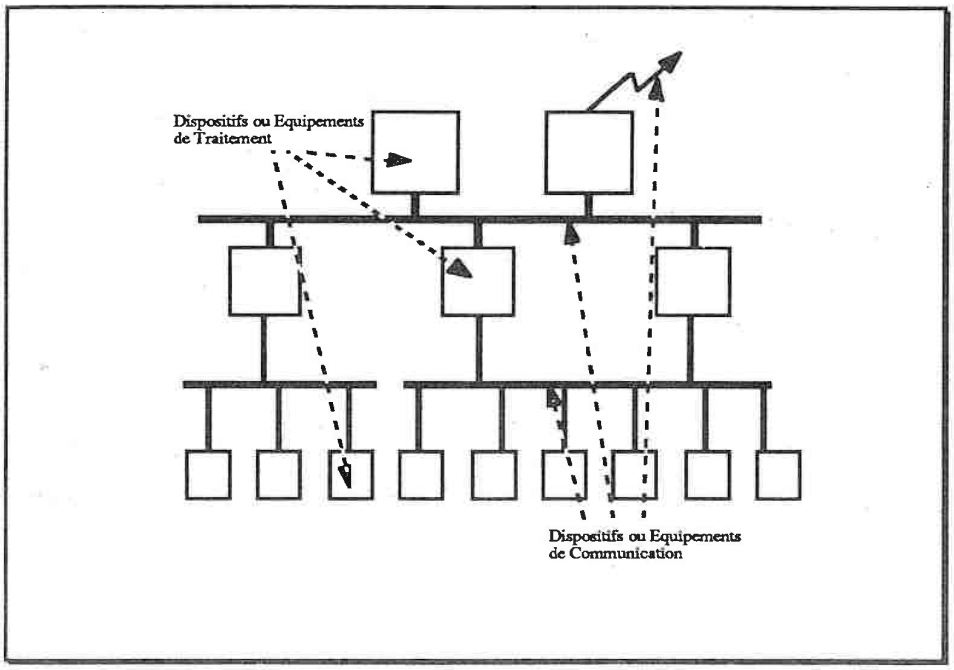
Les Données de Base sur lesquelles s'appuie l'analyse sont:

- une première version (donc partiellement définie) de l'architecture opérationnelle du système,
- la structure fonctionnelle, et
- la table de correspondance entre ces deux modèles représentant l'implantation des entités fonctionnelless sur les dispositifs de traitement de l'architecture opérationnelle.

Ces données de base seront donc fournies par l'utilisateur pour modéliser l'application étudiée. Afin de faciliter au maximum cette tâche préalable, la saisie sera effectuée au moyen d'une **interface interactive** supportant le formalisme **graphique** associé aux objets manipulés.

### 1.2.1. Architecture opérationnelle

L'architecture opérationnelle est constituée uniquement de dispositifs abstraits (ou équipements) de traitement et de communication. Ces dispositifs représentent respectivement soit des calculateurs, des automates et autres systèmes programmables, soit des systèmes de communication (réseau local industriel par exemple) qui seront choisis dans la tâche suivante de "Définition de solutions". Ce sont ces dispositifs qui supportent les fonctions et les échanges d'informations définis dans la structure fonctionnelle (cf paragraphe suivant).

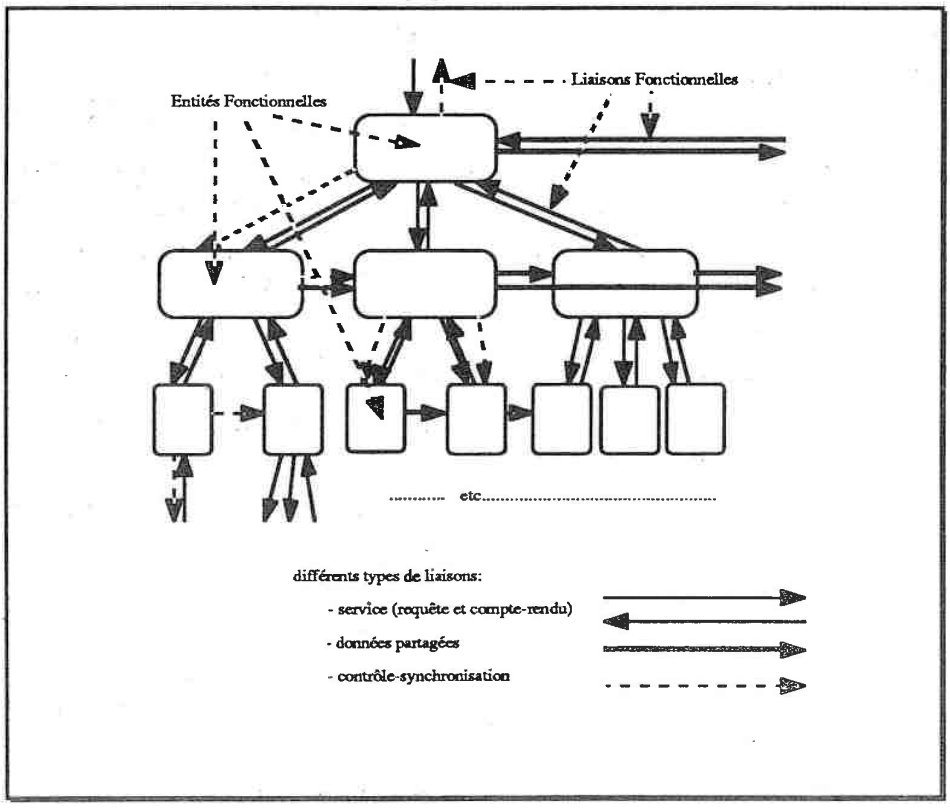



*un exemple d'Architecture Opérationnelle.*

La modélisation des dispositifs de stockage (disques et autres mémoires) n'est pas explicitée, mais est réalisée par les dispositifs de traitement dans la mesure où nous considérons seulement les échanges de données. Ce choix ne doit pas faire négliger l'importance des problèmes posés par la répartition des données dans une architecture distribuée.

### 1.2.2. Structure fonctionnelle

La structure fonctionnelle modélise le système d'information sous la forme de boîtes noires: les entités fonctionnelles, et des liaisons fonctionnelles de flux d'information. Comme il est nécessaire de connaître les échanges d'informations, les fonctions sont supposées être décomposées à un niveau de détail tel que chaque entité fonctionnelle sera implantée sur un seul dispositif de traitement et non pas éclatée sur plusieurs dispositifs.




*un exemple de Structure Fonctionnelle.*

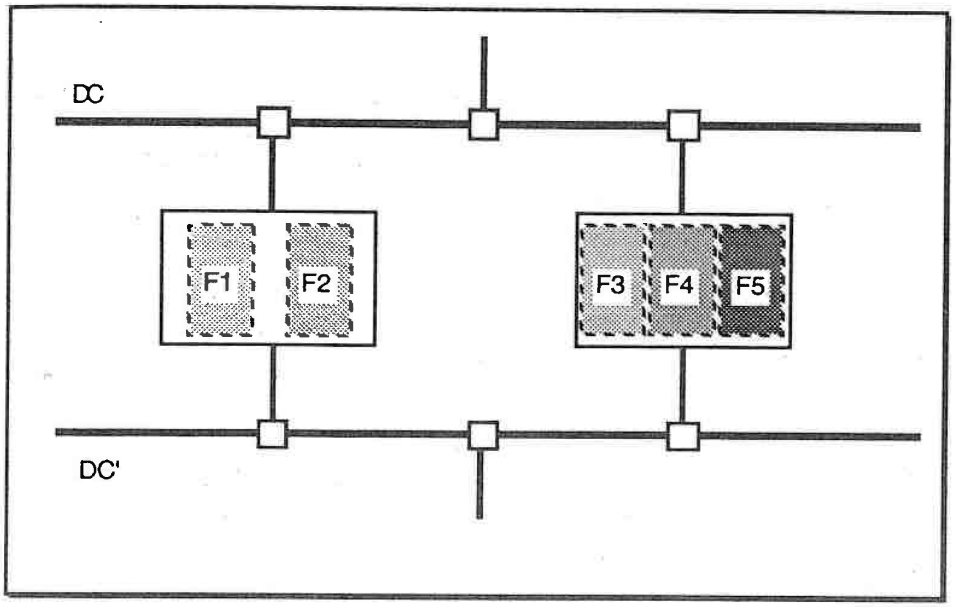
### 1.2.3. Correspondance entre les 2 modèles

La répartition de chacune des entités fonctionnelles sur les dispositifs de traitement est donc spécifiée au départ par une table des correspondances entre les éléments des deux modèles. Un dispositif de traitement peut supporter plusieurs entités fonctionnelles. Afin d'alléger la tâche du concepteur, la répartition des liaisons fonctionnelles sera dans la mesure du possible déduite de la répartition des entités fonctionnelles dans tous les cas où un seul dispositif de communication relie les équipements supportant les entités fonctionnelles. On suppose qu'une communication entre deux équipements ne peut pas passer par un équipement autre qu'un équipement de communication (une passerelle par exemple) dédiée à cet effet.

Plusieurs cas d'implantation des entités fonctionnelles peuvent se présenter:

- une liaison fonctionnelle entre deux entités fonctionnelles F1 et F2 implantées sur un même dispositif opérationnel de traitement ne sera donc pas supportée par un dispositif de communication,

- une liaison fonctionnelle entre deux entités fonctionnelles F1 et F3 implantées sur deux dispositifs opérationnels distincts pourra être supportée par l'un ou l'autre de deux dispositifs de communication DC et DC' reliant les dispositifs de traitement.



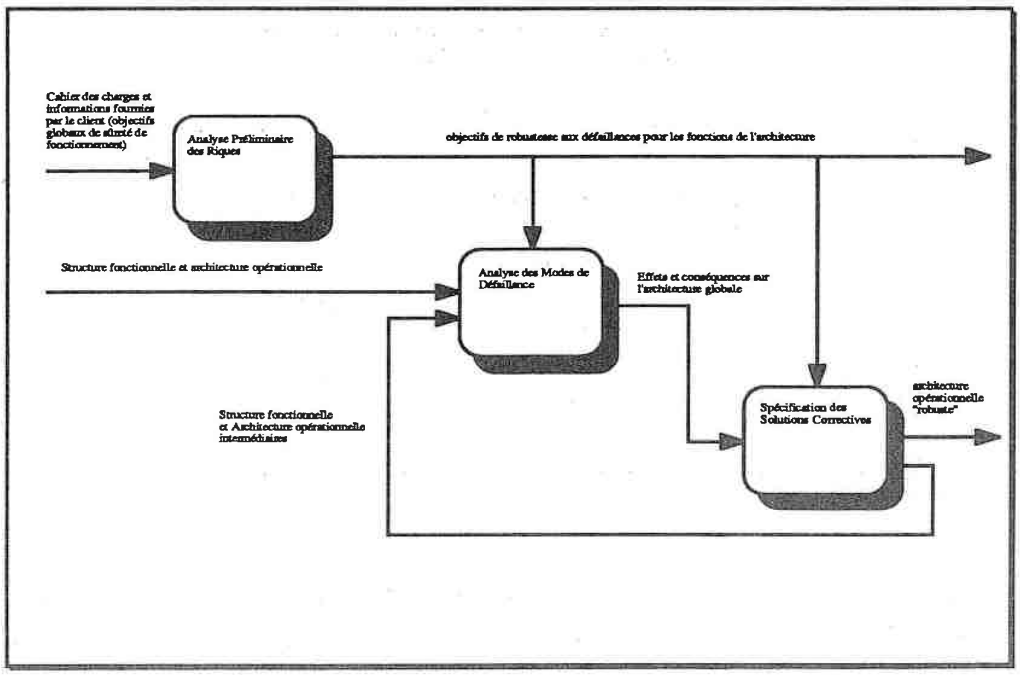

*un exemple d'implantation d'entités fonctionnelles sur des dispositifs de traitement d'une architecture opérationnelle*

## 1.3. Les Etapes de l'étude

L'étude de la robustesse aux défaillances d'une architecture peut être décomposée en trois étapes principales décrites dans les paragraphes suivants:

- Analyse Préliminaire des Risques mettant en évidence les points sensibles en fonction des contraintes du processus de production,
- Analyse des Modes de Défaillances étudiant les effets induits sur tout le système,
- Spécification des Solutions Correctives pour pallier ou limiter ces effets.

Les deux dernières étapes sont parcourues de façon itérative jusqu'à obtention d'une architecture satisfaisante pour le concepteur par rapport aux objectifs définis dans la première étape.




*les étapes de l'Etude de la Robustesse aux Défaillances d'une Architecture.*

### 1.3.1. Analyse Préliminaire des Risques

Avant d'aborder l'analyse détaillée des modes de défaillance, il est important de bien déterminer quels sont les objectifs de sûreté de fonctionnement des différents sous-ensembles fonctionnels de l'architecture. Une architecture étant un système particulièrement complexe et supportant de très diverses fonctionnalités, il n'est pas possible de définir un objectif global unique pour tout le système. Toutes les fonctions n'ont pas la même importance et les conséquences des défaillances de ces fonctions n'ont donc pas la même gravité (au sens défini dans les AMDE).

Il est donc nécessaire de préciser cette importance pour chaque sous-ensemble fonctionnel. La notion de hiérarchie utilisée dans le modèle fonctionnel permet de distinguer différents niveaux de sûreté de fonctionnement. Par exemple le niveau automatisme réflexe en contact direct avec le processus de fabrication doit généralement avoir une fiabilité et une disponibilité plus élevée qu'une fonction comme le suivi des temps de fabrication. Classiquement ce niveau est croissant quand on se rapproche du processus de fabrication, tant pour remplir des objectifs de sécurité que de disponibilité de l'appareil de production. Mais ceci n'est pas une règle générale car les objectifs de sûreté de fonctionnement d'un sous-ensemble du système d'information dépendent de ceux du sous-ensemble du processus de production correspondant. Si par exemple le sous-ensemble du processus de production considéré conditionne la disponibilité de toute l'installation (cas d'une chaîne linéaire de production), le système de commande correspondant devra atteindre un niveau de sûreté de fonctionnement tel qu'il n'entraîne pas la chute de la disponibilité de toute la chaîne de production. D'une façon plus générale on considère que le taux de disponibilité d'un sous-ensemble du système d'information doit être supérieur ou égal à celui du sous-ensemble du processus de production correspondant. Mais des machines pouvant supporter les mêmes opérations de fabrication et traitant en parallèle les flux de produits peuvent autoriser des possibilités de reconfiguration de la production. Dans ce cas il est donc aussi important pour la disponibilité de l'ensemble de l'installation de garantir la disponibilité de la fonction d'ordonnancement / lancement, capable de réaffecter la fabrication des flux de produits sur les machines disponibles, que de garantir la disponibilité des automatismes de chaque machine. Il est donc important de définir au préalable les objectifs de sûreté de fonctionnement de chaque sous-ensemble de l'installation en fonction de la topologie du processus de fabrication.

Au sein d'un même niveau hiérarchique, les objectifs de la sûreté de fonctionnement des différentes entités fonctionnelles peuvent différer en fonction du contexte processus de production. La définition des objectifs de sûreté de fonctionnement du système

d'information est donc indissociable de ceux du processus de production. En conclusion, l'Analyse Préliminaire des Risques est une étape préalable à l'Analyse des Défaillances. Elle permet donc principalement de faire la synthèse des objectifs de sûreté de fonctionnement à partir des informations fournies par le client. La procédure ne s'appuie pas sur une méthode particulière. Les références bibliographiques données dans le chapitre précédent n'ont pas fait apparaître de méthode rigoureuse pour cette phase préalable. Elle ne sera supportée par le prototype que par les attributs permettant de recueillir les résultats de cette analyse préliminaire.

### 1.3.2. Analyse des Modes de Défaillance

#### Approche

Il s'agit d'assister le concepteur dans une analyse inductive, donc partant des défaillances opérationnelles pour déterminer leurs conséquences fonctionnelles sur l'ensemble de l'application. L'approche retenue repose sur une analyse interactive. Le concepteur choisit une combinaison de défaillances élémentaires des composants de l'Architecture Opérationnelle. L'outil procède alors, sur la base de la structure fonctionnelle, à l'analyse de la propagation de ces défaillances jusqu'à la définition de l'effet global.

Deux types de défaillances sont étudiés:

- sur les dispositifs opérationnels de traitement,
- sur les dispositifs opérationnels de communication.

Ces défaillances dites "opérationnelles" sont issues de fautes physiques matérielles, autrement dit de pannes, ou de fautes logicielles. Contrairement aux études détaillées de sûreté de fonctionnement des systèmes informatiques [Hou 87], les risques de défaillance liés aux fautes de conception du logiciel ne sont pas distingués des risques de défaillance liés aux fautes matérielles des dispositifs de traitement ou de communication. Un dispositif de l'architecture opérationnelle sera donc considéré comme globalement défaillant. Les défaillances sont considérées [Coq 88]

- à rapidité de survenance soudaine (et non progressive),
- à étendue complète (et non intermittente ou partielle).

On considère que tout l'équipement est défaillant. En particulier pour les dispositifs de communication, une défaillance d'un des segments entraîne la défaillance de tout le dispositif. Une analyse plus fine des systèmes de communication est possible mais elle nécessiterait des informations supplémentaires concernant l'architecture matérielle: la topologie, les protocoles d'accès et autres caractéristiques de communication. De plus une analyse à un tel niveau de détail est habituellement pas pratiquée car elle n'est pas pertinente dans l'étude d'une architecture de SAP du point de vue de l'utilisateur, mais plutôt dans l'étude du système informatique en tant que produit du point de vue du développeur.

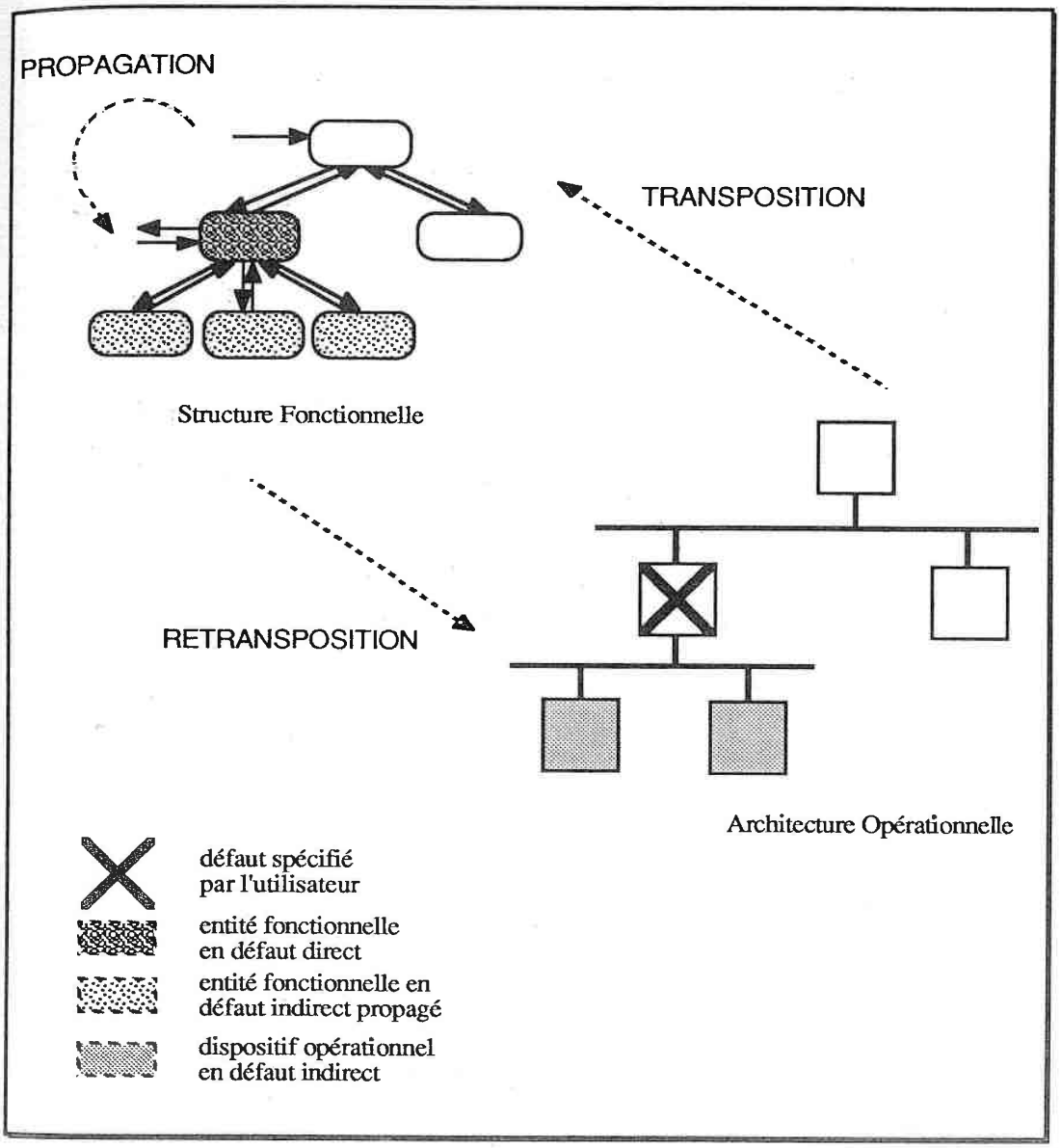
Au niveau fonctionnel, les entités ou liaisons fonctionnelles supportées par des équipements défaillants seront considérées comme étant "directement" en défaut ou défaillants. Ces défaillances "directes" des éléments de la structure fonctionnelle vont

induire des défaillances "indirectes" sur les éléments fonctionnels avec lesquels ils sont en interaction. Ceci suivant des lois dites de "propagation" décrites dans les paragraphes suivants.

### Description de la procédure d'analyse

Pour évaluer les effets locaux et globaux des défaillances sur l'ensemble de l'architecture, il est nécessaire de distinguer trois phases d'analyse (cf schéma):

- analyse de l'effet local: **"Transposition"** des effets des défaillances de l'Architecture Opérationnelle vers la Structure Fonctionnelle.
- analyse de l'effet global: **"Propagation"** des effets des défaillances au sein de la Structure Fonctionnelle,
- analyse de l'effet opérationnel: **"Retransposition"** des effets des défaillances de la Structure Fonctionnelle vers l'Architecture Opérationnelle.




*les trois phases de la procédure d'analyse des effets de défaillances*

L'objectif de l'analyse étant d'interpréter les conséquences de défaillances, il est cependant nécessaire d'examiner les effets à la fois sur la Structure Fonctionnelle et sur l'Architecture Opérationnelle.

**transposition**

Les règles de transposition permettent d'exprimer la dépendance des entités fonctionnelles (respectivement des liaisons fonctionnelles) liées aux relations d'implantation de ces éléments fonctionnels sur les dispositifs opérationnels de traitement (respectivement de communication). Les relations d'implantation étant définies au départ, ces règles s'énoncent simplement:

---

*si* un dispositif opérationnel de traitement est défaillant
*alors* toute entité fonctionnelle implantée dans le dispositif opérationnel de traitement est en défaut.

---

et

---

*si* un dispositif opérationnel de communication est défaillant
*alors* toute liaison fonctionnelle supportée par le dispositif opérationnel de communication est en défaut.

---

**propagation**

La transposition étant établie, les règles de propagation permettent d'exprimer les dépendances entre les éléments fonctionnels, et ainsi d'évaluer les conséquences induites sur l'ensemble du système. Ces relations de dépendance sont directement liées aux interactions fonctionnelles (que constituent les différents échanges d'information) pouvant être modélisées par le formalisme adopté à ce niveau:

- les services (requête et compte-rendu),
- données partagées et
- synchronisation.

Ces règles de propagation expriment que les services de toute entité fonctionnelle peuvent être considérés comme une production d'informations et que ces entités ne peuvent assurer leurs services qu'en possession d'informations nécessaires en entrée. Elles s'énoncent sous la forme générale:

---

*si* une entité fonctionnelle est en défaut
*alors* toute entité fonctionnelle consommatrice de données en provenance de cette entité fonctionnelle est en défaut.

---

*si* une liaison fonctionnelle est en défaut
*alors* toute entité fonctionnelle consommatrice de données en provenance de cette liaison fonctionnelle est en défaut.

Ces règles de propagation peuvent être comparées aux règles d'interaction des travaux sur la Génération de Séquences par Inférence menés à l'EDF [Bou 87]. A partir de telles règles générales, il faut détailler le comportement des entités en fonction du type de liaison fonctionnelle considérée. Des trois liaisons: services, données partagées et contrôle-synchronisation, seules les deux dernières correspondent à des flux unidirectionnels. La propagation de la défaillance d'une entité fonctionnelle par les liaisons de données partagées et de synchronisation se fait donc dans le sens des flux supportés par ces liaisons.

En revanche pour la liaison de service bidirectionnelle, il y a une dissymétrie entre les deux échanges d'information. L'indisponibilité d'une requête ou d'un compte-rendu de service ne conduit pas au même effet. La notion de service permet d'exprimer une relation hiérarchisée "d'utilisation": une entité de niveau supérieur ("utilisateur" ou "user") "utilise" les services d'entités de niveau inférieur ("fournisseur" ou "provider"). Cet "utilisateur" joue aussi le rôle de contrôleur-coordinateur, et définit les objectifs qui paramètrent les requêtes des services utilisés.

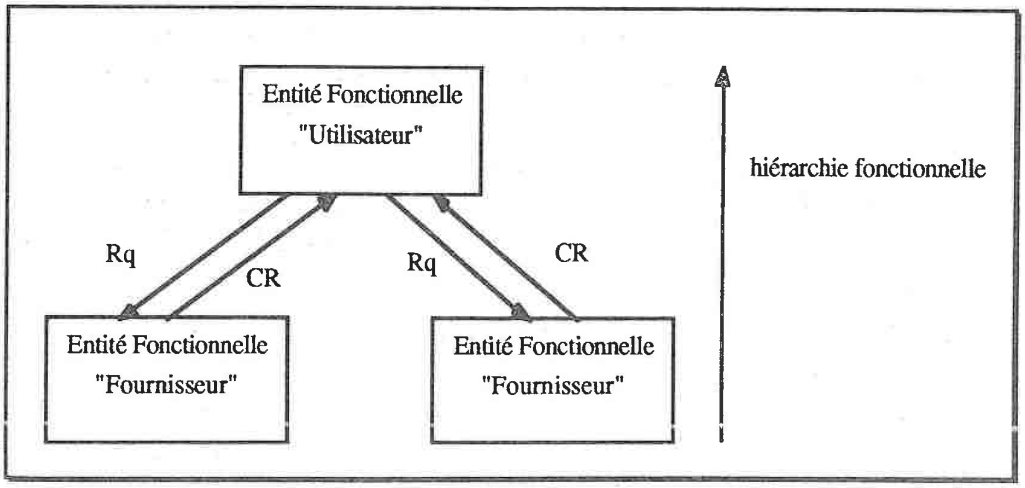



*Schéma de la Hiérarchie Fonctionnelle: "utilisateur-fournisseur".*

Cas où l'utilisateur est défaillant:

La défaillance de l'utilisateur entraîne donc la perte de la définition des objectifs vis à vis du service utilisé. Dans ce cas, l'entité fonctionnelle "fournisseur" de service ne

reçoit plus la requête du service qu'elle doit assurer. Le "fournisseur" n'est plus sous le contrôle fonctionnel de "l'utilisateur" (mode "hors-pilotage"). Nous conviendrons qu'il y a "propagation" de la défaillance dans un sens dit *"descendant"* de l'utilisateur vers le fournisseur du service (que cette défaillance provienne indifféremment de l'entité fonctionnelle "utilisateur" ou de la liaison de service elle-même). Pour pallier les effets de cette propagation, une solution corrective devra remplacer cette requête par une autre source d'information (cf paragraphe suivant).

Cas où le système de communication est défaillant:

Si l'absence de compte-rendu est dû à la perturbation de la liaison de service (c'est-à-dire à cause d'un problème réseau), la requête ne peut pas non plus être échangée, on est donc ramené au cas précédent: mode "hors-pilotage".

Cas où le fournisseur est défaillant:

La perte du fournisseur empêche l'entité fonctionnelle "utilisateur" de recevoir le compte-rendu du service qu'elle doit utiliser. Dans ce cas, le déroulement de la tâche en cours est éventuellement perturbé. Pour l'exécution de son service, l'entité fonctionnelle "utilisateur" doit changer de mode d'exploitation pour tenir compte de cette défaillance. Soit le service offert par le fournisseur est indispensable (pas de reconfiguration possible), soit le service d'un autre fournisseur est utilisable. Cette alternative dépend de la structure topologique du processus de production:

- soit le sous-ensemble du processus de production géré par l'entité fonctionnelle défaillante est indispensable à la continuité de la production,
- soit le sous-ensemble du processus de production géré par l'entité fonctionnelle défaillante peut être remplacé par un autre sous-ensemble géré par une autre entité fonctionnelle.

Dans le premier cas, aucune reconfiguration de l'exploitation des installations de production n'est possible, et la propagation de la défaillance va se faire suivant le processus de production. Mais les entités fonctionnelles ne peuvent pas à proprement parler être considérées comme défaillantes à cause de la défaillance de leur fournisseur.

Dans le deuxième cas, une reconfiguration de l'exploitation des installations de production est possible. Cette notion de mode d'exploitation sort du cadre des modes de défaillances car elle rentre dans la définition interne d'une entité fonctionnelle "utilisateur".

Donc nous considérons qu'il n'y a pas à proprement parler de propagation "ascendante" des défaillances. Ce choix s'explique en d'autres termes par l'intérêt d'étudier les effets des défaillances de l'architecture du système d'information du point de

vue du processus de fabrication, donc plutôt dans le sens "descendant", par préférence au point de vue des entités externes au SAP utilisant les informations qu'il fournit.

**retransposition**

Les règles de retransposition permettent de représenter les effets propagés des défaillances fonctionnelles sur les éléments de l'architecture opérationnelle. Les relations d'implantation de ces éléments fonctionnels sur les dispositifs opérationnels de traitement permettent d'exprimer cette retransposition simplement:

---

*si* une entité fonctionnelle est en défaut
*alors* le dispositif opérationnel de traitement supportant cette entité fonctionnelle est perturbé.

---

### 1.3.3. Spécifications des mesures correctives

Les règles génériques précédemment évoquées permettent d'analyser une architecture qui ne fait que supporter les fonctions du système sans apporter d'amélioration de sûreté de fonctionnement. Pour limiter les effets des défaillances, trois principales classes de solutions peuvent être prises :

- **modifier les modèles** de Structure Fonctionnelle ou d'Architecture Opérationnelle, ou la répartition des entités fonctionnelles sur les dispositifs opérationnels,
- **spécifier le niveau de sûreté de fonctionnement** des dispositifs opérationnels,
- prendre des **solutions correctives fonctionnelles** pour pallier les inévitables défaillances des composants de l'architecture opérationnelle.

La première classe de solutions revient à **modifier les données en entrée**. Les modifications de l'architecture opérationnelle (ajouts ou retraits de dispositifs opérationnels, modification de la structure de l'architecture réseau) et les modifications des choix de répartition des entités fonctionnelles sur les dispositifs opérationnels constituent des mesures courantes prises au vu de l'analyse des défaillances. Dans certains cas la structure fonctionnelle peut elle-même être revue et corrigée par une itération sur la tâche de Structuration Fonctionnelle.

Par exemple une liaison de synchronisation "horizontale", établie entre deux entités fonctionnelles de même niveau hiérarchique assurant des automatismes, entraîne une relation de dépendance qu'il peut être intéressant de remplacer (quand c'est possble) par une synchronisation basée sur un événement procédé et détecté par le niveau inférieur.

Ce type de mesure ne nécessite pas de traitement particulier au niveau de notre outil, les règles précédemment définies étant toujours applicables.

La deuxième classe de solutions consiste à spécifier les taux de disponibilité des dispositifs matériels pour **limiter la probabilité d'apparition des défaillances.** Ces taux de disponibilité peuvent conduire jusqu'au choix de systèmes redondants tant pour les matériels de traitements (automates, calculateurs) que les matériels de communication. Les techniques de redondance mises en oeuvre peuvent être fournies par les fournisseurs d'équipements, le système d'exploitation prenant en charge la gestion de la redondance. La redondance peut aussi être réalisée spécifiquement au niveau de l'application. Il est évident que la mise en oeuvre de telles solutions est très onéreuse et doit donc faire l'objet d'une étude technico-économique pour justifier de tels investissements. En particulier il est très important de comparer les objectifs de disponibilité visés pour le système informatique avec ceux attendus pour les composants du processus de production.

Les règles de transposition deviennent donc:

---

*si* un dispositif opérationnel de traitement est défaillant
et *si* **aucune solution redondante n'est spécifiée pour ce dispositif**
*alors* toute entité fonctionnelle implantée dans le dispositif opérationnel de traitement est en défaut.

---

et

---

*si* un dispositif opérationnel de communication est défaillant
et *si* **aucune solution redondante n'est spécifiée pour ce dispositif**
*alors* toute liaison fonctionnelle supportée par le dispositif opérationnel de communication est en défaut.

---

La troisième classe de solutions consiste à **remplacer les informations manquantes par des informations produites par une autre source adaptée.** Ces solutions correctives se situent au niveau fonctionnel, et doivent pouvoir être modélisées au niveau du modèle fonctionnel. Ces solutions sont assimilables à la notion de modes de marche et arrêt par analogie avec les notions définies dans le GEMMA [GEM]. On distingue généralement les modes de reprise suivants:

- la **conduite locale** par opérateur,
- le fonctionnement en **position de replis** (paramètres de conduite par défaut),

- l'arrêt en **position de sécurité.**

La conduite locale par un opérateur correspond à des modes de marche parfois qualifiés de "semi-automatique" ou "manuel" suivant le niveau hiérarchique auquel l'opérateur intervient. Par rapport aux liaisons hiérarchisées de type service, cette approche des modes de reprise correspond à l'organisation classique des "modes de marche" où un niveau supérieur défaillant est remplacé par une reprise locale assurée par un opérateur.

Les modes de marche et arrêt appliqués au niveau fonctionnel s'accompagnent généralement d'une "dégradation" des services dans la mesure où les informations de remplacement ne sont a priori pas aussi précises, optimisées ou aussi flexibles. (Sinon on serait en droit de s'interroger sur l'apport des fonctions d'origine qui élaborent les informations.) Quand, pour satisfaire des objectifs de sécurité ou de productivité, le dispositif de production ne tolère aucune dégradation, il est nécessaire de choisir une solution de type redondance.

les règles de propagation deviennent donc:

---

*si* une entité fonctionnelle est en défaut
*alors* toute entité fonctionnelle consommatrice de données en provenance de cette entité fonctionnelle et pour lesquelles aucun mode de reprise n'est prévu est en défaut.

---

*si* une liaison fonctionnelle est en défaut
*alors* toute entité fonctionnelle consommatrice de données en provenance de cette liaison fonctionnelle et pour lesquelles aucun mode de reprise n'est prévu est en défaut.

---

### 1.3.4. Démarche de spécification

Les mesures correctives sont vues comme le moyen de stopper la propagation des défaillances dans l'application. Les règles dites de propagation doivent donc tester si des mesures correctives existent et établir les conséquences éventuelles adéquates. Cette approche induit donc une **démarche itérative d'analyse-conception** convergeant vers l'amélioration progressive de la robustesse aux défaillance d'une architecture donnée.

Le choix des mesures correctives a été laissé à l'initiative du concepteur. Cette démarche pourrait être assistée par l'outil en utilisant la notion de criticité des AMDEC. En fonction des risques de non-détection, de fréquence de probabilité d'apparition et de

gravité des conséquences, une pondération serait définie. Il serait donc possible de recommander un type de mesure corrective pour un niveau de criticité donné. Les objectifs sécurité et disponibilité étant généralement contradictoires, des stratégies de reprise des défaillances différentes pourront être mises en oeuvre.

## Conclusion

La démarche proposée pour l'analyse des défaillances d'une architecture de SAP est basée sur l'exploitation du modèle de structure fonctionnelle. En ce sens elle constitue un exemple particulièrement démonstratif de procédure de transformation d'un modèle fonctionnel vers un modèle opérationnel. La progression vers le modèle opérationnel s'effectue par la spécification de solutions correctives aux effets de défaillances, de type définition de solutions redondantes ou spécification de modes de fonctionnement dégradés. L'implantation de cette procédure a été réalisée à l'aide d'un générateur de système expert. La présentation de la maquette se trouve en annexe 2 de ce document.

## CONCLUSION DE LA TROISIEME PARTIE

Nous avons dans cette troisième et dernière partie étudié une procédure de transformation d'un modèle fonctionnel vers un modèle opérationnel. Cette procédure a pour but de montrer comment améliorer l'étude et la prise en compte du point de vue de la sûreté de fonctionnement au cours du processus de conception. Nous avons d'abord examiné quelles étaient les méthodes existantes dans ce domaine et quelle était leur adaptation au problème posé. L'approche proposée pour l'analyse de la sûreté de fonctionnement d'une architecture est donc essentiellement orientée vers la conception d'une solution satisfaisante en terme de comportement en présence de défaillances. Cet aspect est logiquement complémentaire avec la définition des fonctions établie lors de la structuration fonctionnelle en l'absence de toute considération de défaillance. Dans la mesure où l'objectif principal de la conception d'architecture n'est pas d'évaluer le taux de fiabilité ou de disponibilité de l'installation, il n'est pas nécessaire de mettre en œuvre les techniques quantitatives traditionnelles des études de sûreté de fonctionnement. L'étude qualitative assistée par l'exploitation "automatique" du modèle fonctionnel, est dans ce contexte plus adaptée. La maquette d'outil réalisée à l'aide du générateur de système expert G2 et présentée en annexe démontre la faisabilité de cette approche.

## CONCLUSION GENERALE

En conclusion de ce travail de recherche nous voudrions souligner les résultats obtenus et l'apport original de notre contribution, les prolongements d'ores et déjà envisagés et les perspectives à plus long terme.

Au niveau des **résultats** obtenus, notre projet s'est concrétisé par l'élaboration d'une démarche méthodologique définissant les tâches, activités, et modèles utilisés dans le processus de conception des architectures de SAP, et par le développement d'une maquette démontrant la faisabilité d'un outil basé sur une méthode originale d'analyse qualitative de la Robustesse aux Défaillances d'une Architecture.

Bien que le domaine général de l'Automatisation de Production fasse l'objet de nombreux travaux, le sujet traité n'avait pas, au moment du démarrage du projet (1985), été abordé en tant que sujet de recherche appliquée. Ceci dans la mesure où il traite:

- la Phase d'Avant-Projet de Conception d'Architectures, et non pas la phase de développement du logiciel,
- le point de vue de l'Ingénierie d'intégration de systèmes à base de produits matériels et logiciels disponibles sur le marché, et non pas le développement de systèmes matériels,
- le Système d'Information global du Système Automatisé de Production, et non pas seulement les automatismes, la conduite, le suivi de fabrication ou la gestion de production, etc...

Les difficultés de mener à bien ce projet ont donc été:

- la nécessité d'identifier, de classifier et intégrer de nombreux aspects et points de vue concernés par la conception d'architectures,
- la nécessité d'effectuer une large recherche bibliographique sur des spécialités très diverses (méthodes de conception de systèmes, modèles d'architecture de SAP, sûreté de fonctionnement, acquisition de connaissances),
- la nécessité d'approfondir suffisamment chacun de ces points pour obtenir des éléments de réponse concrets à un problème industriel.

La démarche méthodologique est d'abord basée sur la définition précise de modèles formalisant les objets manipulés par les concepteurs aux différents stades de la conception d'architectures. Nous avons montré qu'un modèle simple mais générique permettait de représenter explicitement les fonctionnalités de l'application en respectant les schémas de structuration utilisés mentalement par les concepteurs dans leur raisonnement. Un tel modèle permet de s'assurer de la complétude en intégrant seulement les informations pertinentes nécessaires à la conception d'architecture, et non pas toutes celles qui

concernent le développement de l'application. Il permet également d'effectuer différentes vérifications de cohérence de l'expression des besoins. A partir de cette représentation formelle, il est possible d'aborder de façon rigoureuse, et non plus seulement par analogie à des réalisations antérieures, la conception de solutions opérationnelles satisfaisant les qualités attendues par le demandeur: performances, sûreté de fonctionnement, évolutivité, satisfaction de différentes contraintes, etc... Cette phase permet d'aboutir selon un processus itératif à une ou plusieurs solutions respectant un compromis technico-économique et satisfaisant les objectifs et contraintes du prescripteur.

Cette démarche a été étudiée sur des applications très différentes dans le domaine de la production continue (Mouvements de Produits en Raffinerie) et discontinue (Découpe de Verre Plat). Elle a également été mise en oeuvre dans les domaines du transport de passagers (Projet MAGGALY) et de la gestion technique (Centre d'Essais) par Christian Verlinde [Ver 89]. Il semble donc qu'elle soit suffisamment générique pour être applicable à la conception d'architectures dans de nombreuses catégories d'applications industrielles.

Une des qualités les plus mal appréhendées dans ce domaine s'est avérée être la sûreté de fonctionnement. La difficulté d'utiliser dans ce contexte particulier les méthodes traditionnelles existantes, nous a conduit à proposer une démarche originale dont la faisabilité à été démontrée par le développement d'une maquette. Ce développement a de plus permis d'évaluer les possibilités de maquettage offertes par un environnement de développement de type Générateur de Système Expert (G2). Cet outil combine une structure orientée objets, différentes possibilités de structuration de la base de règles, et surtout une interface développeur et utilisateur évoluée.

S'il fallait résumer l'apport original de notre contribution, nous dirions qu'il se situe dans la formalisation des connaissances concernant d'abord la modélisation des architectures de SAP mais surtout concernant les procédures de transformation permettant de passer d'un modèle de représentation fonctionnelle à une architecture opérationnelle. En l'occurence nous avons approfondi les procédures basées sur la prise en compte de la sûreté de fonctionnement et la définition de solutions de reprise de type redondance ou spécification de mode de fonctionnement dégradé.

Les **prolongements** de ce travail sont d'ores et déjà entrepris. Une action de R&D est menée au sein de la société ESIA pour établir différents documents de référence concernant la Méthodologie d'Avant-Projet de Conception d'Architectures. Ces documents permettront de standardiser la conduite et le contenu des Etudes d'Avant-Projet, avec bien sûr la souplesse d'adaptation nécessaire à la diversité des études. Ils

doivent permettre de rendre opérationnels les résultats de notre travail de recherche en les complétant notamment par la partie technico-économique (chiffrage, évaluation, choix de solution). Cette action est menée avec la participation d'utilisateurs représentant chaque service concerné par ce type de prestations. Elle prend place dans le cadre plus général d'un programme d'Assurance Qualité ayant conduit la société à se doter d'une méthode pour le développement des logiciels d'automatisme et d'informatique industrielle.

Au delà de ces résultats, nous souhaitons positionner notre contribution, évidemment partielle, comme un travail exploratoire sur un aspect industriel de l'automatisation de production longtemps méconnu des équipes de recherche universitaires. Cet aspect est heureusement de plus en plus abordé dans des projets nationaux ou internationaux touchant les domaines des SAP ou du CIM et basés sur un partenariat Recherche-Industrie. Mentionnons l'action de la commission "SAP" de l'EXERA à travers une enquête auprès des utilisateurs prévue pour 1989 sur la définition d'une liste classifiée des besoins à recenser dans une étude d'architecture. Nous voudrions aussi souligner l'importance d'intégrer à terme cette phase de conception d'architecture dans les travaux effectués sur les environnements de développement, parfois appelés "Ateliers de Génie Automatique ou Productique", ou encore "Postes de Travail de l'Automaticien". Une architecture mal définie en amont entraîne de nombreuses difficultés dans le développement et la mise en oeuvre des logiciels d'application. Pour mieux maîtriser le processus de conception d'architecture, certains sujets devront faire également l'objet de recherches plus approfondies :

- la répartition et le partage de données dans une architecture de SAP: prise en compte au cours de la conception, services de communication de niveau application (couche 7) et produits bases de données réparties et temps-réel,
- influence des réseaux locaux industriels de premier niveau appelés "bus de terrain" sur la structure des architectures de SAP.

La nouveauté du sujet et la volonté d'obtenir des résultats concrets susceptibles de déboucher rapidement sur une mise en oeuvre opérationnelle, nous ont conduit à suivre une démarche basée sur l'étude des connaissances utilisées par les experts en Conception d'Architectures, l'intégration de résultats de recherches menées au CRIN, et l'étude bibliographique des connaissances issues de méthodes et modèles adaptés ou adaptables à ce contexte. Il est intéressant de remarquer qu'un tel sujet, d'une part abordé à l'origine comme un problème de méthodologie, et d'autre part nécessitant la prise en compte non seulement d'informations techniques mais aussi d'objectifs, besoins et contraintes plus difficilement formalisables, a naturellement orienté les travaux vers une démarche qualifiable a posteriori d'acquisition de connaissances. Cette démarche a permis de ne pas

se contenter de plaquer une méthode déjà existante pour la conception de système d'information à ce contexte, même si au premier abord, des similitudes apparaîssent de façon évidente. Un savoir-faire n'ayant pas encore été abordé en tant que tel a ainsi pu être en partie explicité. La disponibilité d'un environnement évolué a joué un rôle catalyseur dans le rapprochement d'une démarche orientée "méthode" vers une démarche orientée "connaissances".

# BIBLIOGRAPHIE:

[Abe 88]: ABEL J.,ABEL C.: "Une méthodologie pour le développement des systèmes experts s'appuyant sur la notion de modèle sous-jacent." 8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[Anc 87]: ANCELIN C., LE P., DE SAINT-QUENTIN S.: "Express: la fiabilité des systèmes complexes & les techniques de l'intelligence artificielle."RGN n°4, Juillet-Août 1987

[Anc 88]: ANCELIN C., LE P., DE SAINT-QUENTIN S., VILLATE N.: "A knowledge-based approach for systems safety and reliability analysis."Conférence internationale sur l'interface homme-machine dans l'industrie nucléaire. TOKYO JAPON 15-19 Février 1988

[AnC 87]: ANG C.L.: "Technical planning of factory data communications systems."Computers in Industry, n°9 , p.93, 1987

[Aye 88]: AYEL Marc, CHEIN Michel, PIPARD Eric, ROUSSET Marie-Christine: "De la cohérence dans les bases de connaissances."Actes de Journées Nationales Intelligence Artificielle, PRC-GRECO, Editions Teknea, Toulouse 14-15 Mars 1988

[AFN 84]: "Analyse de la Valeur, vocabulaire." norme AFNOR X50-150

"Guide pour l'élaboration d'un cahier des charges fonctionnel (expression fonctionnelle du besoin)." norme AFNOR X50-151

"Analyse de la Valeur, caractéristiques fondamentales." norme AFNOR X50-152

"Analyse de la Valeur, recommandations pour sa mise en ouevre." norme AFNOR X50-153

[Bal 86]: BALLY P. , LOUET M.: "La sécurité globale d'un programme industriel: la méthode ARP analyse des risques de programme."1986

[Bar 88]: BARBET Jean-François: "Les études de fiabilité des systèmes. "Génie Industriel, n° 170., Mai-Juin 1988

[Bar 88]: BARTHES Jean-Paul: "Intelligence Artificielle: systèmes-experts et temps-réel."Journées de l'Automatique 1988, Rhone Poulenc Industrialisation, Décines, Janvier 1988

[Bar 87]: BARTHES Jean-Paul: "Expert-Systems: Applications to chemical industry."Formula 87 Physico-chimie de la formulation et applications, 1er forum international, Nice, Février 1987

[BeG 86]: BEL Gérard, CAVAILLE Jean-Bernard, LEVEQUE Didier: "OASYS: une nouvelle approche de la simulation de systèmes d eproduction."Rapport interne Valoris-SGN, 1986

[Bel 88]: BELISLE Pierre, JANSSON Hart: "OSI-What is next."1988

[Beo 88]: BEOUNES Christian: "L'analyse qualitative de la sûreté de fonctionnement des matériels et logiciels: méthodes et outils."Séminaire DINOV "Analyse et évaluation de la sûreté de fonctionnement des matériels et des logiciels." Toulouse, Avril 1988

[Ber 84]: BERNAD Jacques: "Approche systémique des structures d'entreprises."Travail et Méthodes, n° 419 et 420, mai, juin-juillet 1984

[Ber 85]: BERTOMEU J.M.: "Simulation de fiabilité de systèmes automatisés."AFCET AUTOMATIQUE 85- Toulouse, 1985

[Bez 84]: BEZIVIN Jean: "Simulation et langages orientés objets."Journée L.O.O. de Brest, Revue Bigre n°41, Novembre 1984

[Bie 88]: BIEMANS F.P., VISSERS C.A.: "Computational tasks in robotics and factory automation."Computers in Industry , vol.10 .n° 2 , Juillet 1988

[Bie 86]: BIEMANS Franck: "Reference model of production control systems."IECON'86, 1986

[Bie 86]: BIEMANS Franck: "The design of distributed transport systems as a major standard interface in Computer Integrated Manufacturing."Computers in Industry , vol.7 .n° 4., 1986

[Bie 86]: BIEMANS Franck, BLONK Pieter: "On the Formal Specification and Verification of CIM architectures using LOTOS."Computers in Industry , vol.7 .n° 4. , 1986

[Bin 84]: BINDER Z.: "Commande multimodèle et commande décentralisée coordonnée."RAIRO Automatique, Vol.18, n°2, 1984

[Bin 78]: BINGHAM J.E., DAVIES G.W.P.: "Handbook of systems analysis."Mac Millan Press Ltd, London, 1978

[Bla 85]: BLANCHARD Michel: "L'Atelier d'Ingénierie Productique."USINICA 85,. Paris 1985

[Bla 79]: BLANCHARD Michel: "Comprendre, maîtriser et appliquer le Grafcet."Cepadues Editions, 1979

[Bod 83]: BODARD François, PIGNEUR Yves: "Conception assistée des applications informations informatiques. Etude d'opportunité et analyse conceptuelle." Editions MASSON, collection Méthode + Programmes, 1983

[Boo 86]: BOOCH Grady: "Object-oriented development."IEEE Transactions on Software Engineering., vol. SE-12, n°2, Février 1986

[Bor 86]: BORELLY J-J., SIMON D.: "Specifications of a sublocal network for robotics." Advanced seminar on real-time local area networks. INRIA. Bandol. France. Avril 1986

[Bou 86]: BOUCON Thierry, BROUILLET Jean-Claude: "Analyse préliminaire fonctionnelle des risques, application à la boucle d'essais CUMULUS."Bertin-EdF. 1986

[Bou 88]: BOUISSOU M.: "Recherche et quantification automatiques de séquences accidentelles pour un système réparable."EDF bulletin de la direction des études & recherches. série A n°1 pp 63-68., 1988

[Bou 88]: BOUISSOU M., VILLATE N.: "GSI2, A simple link betweem the FMEA & the quantification for complex systems ; automatic generation of FMEA by an expert system." 1988

[Boy 88]: BOY Guy, FALLER Benoît, SALLANTIN Jean: "Acquisition et ratification des connaissances."Actes de Journées Nationales Intelligence Artificielle, PRC-GRECO, Editions Teknea, Toulouse 14-15 Mars 1988

[Bra 83]: BRAMS G.W.: "Réseaux de Petri: théorie et pratique."Tomes 1 et 2, Masson, 1983

[Bra 85]: BRAVOCO R.R., YADAV S.B.: "Requirement definition architecture-an overview."Computers in Industry, 6, p237, 1985

[Bra 85]: BRAVOCO R.R., YADAV S.B.: "A methodology to model the functional structue of an organization."Computers in Industry, vol 6, p345, 1985

[Bre 84]: BREUIL: Thèse d'Etat, Université de Bordeaux 1, laboratoire GRAI, 1984

[Bro 88]: BROCHON, GILLE: "Guide d'analyse d'exploitation des moyens d'une ligne. Politique générale."Document interne Régie Renault, Sce 0923, 30 Août 1988

[Bro 87]: BROWN Peter F., Mc LEAN Charles R.: "The architecture of the NBS Factory Automation Research Testbed."IFAC 87, Munich, 1987

[Bru 88] : BRUMENT Jean-Michel, MAIRE F.: "Ingénierie / Explitation: les enjeux d'une coopération." rapport de l'Agence Nationale pour l'Amélioration des Conditions de Travail. Montrouge, Mars 1988.

[Bru 88]: BRUYN W., JENSEN R., KESKAR D., WARD P.: "ESML: an Extended Systems Modeling Language based on the Data Flow Diagram."ACM Sigsoft Software Engineering Notes, vol 13, n°1, Jan. 1988

[Bur 87]: BURBIDGE J.L., FALSTER P., RIIS J.O., SVENDSEN O.M.: "Integration in Manufacturing."Computers in Industry, vol.9 .n° 4, Dec. 1987

[Cam 86]: CAMERON John: "An overview of JSD."IEEE Transactions on Software Engineering., vol. SE-12, n°2, Février 1986

[Cha 80]: CHARESTJacques: "La conception des systèmes: une théorie, une méthode."Gaëtan Morin éditeur, 1980

[Che 76]: CHEN P.P.S: "The entity-relationship model: towards a unified view of data."ACM Transactions on Data Base Systems, vol.1, n°1, 1976

[Cho 86]: CHOPPY Christine: "Techniques et aspects du prototypage."Revue Génie Logiciel, n°3, 1986

[Cnm 87]: CNMA (anonyme): "Users' needs in the field of industrial Local Area Netwoks for Process Control."Questionnaire d'enquête CNMA-PSA., Task 601, octobre 1987

[Cod 79]: CODD E.F.: "Extended the data-base relational model to capture more knowledge."ACM Transactions on Data Base Systems, vol.4, n°4, 1979

[Coq 88]: COQUILLAT M.: "L'AMDEC ou l'AV dans un miroir."La Valeur, n° 36, 1988

[Dar 87]: DART S.A., ELLISON R.J., FEILER P.H., HABERMANN A.N.: "Software development environments."Computer IEEE, vol.20, n°11, Nov. 1987

[Dav 88]: DAVID J-M., KRIVINE J-P: "Acquisition de connaissances expertes à partir de situations types."8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[Dav 82]: DAVIS G.B.: "Strategies for information requirements determination."IBM Systems Journal, vol21, n°1, 1982

[DeM 78]: DE MARCO Tom: "Structured analysis and systems specification."Prentice Hall, 1978

[DeR 75]: DE ROSNAY Joël: "Le macroscope - Vers une vision globale."Editions du Seuil, Collection Points, Paris 1975

[DeV 88]: DE VISMES Arnaud: "La cohabitation des réseaux hétérogènes. Les concepts essentiels de NFS."Décisions, n°186, 16 Mai 1988

[Dec 88]: DECLERFAYT Olivier: ""Software Through Pictures" de IDE: outils et méthodes."Bigre + Globule, n° 58, actes des journées AFCET "Spécification de logiciel, Techniques et outils", Paris, Janv. 1988

[Del 86]: DELAGE A. , BREGER G. , ROTH S.: "Une approche globale originale pour l'optimisation globale des modes de fonctionnement du système."Vème colloque international de fiabilité et de maintenabilité . Biarritz 1986

[Del 86]: DELCUVELLERIE Jean-Luc: "Réseaux Locaux Industriels: Les constats d'une enquête de terrain."Le Monde Informatique, Spécial Télécoms., Juillet -Août 1986

[Der 83]: DERNIAME J.-P., ZAKARI A.: 1983

[Dev 88]: DEVANEY: "Analyse des défaillances des cartes électroniques."Symposium International "Analyse de Défaillances" ASTE, Paris, 31 Mai et 1er Juin 1988

[DGA 86]: "Méthodologie qualité dans les programmes d'armement." STEn-MQ10. Délégation Générale pour l'Armement, 15 septembre 1986

[DiL 87]: DI LEVA Antonio, VERNADET François, BIZIER Daniel: "Information systems analysis and conceptual database design in production environments with M*."Computers in Industry, Vol.9, n°2, 1987

[Dir 87]: DIRNE C.W.G.M.: "The impact of FAMS on overall production control structures."Computers in Industry, vol.9 .n° 4, Dec. 1987

[Dol 83]: DOLLE François, MOALLA Mohamed, RODRIGUES Paul: "Concevoir des systèmes automatisés."Le Nouvel Automatisme, Mars 1983

[Dor 88]: DORBES G.: "Intégration d'une méthode d'analyse de l'information (NIAM) dans une méthode globale d'acquisition de la connaissance pour la création d'un système expert."8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[Dou 85]: DOUMEINGTS G.: "How to decentralize decisions through GRAI model in production management."Computers in Industry, vol. 6 .n° , pages 501 à 514, 1985

[Dou 84]: DOUMEINGTS G.: "Méthode GRAI: Méthode de conception de systèmes en productique."Thèse d'Etat, Université de Bordeaux 1, laboratoire GRAI, 1984

[Dou 86]: DOUMEINGTS G., BREUIL, PUN: "La gestion de production assistée par ordinateur."collection Techniques de base de la XAO, éditions Hermes, Paris 1986

[Dou 87]: DOUMEINGTS G., VALLESPIR B., DARRICAU D., ROBOAM M.: "Design methodology for advanced manufacturing systems."Computers in Industry, vol.9 .n° 4, Dec. 1987

[Dui 87]: DUITSMAN Larry L., PINELLI Madeline M.: "An all-purpose model to aid in all phases of network design."Data Communications. Septembre 1987

[ElF 85]: EL FAZZIKI Abdelaziz: "Contribution à la structuration et à la programmation des applications de contrôle de procédés industriels."Thèse Université de Nancy 1. CRIN, 20 Février 1985

[Ell 88]: ELLOY Jean-Pierre: "Le point sur MAP et sa couche Application MMS."Editions EC2. , Nanterre, 1988

[Ell 88]: ELLOY Jean-Pierre: "Le temps-réel."Rapport établi par le groupe de réflexion temps-réel du CNRS, TSI, vol.7, n°5, 1988

[Eng 85]: ENGELKE, GROTRIAN, SCHEUING,SCHMACKPFEFFER, SCHWARZ, SOLF, TOMANN: "Integrated Manufacturing Modeling System (IMMS)."IBM Journal of Research and Development, vol.29, n°4, juillet 1985

[Fra 88]: FRACHET J-P.: "Modèle conceptuel des données en automatisation: résultats des travaux du groupe BASE-PTA."Congrès Automatique 88, AFCET, Grenoble, 10-12 Octobre 1988

[Gad 86]: GADOIN, COINDOZ: "L'AMDE, un outil essentiel dans le programme de développement d'un grand système, pour en construire la fiabilité."5e colloque international de fiabilité et de maintenabilité, Biarritz, 1986

[Gal 86]: GALARA Dominique: "Besoins en communication dans les systèmes automatisés de production."Enjeux AFNOR, numéro spécial "Productique cap sur MAP". n° 65., Janv. 1986

[Gal 84]: GALARA Dominique, THOMESSE Jean-Pierre: " "Livre Blanc FIP, 1984

[Gal 88]: GALENE Jean-Marie: "Manuel de conception et de réalisation d'applications en environnement GBase."Document interne SGN, version 1, Avril 1988

[Gan 79]: GANE C., SARSON T.: "Structured systems analysis."Editions Prentice Hall, 1979

[Gom 84]: GOMAA H.: "A software design method for real-time systems."Communications of the ACM, vol 27, n°9, Sept. 1984

[Gom 86]: GOMAA H.: "Software development of real-time systems."Communications of the ACM, vol 29, n°7, Juil. 1986

[Gon 88]: GONDRAN Michel: "GENESIA: des outils opérationnels pour le génie logiciel."Génie logiciel et systèmes-experts, n°12, Juin 1988

[Gon 87]: GONDRAN Michel: "Systèmes-experts ....."Revue Générale de l'Electricité, n°4, Avril 1987

[Gon 86]: GONDRAN Michel, HERY J-F, LALEUF J-C: "Une aide aux études de fiabilité par systèmes-experts."5e colloque international de fiabilité et de maintenabilité, Biarritz, 1986

[Gon 87]: GONDRAN Michel, LALEUF Jean-Claude: "Les systèmes experts, outils de l'automatisation: l'expérience d'EDF."RGE n°4 Avril 1987

[Gon 80]: GONDRAN Michel, PAGES Alain: "Fiabilité des systèmes."Editions Eyrolles, collection de la direction des études et recherches d'Electricité de France, 1980

[Gon 78]: GONDRAN Michel, PAGES Alain: "Evaluation de la fiabilité par la méthode des états de marche critique."1978

[Gre 82]: GREENSPAN S.J., MYLOPOULOS J.: "Capturing more knowledge in the requirements specification."Proceedings of the 6th international conference on software enginnering, Tokyo, p.225, 1982

[Gri 88]: GRIFFIN, CAPITANO: "Les besoins du département de la défense, spécifications militaires."Symposium International "Analyse de Défaillances" ASTE, Paris, 31 Mai et 1er Juin 1988

[Gui 86]: GUIDAL C: "L'informatisation des FMECA- Présentation du logiciel FMEGEN."5e colloque international de fiabilité et de maintenabilité, Biarritz, 1986

[Gun 86]: GUNN Thomas: "The CIM connection, computer integrated manufacturing won't succeed without top management's support."Datamation, Février 1986

[Hab 88]: HABRIAS H., BOGDANIUK D., LEPRETRE S.: "La méthode I/A (NIAM) et le logiciel PC-IAST, un exemple d'application."Travail et méthodes, n°465, Juin-Juillet 1988

[Har 87]: HAREL D: "Statecharts: a visual formalism for complex systems."Science of Computer Programming 8, North-Holland Publishing, p.231, 1987

[Hat 88]: HATLEYD., PIRBHAI I.: "Strategies for real-time specification."New-York Dorset House, 1988

[Hen 88]: HENRY Christian, LOTT Michaël: "La méthode de conception MACH2."Journées Internationales du Génie Logiciel et ses Applications, Toulouse, 5-9 Décembre 1988

[Hou]: HOURTOLLE Catherine: ""Conception de logiciels sûrs de fonctionnement: Analyse de la sécurité des logiciels" "Thèse INPToulouse, rapport LAAS n°87.267, octobre1987

[Hut 87]: HUTCHINSON G.K., CHATURVEDI A.: "Information organization in flexible automation systems."Computers in Industry, vol.9 .n° 4, Dec. 1987

[Huy 85]: HUYSENTRUYT J.: "Une architecture PIO (CIM ) basée sur l'approche 'systèmes ouverts' (Open System Approach, ou OSA."revue ENJEUX, AFNOR, n° 65, Janvier 1985

[Jac 75]: JACKSON M.: "Principles of program design."Academic Press, 1975

[Jon 86]: JONES Albert T., Mc LEAN Charles R.: "A proposed Hierarchical Control Model for Automated Manufacturing Systems."Journal of Manufacturing Systems, Vol.5. n1, 1986

[Jos 87]: JOSSERAND, BOEHM: "Les Réseaux Locaux Industriels dans l'automatisation de la production."Automation 87. CETIM, St-Etienne, 1987

[Jou 85]: JOUINEAU Claude: "L'analyse de la valeur: méthodes, mise en œuvre, applications."Entreprise Moderne d'Edition, nouvelle édition, 1985

[Kal 85]: KALLEL G., PELLET Z., BINDER Z.: "Conduite décentralisée coordonnée d'atelier."AFCET-APII, vol.19, n°4, 1985

[Kar 88]: KARCHER Xavier: "Maîtrise des flux de produits, Maîtrise des flux d'informations."Revue française de Gestion Industrielle, n°1, 1988

[Kel 87]: KELLY John C.: "A comparison of four design methods for real-time systems"Comm. ACM , 1987

[Kim 88]: KIMBROUGH S.O., LEE R.M.: "Logic Modeling: A tool for management science."Decision Support Systems International Journal, vol.4, n°1, 1988

[Lan 88]: LANSADE Christophe: "Etude de cas: évaluation de la fiabilité d'un système de contrôle-commande industriel comportant une partie programmée."Séminaire DINOV "Analyse et évaluation de la sûreté de fonctionement des matériels et des logiciels." Toulouse Avril 1988

[Lap 88]: LAPRIE Jean-Claude: "Vers une théorie de la fiabilité du X-iel."TSI-AFCET. Vol.7, n°3, 1988

[Lap 88]: LAPRIE Jean-Claude: "Méthodes et outils de l'évaluation de la sûreté de fonctionnement."Séminaire DINOV "Analyse et évaluation de la sûreté de fonctionnement des matériels et des logiciels." Toulouse Avril 1988

[Lap 85]: LAPRIE Jean-Claude: "Sûreté de fonctionnement des systèmes informatiques et tolérance aux fautes: concepts de base."TSI-AFCET. Vol.4, n°5, 1985

[Lap 88]: LAPRIE Jean-Claude: "Sûreté de fonctionnement et tolérance aux fautes: concepts de base."Rapport LAAS, Toulouse, n°88.287, septembre 1988

[Lap 84]: LAPRIE Jean-Claude: "Towards a taxonomy of dependable computing."4th JCIT, Jerusalem, 22-25 Mai 1984

[Lau 88]: LAURENT Jean-Pierre, FALLER B., LAURIERE J-L., VIVET M., ZIEBELIN D.: "Schéma pour la description et l'évaluation de systèmes experts et d'outils de développement de systèmes experts."Actes de Journées Nationales Intelligence Artificielle, PRC-GRECO, Editions Teknea, Toulouse 14-15 Mars 1988

[Lav 88]: LAVIE René: "The CIM Integrated Data Processing Environment (IDPE) in the European open systems architecture CIM-OSA."MAP/TOP COS Enterprise Networking Event '88 International Baltimore USA , 6-8 juin 1988

[LeM 77]: LE MOIGNE: "La théorie du système général."Presses Universitaires de France, Paris, 1977

[Lec 86]: LECLERCQ P.R. , DELAGE A.: "Matériels de haute sécurité- Analyse de fiabilité et de sécurité "en contexte"."1986

[Lef 88]: LEFRANC Serge: "Productique ou Modernisation: une nouvelle approche."Génie Industriel, n° 170, Mai-Juin 1988

[Lej 86]: LEJON Jean: "Automatisation des processus continus, semi-continus et manufacturiers."Enjeux AFNOR, numéro spécial "Productique cap sur MAP". n° 65., Janvier 1986

[Lej 88]: LEJON Jean, POURCEL: "Etat de l'art et prospective sur l'automatisation des systèmes de production."Etude EXERA-EDF/DER (service Information, Prospective et Normalisation), journée de présentation mars 1988 et résumé dans Instrumentation et Systèmes, n°6, septembre 1988

[Lem 78]: LEMOIGNE J.-L.: "La théorie du système général."Presses Universitaires de France, 1978

[Lho 85]: LHOSTE Pascal: "Exploitation des systèmes automatisés, Ex.A.O.: proposition d'une approche méthodologique et d'outils d'assistance."Thèse Université de Nancy 1. CRAN, 1! Déc. 1985

[Lie 76]: LIEVENS: "Sécurité des systèmes."Editions CEPADUES. Collection SUP'AERO, 1976

[Lon 87]: LONCHAMP J.: "Conception des applications informatiques réparties en commande de procédés industriels: une démarche, des outils."Thèse de Doctorat, Université de Nancy 1. CRIN, 5 Mai 1987

[Lud 87]: LUDEWIG J., MATHEIS H.: "Specification techniques for real-time systems"Computers Standards & Interfaces, n°6, 1987

[Lus 77]: LUSSATO Bruno: "Introduction critique aux théories d'organisation."Dunod, 2ème édition, 1977

[Mah 87]: MAHE Hervé, VESOUL Pierre: "Acquisition des connaissances et adaptation à l'utilisateur: outils et méthodes."7 èmes journées Systèmes Experts d' Avignon, Mai 1987

[Mal 86]: MALAKOOTI B.: "A methodology for the automation of medium or small manufacturing companies."Computers in Industry, vol.7 .n° 4, 1986

[Mam 85]: MAMMERI Z.: "Conception d'applications réparties en commande de processus: une approche par la structuration des données."Thèse de Doctorat en Informatique, INPL, 1985

[Man 87]: MANNINO Paul V.: "A presentation and comparison of four information systems development methodologies."ACM Sigsoft Software Engineering Notes, vol 12, n°2, Avril 1987

[Mar 88]: MARCENAC P., HERIN-AIME D.: "Systèmes experts pour l'enseignement assisté par ordinateur et approché par objets."8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[McL 87]: Mc LEAN C.R.: "Interface concepts for plug-compatible production management systems."Computers in Industry, vol.9 .n° 4, Dec. 1987

[McL 86]: Mc LEAN Charles R.: "A proposed Hierarchical Control Model for Automated Manufacturing Systems."Journal of Manufacturing Systems, Vol.5. n1, 1986

[Mcl 83]: McLEAN Charles R., MITCHELL Mary BARKMEYER Edward: "A computer architecture for small-batch manufacturing."IEEE Spectrum, Mai 1983

[McC 88]: McCONNEL John: "Stepping off the OSI standards bandwagon."Data Communications, Juin 1988

[Mes 70]: MESAROVIC M.D., MACKO D., TAKAHARA Y.: "Theory of hierarchical, multilevel, systems."Academic Press, 1970

[Mey 87]: MEYER W., APPEL Steve: "ESPRIT 932: Knowledge Based Real-Time Supervision in CIM."18 months report 3/1, supplt.A1, Task 1: dedicated AIP modules for CIM, Philips Hamburg, Août 1987

[Moa 86]: MOALLA M.: "Graphes causes-effets appliqués à la vérification des dossiers fonctionnels des systèmes."1986

[Mon 88]: MONTALBAN M., HAREN P., DELCAMBRE B.: "Systèmes experts de conception fondés sur les spécifications."8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[Nay 87]: NAYLOR A.W., VOLZ R.A.: "Design of integrated manufacturing system."IEEE Transactions on Systems, Man, and Cybernetics, vol 17, n°6, nov-dec. 1987

[Nij 77]: NIJSSEN G.M.: "On the gross architecture for the next generation data base management system."Information processing, North Holland, 1977

[Nor 88]: NOREN Chuck: "Rapid prototyping network management systems."IEEE Milcom'88 "21st century military communications-What's possible ?" , Oct. 1988

[Noy 87]: NOYES Daniel: "Approches méthodologiques pour l'aide à la conception et à la conduite des systèmes de production."Thèse de Doctorat d'Etat, INPToulouse, Rapport LAAS n° 87.319, 12 Nov. 1987

[Oda 84]: ODA Y., SATO A., OKUSAWA O.: "A design and maintenance support system for large-scale software."REVIEW of the Electrical Communications Laboratories, vol 32, n°1, 1984

[Ove 88]: OVERWATER R., VEGTER K.J.: "An alternative approach to the development of industrial Systems."Computers in Industry, vol.10 .n° 3, Juillet 1988

[Per 87]: PERKUSICH A., DEEP G.S., CAVALCANTI J.: "Data and control flow diagram and process automation."IECON 87, 1987

[Per 85]: PERRIN Guy-René: "La communication: un outil pour la spécification, la construction et la vérification de systèmes parallèles."Thèse de Doctorat, Université Nancy 1, Octobre 1985

[Pia 87]: PIAT Gérald: "Livre Blanc: OURAI, OUtil de Réalisation d'Automatismes Industriels."rapport de la commission SAP-EXERA, 1987

[Pil 85]: PILTZECKER E.W., SAPITA R.F.: "Modeling management operations systems"Control Engineering, Juin 1985

[Plo 87]: PLOSSL K.R.: "Computer Integrated Manufacturing."Production Engineering, Juin 1987

[Pri 81]: PRINCE S.M., SLOMAN M.S.: "Communications requirements of a distributed computer control system."IEE Proceedings, Vol. 128. Pt.E,n°1, Jan 1981

[Pun 84]: PUN: "Systèmes Industriels d'Intelligence Artificielle."EdiTests, Paris, 1984

[RaS 87]: RAY S.T.: "Process reasoning."Computers in Industry, vol.9 .n° 4, Dec. 1987

[Rob 88]: ROBINET E.: "Le projet EUREKA FIABEX." 1988

[Roe 87]: ROESCH M., LAURENT C.: "Structuration de la partie commande d'un système automatisé."Electronique industrielle, n°119, Février 1987

[Rol 86]: ROLLAND Colette: "REMORA: une méthode de conception des systèmes d'information."Génie Logiciel, n°4, 1986

[Rol 86]: ROLLAND Colette: "Introduction à la conception des systèmes d'information et panorama des méthodes disponibles."Génie Logiciel, n°4, 1986

[Ros 77]: ROSS D.T.: "Structured Analisys (SA) : a language for communicating ideas."IEEE Transactions on Software Engineering., vol. SE-3, n°1, Janvier 1977

[Ros 77]: ROSS D.T., SCHOMAN K.E.: "Structured analysis for requirements definition."IEEE Transactions on Software Engineering., vol. SE-3, n°1, Janvier 1977

[Rou 88]: ROUTT Thomas J.: "From TOP (3.0) to bottom: Architectural close-up."Data Communications, Mai 1988

[Rub 87]: RUBIELLO Luc: "Systèmes à base de connaissances: au-delà de la mode."Le Monde Informatique, 31Août 1987

[Sar 87] : SARRIEU Valérie: "Description graphique des systèmes d'informations" Stage DEA ENSEEIHT. 1987

[Sau 88]: SAUGE Pierre: "Création d'un atelier de génie logiciel à partir d'une boîte à outils: GraphTalk."Rank Xerox, Journées Internationales du Génie Logiciel et ses Applications, Toulouse, 5-9 Décembre 1988

[Sav 88]: SAVOLAINEN Tapani: "Software technologies in Computer Integrated Manufacturing."Computers in Industry, vol 11, n°1 , p.93, nov. 1988

[Sch 88]: SCHREIBER G., BREUKER J., BREDEWEG B., WIELINGA B.: "Modelling in KBS development."8 èmes journées Systèmes Experts d' Avignon, Mai 1988

[Sco 88]: SCOTT P.C.: "Requirement analysis assisted by Logic Modeling."Decision Support Systems International Journal, vol.4, n°1, 1988

[She 87]: SHEMER I.: "Systems analysis: a systemic analysis of a conceptual."Communications of the ACM, vol 30, n°6, juin 1987

[Sok 80]: SOK S.: "Evolutivité du logiciel" Thèse de 3ème cycle, Université de NANCY 1, Septembre 1980

[Sou 85]: SOURISSE C., FRACHET J-P: "L'expression des besoins dans l'automatisation des processus: vers un outil d'assistance à l'établissement d'un cahier des charges."Convention Automatique-Productique, Méthodologie générale d'automatisation pour les industries manufacturières ( repris dans Bureaux d'études Automatismes n°27 .1986) Paris, Décembre 1985

[Spa 87]: SPANIER Steve: "Comparing distributed file systems."Data Communications, Déc. 1987

[Ste 82]: STEVENS W.P.: "How data flow can improve application development productivity"IBM Systems Journal, vol 21, n°2, 1982

[Ste 74]: STEVENS W.P., MYERS, CONSTANTINE: IBM Systems Journal, vol , n°2, 1974

[Tar 83]: TARDIEU Hubert, ROCHFELD Arnold, COLLETTI René: "La méthode MERISE. Tome 1. Principes et Outils."Editions d'Organisation, Paris, 1983

[Tar 85]: TARDIEU Hubert, ROCHFELD Arnold, COLLETTI René, PANET Georges, VAHEE Gérard: "La méthode MERISE. Tome 2. Démarche et pratique."Editions d'Organisation, Paris, 1985

[Tei 77]: TEICHROEW D., HASCHEY E.A.: "PSL/PSA, a computer aided technique for structured documentation and analysis of information processing system."IEEE Transactions on Software Engineering., vol. SE-3, n° , Fevrier 1977

[Thi 86]: THIREAU Ph.: "Méthodologie d'analyse des effets des erreurs logiciel (AEEL) appliquée à l'étude d'un logiciel de haute sécurité."Vème colloque international de fiabilité et de maintenabilité . Biarritz 1986

[Tho 88]: THOME Franck: "SAGP, un système d'audit en gestion de production."Présentation SGN, St Quentin en Yvelines, 30 Septembre 1988

[Tho 80]: THOMESSE J-P: "SYGARE: une structuration pour la conception d'applications en temps réel et réparties."Thèse de Doctorat, INPL Nancy, Mai 1980

[Tho 85]: THOMESSE J-P: "Pour un système générique de commande d'atelier flexible."Méthodologie générale d'automatisation pour les industries manufacturières, Convention Automatique-Productique, Paris, Décembre 1985

[Tho 86]: THOMESSE Jean-Pierre, DELCUVELLERIE Jean-Luc: "Les Réseaux Locaux Industriels."Revue Axes-Robotique, 1986

[Tho 86]: THOMESSE Jean-Pierre, DELCUVELLERIE Jean-Luc: "MAP, TOP et FIP: trois candidats à la normalisation."Le Monde Informatique, Spécial Télécoms., Juillet -Août 1986

[Tho 87]: THOMESSE Jean-Pierre, DELCUVELLERIE Jean-Luc: "FIP: a fieldbus proposal."Workshop on factory communications, IEEE-NBS, Gaithersburg, Maryland, Mars 1987

[Tho 86]: THOMESSE Jean-Pierre,DELCUVELLERIE, DERNIAME, LEPAGE, PICHON, SCHAFF: "Les Réseaux Locaux Industriels."Editions EC2. , Nanterre, 1986

[Tit 75]: TITLI: "Commande hiérarchisée et optimisation des processus complexes."Dunod, 1975

[Tse 86]: TSE T.H., PONG L.: "An application of Petri Nets in Structured Analysis."ACM Sigsoft Software Engineering Notes, vol 11, n°5, Oct 1986

[Ver 89]: VERLINDE Christian: "Contribution à l'étude des arcitectures de systèmes automatisés."Thèse de Doctorat INPL, Nancy, Février 1989

[Vog 88]: VOGEL Claude: "Génie cognitif."Editions MASSON, collection sciences cognitives, 1988

[Vog 86]: VOGIN R. , BERMEJO L. , SUDRET P.: "L'arbre fonctionnel : une méthode éprouvée d'analyse prévisionnelle de la fiabilité."V ème colloque international de fiabilité et de maintenabilité. Biarritz 1986

[Vog 87]: VOGRIG R., MOREL G., DEI-SVALDI D., VAUTRIN J-P.: "Analyse de la sécurité par SADT: application sur un ilôt automatisé."Electronique industrielle, n°115, Novembre 1987

[War 86]: WARD P.T.: "The transformation schema: an extension of the data flow diagram to represent control and timing."IEEE Transactions on Software Engineering., vol. SE-12, n°2, Fevrier 1986

[War 88]: WARD Paul.T.: "Real-Time specification methods."Software Development Concepts, support de cours, Journées Internationales du Génie Logiciel et ses Applications, Toulouse, 5-9 Décembre 1988

[Was 85]: WASSERMAN A.I.: "Extending state transition diagrams for the specification of human-computer interaction." IEEE Transactions on Software Engineering, Vol.11, n°8, pp.699, 1985

[Was 86]: WASSERMAN A.I., PIRCHER P.A.: "A graphical, extensible integrated environment for software engineering." ACM Sigplan Notice, Vol.22, n°1, pp.131, Janvier 1986

[Wil 87]: WILLIAMS Theodore J.: "The Integrated Approach to Industrial Control."IFAC 87, Munich, 1987

[Wil 87]: WILLIAMS Theodore J.: "Recent development in the application of Plant-Wide Computer Control."Computers in Industry, 8 , 1987

[Win 84]: WINSTON Patrick Henry: "Artificial intelligence."Addison Wesley Publishing Company, 2nd edition, 1984

[YaS 86]: YAU Stephen, TSAI Feffery J.-P.: "A survey of software design techniques."IEEE Transactions on Software Engineering., vol. SE-12, n°6, Juin 1986

[Yeo 86]: YEOMANS R.W., CHOUDRY A., ten HAGEN P.J.W.: "Design rules for a CIM system."ESPRIT pilot Project (n° 5.1/34), North-Holland, 1986

[You 79]: YOURDON, CONSTANTINE: "Structured design: fundamentals of a discipline of computer program and systems design."Prentice Hall, 1979

[Zac 87]: ZACHMAN J.A.: "A framework for information systems architecture."IBM Systems Journal, vol 26, n°3, 1987

[Zak 84]: ZAKARI Abdelouahed: "FLEXI: langage de conception d'application de conduite de procédés industriels, contribution à l'étude de la communication entre processus coopérants."Thèse Université de Nancy 1. CRIN, 2 Mars 1984

[Zwi 87]: ZWINGELSTEIN : "Application industrielle de l'I.A.: des débuts prometteurs."Revue Générale de l'Electricité, n°4, Avril 1987

SofTech Inc. Waltham"SADT advanced topics, author guide supplement."SofTech Inc. Waltham, Massachusetts, TP 083, Decembre 1978

Digital Equipment Corp."Digital Industrial Networks Guidebook."Digital Equipment Corp., Order Number EB-31782-70, Juin 1988

Esprit 932: Knowledge-based real-time supervision in CIM"Extension of chosen methods towards AI."Esprit 932: Knowledge-based real-time supervision in CIM, 6 months report 1/1, supplt.B1, Task 2.1.1.2., Philips Hamburg, 22 Mai 1986

EPOS description générale"EPOS Engineering and project-management oriented support system."EPOS description générale, Document 3IP,104, rue Castagnary, 75 015 PARIS, octobre 1987

AMICE ESPRIT Project 688"CIM-OSA: A primer on key concepts and purpose."AMICE ESPRIT Project 688, Document de présentation, 1987

ADEPA"GEMMA: Guide d'Etude des Modes de Marches et d'Arrêts."ADEPA

G2. Version 1.11"G2 user's manual. Real time expert systems."G2. Version 1.11, Gensym Corporation, Cambridge, 1988

CISI Ingénierie Document de présentation"HOOD, une méthode de conception hiérarchisée orientée objets poyur le développement des logiciels techniques et temps réels."CISI Ingénierie Document de présentation, 1987

ISO 7498 "Système de traitement de l'information. Interconnexion de systèmes ouverts. Modèle de référence. "ISO 7498, 1984

GAME T17.version 2"Méthodologie de développement de logiciels intégrés."GAME T17.version 2, Ministère de la Défense-DGA, Mai 1987

ESIA document interne. Note technique 1006-A.TMA.0003.DTD"Assurance qualité-Cycle de vie de l'étude préalable."ESIA document interne. Note technique 1006-A.TMA.0003.DTD, Juillet 1988

CETIM "Etude et description du cahier des chyarges d'automatismes: le grafcet - le gemma."CETIM, Rapport d'étude, n°1 - 4N -08 - 01, Février 1983

## publications de l'auteur:

- "Connaissances en Ingénierie des Systèmes Automatisés de Production: conception des systèmes informatisés d'automatisation et analyse de la robustesse aux défaillances d'une architecture répartie." thèse de Doctorat INPL, Mars 1989

- "Outil d'Analyse de la Robustesse aux Défaillances d'une Architecture d'un Système Automatisé de Production." journée "Quelle CAO pour les automatismes ?" SEE-EXERA, Janvier 1989

- "Réflexions d'utilisateurs sur les Réseaux Locaux Industriels." avec J-P. Thomesse, congrès AFCET 88 Grenoble, Octobre 1988

- "FIP: a field bus proposal." avec J-P. Thomesse Workshop on factory communications. IEEE-NBS, National Bureau of Standard, Washington, Mars 1987

- "Les Réseaux Locaux Industriels." avec J-P. Thomesse, revue Réseaux d'Entreprise n°4, 1987

- "Réseaux Locaux: les constats d'une enquête de terrain." avec J-P. Thomesse, Le Monde Informatique, Numéro spécial Télécoms Juillet-Août 1986

- "Les Réseaux Locaux Industriels." collectif avec J-P. Thomesse, éditions EC2, Nanterre, Juin 1986

# ANNEXES

**annexe 1:**

# APPLICATION A LA PRODUCTION DE VERRE PLAT SELON LE PROCEDE FLOAT

## INTRODUCTION

Ce chapitre présente l'application des principes de bases retenus dans le travail de recherche à la conception d'une architecture de conduite d'unité de production de verre plat. Ces études ont été développées dans le cadre particulier d'une action de R&D interne à la société ESIA dont l'objectif était:

- de capitaliser l'expérience acquise dans la mise en oeuvre de tels systèmes en définissant une architecture "standard",
- d'approfondir la réflexion sur la possibilité d'intégrer les communications à l'aide de Réseaux Locaux Industriels,
- de mieux percevoir l'impact induit par différentes options technologiques prises au niveau du processus de production sur les fonctions d'automatisme.

Le contenu et les résultats de cette étude font l'objet du rapport interne "Note Technique 5320 A - AUT 007- 092" du 14 Janvier 1988.

La particularité de cette étude était donc de ne pas traiter une application particulière mais de concevoir un canevas général d'architecture intégrant différentes options au niveau de l'organisation ou au niveau des machines.

La démarche a consisté à suivre les principes développés dans l'étude théorique, c'est-à-dire à aborder la conception du système selon une approche globale à partir de ses relations et de ses échanges d'informations avec son environnement (le processus physique de production, les opérateurs, et la gestion de production). Cette étude s'est déroulée en parallèle avec les travaux de recherche et a constitué un cas industriel pour la mise au point des différents concepts de modélisation et du découpage en tâches de la conception. C'est pourquoi le lecteur ne devra pas chercher dans ce travail l'application parfaitement rigoureuse de tous les détails de la méthode de conception, mais plutôt une application opérationnelle et systématique de son "esprit" et de ses principes directeurs. Nous avons en effet choisi de laisser aux ingénieurs analystes une liberté d'utilisation suffisante pour :

- permettre un apprentissage progressif des concepts de base de la méthode,
- éviter un rejet lié au côté rébarbatif de l'application "d'outils" contraignants,

- recueillir le maximum de suggestions liées à l'adaptation opérationnelle sur une étude industrielle,
- accroître la motivation et la participation par une ouverture suffisante.

Nous avons estimé qu'il était préférable de faire passer la "philosophie" de la méthode, plutôt que faire appliquer des "outils formels" encore imparfaits. Ceci devant permettre en cas de succès de susciter l'envie d'outils plus opérationnels et rigoureux.

Cette étude constitue un exemple très représentatif du type d'étude d'architecture dont nous avons souhaité améliorer la qualité.

La première étape correspond à une identification détaillée des opérateurs du processus physique de production. Cette identification est basée sur la décomposition du processus en différentes entités ou machines de transformation, manutention, stockage ou contrôle du produit. Cette orientation reflète l'importance accordée par les concepteurs (et par la méthode proposée) à la structure du processus.

La seconde étape correspond à la définition des fonctions et des échanges d'informations entre ces fonctions. Nous donnons des exemples à différents niveaux pour illustrer la structure fonctionnelle.

La troisième étape correspond à la répartition des fonctions sur des dispositifs de traitement de l'information (calculateurs ou automates), et à la synthèse des échanges d'informations en flux cohérents. La répartition est basée sur différents critères: contraintes géographiques, contraintes liées aux conditions d'exploitation. Une analyse des couplages induits par les échanges d'informations entre ces dispositifs permet de vérifier que la répartition est satisfaisante par rapport à la disponibilité requise.

# 1. STRUCTURE DU PROCESSUS PHYSIQUE DE PRODUCTION

## 1.1. DESCRIPTION GENERALE DU PROCEDE

Le processus de production a pour but de produire des "volumes" (panneaux) de verre plat à partir d'un ruban continu de verre. Ce ruban est récupéré à la sortie du four où la planéité est obtenue suivant le procédé "FLOAT" ou "FLOATING GLASS". Ce procédé consiste à couler le verre presque liquide à haute température sur un bain d'étain liquide. Ceci permet d'obtenir naturellement un film de verre sans contraintes. Le ruban en sortie du bain d'étain est refroidi sur l'étenderie constituée de convoyeurs à rouleaux.

Le produit traité **en entrée** est un ruban continu de verre refroidi (60°), donc solide, mais fragile, dont la largeur est comprise entre 2,6 et 3,60 m et l'épaisseur entre 1,5 et 30 mm. La vitesse d'avancement du ruban peut aller jusqu'à 30 m/mn. Cette vitesse dépend de l'épaisseur du ruban car le débit du four est à peu près constant: de l'ordre de 400 à 600 tonnes / jour, la production étant continue 24h sur 24.

Le produit **en sortie** est un volume (plateau) de 1 * 1 m à 6 * 3,60 m.

Le procédé EQUARRI consiste donc à découper un ruban de verre (dont l'avance est continue) dans le sens transversal (axe X) par rapport à l'axe du ruban, mais aussi dans le sens longitudinal (axe Y) pour d'une part éliminer les bords non-rectilignes du ruban et d'autre part découper des panneaux de dimension inférieure à la largeur du ruban. Cette découpe se fait en deux étapes: d'abord par **tracé** puis **rompage** suivant ce tracé (le terme "rompage" est en fait traditionnellement réservé à la découpe transversale et longitudinale sauf pour les bords) et on appelle "débandage" la découpe longitudinale des bords du ruban).

En plus des **mesures** de dimension, de position ou de vitesse d'avancement, des opérations de **contrôle-qualité** détectent les défauts éventuels du ruban de verre. Aussitôt après la découpe, les volumes doivent être écartés afin d'éviter les chocs entre les bords de volumes adjacents. Cette séparation est effectuée par une "**accélération**" des volumes par des convoyeurs spécialisés. Après découpe, des opérations complémentaires assurent le **nettoyage** et la **protection** des volumes vis à vis des risques de rayures par une pose de papier ou un "poudrage".

Les volumes doivent ensuite être acheminés vers des dispositifs permettant de constituer des "**piles**" de volumes à expédier au client. Cette manutention est réalisée par des **convoyeurs** à rouleaux et différents types d'**aiguillages** (transfert, transbordeur) permettant de dispatcher les volumes sur les différentes lignes. Des machines de relevage ("**releveuses**") permettent la saisie des volumes pour la mise en pile. Une

"**focalisation**" selon l'axe Y, et un "**positionnement**" selon l'axe X, suffisamment précis des volumes sont nécessaires pour un empilage correct des volumes.

En différents points de la ligne de transport, des "**trappes**" permettent de recycler les produits présentant des défauts non-récupérables.

Tous les volumes ne sont pas directement découpés aux dimensions finales. En plus de la découpe en continu, des installations de "**redécoupe statique**" permettent de reprendre des volumes "bruts" pour une découpe aux dimensions définitives. Il est donc nécessaire d'assurer la manutention des volumes vers et en provenance de ces installations de redécoupe statique.

## 1.2. DECOMPOSITION DES ENTITES DE PRODUCTION

La classification suivante a été adoptée:

- entités de **mesure** et **contrôle** des produits:
  - mesure et contrôle en continu,
  - mesure et contrôle statique;
- entités de **transformation**:
  - découpe,
  - nettoyage,
  - protection;
- entités de **manutention**:
  - transport,
  - aiguillage;
- entités de **stockage**.

Les tableaux suivants listent les différentes entités du procesuus de production:

Entités de **mesure** et **contrôle** des produits:

| entité | variable | produit | dispositif | référence |
|---|---|---|---|---|
| mesure | | | | |
| / continu | dévirage | ruban, | capteurs asservis | MESDEV |
| / continu | largeur | ruban, | capteurs asservis | MESLAR |
| / continu | vitesse | ruban, | roue | ROUMES |
| / continu | épaisseur | ruban, | capteurs épaisseur et longueur | MESEPA |
| contrôle | défaut | ruban, | opérateur ou système optique | DETDEF |
| / continu | dévirage | ruban, | au niveau tracé longitudinal | DEVPTY |
| / continu | dévirage | ruban, | au niveau tracé longitudinal bords | DEVPTB |
| / continu | dévirage | volumes, | au niveau rives de débandage | DEVRIV |
| / statique | défaut | volumes, | prise d'échantillon | PRECHA |
| / statique | défaut | volumes, | contrôle au fil | CONFIL |
| / statique | défaut | volumes, | contrôle contrainte statique | CONCON |

Entités de **transformation** des produits:

| entité | variable | produit | dispositif | référence |
|---|---|---|---|---|
| **découpe** | | | | |
| / tracé transversal | longueur | ruban | pont de tracé | DETRAN |
| / rompage | déplacement | rub.tracé | | DEROMP |
| / tracé longitudinal | largeur | ruban | pont de tracé | DECPTY |
| / tracé bords | largeur | ruban | pont de tracé | DECPTB |
| / débandage | déplacement | rub.tracé | convoyeur | CONDEB |
| **nettoyage** | | | | |
| lavage | | ruban | machine à laver avec convoyeur sous machine | MALARU |
| soufflage-brossage | | ruban | | SOUBRO |
| **protection** | | | | |
| / pose papier | | volume | | MACHPA |
| / pose papier | | ruban | | MAPARU |
| / enduit | | volume | | MACHEN |
| / poudrage | | volume | rouleau poudreur | ROUPOU |

Entités de **manutention** des produits:

| entité | variable | produit | dispositif | référence |
|---|---|---|---|---|
| **transport** | | | | |
| / accélération | vitesse | volume | convoyeur asservissement de vitesse | ZONACC |
| / déplacement | vitesse | volume | convoyeur circulation | CONSTA |
| / focalisation | position | volume | convoyeur focalisation latérale en continu | FOCLAT |
| / focalisation | position | volume | focalisation pour relevage | FOCPLF |
| / centrage | position | volume | centrage pour relevage | CENVOL |
| / retourneur | position | volume | convoyeur | CONRET |

| **aiguillage** | | | | |
|---|---|---|---|---|
| / vertical | direction | volume | deux niveaux vers ≠ lignes convoyeurs | AIGNIV |
| / transfert | direction | volume | à roues | TRAROU |
| / free-way | direction | volume | | FREWAY |
| **/ trappe** | | | | |
| / ruban | direction | ruban | | TRARUB |
| / plateau | direction | volume | | TRAPLA |
| / extensible | direction | volume | | TRAEXT |
| / levante | direction | volume | | TRALEV |

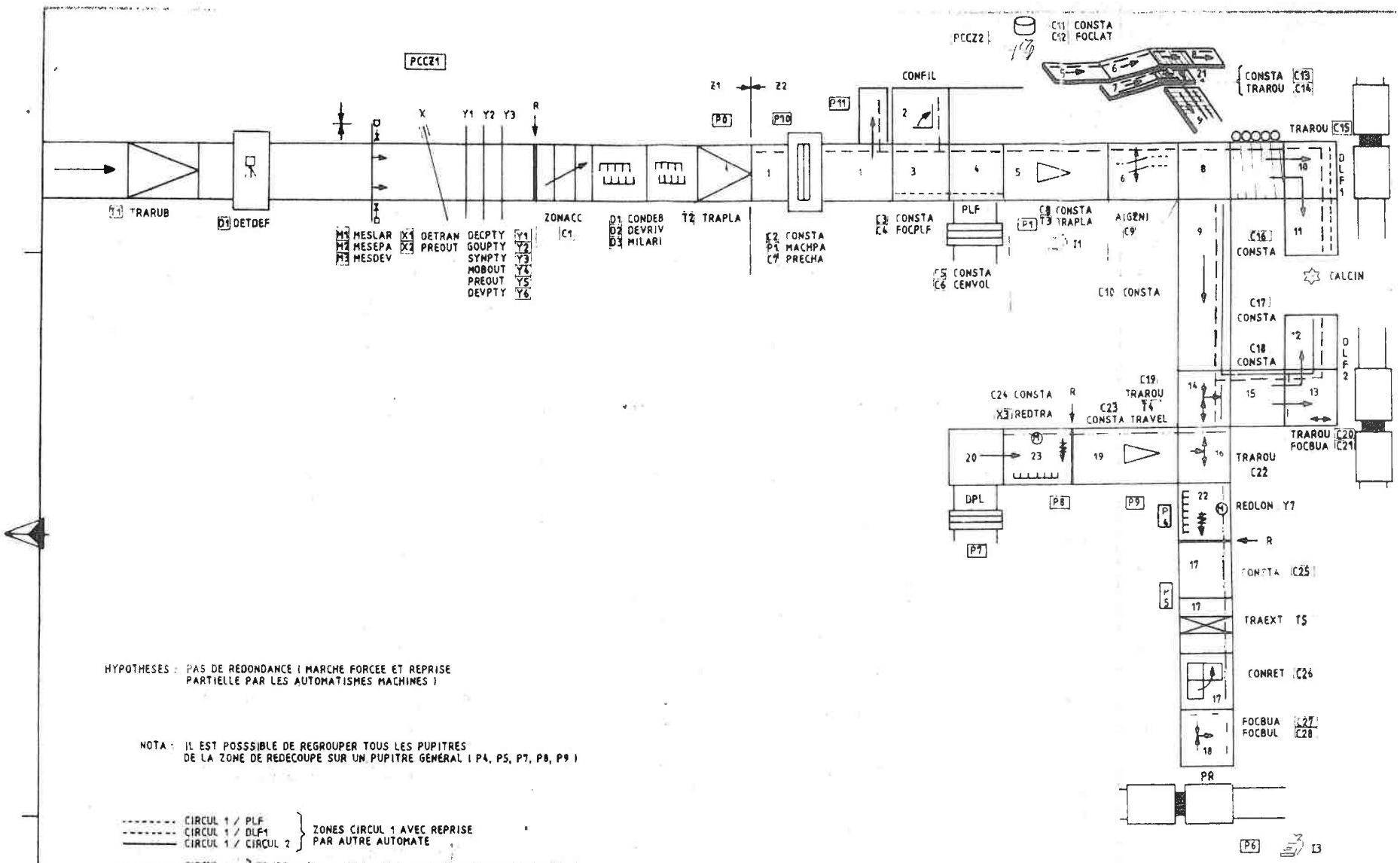
Entités de **stockage** des produits:

| **entité** | **variable** | **produit** | **dispositif** | **référence** |
|---|---|---|---|---|
| **stockage** | n° pile | | | |
| / releveuse | | PLF | | RELPLF |
| / releveuse | | DLF | | RELDLF |
| / glissière | | PLF | | GLIPLF |
| / glissière | | DLF | | GLIDLF |
| / va et vient | | DLF | | VAVIEN1 |
| / va et vient | | primitif | | VAVIEN2 |
| / empilage | | primitif | empilage vertical | RELPRI |

## 1.3. STRUCTURE DU PROCESSUS

Les entités précédemment définis sont organisés suivant deux zones principales appelées zones 1 et 2. La zone 1 est constituée du centre de découpe et des convoyeurs d'accélération des volumes. La zone 2 est constituée des dispositifs de circulation et de stockage des volumes.

Le schéma ci-joint résume l'organisation topologique du processus de production.




HYPOTHESES : PAS DE REDONDANCE ( MARCHE FORCEE ET REPRISE PARTIELLE PAR LES AUTOMATISMES MACHINES )

NOTA : IL EST POSSSIBLE DE REGROUPER TOUS LES PUPITRES DE LA ZONE DE REDECOUPE SUR UN PUPITRE GENERAL ( P4, P5, P7, P8, P9 )

- CIRCUL 1 / PLF
- CIRCUL 1 / DLF1
- CIRCUL 1 / CIRCUL 2

} ZONES CIRCUL 1 AVEC REPRISE PAR AUTRE AUTOMATE

- CIRCUL 1
- CIRCUL 2

} ZONES A COMMANDE UNIQUE ( MARCHE FORCEE POUR CIRCUL 1) ( RIEN POUR CIRCUL 2 )

## 2. DEFINITION DES FONCTIONS

Toutes les fonctions de traitement de l'information ont été décrites en détail selon le modèle générique de structuration fonctionnelle préconisé. On trouve pour chaque fonction la description des différentes liaisons supportant les échanges d'informations. Ces liaisons sont classifiées en:

- liaisons de service avec le niveau hiérarchique supérieur,
- liaisons de service avec le niveau hiérarchique inférieur,
- liaisons de partage de données avec d'autres fonctions,
- liaisons de synchronisation entre fonctions.

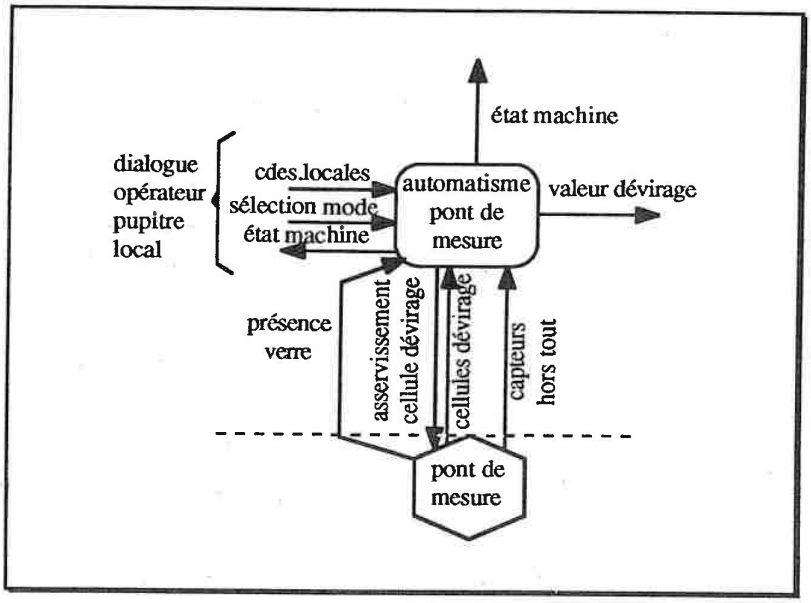
Nous ne donnerons ici que quelques exemples de cette représentation: les fonctions suivi de dévirage, conduite automatique du pont de tracé longitudinal, du déplacement des rives de débandage, fonction de conduite du centre de découpe et fonctions de gestion et de suivi de production.

### 2.1. fonction de suivi de dévirage

**description du rôle de la fonction par rapport au processus associé:**

La fonction de suivi de dévirage a pour but de déterminer la valeur du dévirage, c'est-à-dire la distance entre les cellules de détection de la position du bord du ruban par rapport à son axe central. Pour réaliser cette fonction, l'automatisme doit déplacer des cellules optique pour "suivre" le déplacement latéral du ruban.

remarque: Cette entité est un exemple de mesure ou de contrôle nécessitant un automatisme dédié même si, comme évoqué au chapitre sur la modélisation du processus de production, l'opération n'effectue aucune modification sur le produit.




*structure fonctionnelle de la fonction de suivi de dévirage (MESDEV)*

**échanges d'informations:**

Les informations à destination du niveau supérieur, conduite de la zone 1:

- état machine de mesure dévirage.

Les informations en provenance du niveau inférieur, en l'occurence constitué par le niveau 0 du processus de production, sont les suivantes:

- consignes des moteurs de déplacement des cellules (2 vitesses: petite vitesse pour la commande automatique, grande vitesse pour la commande manuelle),
- signaux des cellules de dévirage: 2 cellules optiques écartées de 10 mm,
- signaux des capteurs hors-tout: 2 fins de course,
- signal de présence verre.

Les informations partagées sont:

- la valeur du dévirage à destination du pont de tracé X, du pont Y et des rives de débandage.

Cet automatisme est prévu pour deux modes de fonctionnement:

- un mode automatique, où le positionnement des cellules optiques est effectué par l'automatisme lui-même,
- un mode de conduite local, où le positionnement des cellules est effectué par l'opérateur.

Le dialogue opérateur est donc constitué :
- de la sélection du mode de marche,
- de la visualisation de l'état machine,
- en mode local: commande du dévirage à droite et commande à gauche des moteurs de déplacement des cellules.

## 2.2. fonction conduite automatique du pont de tracé longitudinal

**description du rôle de la fonction par rapport au processus associé:**

Cette fonction reçoit le plan de fabrication, c'est-à-dire les dimensions des "primitifs" à découper. L'option présentée ici implante la synchronisation avec le tracé transversal (pont de tracé X). Cette option permet donc de passer directement sans perte de fabrication du tracé de volumes d'une largeur donnée à celui d'une autre largeur entre deux tracés transversaux immédiatement successifs par basculement d'un chariot d'outils à l'autre.

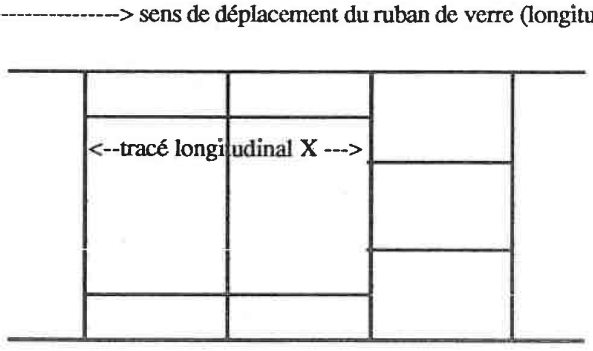



*schéma du tracé des volumes (vue de dessus)*

Une machine de tracé longitudinal se compose en effet d'un ensemble de chariots porte-outils motorisés se déplaçant sur des supports individuels. Chaque outil peut être déplacé et sélectionné (mis en service) de façon individuelle. L'ensemble des outils sélectionnés dans une même rangée peut être mis en contact avec le verre par un dispositif de montée/baisse. C'est cette montée/baisse qui sera ou non synchronisée avec le pont de tracé transversal. Une rangée d'outils constitue une "machine de tracé". Chaque machine possède ses asservissements et son système de montée/baisse. Un pont de tracé est constitué de une (pas de synchronisation possible) à quatre machine.

La première fonction du pont de tracé est donc, à partir des dimensions des volumes à découper, de commander les différentes rangées d'outils (les machines de tracé) en fonction de l'usure des outils et du temps de positionnement de ces outils, ceci en

synchronisation avec le pont de tracé transversal. Le schéma suivant ne représente pour plus de lisibilité qu'une machine de tracé "générique".

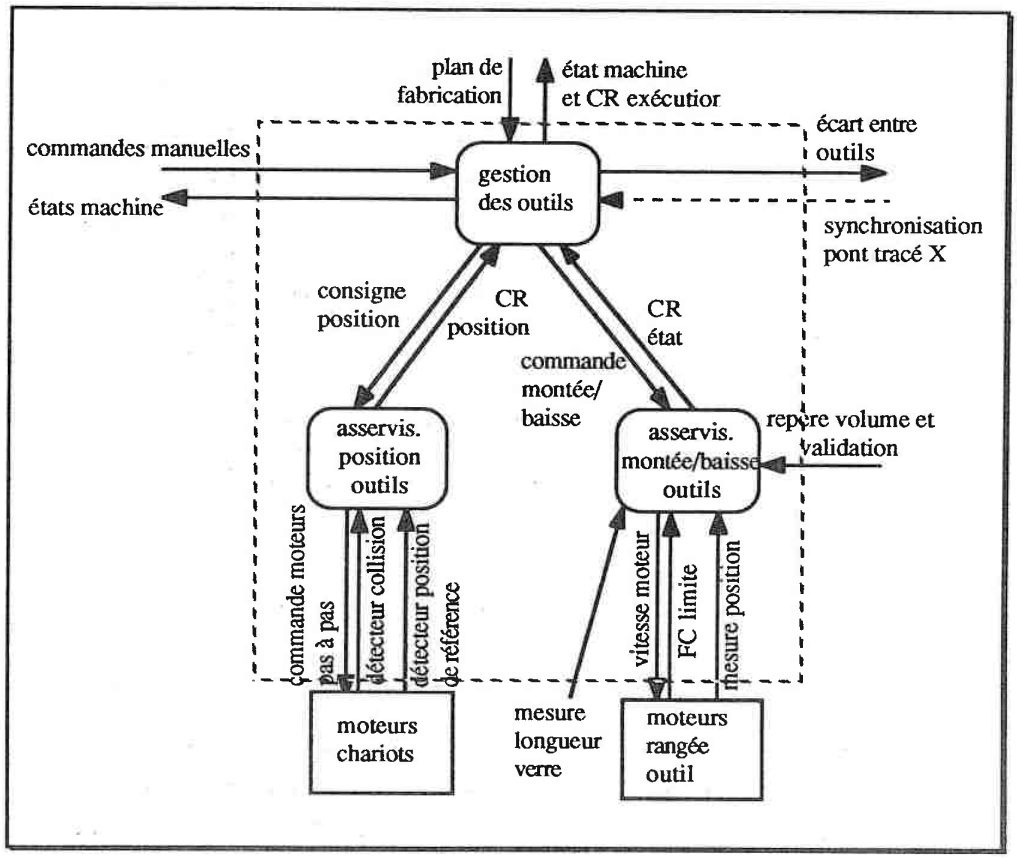



*schéma de structure fonctionnelle de la conduite automatique du pont de tracé longitudinal (DECPTY)*

**échanges d'informations de la fonction gestion outils (GOUPTY):**

Les informations échangées avec le niveau supérieur de conduite sont

- la définition des plans de découpe ainsi que leur validation,
- le compte-rendu de positionnement des outils.

Les informations partagées avec les autres entités fonctionnelles de même niveau sont:

- synchronisation avec le top de tracé transversal,
- écart entre les outils (données produites partagées avec le débandage et le suivi intégral de produits).

Les informations en provenance et à destination du niveau inférieur (asservissements) sont:

- consignes de déplacement vers les asservissements des moteurs de positionnement des outils,
- commande des asservissements des électrovannes de mise service/hors service des outils de tracé,
- compte-rendus et état des asservissements.

Deux modes de fonctionnement sont possibles:
- un mode normal où la fonction de gestion des outils pilote (positionnement et mise en service) les différentes machines de tracé,
- un mode dégradé où l'opérateur positionne lui-même les outils où il le désire et décide de leur mise en service. L'opérateur peut sélectionner le mode de fonctionnement.

## 2.3. fonction conduite automatique du déplacement des rives de débandage

**description du rôle de la fonction par rapport au processus associé:**

Cette fonction permet d'ajuster la position des rives de débandage assurant la rupture du verre dans le sens longitudinal en volumes et bords. Du point de vue mécanique, il s'agit de convoyeurs de débandage pourvus de têtes de débandage solidaires des rives. Chaque rive est motorisée et possède son propre codeur.

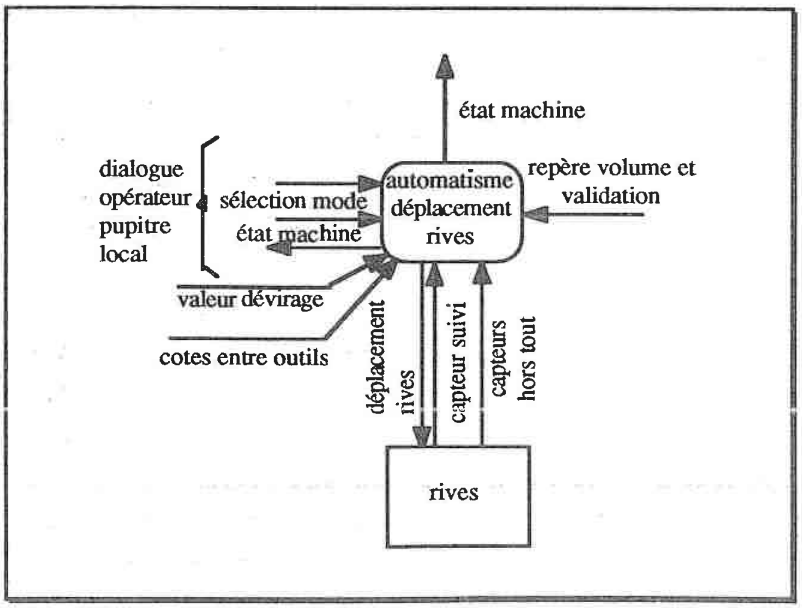



*schéma de structure fonctionnelle de la conduite du déplacement des rives de débandage (MILARI)*

**échanges d'informations:**

Les informations nécessaires partagées (en lecture) avec les autres entités fonctionnelles de même niveau sont:

- la valeur de dévirage en provenance du pont de mesure,
- la valeur des écarts entre outils en provenance du pont de tracé longitudinal.

Les informations en provenance et à destination du processus sont:

- les consignes des moteurs de déplacement des rives (2 sens, 1 vitesse),
- les signaux des capteurs hors-tout: 2 fins de course,
- les signaux des capteurs de suivi.

Les informations partagées avec la fonction de suivi intégral des volumes sont:

- le repère du volume se trouvant au pont de tracé longitudinal et la validation pour la mémorisation des cotes Y affectées à ce volume,
- le repère du volume se arrivant sous les têtes de débandage et la validation pour la prise en compte de ce repère.

Cet automatisme est prévu pour deux modes de fonctionnement: le choix du mode est commun avec la fonction de suivi de dévirage (même poste de conduite):

- un mode automatique, où le positionnement des rives est effectué par l'automatisme lui-même,
- un mode de conduite local, où le positionnement des rives est effectué par l'opérateur.

Le dialogue opérateur est donc constitué :

- de la sélection du mode de marche,
- de la visualisation de l'état machine,
- en mode local: commande du dévirage à droite et commande à gauche des moteurs de déplacement des rives.

## 2.4. fonction de conduite du centre de découpe

**description du rôle de la fonction par rapport au processus associé:**

Cette fonction a pour but de réaliser la conduite du centre intégré de découpe en coordonnant les différents entités du processus de tracé-découpe du ruban de verre. Le schéma de structure fonctionnelle (de niveau 2) suivant détaille les échanges d'information entre cette fonction et celles de niveau inférieur.

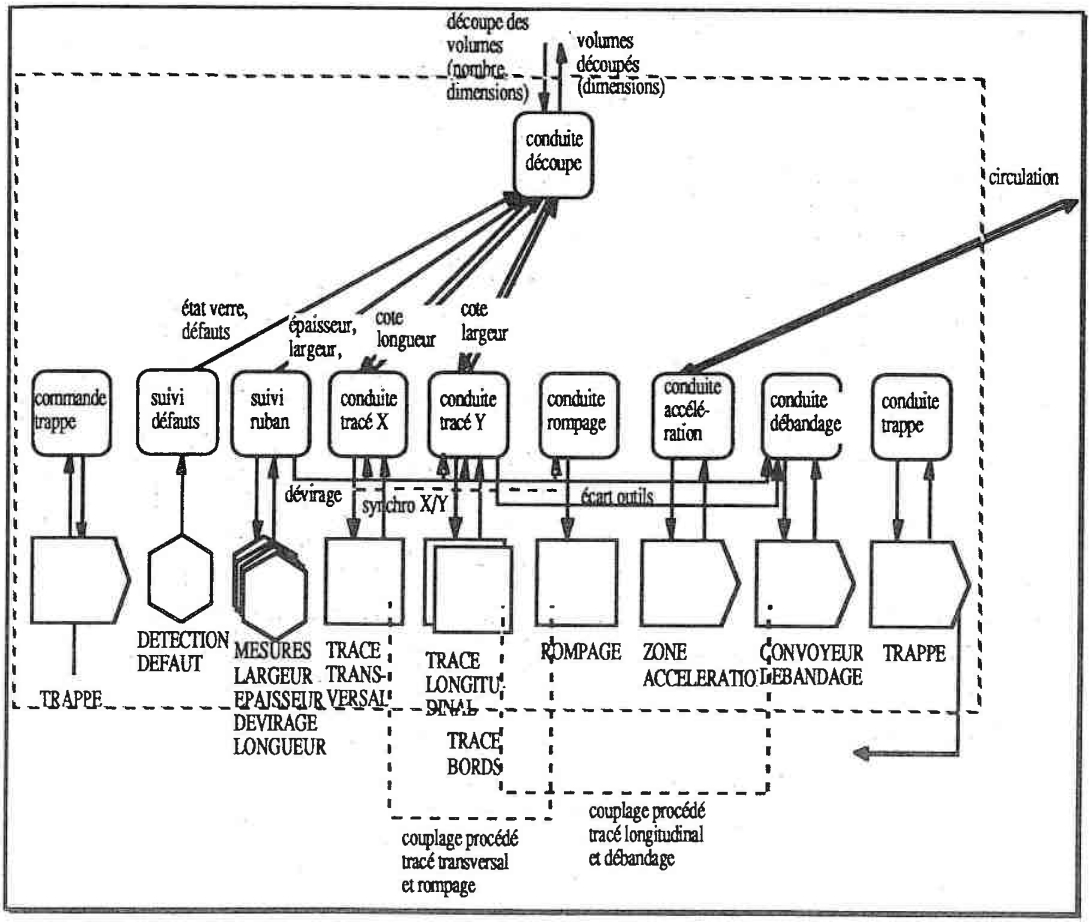



*schéma de la structure fonctionnelle (niveau 2) de la conduite automatique du centre de découpe*

Ce schéma permet d'illustrer l'apport essentiel d'une représentation de la structure fonctionnelle d'un SAP: les analystes peuvent situer précisément la position fonctionnelle de chaque fonction de traitement de l'information dans la structure d'ensemble et choisir le niveau hiérarchique le plus pertinent pour positionner chaque fonction. Dans le cas présent, les concepteurs ont choisi de ne pas matérialiser les relations de service des fonctions rompage et débandage par des échanges d'information de type requête-compte-rendu avec la fonction de niveau supérieur. On constate en effet qu'il n'y a pas de

fonction chargée de coordonner tracé X et rompage (respectivement tracé Y et débandage). Ceci n'exprime pas le fait de l'absence de coordination (des échanges d'informations permettent d'ailleurs de réaliser cette coordination). Ce choix exprime le fait que cette coordination ne nécessite pas l'implantation d'une fonction de traitement de l'information dédiée à cet effet.

La représentation fonctionnelle conduit donc à prendre des décisions de conception que l'on doit justifier. Le schéma de niveau 2 détaille les échanges d'informations qui seront effectivement supportés par l'architecture opérationnelle. Les informations nécessaires à l'exécution de ces fonctions matérialisent les couplages procédé avec les fonctions de tracé respectivement transversal et longitudinal, elles sont donc directement échangées avec ces mêmes fonctions. Les couples d'opérations tracé transversal / rompage et tracé longitudinal / débandage ne constituent en effet que des opérations globales de découpe dont les deux opérations élémentaires sont réparties géographiquement. De tels couplages sont mis en évidence dans le schéma de niveau 1 où l'accent est d'abord mis sur la structure hiérarchisée du système d'information par rapport à celle du processus de production. L'intérêt de cette représentation "orientée processus" est de mettre immédiatement en évidence cette notion de couplage. Dans le cas étudié ici la remontée des informations au niveau conduite découpe est inutile car aucune prise de décision n'est nécessaire à ce niveau de coordination des activités. De plus une coordination hiérarchisée et centralisée au niveau conduite découpe ou conduite découpe X (respectivement Y) rendrait technologiquement parlant le respect des contraintes de temps difficile: les concepteurs seront donc amenés à réaliser des synchronisation par l'intermédiaire de liaisons point à point en l'absence d'un système de communication capable de fournir un service satisfaisant de telles contraintes de temps de transfert.

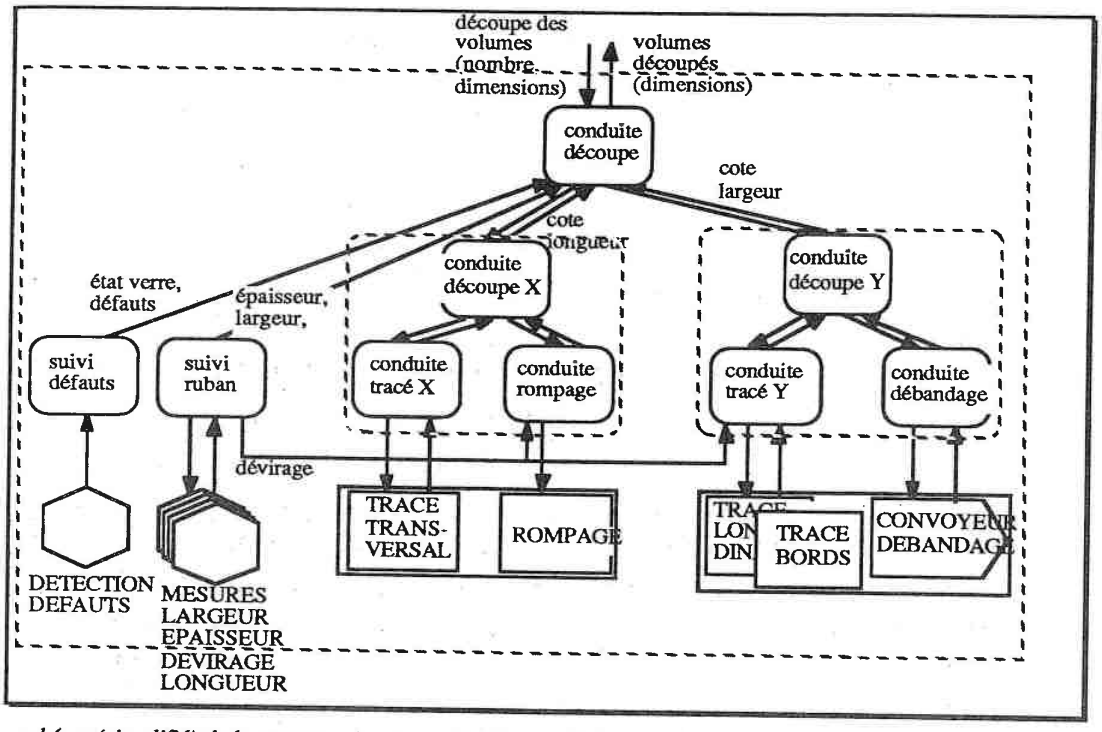



*schéma (simplifié) de la structure fonctionnelle (niveau 1) de la conduite automatique du centre de découpe*

## 2.5. fonctions de gestion et de suivi de production

**description du rôle de la fonction par rapport au processus associé:**

Ces fonctions assurent la gestion / planification et le suivi de la production pour toute l'unité considérée. Le schéma fonctionnel représente donc tout le système considéré à un niveau de détail peu élevé mais avec les interactions entre le système et son environnement.

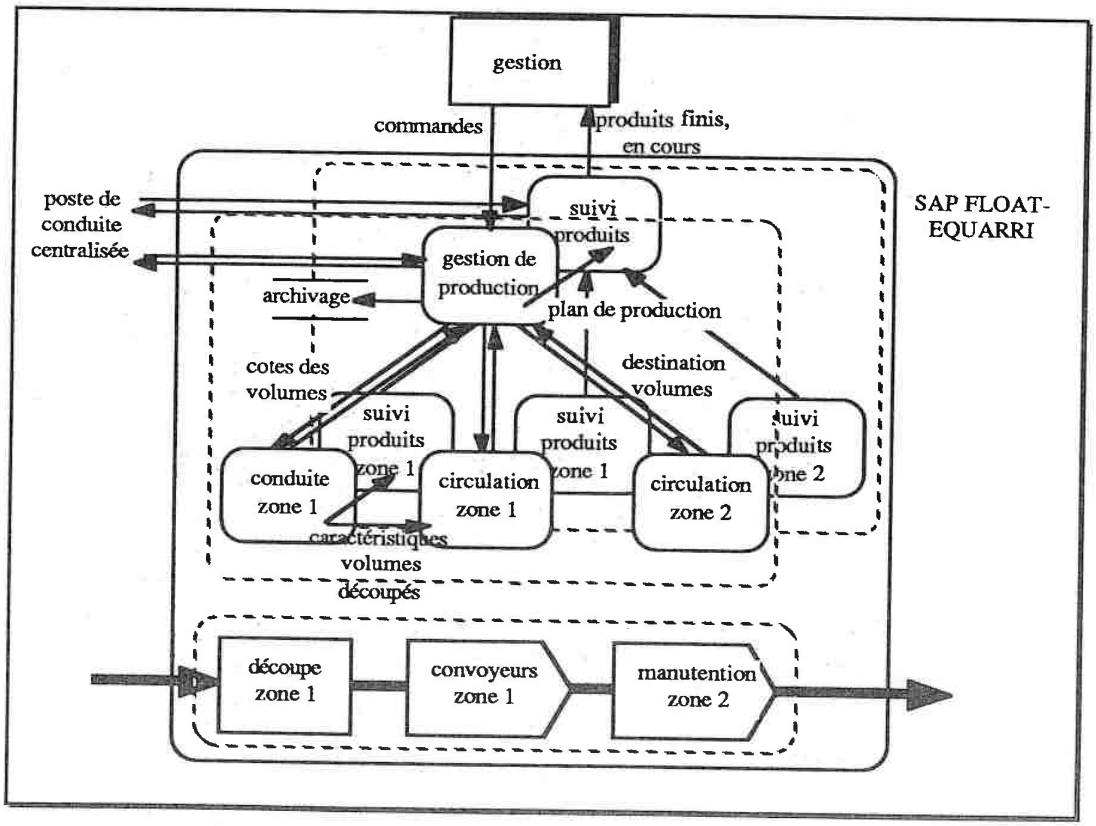



*schéma de structure fonctionnelle de niveau 1 du SAP FLOAT (schéma de contexte)*

Le schéma de niveau 1 fait apparaître deux "plans fonctionnels" principaux pour les deux fonctions de base du système que sont la conduite du processus de production et le suivi des produits. Dans le schéma de niveau 2 des regroupements et des choix d'organisation de la structure de contrôle ont été effectués par les concepteurs: notamment le regroupement des fonctions de suivi de produit et des fonctions de conduite de la circulation étant données leurs fortes interactions. Les fonctions de conduite de la circulation ne sont pas directement pilotées par le niveau supérieur dans le cas où les produits sont acheminés vers leur lieu de relevage et stockage uniquement en fonction de leur dimension (option de suivi dimensionnel). Dans le cas d'un "suivi intégral" un

dialogue plus important reflétant cet accroissement de flexibilité est nécessaire entre les différents dispositifs de convoyage (convoyeurs et aiguillages) et les fonctions circulation et suivi pour déterminer le cheminement et la position des volumes dans la zone 2. Dans le cas où des paramètres tels que la qualité interviennent, la seule dimension des plateaux n'est plus suffisantes pour leur l'identification. Le suivi intégral devient alors indispensable.

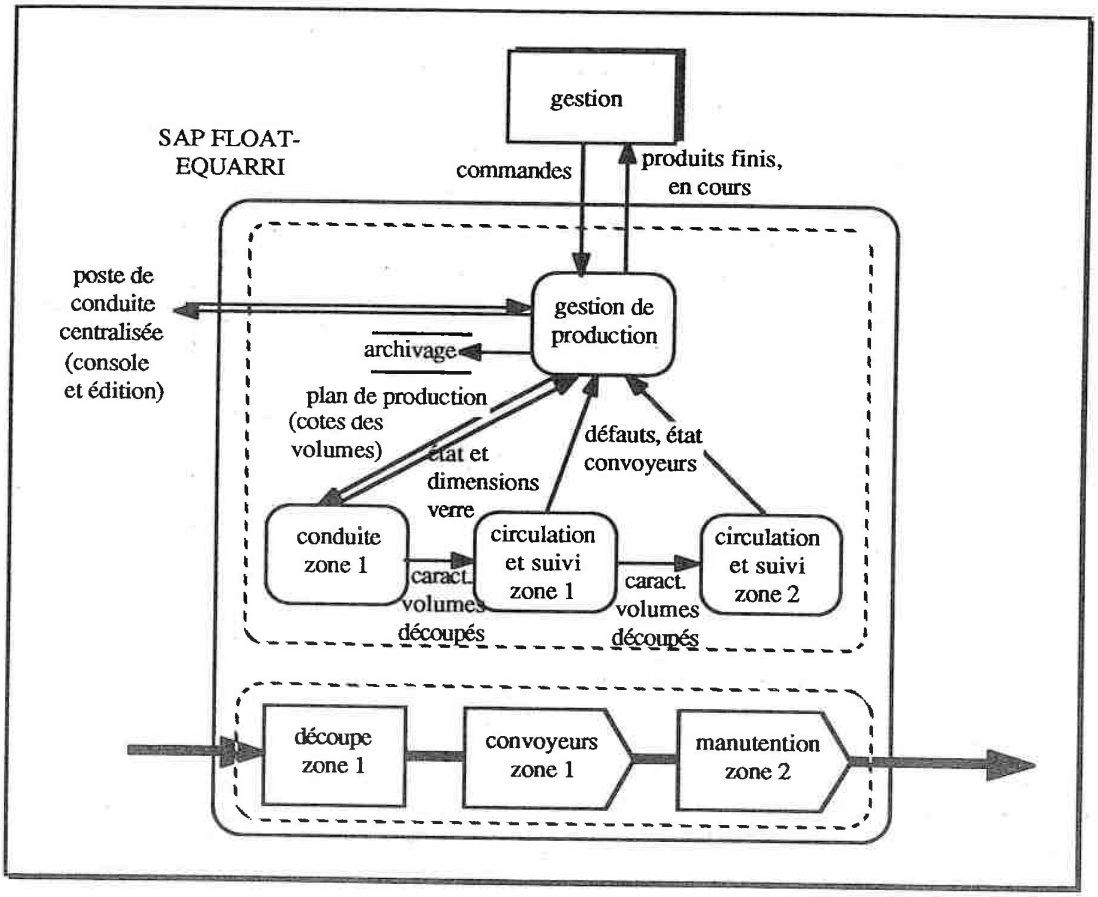



*schéma de structure fonctionnelle de niveau 2 du SAP FLOAT (schéma de contexte)*

# 3. ARCHITECTURE GLOBALE

## 3.1. STRUCTURE FONCTIONNELLE

Les entités fonctionnelles d'une unité float-équarri ont été organisées en quatre niveaux:

**4 - niveau de décision et gestion des ressources,**
**3 - niveau de conduite et suivi par zone,**
**2 - niveau de de conduite des différentes machines,**
**1 - niveau des automatismes réflexes .**

Ce découpage est basé sur des critères :
- de structure du processus de production,
- du niveau de détail par rapport à la production,
- de temps de réponse et d'horizon,
- de conditions d'exploitation par les opérateurs.

| NIVEAU | PROCESSUS | DETAIL | HORIZON | EXPLOITATION |
|---|---|---|---|---|
| 4 | ligne complète | série de volumes | de la mn à l'heure | centralisée |
| 3 | zone | volume logique | la s | par zone |
| 2 | machine | volume physique | 10 à 100 ms | reprise locale |
| 1 | outil / actionneur | variable élémentaire | quelques ms | reprise manuelle |

**niveau 1:**

Les automatismes réflexes doivent assurer une précision de l'ordre du mm sachant qu'un plateau parcours 1 cm en 20 mS. Trois exemples types de ces automatismes sont les:

- variateurs de vitesse,
- asservissements des chariots porte-outils du tracé transversal,
- asservissements de position sous releveuse.

**niveau 2:**

Les automatisme de conduite des machines assurent les mouvements (séquences) en liaison avec les automatismes réflexes. La gestion des modes de marche est assurée à ce niveau et bien-sûr les transitions entre ces modes. Le mode automatique élabore les requêtes à destination des automatismes réflexes, et dans certains cas directement vers les actionneurs, à partir des requêtes reçues en provenance du niveau supérieur. En cas d'indisponibilité du niveau supérieur: mode reprise locale, les requêtes sont rentrées

directement par l'opérateur sur un pupitre local, qui permet également la visualisation des compte-rendus d'exécution et états des dispositifs commandés.

**niveau 3:**

Les fonctions de conduite et de suivi permettent l'**exploitation centralisée** des différentes zones de la ligne de production. Ces fonctions sont donc essentiellement concernées par le système de transport, la position des produits sur la ligne de production et l'état des différentes machines.

**niveau 4:**

Les fonctions de décision et de gestion permettent d'élaborer le plan de fabrication et d'en assurer le suivi en tenant compte:

- de la qualité du produit,
- des ressources disponibles (machines du processus de production et leurs automatismes de conduite),
- des caractéristiques des volumes présents dans la ligne.

## 3.2. REGROUPEMENTS DE FONCTIONS

Les fonctions d'automatisme sont regroupées de façon à minimiser les liaisons inter-automates. Certaines machines possèdent leurs propres automatismes comme les releveuses par exemple.

Pour la zone 1, les regroupements sont les suivants:

- mesure ruban, mesure dévirage, conduite pont de tracé des bords et mise en service des outils correspondants, mise à largeur des rives de débandage,
- conduite découpe transversale: gestion des outils pont de tracé transversal, rompage,
- conduite découpe longitudinale: gestion des outils pont de tracé longitudinal, débandage,
- asservissements tracé outils X,
- asservissements position outils Y,
- asservissements montée / baisse outils Y,
- automatismes de circulation et de suivi.

Les commandes opérateurs des différents dispositifs de la zone 1 ont été regroupés sur un pupitre de conduite banalisé local , sauf pour la montée / descente des trappes qui ont été implantées à proximité immédiate des trappes.

exemple de regroupement de fonctions:

Les fonctions:

- mesure de dévirage,
- conduite du pont de tracé des bords avec suivi de dévirage,
- suivi de dévirage des rives de débandage,

ont été regroupées sur un même dispositif de traitement étant donnée l'importance du couplage induit par les liaisons entre ces fonctions. (Ce couplage étant lui-même induit par le couplage procédé.)

## 3.3. SURETE DE FONCTIONNEMENT

Certains opérateurs du processus ont un caractère **critique** étant donné:

- la nature **continue** de la première partie du processus jusqu'à la découpe du ruban,
- la **ligne unique** de convoyage des volumes découpés.

Les automatismes des machines de découpe et des convoyeurs doivent donc être repris en secours:

- soit par des automatismes en "mode dégradé", n'assurant qu'une partie de la fonction: marche forcée des moteurs des convoyeurs par exemple,
- soit par un automatisme "redondant",
- soit par la reprise locale en "mode manuel".

La redondance des automatismes est possible mais celle-ci n'est pas indispensable, du fait de la fiabilité des matériels de contrôle-commande, ici des automates programmables industriels (API), supérieure à celle des matériels de puissance et des éléments mécaniques. Pour pallier les défaillances toujours possibles des automates, il a été jugé préférable de prévoir un découpage en fonctions présentant une autonomie maximum et la définition de modes de marche dégradés: manuels ou semi-automatique par une position de replis "forcée" au niveau des asservissements.

exemple des ponts de tracé:

En cas d'indisponibilité du niveau conduite de centralisée, les consignes de la découpe peuvent être saisies directement par l'opérateur au niveau de l'automatisme gérant les différentes rangées d'outils selon un mode de fonctionnement semi-automatique. En cas d'indisponibilité de cet automatisme, une reprise en fonctionnement manuel est possible au niveau des asservissements de chaque rangée d'outil. Au niveau mécanique, une redondance de deux à quatre ponts de tracé permet de garantir une disponibilité

suffisante. La disponibilité globale des automatismes est renforcée par la redondance des capteurs roue de mesure et compteur.

exemple du convoyage:

Pour les automatismes du convoyage, un fonctionnement dit "en marche forcé" est prévu afin de pouvoir évacuer les volumes sans arrêter toute la ligne de fabrication même en cas d'incident.

## 3.4. ARCHITECTURE MATERIELLE

Comme nous l'avons vu l'architecture est en grande partie hiérarchisée mais des informations circulent entre fonctions de même niveau. Certaines d'entre elles sont des informations de synchronisation à la ms qui ne peuvent transiter que par des entrées sorties classiques. Le plus grand nombre de ces informations peut cependant supporter des temps de transit de l'ordre de 100 ms, ce qui permet d'utilise des réseaux locaux industriels.

Les matériels choisis pour assurer les différentes fonctions des niveaux 2 à 4 ont été des automates programmables industriels afin de bénéficier:

- de la fiabilité et de l'adaptation au milieu industriel,
- de la facilité de mise au point,
- de la possibilité pour le client de faire face à des modifications sur son installation sans gros investissements en formation et en matériel.

Pour le niveau 2, des automates d'environ 100 E/S sont affectés à chaque machine afin d'assurer un découpage modulaire.

Un automate haut de gamme (800 E/S) peut assurer la gestion des fonctions de niveau 3 et 4.

Les automates sont interconnectés par le réseau du constructeur.

Les automatismes asservissements de niveau 1 sont implantés en automatismes cablés ou à base de microprocesseurs pour des raisons liées au temps de réponse et à la nature des traitemets à réaliser.

La conduite est prévue pour que les commandes soient regroupées au maximum pour une zone géographique donnée, de façon à faciliter le travail des opérateurs sans dégrader le niveau des performances.

Chaque machine dispose d'un pupitre de commandes locales sauf en zone 1 où la concentration géographique a conduit à un regroupement en un seul pupitre. Ces pupitres de commandes locales seront constitués d'écran / clavier et d'un arrêt d'urgence. Les critères ayant conduit à ce choix sont:

- la standardisation du matériel (tous les pupitres locaux sont identiques),
- la souplesse de modifications (ajouts de commandes ou de signalisation) par rapport à une solution câblée boutons poussoirs / voyants ,
- la possibilité d'extensions pour des fonctions d'aide à la maintenance au pied des machines,
- la diminution du câblage sur le site,
- un coût comparable par rapport à un pupitre conventionnel.

Les postes de conduite centralisée seront constitués de matériels type micro-ordinateurs offrant une intelligence locale permettant de proposer des dialogues opérateurs plus élaborés.

## 4. RESULTATS

Cette étude a permis ["Point sur l'action recherche et développement Float", Note Technique ESIA 5320 A, AUT 005, Fév.88]:

- une identification précise et détaillée des principales fonctions,
- la spécification des E/S et des liaisons entre toutes ces fonctions, opérateurs et machines,
- une description-explication des éléments du processus physique associés aux différentes fonctions,
- un résumé des traitements automatisme.

L'objectif de cette étude était de prendre du recul par rapport aux différentes réalisations, en faisant abstraction des choix d'implantation et de développement des logiciels.

L'apport de l'étude est de permettre une validation des choix de répartition des fonctions par rapport à des architectures matérielles. Cela signifie en clair:

- une structuration de l'architecture permettant de minimiser les couplages et d'accroître l'autonomie des fonctions (optimisation permise par la définition rigoureuse des échanges d'informations);
- une répartition plus explicite des fonctions supportées par les différents partenaires: asservissements machines, automatismes, informatique, et des interfaces correspondants;
- une définition des services et performances de systèmes de communication à mettre en oeuvre.

Les prolongements de cette étude seront:

- analyse des fonctions d'automatismes non traitées,
- étude détaillée des fonctions de gestion, et de diagnostic et aide à la maintenance
- étude des coûts matériels et de développement par fonction (analyse de la valeur).

## 5. ENSEIGNEMENTS TIRES DE CETTE APPLICATION

Démarche suivie par les concepteurs:

La démarche de définition des différentes fonctions n'a pas été imposée a priori. La démarche suivie en pratique a spontanément été à la fois descendante et ascendante. En fait le niveau de départ n'est pas le niveau inférieur 1 ou le niveau supérieur 4, mais le niveau 2. Partant des fonctions du niveau 2, la démarche a été d'abord descendante pour préciser le niveau 1 qui était bien connu. La démarche a ensuite été ascendante pour traiter la conduite et le suivi au niveau 3 avant de définir les relations avec le niveau 4 actuellement peu intégré dans les réalisations existantes. On peut donc dire que la démarche a été "globalement" ascendante.

Ce choix qui peut sembler paradoxal s'explique par les faits suivants:

- le niveau de décomposition du processus de production privilégié par les analystes est le niveau machine décrit dans la première partie. Il permet en effet d'obtenir une bonne connaissance du procédé et de l'organisation des flux de production sans rentrer dans un niveau de détail superflu. Or ce niveau de décomposition correspond au découpage des fonctions au niveau 2;
- les spécialistes ayant travaillé sur cette étude sont des automaticiens. Leur connaissance de l'application porte d'abord sur les niveaux inférieurs;
- l'application d'une approche méthodologique nouvelle est plus facile à assimiler sur des fonctions d'automatisme proches du processus de production;
- le stade de développement et d'intégration des niveaux supérieurs est assez peu avancé dans les applications existantes, même si des publications sur l'ordonnancement dans les industries verrières [Gon 88] montrent que les difficultés en particulier liées aux changements de fabrication devrait certainement progresser à l'avenir.

Bien que cette étude ne constitue qu'un cas particulier, on peut constater que:

- les concepts retenus pour la modélisation de l'architecture permettent une approche mixte ascendante et descendante,
- la démarche particulière suivie pour une application donnée doit pouvoir être adaptée aux objectifs de l'étude et au profil des concepteurs.

Ainsi une étude portant essentiellement sur l'intégration de la gestion de production ou sur l'exploitation avancée de l'unité devra plutôt suivre une démarche globalement descendante.

Par rapport à la présentation théorique des différentes tâches proposées dans la méthode, la mise en pratique conduit à une procédure beaucoup plus itérative, où la distinction entre les différentes tâches n'est pas séquentielle. Ainsi les dialogues opérateurs pour l'exploitation ont bien été définis au niveau fonctionnel, mais les dialogues opérateurs pour les modes de fonctionnement dégradés ont aussi été spécifiés à ce niveau sans attendre une étude globale de la robustesse aux défaillances de l'architecture opérationnelle. Il apparaît en effet que les conditions d'exploitation ("normales") ne peuvent être spécifiées sans traiter les conditions d'exploitation liés aux modes perturbés. Cette souplesse est nécessaire afin de permettre aux concepteurs de préciser sans attendre les étapes suivantes certaines spécifications. Dans cette étude de cas, les concepteurs connaissaient précisément ces conditions d'exploitation. Ceci ne les empêche pas de modifier cette spécification dans la suite de l'étude.

Une telle approche n'est pas contradictoire avec le découpage en tâches que nous avons proposé. En effet, et nous l'avons déjà précisé, ce découpage est fonctionnel au sens où il correspond à des préoccupations différentes et ne correspond pas nécessairement à un découpage séquentiel ou temporel des activités de conception. D'une façon générale, il est apparu souhaitable de laisser une liberté suffisamment large au concepteur lui permettant d'anticiper ou d'itérer certains points de son analyse ou de ses choix de conception. Ce besoin de marge de manoeuvre peut différer suivant la personnalité du concepteur, mais elle doit être d'autant plus large que la méthode est nouvelle pour faciliter son acceptation. Nous avons constaté en effet un besoin de rigueur croissante au fur et à mesure de l'assimilation des concepts de base de la méthode.

## annexe 2:

# MAQUETTE D'OUTIL D'AIDE A L'ANALYSE DE LA ROBUSTESSE AUX DEFAILLANCES

## INTRODUCTION

Ce chapitre présente l'application des principes de bases proposés pour la prise en compte de l'aspect sûreté de fonctionnement au cours du processus de conception d'une architecture de SAP. La maquette développée à l'aide du produit générateur de systèmes-experts G2 a pour but de montrer la faisabilité de l'approche "Analyse de la Robustesse aux Défaillances" exploitant les modèles génériques de Structure Fonctionnelle et d'Architecture Opérationnelle. Cette annexe présente succinctement le produit G2 en mettant l'accent sur les fonctionnalités utilisées. La seconde partie présente la structure des objets de la base de connaissance et celle des règles d'analyse conformément à l'analyse fonctionnelle de l'outil exposée au chapitre 9.

## 1. PRESENTATION DU PRODUIT G2

Le produit G2 de la société GENSYM est un générateur ("shell") de systèmes-experts destinés à l'aide à la conduite temps-réel de procédés industriels ou au diagnostic en ligne de systèmes automatisés de production. Cette vocation première vaut au produit d'être doté d'outils logiciels dont nous n'avons utilisé qu'un certain nombre. N'ont en particulier pas été utilisés:

- le moteur de simulation intégré,
- les logiciels d'intégration et de communication du produit avec d'autres systèmes informatiques de contrôle-commande,
- les possibilités temps-réel, c'est-à-dire le raisonnement sur les durées de validité et l'évolution temporelle des variables ainsi que l'invocation des règles par l'arrivée d'alarmes ou événements.

Nous avons en revanche largement tiré parti des possibilités suivantes fournies d'origine avec le produit :

- structuration de la base de connaissance orientée objet,

- existence d'objets particuliers de type "connexion" et possibilités de référence par l'intermédiaire de ces connexions dans les règles,
- graphisme intégré à ces notions d'objets et de connexions,
- possibilités de structuration et de généricité de la base de règles,
- interfaces de développement en langage naturel et d'édition syntaxique,
- outils de mise au point et de documentation de l'application ,
- outils de dialogue utilisateur (dispositifs de visualisation et de saisie, multifenêtrage, graphisme).

Toutes ces possibilités, alliées aux performances du produit tournant sur toutes les principales stations de travail du marché, font de G2 un outil particulièrement adapté au maquettage de systèmes à base de connaissances. ["Rapid prototyping with G2" Bob MOORE. GENSYM, 1988]

On trouvera le détail de ces possibilités dans le ["G2 user's manual" GENSYM, Version 1.11, 1988].

# 2. STRUCTURE ET COMPOSITION DE LA MAQUETTE

## 2.1. OBJETS

```
** Gensym G2 Knowledge Base Inspection Output

** From KB:     DUA0:'USERS.JLUC$these.KB;
** File:        DUA0:'USERS.JLUC$THESE.QP;1
** Written at: 23 Feb 89  9:11:01 p.m.


write to the file "these" every object-
   definition; every connection-definition; every
   tabular-function; every rule; every generic-
   formula
```

objet-reference, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | objet-reference |
| Superior class | object |
| Attributes specific to class | reference is given by a text-variable; comment is given by a text-variable; default is giv by a defaut-var; specif-def is given by a defaut-var |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | none |
| Default settings for variables | none |
| Stubs | none |
| Icon height | 20 |
| Icon width | 40 |
| Icon description | solid rectangle (0, 0) (40, 20) |
| Color | green |

### 2.1.1. STRUCTURE FONCTIONNELLE

entités fonctionnelles

liaisons fonctionnelles

liaisons de service

liaisons de données partagées

liaisons de synchronisation

entite-fonctionnelle, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | entite-fonctionnelle |
| Superior class | objet-reference |
| Attributes specific to class | none |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | reference is given by a text-variable; comment is given by a text-variable; default is giv by a defaut-var; specif-def is given by a defaut-var |
| Default settings for variables | none |
| Stubs | an input service located at top 20; an output service located at bottom 20; an input donnees-partagees located at left 10; an output donnees-partagees located at right 1 |
| Icon height | inherited |
| Icon width | inherited |
| Icon description | inherited |
| Color | black |

connexion-reference, a connection-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | connexion-reference |
| Superior class | connection |
| Attributes specific to class | reference is given by a text-variable; comments is given by a text-variable; defaut is given by a symbolic-variable |
| Capabilities and restrictions | none |
| Change | none |
| Inherited attributes | none |
| Cross section pattern | 2 red, 1 yellow, 2 red |
| Stub length | 10 |
| Junction block | junction-block-for-functional-liaison |

liaison-fonctionnelle, a connection-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | liaison-fonctionnelle |
| Superior class | connexion-reference |
| Attributes specific to class | reprise has values aucune or indefinie (default is aucune); reseau has values indefini or ma (default is map); absence-info is given by a defaut-var; propagation is given by a defaut var; default is given by a defaut-var |
| Capabilities and restrictions | none |
| Change | none |
| Inherited attributes | reference is given by a text-variable; comments is given by a text-variable; defaut is given by a symbolic-variable |
| Cross section pattern | inherited |
| Stub length | 10 |
| Junction block | none |

service, a connection-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | service |
| Superior class | liaison-fonctionnelle |
| Attributes specific to class | none |
| Capabilities and restrictions | none |
| Change | none |
| Inherited attributes | reprise has values aucune or indefinie (default is aucune); reseau has values indefini or ma (default is map); absence-info is given by a defaut-var; propagation is given by a defaut var; default is given by a defaut-var; reference is given by a text-variable; comments is given by a text-variable; defaut is given by a symbolic-variable |
| Cross section pattern | 4 red |
| Stub length | 10 |
| Junction block | junction-block-for-service |

donnees-partagees, a connection-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | donnees-partagees |
| Superior class | liaison-fonctionnelle |
| Attributes specific to class | none |
| Capabilities and restrictions | none |
| Change | none |
| Inherited attributes | reprise has values aucune or indefinie (default is aucune); reseau has values indefini or ma (default is map); absence-info is given by a defaut-var; propagation is given by a defaut var; default is given by a defaut-var; reference is given by a text-variable; comments is given by a text-variable; defaut is given by a symbolic-variable |
| Cross section pattern | 4 blue |
| Stub length | 10 |
| Junction block | junction-block-for-liaison-donnees-partagees |

liaison-operationnelle, a connection-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | liaison-operationnelle |
| Superior class | connexion-reference |
| Attributes specific to class | none |
| Capabilities and restrictions | none |
| Change | none |
| Inherited attributes | reference is given by a text-variable; comments is given by a text-variable; defaut is given by a symbolic-variable |
| Cross section pattern | 3 green |
| Stub length | 10 |
| Junction block | junction-block-for-liaison-operationnelle |

```
                          defaut-var, an object-definition
Notes                                 OK
User restrictions                     none
Class                                 defaut-var
Superior class                        quantitative-variable
Attributes specific to class          none
Capabilities and restrictions         none
Change                                none
Menu option                           a final menu choice
Inherited attributes                  none
Default settings for variables        validity interval: supplied; chaining options:
                                         forward chain, breadth first backward chair
Stubs                                 none inherited
Icon height                           inherited
Icon width                            inherited
Icon description                      inherited
Color                                 black

                           defaut, an object-definition
Notes                                 OK
User restrictions                     none
Class                                 defaut
Superior class                        object
Attributes specific to class          entite-en-defaut is given by a symbolic-variab
Capabilities and restrictions         none
Change                                none
Menu option                           a final menu choice
Inherited attributes                  none
Default settings for variables        none
Stubs                                 none inherited
Icon height                           30
Icon width                            30
Icon description                      lines (0, 0) (30, 30); lines (0, 30) (30, 0);
                                         lines (1, 0) (30, 29); lines (30, 1) (1, 3(
Color                                 brown
```

### 2.1.2. ARCHITECTURE OPERATIONNELLE

dispositifs opérationnels

réseaux

liaisons opérationnelles

dispositif-operationnel, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | dispositif-operationnel |
| Superior class | objet-reference |
| Attributes specific to class | redondance has values aucune or indefinie (default is aucune) |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | reference is given by a text-variable; comment is given by a text-variable; default is giv by a defaut-var; specif-def is given by a defaut-var |
| Default settings for variables | none |
| Stubs | a liaison-operationnelle located at bottom 20 |
| Icon height | 40 |
| Icon width | 40 |
| Icon description | solid rectangle (0, 0) (40, 40) |
| Color | blue |

reseau, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | reseau |
| Superior class | objet-reference |
| Attributes specific to class | redondance has values aucune or indefinie (default is aucune) |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | reference is given by a text-variable; comment is given by a text-variable; default is giv by a defaut-var; specif-def is given by a defaut-var |
| Default settings for variables | none |
| Stubs | a liaison-operationnelle located at right 20 |
| Icon height | 40 |
| Icon width | 40 |
| Icon description | solid rectangle (0, 0) (40, 40) |
| Color | yellow |

correspondance, a tabular-function-of-1-arg

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Names | CORRESPONDANCE |
| Keep sorted | by args |
| Interpolate? | yes |

```
                    defaut-operationnel-et-redondance-mal, a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   DEFAUT-OPERATIONNEL-ET-REDONDANCE-MA
Tracing and breakpoints                 tracing message level 3 (trace messa
                                           step); breakpoint level 3 (breakp
                                           every step)
for any dispositif-operationnel DO1
if the specif-def of DO1 = 1
and the redondance of DO1 /= the symbol aucune
then conclude that the default of DO1 =
 the default of the dispositif-operationnel DO2
 named by the redondance of DO1
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              defaut-operationnel


Names                                   none
Tracing and breakpoints                 default
for any reseau R
if the specif-def of R = 1
and the redondance of R = the symbol aucune
then conclude that the default of R = 1
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              defaut-operationnel
Rule priority                           6
Depth first backward chaining precedence 1
Timeout for rule completion             use default

                                        a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   none
Tracing and breakpoints                 default
for any reseau R
if the specif-def of R = 0
then conclude that the default of R = 0
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              defaut-operationnel
Rule priority                           6
Depth first backward chaining precedence 1
Timeout for rule completion             use default
```

### 2.2.2. TRANSPOSITION

Quand tout dispositif opérationnel change d'état (transition entre défaut ou pas de défaut opérationnel), ces règles mettent en défaut les entités fonctionnelles qu'il supporte. De même pour un réseau en défaut les liaisons fonctionnelles supportées sont alors déclarées en défaut si aucun réseau n'est prévu pour reprendre les communications.

```
                                        a rule
                                transposition-ma1, a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   TRANSPOSITION-MA1
Tracing and breakpoints                 default
for any entite-fonctionnelle EF
if the name of any dispositif-operationnel DO =
correspondance(the name of EF) and the default of DO = 1
then conclude that the specif-def of EF = 1
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              transposition
Rule priority                           6
Depth first backward chaining precedence  1
Timeout for rule completion             use default


                                transposition-raz, a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   TRANSPOSITION-RAZ
Tracing and breakpoints                 default
for any entite-fonctionnelle EF
if the name of any dispositif-operationnel DO =
correspondance(the name of EF) and the default of DO = 0
then conclude that the specif-def of EF = 0
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              transposition
Rule priority                           6
Depth first backward chaining precedence  1
Timeout for rule completion             use default
```

```
                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any reseau R
for any liaison-fonctionnelle LF
if the name of R = the reseau of LF
and the default of R = 0
then conclude that the default of LF = 0
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                transposition
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default

                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any reseau R
for any liaison-fonctionnelle LF
if the name of R = the reseau of LF
and the default of R = 1
then conclude that the default of LF = 1
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                transposition
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default
```

### 2.2.3. PROPAGATION

Ces règles examinent pour chaque entité fonctionnelle si certaines des informations nécessaires sont indisponibles. L'absence d'information peut provenir soit de la défaillance de l'entité fonctionnelle responsable de la production de cette information ou de la défaillance de la liaison. Le cas échéant et si aucune solutions correctives n'est spécifiée au niveau de la structure fonctionnelle (reprise en conduite locale, valeur par défaut ou position de replis), alors l'entité fonctionnelle est déclarée en défaut induit.

```
                                        a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   none
Tracing and breakpoints                 default
for any liaison-fonctionnelle LF
if the default of the entite-fonctionnelle EFP at an input end of LF = 1
or the specif-def of EFP = 1
or the default of LF = 1
then conclude that the absence-info of LF=1
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              none
Rule priority                           6
Depth first backward chaining precedence 1
Timeout for rule completion             use default

                                        a rule
Options                                 invocable via backward chaining,
                                        invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   none
Tracing and breakpoints                 default
for any liaison-fonctionnelle LF
if the default of the entite-fonctionnelle EFP at an input end of LF = 0
and the specif-def of EFP = 0
and the default of LF = 0
then conclude that the absence-info of LF = 0
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              none
Rule priority                           6
Depth first backward chaining precedence 1
Timeout for rule completion             use default
```

```
                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any liaison-fonctionnelle LF
if the absence-info of LF = 1
and the reprise of LF = the symbol aucune
then conclude that the propagation of LF=1
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default
                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any liaison-fonctionnelle LF
if the absence-info of LF = 1
and the reprise of LF /= the symbol aucune
then conclude that the propagation of LF=0
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default
                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
for any liaison-fonctionnelle LF
if the absence-info of LF = 0
then conclude that the propagation of LF=0
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default
```

```
                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK but not active.
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
initially conclude that the default of every liaison-fonctionnelle = 0
and conclude that the absence-info of every liaison-fonctionnelle = 0
and conclude that the propagation of every liaison-fonctionnelle = 0
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default

                                          a rule
Options                                   invocable via backward chaining,
                                          invocable via forward chaining,
                                          may cause data seeking,
                                          may cause forward chaining
Notes                                     OK but not active.
User restrictions                         none
Names                                     none
Tracing and breakpoints                   default
initially conclude that the default of every reseau = 0
Scan interval                             none
Focal classes                             none
Focal objects                             none
Categories                                none
Rule priority                             6
Depth first backward chaining precedence  1
Timeout for rule completion               use default
```

```
                              initialisation-defaut-op, a rule
Options                                       invocable via backward chaining,
                                              invocable via forward chaining,
                                              may cause data seeking,
                                              may cause forward chaining
Notes                                         OK but not active.
User restrictions                             none
Names                                         INITIALISATION-DEFAUT-OP
Tracing and breakpoints                       default
initially conclude that the default of every dispositif-operationnel = 0
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    initialisation
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default

                                              a rule
Options                                       invocable via backward chaining,
                                              invocable via forward chaining,
                                              may cause data seeking,
                                              may cause forward chaining
Notes                                         rule-xxx-5: OK but not active.
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
initially conclude that the specif-def of every entite-fonctionnelle = 0
and conclude that the default of every entite-fonctionnelle = 0
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    none
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default

                                               a rule
Options                                       invocable via backward chaining,
                                              invocable via forward chaining,
                                              may cause data seeking,
                                              may cause forward chaining
Notes                                         OK
User restrictions                             none
Names                                         none
Tracing and breakpoints                       default
for any reseau R
if the specif-def of R = 1
and the redondance of R /= the symbol aucune
then conclude that the default of R =
the default of the reseau named by the redondance of R
Scan interval                                 none
Focal classes                                 none
Focal objects                                 none
Categories                                    defaut-operationnel
Rule priority                                 6
Depth first backward chaining precedence      1
Timeout for rule completion                   use default
```

### 2.2.4. RETRANSPOSITION

Tous les dispositifs opérationnels supportant au moins une fonction défaillante sont signalés comme ayant un fonctionnement perturbé.

### 2.2.5. AFFECTATION DES LIAISONS AUX RESEAUX

Ces règles complémentaires et optionnelles permettent dans une phase d'initialisation de déterminer automatiquement et pour tous les cas non-ambigüs, quel est le réseau qui permet de supporter la communication des données de chaque liaison fonctionnelle. Le travail de spécification du concepteur est ainsi réduit aux cas où une liaison peut être supportée par plusieurs réseaux.

### 2.2.6. COHERENCE et BILANS DE PERFORMANCES

Différentes règles permettent d'effectuer automatiquement certaines vérifications sur la cohérence des modèles fonctionnels et opérationnels. On peut ainsi que pout toute liaison fonctionnelle, un système de communication a été prévu; que toute entité fonctionnelle possède au moins une entrée et une sortie. Il est également possible, si toutes les données élémentaires quantifiant les différentes liaisons fonctionnelles ont été spécifiées, d'effectuer la sommation des bilans sur les débits des flux supportés ou de déterminer la contrainte de temps de transfert la plus élevée par chaque réseau.

```
                                        absence-entree, a rule
Options                                         invocable via backward chaining,
                                                invocable via forward chaining,
                                                may cause data seeking,
                                                may cause forward chaining
Notes                                           OK
User restrictions                               none
Names                                           ABSENCE-ENTREE
Tracing and breakpoints                         default
for any entite-fonctionnelle FE
if not (there exists a entite-fonctionnelle connected at an input of FE)
then inform me that
"l'entite fonctionnelle `the name of FE$ n'est pas connectee en entree"
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      coherence
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default


                                        absence-sortie, a rule
Options                                         invocable via backward chaining,
                                                invocable via forward chaining,
                                                may cause data seeking,
                                                may cause forward chaining
User restrictions                               none
Names                                           ABSENCE-SORTIE
Tracing and breakpoints                         default
for any entite-fonctionnelle FE
if not (there exists a entite-fonctionnelle connected at an output of FE)
then inform me that
"l'entite fonctionnelle `the name of FE$ n'est pas connectee en sortie"
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      coherence
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default
```

## 2.3. INTERFACE UTILISATEUR

### 2.3.1. OBJETS
### 2.3.2. REGLES
**déclaration des défauts**
**visualisation des défauts**
**saisie et visualisation des attributs**
### 2.3.3. DISPOSITIFS DE VISUALISATIONS

defaut-reseau, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | defaut-reseau |
| Superior class | object |
| Attributes specific to class | entite-en-defaut is given by a symbolic-variab |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | none |
| Default settings for variables | none |
| Stubs | none inherited |
| Icon height | 30 |
| Icon width | 30 |
| Icon description | lines (0, 0) (30, 30); lines (0, 30) (30, 0); lines (1, 0) (30, 29); lines (30, 1) (1, 30 outline (0, 0) (0, 30) (30, 30) (30, 0) |
| Color | red |

defaut-operationnel, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | defaut-operationnel |
| Superior class | object |
| Attributes specific to class | entite-en-defaut is given by a symbolic-variab |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | none |
| Default settings for variables | none |
| Stubs | none inherited |
| Icon height | 30 |
| Icon width | 30 |
| Icon description | lines (0, 0) (30, 30); lines (0, 30) (30, 0); lines (1, 0) (30, 29); lines (30, 1) (1, 30 |
| Color | red |

specif, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | specif |
| Superior class | object |
| Attributes specific to class | entite-specifiee is given by a symbolic-variab |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | none |
| Default settings for variables | none |
| Stubs | none inherited |
| Icon height | 30 |
| Icon width | 30 |
| Icon description | lines (30, 0) (0, 30); lines (29, 0) (0, 29); lines (28, 0) (0, 28); lines (20, 0) (30, 0 lines (20, 1) (30, 1); lines (20, 2) (30, 2 lines (30, 0) (30, 10); lines (29, 0) (29, 10); lines (28, 0) (28, 10) |
| Color | black |

specif-op, an object-definition

| | |
|---|---|
| Notes | OK |
| User restrictions | none |
| Class | specif-op |
| Superior class | specif |
| Attributes specific to class | entite-specifiee is given by a symbolic-variab |
| Capabilities and restrictions | none |
| Change | none |
| Menu option | a final menu choice |
| Inherited attributes | none |
| Default settings for variables | none |
| Stubs | none inherited |
| Icon height | 30 |
| Icon width | 30 |
| Icon description | lines (30, 0) (0, 30); lines (29, 0) (0, 29); lines (28, 0) (0, 28); lines (20, 0) (30, 0 lines (20, 1) (30, 1); lines (20, 2) (30, 2 lines (30, 0) (30, 10); lines (29, 0) (29, 10); lines (28, 0) (28, 10) |
| Color | blue |

```
                        specif-fct, an object-definition
Notes                                 OK
User restrictions                     none
Class                                 specif-fct
Superior class                        specif
Attributes specific to class          entite-specifiee is given by a symbolic-variab
Capabilities and restrictions         none
Change                                none
Menu option                           a final menu choice
Inherited attributes                  none
Default settings for variables        none
Stubs                                 none inherited
Icon height                           30
Icon width                            30
Icon description                      lines (30, 0) (0, 30); lines (29, 0) (0, 29);
                                         lines (28, 0) (0, 28); lines (20, 0) (30, 0
                                         lines (20, 1) (30, 1); lines (20, 2) (30, 2
                                         lines (30, 0) (30, 10); lines (29, 0) (29,
                                         10); lines (28, 0) (28, 10)
Color                                 green



                      detection-action-operateur, a rule
Options                                  not invocable via backward chaining,
                                         not invocable via forward chaining,
                                         may cause data seeking,
                                         may cause forward chaining
Notes                                    OK
User restrictions                        none
Names                                    DETECTION-ACTION-OPERATEUR
Tracing and breakpoints                  default
whenever any defaut D is moved
then conclude that the specif-def of every entite-fonctionnelle EF =
 (if the distance between EF and the defaut nearest to EF < 30 then 1 else 0)
Scan interval                            none
Focal classes                            none
Focal objects                            none
Categories                               none
Rule priority                            2
Depth first backward chaining precedence 1
Timeout for rule completion              use default
```

detection-action-operateur-op, a rule

| | |
|---|---|
| Options | not invocable via backward chaining, invocable via forward chaining, n seeking, may cause forward chaini |
| Notes | OK, and note that this is one of 2 c named detection-action-operateur- |
| User restrictions | none |
| Names | DETECTION-ACTION-OPERATEUR-OP |
| Tracing and breakpoints | default |

```
whenever any defaut-operationnel DEF is moved
then in order conclude that the specif-def of every dispositif-operationnel DC
 (if the distance between DO and the defaut-operationnel nearest to DO > 30
 then 0 else 1)
and invoke defaut-operationnel rules
```

| | |
|---|---|
| Scan interval | none |
| Focal classes | none |
| Focal objects | none |
| Categories | none |
| Rule priority | 6 |
| Depth first backward chaining precedence | 1 |
| Timeout for rule completion | use default |

saisie-donnees-op, a rule

| | |
|---|---|
| Options | not invocable via backward chaining, invocable via forward chaining, n seeking, may cause forward chaini |
| Notes | OK |
| User restrictions | none |
| Names | SAISIE-DONNEES-OP |
| Tracing and breakpoints | default |

```
whenever any specif-op SO is moved
and when the distance between SO and the nearest dispositif-operationnel < 30
then show the workspace named saisie-des-donnees-operationnelles
 at three-quarter scale with its bottom left corner
 at the bottom left corner of the screen
```

| | |
|---|---|
| Scan interval | none |
| Focal classes | none |
| Focal objects | none |
| Categories | none |
| Rule priority | 6 |
| Depth first backward chaining precedence | 1 |
| Timeout for rule completion | use default |

detection-action-operateur-reseau, a rule

| | |
|---|---|
| Options | not invocable via backward chaining, |
| Names | DETECTION-ACTION-OPERATEUR-RESEAU |
| Tracing and breakpoints | default |

```
whenever any defaut-reseau DR is moved
then conclude that the specif-def of every reseau R =
 (if the distance between R and the
  defaut-reseau nearest to R > 30 then 0 else 1)
and invoke transposition rules for every reseau
```

| | |
|---|---|
| Scan interval | none |
| Focal classes | none |
| Focal objects | none |
| Categories | none |
| Rule priority | 2 |
| Depth first backward chaining precedence | 1 |
| Timeout for rule completion | use default |

```
        detection-action-operateur-op, a generic-formula
Notes                 OK, and note that this is one of 2 distinct items
                        named detection-action-operateur-op
User restrictions     none
Names                 DETECTION-ACTION-OPERATEUR-OP
let the specif-def of any dispositif-operationnel DO= (if the
  distance between DO and the defaut-operationnel nearest to DO >
  30 then 0 else 1)

      detection-action-operateur-reseau, a generic-formula
Notes                 OK, and note that this is one of 2 distinct items
                        named detection-action-operateur-reseau
User restrictions     none
Names                 DETECTION-ACTION-OPERATEUR-RESEAU
let the specif-def of any reseau R = (if the distance between R and
  the defaut-operationnel nearest to R > 30 then 0 else 1)


                                                a rule
if the default of any dispositif-operationnel DO=1
then change the icon-color of DO to orange
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      none
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default

                                                a rule
if the default of any dispositif-operationnel DO=0
then change the icon-color of DO to blue
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      none
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default


                                                a rule
if the specif-def of any reseau R = 1
then change the icon-color of R to red
Scan interval                                   none
Focal classes                                   none
Focal objects                                   none
Categories                                      none
Rule priority                                   6
Depth first backward chaining precedence        1
Timeout for rule completion                     use default
```

| | |
|---|---|
| | a rule |
| Options | invocable via backward chaining, |
| | invocable via forward chaining, |
| | may cause data seeking, |
| | may cause forward chaining |
| Notes | OK |
| User restrictions | none |
| Names | none |
| Tracing and breakpoints | default |
| for any reseau R | |
| if the default of R = 0 | |
| then change the icon-color of R to yellow | |
| Scan interval | none |
| Focal classes | none |
| Focal objects | none |
| Categories | none |
| Rule priority | 6 |
| Depth first backward chaining precedence | 1 |
| Timeout for rule completion | use default |
| | |
| | a rule |
| Options | invocable via backward chaining, |
| | invocable via forward chaining, |
| | may cause data seeking, |
| | may cause forward chaining |
| Notes | OK |
| User restrictions | none |
| Names | none |
| Tracing and breakpoints | default |
| for any reseau R | |
| if the default of R = 1 | |
| then change the icon-color of R to red | |
| Scan interval | none |
| Focal classes | none |
| Focal objects | none |
| Categories | none |
| Rule priority | 6 |
| Depth first backward chaining precedence | 1 |
| Timeout for rule completion | use default |

```
                                        purple, a rule
Options                                     invocable via backward chaining,
                                            invocable via forward chaining,
                                            may cause data seeking,
                                            may cause forward chaining
Notes                                       OK
User restrictions                           none
Names                                       PURPLE
Tracing and breakpoints                     default
if the specif-def of any entite-fonctionnelle FE=1
then change the icon-color of FE to purple
Scan interval                               none
Focal classes                               none
Focal objects                               none
Categories                                  couleur
Rule priority                               6
Depth first backward chaining precedence    1
Timeout for rule completion                 use default

                                         green, a rule
Options                                     invocable via backward chaining,
                                            invocable via forward chaining,
                                            may cause data seeking,
                                            may cause forward chaining
Notes                                       OK
User restrictions                           none
Names                                       GREEN
Tracing and breakpoints                     default
if the specif-def of any entite-fonctionnelle FE=0 and the default of FE =0
then change the icon-color of FE to green
Scan interval                               none
Focal classes                               none
Focal objects                               none
Categories                                  couleur
Rule priority                               6
Depth first backward chaining precedence    1
Timeout for rule completion                 use default

                                       orange, a rule
Options                                     invocable via backward chaining,
                                            invocable via forward chaining,
                                            may cause data seeking,
                                            may cause forward chaining
Notes                                       OK
User restrictions                           none
Names                                       ORANGE
Tracing and breakpoints                     default
if the default of any entite-fonctionnelle FE = 1
and the specif-def of FE = 0
then change the icon-color of FE to orange
Scan interval                               none
Focal classes                               none
Focal objects                               none
Categories                                  couleur
Rule priority                               6
Depth first backward chaining precedence    1
Timeout for rule completion                 use default
```

```
                                regle-saisie-operateur, a rule
Options                                 not invocable via backward chaining,
                                        not invocable via forward chaining,
                                        may cause data seeking,
                                        may cause forward chaining
Notes                                   OK
User restrictions                       none
Names                                   REGLE-SAISIE-OPERATEUR
Tracing and breakpoints                 default
whenever any specif-fct SF is moved
and when the distance between SF and the nearest entite-fonctionnelle < 30
then show the workspace named SAISIE-DES-DONNEES-FONCTIONNELLES
 at three-quarter scale with its bottom left corner
 at the bottom left corner of the screen
Scan interval                           none
Focal classes                           none
Focal objects                           none
Categories                              none
Rule priority                           6
Depth first backward chaining precedence 1
Timeout for rule completion             use default
```

## TABLE des MATIERES :


| | | |
|---|---|---|
| Remerciements | | 5 |
| Sommaire | | 7 |
| Introduction | | 9 |
| **1ère partie** | **PROBLEMATIQUE** | **13** |
| **chapitre 1:** | **Définition du cadre de travail** | **15** |
| 1. | Système Automatisé de Production | 15 |
| 1.1. | Définitions | 15 |
| 1.2. | L'environnement du SAP: Production Intégrée par Ordinateur ou Computer Integrated Manufacturing (PIO/CIM) | 19 |
| 2. | Les constituants du SAP | 23 |
| 2.1. | Le processus de production | 23 |
| 2.2. | Le système d'information | 23 |
| 2.3. | Les hommes | 24 |
| 3. | Architecture | 25 |
| 3.1. | Définitions | 25 |
| 3.2. | Architecture et système de communication | 28 |
| | Conclusion | 29 |
| **chapitre 2:** | **Place des Réseaux Locaux Industriels dans les Architectures de Systèmes Automatisés de Production (SAP)** | **31** |
| 1. | L'apport des RLI | 32 |
| 2. | Influence des RLI sur les architectures | 37 |
| 3. | Besoins en conception d'architectures induits par la mise en oeuvre de solutions réseaux | 41 |
| | Conclusion | 44 |
| **chapitre 3:** | **Contexte des Etudes d'Avant-Projet en Ingénierie** | **49** |
| 1. | Etudes d'avant-projet | 50 |
| 2. | Contraintes liées aux appels d'offres | 51 |
| 3. | Délais | 53 |
| 4. | Expression du besoin du client | 54 |
| 5. | Evaluation d'une architecture | 55 |
| 6. | Méthodes de travail | 56 |
| 7. | Complexité croissante des architectures | 58 |

8. Besoins 59
Conclusion 59

**chapitre 4: Objectifs et démarche 61**
1. Objectifs 61
2. Démarche suivie 63
3. Développement de systèmes à base de connaissances (SBC) 65
4. Plan de travail 72
Conclusion 74
Conclusion de la 1ère PARTIE 75

**2ème partie: CONNAISSANCES EN CONCEPTION D'ARCHITECTURE 77**

**chapitre 5: Concepts de modélisation d'une Architecture 79**
1. Modèles de conception des systèmes d'information 80
1.1. Classification 80
1.2. Adaptation à la modélisation des architectures de SAP en phase d'avant-projet 83
1.3. Synthèse des concepts retenus 92
2. Modèles d'Architecture de SAP 95
2.1. Les principaux modèles d'architecture de SAP 95
2.2. Synthèse des concepts retenus 121
3. Concepts utilisés en ingénierie 125
3.1. Présentation de l'étude 125
3.2. Critères de structuration 127
3.2.1. Structuration basée le processus 127
3.2.2. Structuration basée sur les produits 135
3.2.3. Décomposition hiérarchisée 137
3.2.4. Structuration basée sur les fonctionnalités 138
Conclusion 142

**chapitre 6: Formalisation des concepts 143**
1. Concepts de hiérarchie et de communication 143
1.1. Différents concepts de hiérarchie 143
1.2. Communications 146
1.3. Formalisation par le concept de service 150
1.4. Exemple 155
2. Représentation des connaissances 161

| | | |
|---|---|---|
| 2.1. | Caractéristiques attendues du formalisme | 161 |
| 2.2. | Classification des connaissances | 161 |
| 2.3. | Comparaison de différents modèles conceptuels de représentation de connaissances | 164 |
| 2.4. | Choix retenu | 165 |
| 2.5. | Notations | 167 |
| 3. | Formalisation des modèles | 171 |
| 3.1. | Elements de base | 171 |
| 3.2. | Système Automatisé de Production | 173 |
| 3.3. | Système d'Information | 176 |
| 3.4. | Processus de Production | 196 |
| | Conclusion | 216 |

**chapitre 7: Tâches d'Analyse et de Conception** **217**

| | | |
|---|---|---|
| 1. | Découpage en tâches des études d'avant-projet de conception d'architecture | 218 |
| 1.1. | Position de l'avant-projet de Conception d'Architecture dans le cycle de vie d'un projet | 218 |
| 1.2. | Objectifs du découpage en tâche | 218 |
| 1.3. | Etude comparative de quelques méthodes de spécification des besoins, d'analyse et de conception de systèmes | 220 |
| 1.4. | Présentation des tâches retenues | 229 |
| 2. | Analyse du contexte | 235 |
| 2.1. | Description des activités | 235 |
| 2.2. | Classification des données de base | 236 |
| 2.3. | Techniques d'identification et de formulation des besoins | 248 |
| 3. | Structuration Fonctionnelle | 251 |
| 3.1. | Description des activités | 251 |
| 3.2. | Validation de la structure fonctionnelle | 257 |
| 4. | Conception d'Architecture Opérationnelle | 267 |
| 4.1. | Description des activités | 267 |
| 4.1.1. | Répartition des fonctions | 268 |
| 4.1.2. | Spécification des fonctions techniques | 274 |
| 4.1.3. | Analyse de la robustesse aux défaillances | 276 |
| 4.1.4. | Spécification des performances opérationnelles | 276 |
| 5. | Définition de solutions | 285 |
| | Conclusion | 286 |

Conclusion de la deuxième partie 287

**annexes:** **361**

1. Exemple d'application: architecture de conduite et suivi de production de verre plat selon le procédé FLOAT-EQUARRI. 347

2. Environnement de développement et présentation de la maquette. 389

table des matières 417

INPL
INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE

**AUTORISATION DE SOUTENANCE DE THESE**
**DU DOCTORAT DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE**

---

VU LES RAPPORTS ETABLIS PAR :

**Monsieur QUENEC'HDU, Professeur, Supelec Rennes,**
**Monsieur HATON, Professeur, CRIN/Université de Nancy I.**

Le Président de l'Institut National Polytechnique de Lorraine, autorise :

**Monsieur DELCUVELLERIE Jean-Luc**

à soutenir devant l'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE, une thèse intitulée :

**"Ingéniérie des systèmes d'automatisation de production"**

BIBLIOTHÈQUE UNIVERSITAIRE NANCY SCIENCES

en vue de l'obtention du titre de :

**DOCTEUR DE L'INSTITUT NATIONAL POLYTECHNIQUE DE LORRAINE**

Spécialité : **"Informatique"**

Fait à Vandoeuvre le, **20 Février 1989**
Le Président de l'I.N.P.L.,

**M.GANTOIS**

2, avenue de la Forêt de Haye - B.P. 3 - 54501 VANDŒUVRE CEDEX
Téléphone : 83. 57. 48. 48 - Télex : 961 715 F - Télécopie : 83. 57. 49. 55