ONERA

DÉPARTEMENT TRAITEMENT DE L'INFORMATION ET MODÉLISATION

Pierre Siron (Dr)

Mémoire d'Habilitation à diriger des recherches

Architectures distribuées
pour les hautes performances, la sécurité, la simulation

Discipline : Informatique

Soutenue le 7 janvier 2003 devant la Commission d'Examen

Christian Fraboul, Pr. ENSEEIHT, Toulouse

William Jalby, rapporteur, Pr. Université de Versailles

Bernard Lécussan, Pr. SUPAERO, Toulouse

Thierry Priol, rapporteur, Directeur de Recherche INRIA/IRISA, Rennes

Patrick Sallé, correspondant, Pr. ENSEEIHT, Toulouse

Bernard Toursel, rapporteur, Pr. Ecole Universitaire Polytechnique de Lille

Table des matières

1 Renseignements administratifs

Ingénieur de recherches au CERT-ONERA[1], Département d'Informatique depuis 1984

- architectures parallèles, distribuées
- sécurité des systèmes informatiques
- simulation distribuée

Activités d'enseignement à l'ENSAE depuis 1982

- cursus ingénieur, ex. algorithmique et programmation, langages C, Pascal, Fortran
- mastère et DEA : systèmes d'exploitation, systèmes répartis, sécurité, validation de systèmes
- encadrement de stagiaires

Thèse de docteur-ingénieur de l'ENSAE le 24 janvier 1984

Centre d'Essais de la Méditerranée, scientifique du contingent 1980 - 1981

DEA informatique de l'Université Paul Sabatier 1980

Ingénieur ENSEEIHT, filière informatique et mathématiques appliquées 1980

Né le 5 octobre 1957 à Audincourt (Doubs), marié

Divers : pilote privé, handicapé au golf.

[1] Les sigles utilisés dans ce document font l'objet d'un glossaire

2 Activités de recherches

2.1 Contexte

J'ai mené mes activités de recherches à l'ONERA au DERI, puis au DTIM.

La mission de l'ONERA est de définir et de réaliser des recherches appliquées pour les besoins de l'aéronautique, du spatial ou de la défense.

Le mode de financement de l'ONERA (EPIC) est particulier, en ce sens que, si certains travaux ont pu être menés sur fonds propres (la subvention), la plupart des travaux ont été réalisés sur contrats émanant, en particulier, des services de la DGA (principalement de la DRET, puis du STTC), ou d'autres organismes comme le CNES ou d'entreprises, EADS, France Télécom, etc.

Ce mode de financement est contraignant, tout en étant considéré comme une garantie de l'application des recherches. Cette contrainte se traduit d'une part en un temps conséquent passé dans la recherche des contrats, d'autre part en une obligation de changer de sujet quand le financement sur le thème n'est plus trouvé. Au fil des années, le manque de suivi dans le soutien des thèmes de recherche s'est accru, une explication étant les diverses réorganisations de la DGA.

Mon parcours a concerné les architectures parallèles, ensuite les architectures distribuées, et aujourd'hui la sécurité des systèmes informatiques et la simulation distribuée.

2.2 Thème architectures parallèles et distribuées

J'ai approfondi ce thème depuis 1982 jusque en 1995.

Le besoin au départ concerne les calculateurs à hautes performances ou supercalculateurs. Dans de nombreux domaines scientifiques (aérodynamique, structure, acoustique, etc.), la simulation numérique des phénomènes physiques occupe une place prépondérante. La mise en œuvre de ces simulations nécessite des calculateurs de grandes performances et conduit, dans les années 80, à la définition de nouvelles architectures de machines pour satisfaire les besoins présents et futurs.

L'approche choisie a concerné les multiprocesseurs dits MIMD[2], ceci constituait une continuité pour le groupe Structures de calculateurs que j'ai alors intégré et qui avait développé la première machine data flow. D'autres laboratoires travaillaient davantage sur les nouvelles technologies des processeurs ou sur les calculateurs pipeline ou vectoriels.

Ces études ont été soutenues par la DGA dans le cadre du programme Grands Calculateurs,

[2] Dans la présentation des thèmes abordés, je ne ferai que citer les études menées et je réserverai les explications correspondantes dans le chapitre Synthèse des travaux.

J'ai participé en particulier aux projets Marianne, Multi-AP, ISIS.

Les travaux effectués dans le cadre des architectures parallèles ont porté, outre sur la conception de nouvelles architectures à mémoires locales, sur les moyens nécessaires pour exprimer (langage de synchronisation de tâches) et exploiter le parallélisme. Ils ont été reconnus mais le thème a du être mis en sommeil pour de nombreuses raisons :

- les efforts des industriels dans le cadre du plan supercalculateur n'ont pas été couronnés de succès donc nos études d'accompagnement n'étaient plus justifiées,

- l'accès à la technologie américaine n'était plus menacé (peut-être une conséquence des actions entreprises),

- une thèse souvent entendue était qu'il ne fallait pas demander d'efforts aux utilisateurs, ou alors que ceux-ci étaient prêts à leur tour à "penser parallèle" que si le gain attendu était important. Ce gain dépendait manifestement du nombre de processeurs et il fallait passer au parallélisme dit "massif" (que l'on fixera à plus d'une centaine de processeurs par rapport à quelques unités ou dizaines).

Pour arriver à cela, j'ai alors travaillé dans le domaine des architectures distribuées : des processeurs conventionnels sont reliés par des structures d'interconnexion performantes. Ces structures peuvent être fixes (grilles, hypercubes, ...) ou intégrer des réseaux de commutation. Dans ces deux cas, le principe de la communication repose sur l'envoi et la réception de messages, plutôt que des accès à une mémoire physique partagée réservée arbitrairement aux architectures parallèles. La mode actuelle des clusters n'utilise pas des principes éloignés des architectures distribuées alors étudiées.

Un processeur européen, le Transputer de la société Inmos, et le modèle de programmation CSP (Concurrent Sequential Programming) étaient particulièrement adaptés et ont servi de moteur. Un autre point clé a été le regain des technologies optiques.

J'ai alors réalisé deux gros projets : MILORD, un multiprocesseur avec un réseau d'interconnexion optique, et Modulor, une architecture modulaire et reconfigurable à base de Transputers.

La nouveauté était que ces architectures étaient accessibles pour des laboratoires comme le DERI, elles n'étaient plus réservées, du fait de leur coût ou encombrement, aux gros centres de calcul. En affinant ce raisonnement, j'ai contribué à l'idée d'en faire des architectures embarquées.

2.3 Thème sécurité informatique

J'ai intégré l'équipe Sécurité des Systèmes Informatiques en 1993, et continué à mener des projets dans ce thème jusque en 2000. Cette équipe avait pour objectif de concevoir des systèmes de haute sécurité, aptes à protéger tout à la fois la confidentialité, l'intégrité des informations et la continuité de service. Dans un projet global, qui inclut en particulier la

construction de fondements théoriques efficaces, je me suis impliqué dans la construction de systèmes sûrs, et, par profession, dans la construction de systèmes distribués sûrs.

Les activités de construction de systèmes sûrs cherchent à proposer des techniques et à concevoir des structures d'implantation pour garantir des propriétés de sécurité. Pour les nouveaux systèmes distribués, un travail important a consisté à recenser les différentes alternatives de mise en œuvre, c'est-à-dire au niveau :

- du matériel,
- du système d'exploitation, avec prise en considération des propositions des nouveaux systèmes d'exploitation répartis et des nouveaux protocoles de communication,
- du niveau logiciel intermédiaire ou "middleware",
- des applications.

Il a fallu également suivre l'évolution rapide de ces systèmes répartis.

Les problèmes, qui trouveront plusieurs solutions dans les projets décrits dans la suite, sont :

- la détermination du niveau le plus efficace de mise en œuvre des fonctions de sécurité pour chaque application,
- l'assurance de non-contournabilité des fonctions de sécurité,
- la réutilisation de composants sur étagère ou COTS pour minimiser les coûts de réalisation de ces fonctions,
- la compatibilité entre sécurité et confort de l'utilisateur (maintien des performances en particulier).

Plusieurs solutions sont acceptables, voir les projets de réseau local sûr (SMAC), de système Chorus sûr, ou les projets SAIDA et PEDALO. La conception de nouvelles architectures de sécurité a cependant des points communs :

- ces architectures seront distribuées par nature ou pour des raisons de réduction des coûts et d'efficacité,
- ces architectures respecteront le principe de base de définition d'un sous-système de sécurité (SSS) minimal, évaluable, soit isolé soit réparti soit les deux, qui assure la sécurité de l'ensemble du système.

2.4 Thème simulation distribuée

Après avoir étudié de nombreuses formes d'architectures distribuées, après avoir parfois, pour cela, utilisé des techniques de simulation, il était naturel que je m'intéresse aux nouvelles techniques de simulation distribuée. Je travaille sur ce thème depuis 1996 et il constitue aujourd'hui un des axes scientifiques du département.

Il y aura, comme souvent, un double aspect : les problèmes techniques derrière la simulation distribuée (un champ de recherche et d'application des algorithmes répartis et des architectures distribuées) et le point de vue de l'utilisateur et l'apprentissage pour réussir des simulations distribuées.

Les besoins sont clairement identifiés. La modélisation et simulation à événements discrets est une aide depuis la conception des systèmes complexes (systèmes de défense, systèmes aéronautiques) jusqu'à leur utilisation. Par la suite, j'emploierai le terme de simulation à la place du couple modélisation et simulation. Cette technique, abondamment utilisée, se tourne maintenant vers la distribution car cette distribution permet de multiplier les performance de calcul, elle permet également de rendre modulaire la simulation soit selon les objets simulés, soit selon les métiers ou entreprises des utilisateurs. L'argument clé est qu'il sera plus facile de faire interopérer des modules existants, de les réutiliser, au sein d'une architecture distribuée qu'au sein d'un programme monolithique. Mais, pour cela, il faut également disposer de standards.

C'est pour cela que l'essentiel des travaux a été mené avec et pour le standard HLA, la définition d'une architecture de haut niveau pour la simulation distribuée à événements discrets.

J'ai en particulier réalisé une des premières mises en œuvre de standard avec la conception d'une infrastructure d'exécution adéquate (RTI), et, avec ce RTI, développé des applications significatives : application de défense anti-aérienne.

Un autre besoin a été considéré : la sécurité des simulations distribuées, que ce soit pour les besoins d'une collaboration entre plusieurs entreprises qui désirent conserver leur savoir-faire (projet SAIDA), ou pour exécuter des simulations distribuées qui tiennent compte des infrastructures de sécurité existantes. Notons que mes travaux dans ce thème ont nécessité à la fois la maîtrise de la conception et de la programmation des architectures distribuées, acquise dans le premier thème architectures distribuées, et la maîtrise des techniques de sécurité informatique, acquise dans le second thème sécurité informatique.

La promotion de ce standard à l'ONERA, ou au sein de la Défense, ou dans la fédération de laboratoires FéRIA est une tâche qui se poursuit.

2.5 Problématique commune

Mon parcours a également sa propre unité. En effet, les points communs dans les différents thèmes abordés sont nombreux :

- il s'agit toujours de systèmes parallèles ou distribués, qui nécessitent pour les concevoir ou les utiliser une adaptation de la façon de penser particulière,

- il s'agit de recherche en architecture au sens large. Les aspects matériels ont quelquefois été considérés mais toujours avec le souci de concevoir des architectures générales avec le moyen de les exploiter et pour des applications,

- il s'agit d'utiliser la distribution soit comme une solution, soit comme une contrainte, pour des recherches en performances, sécurité, simulation, etc. , d'où le titre de ce mémoire.

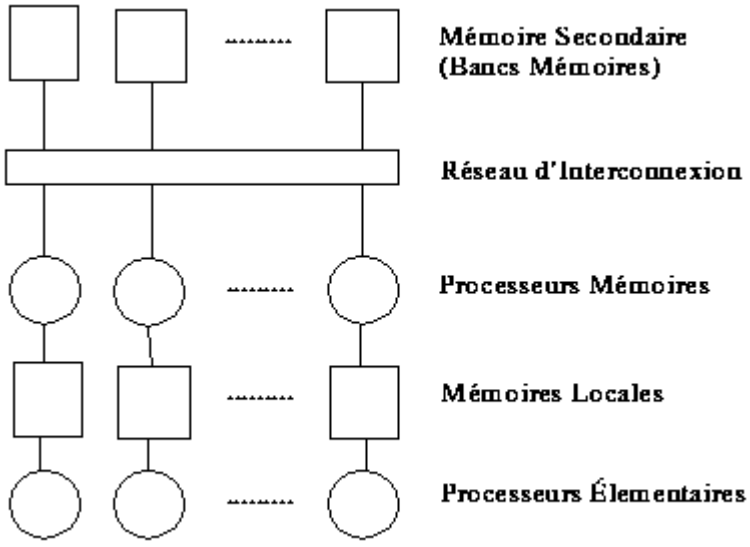
3 Synthèse des travaux

3.1 Architectures parallèles

3.1.1 Introduction : architectures à mémoires locales de type MIMD

Outre les progrès technologiques, la solution pour atteindre les performances demandées réside à la fois dans la définition de nouvelles architectures et dans leur exploitation optimale. J'ai étudié en particulier les architectures multiprocesseurs où l'amélioration de performances est recherchée dans le parallélisme des algorithmes numériques et son exploitation au niveau de la structure de la machine.

Une classification générale de ces nouveaux supercalculateurs a été réalisée et les études évoquées ici se situent dans le contexte de la définition et de l'évaluation des nouvelles architectures à mémoires locales de type MIMD (Multiple Instruction Multiple Data stream), architectures schématisées par la figure suivante.



Architectures MIMD à mémoires locales

Dans ce contexte, j'ai réalisé différents travaux qui seront détaillés dans les paragraphes suivants :

- étude des problèmes d'accès aux données,
- proposition du langage d'exploitation du parallélisme Xanadu,

- mise en œuvre du système expérimental multi-AP,
- mise en œuvre d'applications,
- participation au projet de supercalculateur Isis.

3.1.2 Étude des problèmes d'accès aux données [82-84]

Dans les architectures multiprocesseurs, le parallélisme est exploité non seulement au niveau du traitement (processeurs multiples et indépendants selon le principe dit MIMD), mais également au niveau des transferts de données entre mémoires et processeurs. Mais, comme dans les organisations classiques, les liaisons mémoires-processeurs constituent le principal goulot d'étranglement durant l'exécution des programmes. Ce problème de l'accès aux données prend ici une autre dimension lorsqu'il s'agit d'alimenter avec un maximum d'efficacité un ensemble de processeurs, non synchronisés, avec des données situées à des adresses a priori quelconques en mémoire (il y a des solutions à ce problème dans le cas des architectures SIMD). L'étude entreprise a permis de proposer et d'évaluer différentes stratégies d'accès aux données dans une architecture à mémoires locales de type MIMD.

La première étape a été d'étudier la classification des supercalculateurs et les organisations mémoires associées les plus couramment utilisées : mémoires parallèles, mémoires à deux niveaux, mémoires locales simples ou doubles, réseaux d'interconnexion sans ou avec blocages (crossbar, omega). Le réseau d'interconnexion peut être une couche intermédiaire entre les bancs mémoires et les processeurs mémoires, mais encore entre les mémoires locales et les processeurs élémentaires.

Puis il a été décidé d'effectuer une évaluation de performances par simulation, une prédiction de performances puisque les systèmes étaient en cours de définition. J'ai du pour cela :

- penser à la charge de travail à simuler. L'utilisation de multiprocesseurs de type MIMD à mémoires locales a nécessité la définition d'un langage permettant d'expliciter le déroulement parallèle des programmes. Il s'agit du langage Xanadu présenté dans le paragraphe suivant et qui a été un point d'entrée de cette étude permettant de définir les applications à évaluer.
- réaliser un simulateur à événements discrets, qui a servi à l'étude de ces architectures et applications en général et à l'étude des transferts de données en particulier.

Les résultats de simulation sur une version simple de l'architecture mettent en valeur les difficultés d'allocation des données en mémoire commune et les problèmes de transfert qui sont à résoudre.

J'ai alors proposé diverses optimisations des transferts, qui peuvent se traduire soit par des anticipations, soit par des rémanences de données en mémoire, soit par des algorithmes de

contrôle plus complexes. Ces optimisations sont accessibles à l'utilisateur et intégrées au langage de programmation, ou encore elles font partie du superviseur du système.

Sur un benchmark de programmes représentatifs, les simulations ont montré un gain moyen de 40 % qui permet de conclure sur la bonne compatibilité des optimisations proposées et leur efficacité.

Rapports effectués sur ce thème : [AdCoSi82], [AdCoDoSi84], [Sir84]

Publications effectuées sur ce thème : [AdCoSiSy83], [AdBeCoCrFrHiSiSy84] [AdBeCoSy84], [AdCrFrHiSi85], [AdSi85]

3.1.3 Le langage Xanadu [82-86]

Le langage Xanadu est un langage de description d'applications parallèles, il a été imaginé en même temps que les nouvelles propositions d'architectures parallèles car il n'était pas concevable d'effectuer ces propositions en éludant le problème de leur exploitation.

Une première caractéristique de ce langage est que la description explicite du parallélisme. Une deuxième caractéristique essentielle est qu'il repose sur la notion de tâches, tâches en particulier issues de la structure ou de la décomposition de l'application à exécuter.

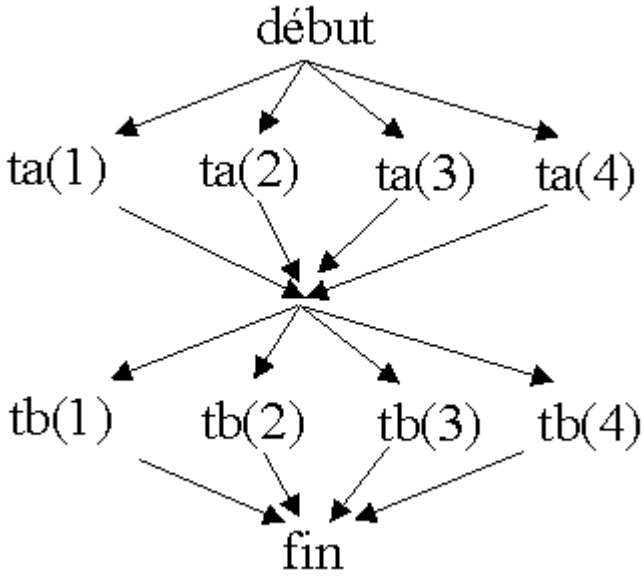
Trois types de tâches ont été définies :

- les tâches de calcul (ou CTASK) correspondent à des étapes algorithmiques exécutables sur les processeurs élémentaires. Elles ont des données d'entrée, de sortie, et une référence à un code de calcul, qui, dans le domaine numérique, est généralement écrit dans le langage Fortran.

- les tâches de mouvement (ou MTASK) offrent la possibilité de réorganiser l'information contenue en mémoire commune, pour, en particulier, optimiser les accès ultérieurs.

- la tâche de séquencement (ou STASK) correspond au code d'enchaînement des autres tâches. Elle est interprétée par le superviseur de la machine MIMD. Elle constitue une partie originale du langage (en 1982), elle est constituée d'un ensemble de formules décrivant les conditions sous lesquelles un ensemble d'actions peut être lancé. Une formule est interprétable sur l'arrivée d'événements, généralement la fin d'autres tâches.

Je peux illustrer les possibilités d'ordonnancement sur deux exemples simples. Le premier est un exemple dit de synchronisation forte (ou du type fork-join), le graphe de tâche est :

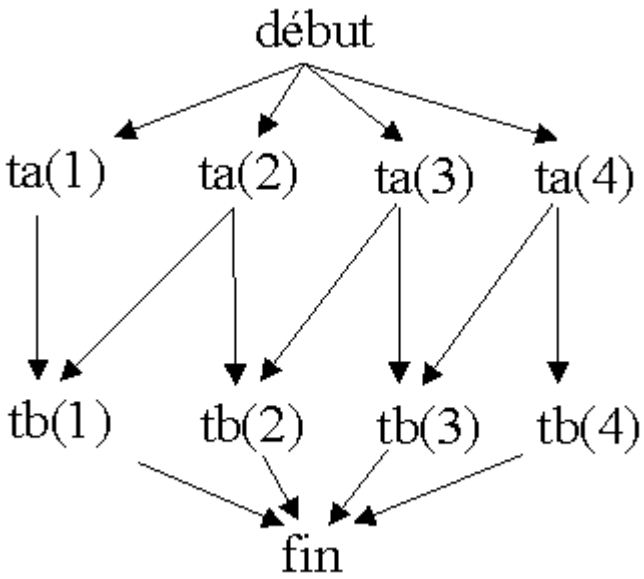


Exemple de graphe de tâches fortement synchronisées

Le code de la STASK correspondante est :

```
STASK

  START => EXPAND ta(x) x=1:4

  ALL x[1:4] ta(x) => EXPAND tb(x) x=1:4

  ALL x[1:4] tb(x) => OVER

END
```

Le deuxième exemple illustre une synchronisation dite faible du même ensemble de tâches, avec moins de contraintes donc susceptible d'être exécuté plus efficacement. Le graphe est :



Exemple de graphe de tâches faiblement synchronisées

Le code de la STASK reste facilement compréhensible :

STASK

START => EXPAND ta(x) x=1:4

EACH x ta(x)AND ta(x+1) => tb(x)

ALL x[1:4] tb(x) => OVER

END

3.1.4 Le projet multi-AP [84-86]

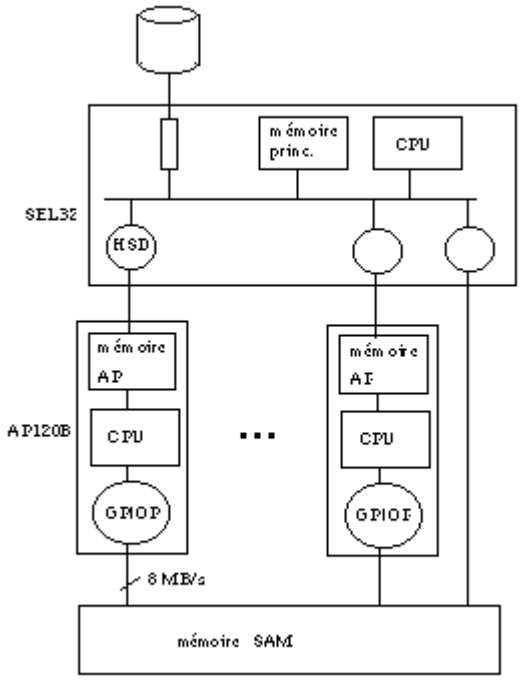
Lors de l'étude précédente, un langage pour exploiter le parallélisme a été défini, ainsi qu'une chaîne de développement de programmes et les outils de supervision associés.

J'ai mis en œuvre ces outils ont été mis dans le contexte de la configuration multi-AP de l'ONERA, qui, malgré quelques contraintes, peut être vue comme une maquette d'architecture MIMD à couplage lâche.

Le développement d'applications numériques à l'aide de ces outils a permis d'évaluer les possibilités d'expression et de supervision offertes, ainsi que le gain apporté par l'exploitation du parallélisme de type MIMD.

Le schéma suivant représente l'architecture. Elle est composée de quatre processeurs

vectoriels AP-120B de FPS, contrôlés par un mini-calculateur hôte SEL-32 de Gould. Chaque processeur vectoriel a sa mémoire locale et est connecté, ainsi que le mini-calculateur, à une mémoire secondaire partagée, SAM, de Texas Memory System. Le parallèle avec l'architecture générique de type MIMD est immédiat.



Architecture multi-AP

Le succès n'a pas été complet pour une raison inattendue. Pour programmer les tâches de calcul sur le processeur vectoriel, le langage Fortran a été à la fois choisi et imposé. Or le compilateur AP-Fortran n'était pas assez mûr et l'ensemble des performances a été ainsi dégradé. Des compilateurs vectoriseurs de bonne qualité allaient voit le jour dans les années suivantes.

Rapports : [CoFrHiSi84], [AdFrHiSi86], [Sir86], [FrSi86]

Articles :[AdFrHiSiBeDa88]

Stages :[Bou83]

3.1.5 Exemple d'application [82-89]

L'algorithme de Jacobi est une des méthodes numériques choisies en collaboration avec les mathématiciens de l'ONERA car il est représentatif pour les futurs utilisateurs des

supercalculateurs. Pour les non numériciens, il a l'avantage d'être simple à comprendre et à exposer, tout en offrant une multitude de mises en oeuvre possibles. J'en n'ai pas encore épuisé toutes les subtilités et il m'accompagnera au fil des années. D'autres algorithmes ont été étudiés, qui ne seront pas présentés : Gauss, Gauss-Seidel, réduction cyclique, etc.

Le but est de résoudre l'équation matricielle suivante, représentant un gros système linéaire issu d'équations différentielles :

$$B \cdot x = b$$

L'algorithme de Jacobi consiste à approcher le vecteur inconnu X par une suite de secteur $X^{(p)}$.

Si A est la matrice déduite de B par suppression des termes de la diagonale, si V est le vecteur formé des termes diagonaux de B, la méthode itérative utilisée correspond à l'algorithme est :

répéter

pour i de 0 à N-1 faire

(1) Y_i <- somme($A_{ij} \cdot X_j^{(p-1)}$) j de 0 à N-1

(2) X_i <- $(B_i - Y_i) / V_i$

(3) Z_i <- $X_i^{(p)} - X_i^{(p-1)}$

(4) $X_i^{(p-1)}$ <- $X_i^{(p)}$

fin pour

jusqu'à $||Z|| < epsilon$

Si on prend un découpage de la matrice initiale en blocs de lignes, la parallélisation de ce programme donne les deux formes de STASK décrites dans la présentation du langage Xanadu.

L'algorithme de Jacobi a d'abord fait l'objet des programmes de test du simulateur d'architectures MIMD, puis il a été mis en œuvre sur l'architecture multi-AP (Xanadu et Fortran).

3.1.6 Le projet ISIS [85-89]

Le supercalculateur Isis a été développé par la société Bull. Son architecture peut être résumée comme étant composée de quatre processeurs scalaires indépendants se partageant

un opérateur vectoriel et une mémoire principale.

J'ai étudié l'exploitation d'Isis en mode de fonctionnement mono-usager, et plus particulièrement l'exploitation du parallélisme de type MIMD entre des tâches de l'application se déroulant sur des flots distincts (les processeurs scalaires). L'exploitation du parallélisme de type SIMD est automatique grâce à un compilateur Fortran vectoriseur. Le système d'exploitation d'Isis est un Unix System V.

J'ai proposé puis réalisé :

- une bibliothèque de "macrotasking" inspirée de celle disponible sur les Cray,
- un système dit de "supertasking" autorisant une plus grande complexité pour la synchronisation des tâches,
- des outils de mise au point.

3.1.6.1 Mise en œuvre du macrotasking

Le macrotasking permet d'exprimer des traitements parallèles au niveau de la subroutine. Une bibliothèque a été proposée sur la gamme Cray. Il apparaît dès l'appel explicite à la procédure de création de tâche TSKSTART, inversement il est possible d'attendre la fin d'exécution des tâches créées par TSKWAIT. Ces deux primitives permettent d'exprimer facilement une synchronisation forte des tâches.

Une synchronisation plus fine est possible mais son expression est aussi compliquée qu'une programmation système (imposée à des numériciens) puisqu'elle repose sur les notions d'événements (EVPOST, EVWAIT) et de verrous (LOCKON, LOCKOFF).

J'ai développé cette bibliothèque en utilisant les fonctions systèmes d'Unix (fork, wait, ...), et en particulier les IPC (Communications entre les processus) du System V (mémoire partagée, sémaphores, messages). Une version optimisée permet d'une part la mise en oeuvre de COMMON Fortran partagé, d'autre part de limiter le nombre de processus créés.

3.1.6.2 Définition et mise en oeuvre du supertasking

Il s'agit de reprendre et améliorer ce qui a été fait avec le langage Xanadu. J'ai proposé d'inclure des directives dites de supertasking dans le code afin d'expliciter des graphes de synchronisation de tâches. Cray avait déjà défini le macrotasking et le microtasking (pour un parallélisme à grain fin).

Le module interpréteur séparé a été supprimé. J'ai alors procédé à la traduction des directives de supertasking en un code Fortran, qui contient des appels à une bibliothèque optimisée de macrotasking et qui contient le code pour l'interprétation de chaque formule. L'exercice était délicat mais nous avons atteint un haut niveau de portabilité.

Si je reprends les graphes de tâches qui me servent d'exemples. Pour la synchronisation forte, le code à écrire devient :

```
C     programme fortran principal
      n=4
[
      debut :- (ta(i),i=1,n)
      (ta(i),i=1,n) :- (tb(i),i=1,n)
      (tb(i),i=1,n) :- fin
]
      stop
      end
```

Avec une synchronisation faible, le programme devient :

```
C     programme fortran principal
      n=4
[
      debut :- (ta(i),i=1,n)
      (ta(i), ta(i+1) :- tb(i)),i=1,n-1
      ta(n) :- tb(n)
      (tb(i),i=1,n) :- fin
]
      stop
      end
```

3.1.6.3 Outils de suivi et de mise au point

Mettre au point un programme parallèle présente de grosses difficultés. Il s'agit de développer un programme à la fois correct et efficace. J'ai réalisé trois fonctions principales :

- un outil de trace permettant d'observer en temps réel ou en différé le comportement de l'application,
- un détecteur de blocage (deadlock),
- un outil facilitant la prise en compte des entrées-sorties dans une application parallèle.

3.1.6.4 Exemple d'application

Ces travaux ont été effectués sans attendre la disponibilité du prototype du supercalculateur Isis. Pour cela une machine Bull-SPS7 avec quatre processeurs a été utilisée. Les performances ne sont pas comparables mais le système d'exploitation est semblable.

Avec la bibliothèque de macrotasking, j'ai obtenu des durées d'exécution (212 inconnues, 28 itérations) du programme séquentiel de 212 s et du programme sur quatre processeurs

de 69 s, soit un speedup légèrement supérieur à 3.

Les performances obtenues avec des directives de supertasking sont semblables pour ce programme. En effet le découpage en paquets de quatre tâches bien équilibrées est très adapté au nombre de processeurs qui était disponible.

La machine Isis n'a pas été un succès car les difficultés techniques de réalisation étaient nombreuses. Les outils proposés auraient pu être utiles et portés sur d'autres machines comme les Cray multiprocesseurs. Des contacts encourageants ont eu lieu, la diffusion n'a pas été faite pour des raisons non techniques.

Rapports : [DiRoSi86], [RoSiRa86], [RoSi87], [BoRoSi89]

Articles :[RoSi88], [RoSiBo89]

Stages :[Bon88]

3.2 Architectures distribuées

Les machines dites massivement parallèles sont des machines à grand nombre de processeurs, elles ont été introduites pour obtenir un gain significatif de performances entre l'exécution d'une application sur un calculateur séquentiel ou parallèle (le code de cette application pouvant être très différent).

Les processeurs utilisés sont des processeurs existants, donc l'effort peut porter essentiellement sur l'assemblage de ces processeurs (la proposition d'architectures distribuées) et sur la capacité à exploiter ce nouveau type de parallélisme.

J'ai principalement dirigé et réalisé deux projets :

- le projet MILORD sur l'apport de l'optique dans les architectures massivement parallèles,
- le projet MODULOR sur la proposition d'architectures avec des qualités particulières de reconfigurabilité et de modularité.

3.2.1 Le projet Milord [86-90]

3.2.1.1 Apports de l'optique

Les architectures des calculateurs massivement parallèles se heurtent en particulier au problème des communications que l'électronique a du mal à résoudre. Par nature, l'optique offre des avantages qui permettent de réaliser des systèmes hautement parallèles ou très

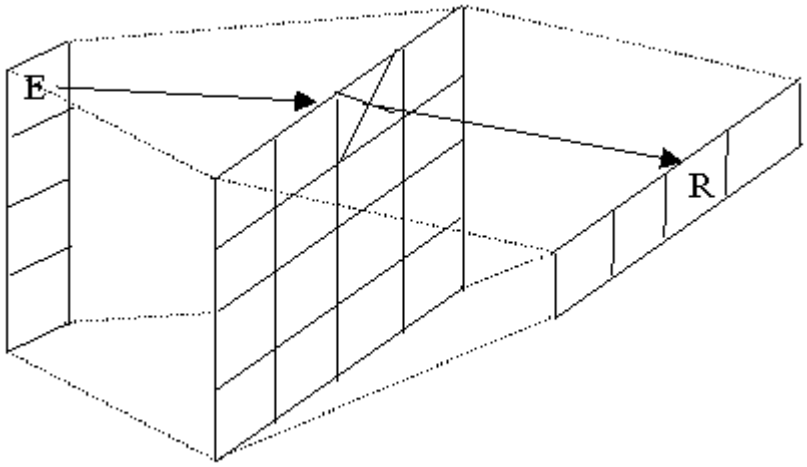
rapides.

En étudiant les différents apports possibles de l'optique, j'ai écarté le "calcul optique" proprement dit. Certes, des transistors commutant à la picoseconde (voire à la femtoseconde) ont été réalisés, mais à l'unité et sur des bancs de laboratoire. Les interfaces avec le monde extérieur sont également limités aux capacités de l'électronique, sauf dans certains cas particuliers de traitement d'images.

J'ai regardé l'apport de l'optique dans son domaine d'application conventionnel : les systèmes de communication. Les avantages qualifiés d'inhérents sont la possibilité de hauts débits, les communications possibles en trois dimension (en espace libre), le parallélisme, l'insensibilité aux interférences, etc.

3.2.1.2 Un réseau crossbar optique

Diverses solutions ont été étudiées pour réaliser un réseau crossbar optique, c'est à dire un réseau sans risque de blocage. Le principe retenu est celui du produit d'une matrice par un vecteur (voir schéma).



Crossbar optique

Chaque émetteur optique est imagé sur une ligne de la matrice. En retour, chaque colonne de cette matrice est imagée sur un récepteur (tout ceci ne pose aucune difficulté à des opticiens). Par différents procédés physiques, il est possible de commander chaque élément de la matrice et de faire en sorte qu'il soit considéré comme "opaque" ou "transparent" vis à vis du récepteur (quand bien même seule la polarisation de la lumière soit affectée). Un réseau crossbar est ainsi obtenu avec les commandes adéquates (le véritable produit analogique serait difficilement exploitable). Le composant central, la matrice, est appelé modulateur spatial de lumière ou SLM (Spatial Light Modulator).

3.2.1.3 Choix d'une architecture reconfigurable

Une des difficultés est la lenteur de réponse des composants actuels, en particulier de la commande du SLM, mais un grand nombre de voies est bien envisageable. Ceci nous oriente vers une architecture du type multiprocesseur, à passage de messages, dont les capacités de communication peuvent varier au fil du temps, d'où le sigle du projet : MILORD ou Multiprocesseur Interconnecté par Liaisons Optiques Reconfigurables Dynamiquement.

Même si un premier objectif de l'étude est la coopération entre opticiens et architectes, il n'est pas question de proposer une architecture autour d'un composant sans avoir à l'esprit les applications (son utilité) et les moyens de l'exploiter. Je me suis donc intéressé à la définition et à l'exploitation d'architectures générales et reconfigurables.

Ces architectures offrent beaucoup de souplesse en adoptant la structure d'interconnexion aux besoins des applications. Le routage coûteux de messages, comme dans le cas d'un hypercube, ne sera plus nécessaire, les communications entre deux processeurs se feront sans intermédiaire.

Deux modes de fonctionnement ont été définis :

- reconfiguration statique : une seule topologie d'interconnexion est utilisée pour une application donnée,

- reconfiguration dynamique : plusieurs topologies sont utilisées et adaptées à chaque phase de l'application. Les reconfigurations auront lieu en des points de synchronisation déterminés dans le programme.

3.2.1.4 Maquettage de Milord

Afin de se confronter aux réelles difficultés, un maquettage de la machine Milord a été réalisé. Je peux celui-ci brièvement :

- les processeurs élémentaires sont des transputers T800 de la société Inmos. Cette brique de base est pour nous idéale puisque elle intègre sur une seule puce processeur de calcul, mémoire, horloges, superviseur et quatre liens de communication série bidirectionnels à un débit de 10 ou 20 Mbits/s. 8 processeurs de calcul ont été utilisés.

- tous les liens des processeurs utilisés, ainsi que deux liens d'un processeur de supervision, sont connectés en entrée et en sortie au réseau crossbar. Il faudra donc un réseau 34 x 34.

- l'opto-électronique d'émission comprend des diodes lasers couplées à des fibres optiques organisés en peigne à l'approche du réseau. Paradoxalement, une commutation en espace libre permet de s'affranchir de certaines contraintes d'encombrement.

- le SLM est une "valve à cristaux liquides". Sa commande est également optique par le

biais d'un tube cathodique, qui génère une image déterminée par le processeur de supervision.

- une barrette de protodétecteurs est utilisée, suivie d'une électronique d'amplification et de remise en forme des signaux.

3.2.1.5 Programmation de Milord

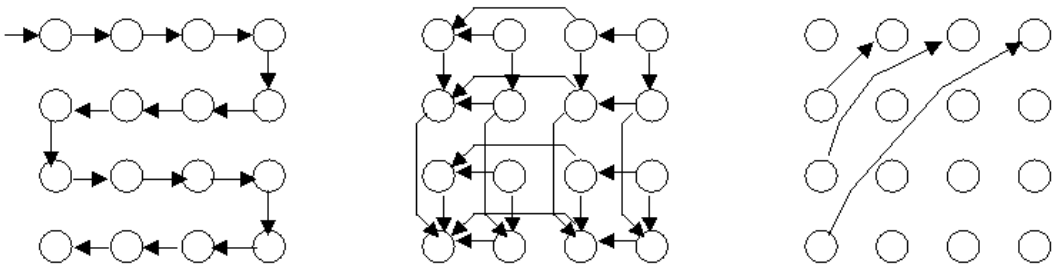
Le langage de programmation associé aux transputers est le langage Occam, qui reprend avec efficacité les principes du langage CSP de Hoare. Pendant une phase, il s'agit simplement d'un graphe de processeurs interconnectés, ce modèle nous convient.

La synchronisation entre les phases peut être également programmée avec ce modèle, la reconfiguration physique est commandée par un processeur de supervision pendant que les processeurs de traitement sont passifs, c'est-à-dire dans l'attente d'une communication sur un canal indéterminé.

3.2.1.6 Exemple d 'application

L'application Jacobi a été reprise. Un ensemble de n² processus disposés en grille a été utilisé. La matrice A est découpée en blocs de côté N/n et répartie régulièrement sur la grille de processus. Les vecteurs x, v et b sont découpés en sous-vecteurs de taille N/n et alloués à la première ligne de processus.

La phase 0 de l'application consiste à charger toutes ces données depuis la machine hôte. Une topologie en pipeline a été proposée pour cela.



Différentes phases de l'application

La boucle principale de calcul consiste, dans la phase 1, à faire le produit de A par x, et d'obtenir le vecteur résultat sur la première colonne de processus. Pour cela, le vecteur x doit être diffusé verticalement sur chaque colonne de processus. Je propose un arbre binaire. Les produits locaux sont effectués, puis une somme est nécessaire sur chaque ligne (opération de réduction). J'utilise pour cela d'autres arbres binaires.

La suite du calcul nécessite d'abord la phase 2 pour transposer le vecteur résultat de la première colonne de processus vers la première ligne.

Le reste de l'itération concerne la première ligne (calcule de la norme par exemple) et peut utiliser la configuration de la phase 1.

Il faut souligner que chaque processus a au plus 4 canaux de communication (cas n=4) et pourra être alloué à un processeur du type Transputer sur Milord.

3.2.1.7 Discussion

Un réseau 24 x 24 a été achevé. Des programmes de test ont tourné. Des difficultés techniques ont été rencontrées qui ont empêché l'exécution des applications de calcul initialement projetées.

Les problèmes ont concerné le bruit modal des fibres d'entrée et des mesures sur les caractéristiques spectrales des diodes lasers pour divers taux de modulation ont été entreprises. Un nouveau module de réception destiné à court terme la barrette actuelle a été caractérisé en ce qui concerne la puissance optique minimale nécessaire pour un taux d'erreur de 10-9 à 100 Mbits/s, puissance qui a fait défaut sur la maquette. Enfin un nouveau modulateur était disponible à la fin de l'étude permettant de diminuer d'un facteur 20 le temps de reconfiguration global.

Les conclusions ont donc été variables :

- la maquette n'a pas fonctionné comme prévu mais présentait des possibilités d'évolution intéressantes,

- il y a eu un choc de culture et des dialogues réussis et intéressants entre opticiens et informaticiens. Il n'était plus l'habitude de devoir caractériser un matériel, d'étudier ses défauts avant de l'utiliser. La partie opto-électronique est bien le maillon faible.

- le projet Milord a eu plusieurs suites, d'autres collaborations entre les départements d'optique et d'informatique, le projet Modulor.

Rapports : [CoSiTh87], [CoSiTh88], [CoSiTh89], [CoSi90]

Articles : [CoSiTh87], [CoSiThChFrLa87], [ChCoFrLaSiTh87], [ChFrLaCoSiTh87], [FrBoChLaCoSiTh88], [ThCoSi88], [CoSiThFrBo89], [FrBoBoCoSi90]

Thèse (participation) : [Thi89]

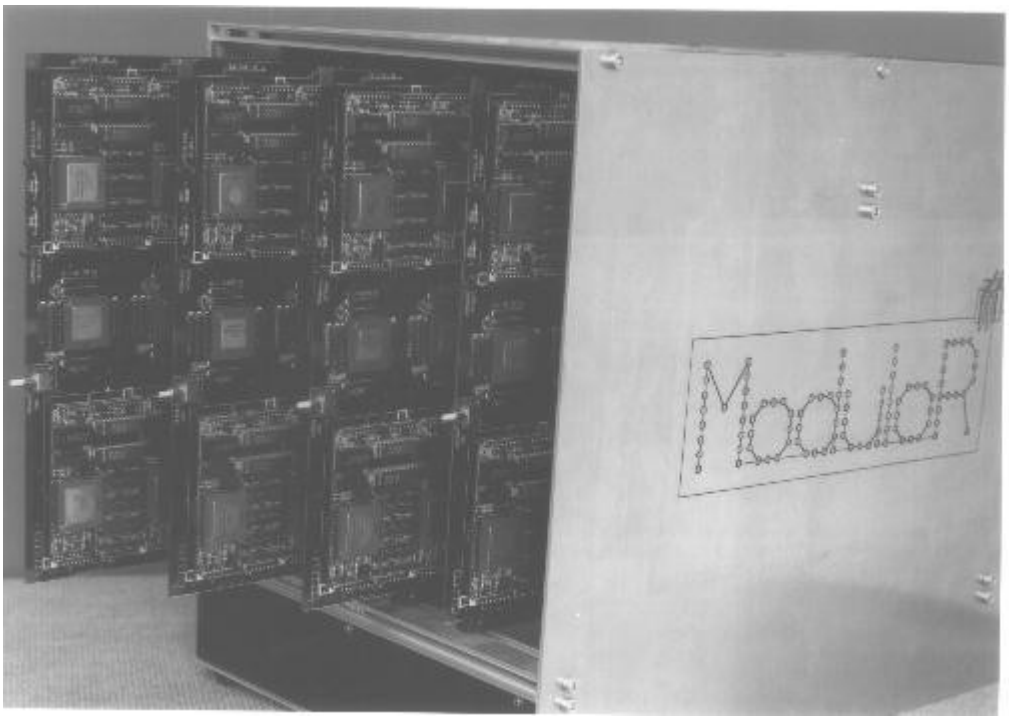
3.2.2 Le projet MODULOR [89-95]

3.2.2.1 Présentation générale

Le projet MODULOR fait suite au projet MILORD en abandonnant la solution optique. En profitant des derniers progrès en intégration de circuits, un maquettage matériel et logiciel a permis de réelles applications.

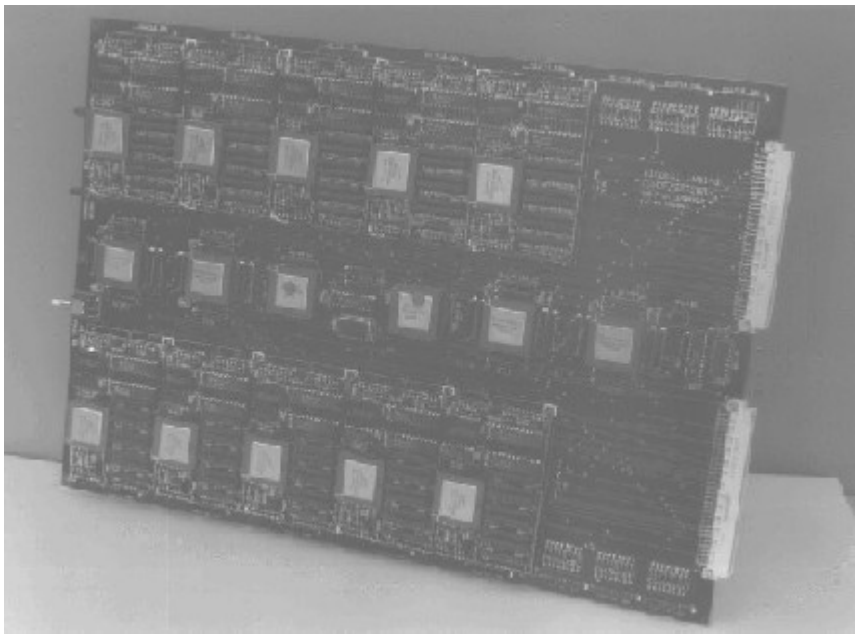
Ces travaux ont permis la spécification et la mise en oeuvre d'une machine massivement parallèle, modulaire et reconfigurable dynamiquement. La reconfiguration de l'architecture a été étudiée, dans un premier temps, essentiellement sous l'aspect fonctionnel. Il s'agit d'adapter automatiquement la topologie d'interconnexion pour mettre en oeuvre le plus efficacement possible une application (ou une étape d'une application) donnée. Deux niveaux de problèmes ont été abordés :

- au niveau architectural, ont été pris en compte les contraintes technologiques telles que la taille des commutateurs utilisables pour construire la structure d'interconnexion, le nombre de liens de communication disponibles par processeur, les moyens de commande de la structure d'interconnexion. Ces contraintes ont conduit à la définition d'une architecture modulaire mettant en œuvre deux niveaux de réseaux d'interconnexion commandables par programme.



La machine Modulor

Une maquette de la machine MODULOR, constituée de 4 modules de 20 processeurs, a été réalisée à partir de composants standard INMOS (le Transputer T800 qui dispose de 4 liens de communication et le commutateur crossbar C004 qui permet de connecter 32 voies d'entrée sur 32 voies de sortie).



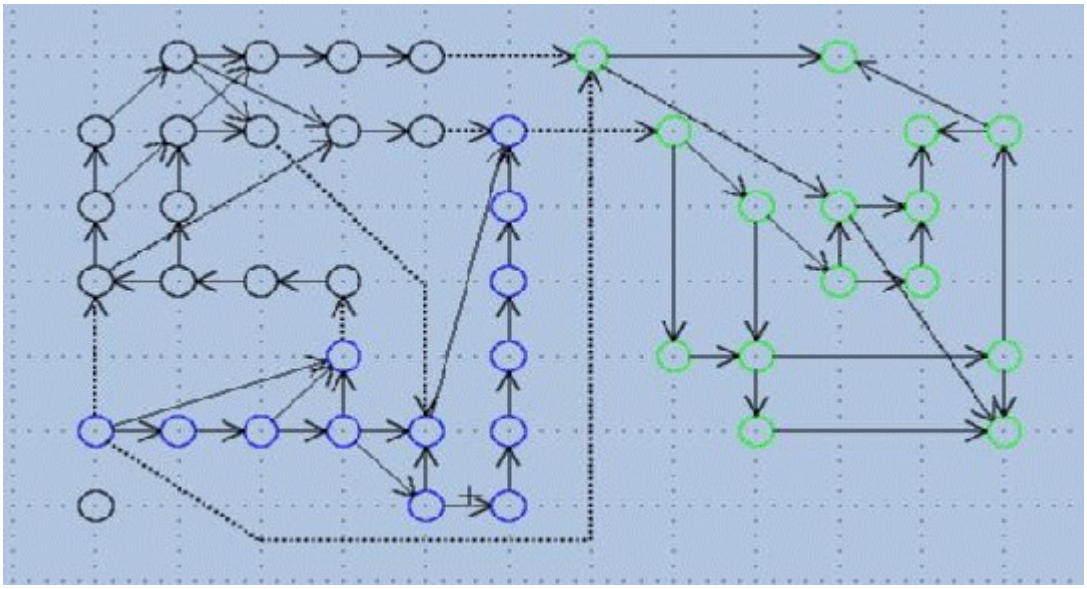
Un module de Modulor

- au niveau logiciel, le problème était d'offrir à l'utilisateur les outils permettant l'exploitation de la reconfiguration de la topologie de l'architecture d'une manière aussi transparente que possible. Une chaîne complète de développement d'applications reconfigurables permet la mise en œuvre d'applications décrites en phases successives sur la maquette réalisée. Cette chaîne comporte :

  - un interface graphique permettant la description d'une application reconfigurable sous forme de graphes de processus communiquant en mode message,

  - un outil de partitionnement (voir exemple) et placement de graphes de processus sur un réseau de processeurs reconfigurable qui conduit à la détermination de la topologie nécessaire et à la détermination des commandes des différents commutateurs,

  - un logiciel de génération de code exécutable sur la maquette réalisée à partir de processeurs standard qui assure tous les mécanismes de synchronisation nécessaires avant tout changement de la topologie d'interconnexion.



Un module de Modulor

3.2.2.2 La reconfigurabilité

La spécification de mécanismes de reconfiguration permet de tirer parti à la fois :

- d'une reconfiguration fonctionnelle qui vise essentiellement l'adaptation (la plus automatique et la plus dynamique possible) de la topologie d'interconnexion à un problème (ou un sous-problème) donné,

- et d'une reconfiguration en cas de panne qui a principalement comme objectif d'isoler un élément défaillant de l'architecture pour permettre la poursuite des traitements en cours après connexion d'un élément de secours ou basculement en mode dégradé.

De tels mécanismes sont nécessaires pour définir une architecture assurant un maximum d'efficacité de communication, dans un environnement temps réel qui nécessite de plus de pallier la défaillance de certains des éléments de la machine. Dans la plupart des cas, il s'agit principalement d'isoler un élément défaillant de l'architecture pour permettre la poursuite des traitements en cours. La tolérance aux pannes conduit ainsi à une forme de reconfiguration de la topologie d'interconnexion visant le masquage de la défaillance d'un ou plusieurs processeurs rendu possible dans la mesure où tous les processeurs sont banalisés et que des processeurs de secours sont également reliés à cette structure d'interconnexion.

Cette étude conduit à la validation de solutions pouvant être apportées au niveau logiciel pour permettre, en fonction des hypothèses architecturales, de poursuivre l'exécution d'une application après détection d'une anomalie et reconfiguration de l'architecture. Des mécanismes de détection d'anomalie et de caractérisation d'une panne ont été spécifiés en liaison avec les possibilités de détection matérielle. Des procédures de synchronisation avant toute reconfiguration matérielle de l'architecture ont également définies. Enfin des stratégies de duplication ou de réarrangement de données permettent la reprise de

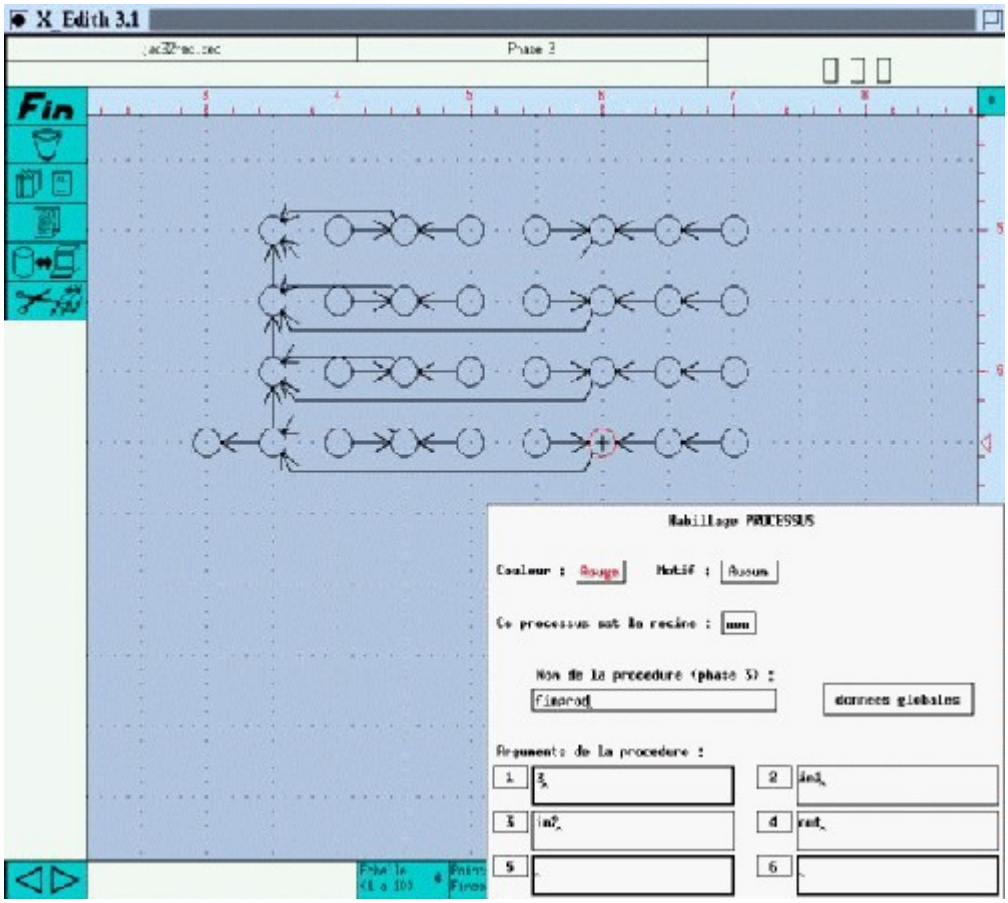
traitements interrompus sur un processeur défaillant. Ces mécanismes assurant la tolérance aux pannes des processeurs ont été intégrés aux outils existants, nécessaires pour la gestion d'une reconfiguration fonctionnelle de la topologie d'interconnexion. Ces outils reposent sur le découpage de l'application en phases successives séparées par des points de reconfiguration qui servent également de points de reprise.

En prévision des applications avioniques futures, et pour accumuler des techniques qui seront nécessaires à l'unité de recherche CVSI, des méthodes formelles ont été utilisées pour valider ces solutions. La validation des algorithmes de reconfiguration fonctionnelle a été effectuée, ainsi que la validation de quelques algorithmes utilisés pour la reconfiguration en cas de panne. Ce dernier travail de validation étant plus délicat à accomplir car devant considérer explicitement le temps. Mon travail a essentiellement consisté à expliquer les problèmes à considérer à mes collègues « valideurs », à spécifier les propriétés à démontrer et également à fournir des contre-exemples (mauvais algorithmes répartis).

3.2.2.3 Exemples d'applications

La conception d'un programme reconfigurable pour l'application Jacobi avait été faite pour le projet Milord, j'ai eu, avec la maquette du Modulor, une application opérationnelle.

L'image suivante représente la description d'une phase de cette application avec l'interface graphique réalisé X_Edith :



Exemple éditeur de graphes

Je peux comparer les résultats de cette version reconfigurable avec une version plus classique (statique) sur une grille des mêmes processeurs. Avec 32 transputers T800, pour un système de 120 inconnues, une itération dans la version statique dure 8.5 ms, dans la version reconfigurable 7.4 ms.

Un calcul de complexité permet également de comparer les deux versions programmées.

Des mesures ou des calculs permettent de savoir quand il faut changer de topologie d'interconnexion ou quand il vaut mieux programmer des routages. Un parti pris évident me permet de dire que la programmation de la version reconfigurable est plus facile.

Enfin, des fautes ont été volontairement injectées dans la version reconfigurable afin de tester les mécanismes de tolérance mis en œuvre.

Plus que le gain de performances évoqué précédemment, la versatilité de Modulor a permis d'étudier et de mettre en œuvre la distribution d'applications dans les domaines les plus variés. Ces études ont nécessité la collaboration d'autres groupes ou départements de recherche de l'ONERA, ainsi que d'autres laboratoires.

Dans le domaine des mathématiques appliquées, d'autres algorithmes de résolution de grands systèmes linéaires ont été étudiés : méthode de Gauss, réduction cyclique par exemple (collaboration ENSEEIHT).

Dans le domaine du traitement d'images (collaboration DERO), la reconfiguration a permis d'enchaîner aisément des traitements locaux (convolutions, filtrages divers) et des traitements globaux (transformées de Fourier, transpositions). Comme bien d'autres personnes l'ont fait, l'efficacité de la distribution est grande pour la génération de belles images fractales.

Le domaine de l'intelligence artificielle (au sens large) a été également prometteur :

- émulation de réseaux neuronaux,

- algorithmes d'optimisation pour la gestion du trafic routier (collaboration DERA),

- développement d'un environnement de programmation logique distribuée (collaboration ALL-MULTILOGIC, Hongrie). Un programme CS-PROLOG (Communicating Sequential PROLOG), consiste en plusieurs programmes PROLOG exécutés en parallèle (processus PROLOG) qui communiquent entre eux par l'intermédiaire de messages échangés sur des canaux de communication en utilisant une synchronisation de type rendez-vous. Les processus PROLOG exécutés simultanément peuvent effectuer des retours arrière, par contre les émissions et les réceptions de messages sont déterministes (pas de retour arrière possible). Des extensions temps réel du système CS-PROLOG ont également été étudiées.

- avec l'environnement CS-PROLOG sur Modulor, algorithmes distribués de renforcement de cohérence pour des problèmes de satisfaction de contraintes (CSP),

- modèle acteurs appelé PLASMA (Planner like Language And System Modeled on Actors) (collaboration IRIT). La machine virtuelle distribuée spécifiée pour supporter le langage d'acteurs parallèle, a été implémentée, dans un premier temps, au dessus d'un système distribué, orienté Transputer (HELIOS) sur TNODE.

- algorithmes parallèles sur les arbres de décision et raisonnement en temps contraint. Etude et application au minimax (et jeu d'échecs).

3.2.2.4 Soutiens et collaborations

Les soutiens et les collaborations (déjà partiellement citées dans le paragraphe précédent) ont été particulièrement nombreux. C'est pourquoi ce paragraphe n'apparaît que dans la présentation de Modulor.

Ce projet a été soutenu principalement par la DGA/DRET, et par des contrats :

- CEE/ESPRIT/PCA. Une machine TNODE, développée dans le cadre du projet européen SUPERNODE, a pu être obtenue et la chaîne de développement

d'applications reconfigurables a été adaptée. Ceci démontre une certaine généralité de l'architecture logicielle proposée.

- Région Midi-Pyrénées. Le soutien a principalement porté sur l'achat des composants matériels. En retour, deux versions consécutives de MODULOR ont été montrées à deux salons SITEF.

- MRT/PRC-ANM. Ceci a été l'occasion de nombreux échanges dans le cadre de la communauté universitaire travaillant dans le domaine des architecture parallèles à mémoire distribuée.

- INSA. Le laboratoire d'électronique a été consultant, puis "sous-traitant" de l'ONERA, pour la conception et la réalisation des cartes intégrées.

Rapports : [FrRoSi89], [BoFrRoSi90], [DaFrRoSi91], [FrSi92], [FrLaSiFuPaSz93], [FrMaSi93], [FrSiFuPaSzNo93]

Articles : [BoCoFrRoSi89], [FrRoSi89], [FrRoSi89b], [FrRoSi89c], [FrRoSi90], [FrRoSi91], [FrRoSi91b], [DaFrRoSi91], [FrSi91], [DaFrRoSi92], [DaFrRoSi92b], [DaFrRoSi92c], [DaFrRoSi92d], [DaFrRoSi92e], [FrSi93], [FrSi93b], [FrSi94], [PaSeSiWi95]

Stages : [Que88], [Fou89], [Rak89], [AdRe90], [Ona90], [Bai90], [NoRo91], [RiBo91], [Hen91], [Goi91], [Bas92], [Cho92], [Lan92], [Fui93], [Cho93]

Thèse (participation) : [Dav93]

3.3 Sécurité informatique des systèmes répartis

Le besoin en sécurité était bien connu, sans être totalement maîtrisé, dans le domaine de la défense pour des systèmes informatiques traditionnels. Il s'est accru quand il a fallu utiliser des systèmes distribués. Je me suis donc intéressé à ce nouveau thème de recherches : ajouter des propriétés de sécurité aux systèmes distribués.

L'ONERA était déjà en avance sur le développement de fondements théoriques de la sécurité permettant la construction efficace de systèmes sûrs. En particulier, un modèle de sécurité multi-niveau a conduit à produire des conditions suffisantes de sécurité permettant de contrôler efficacement toute forme de flux d'informations (y compris les canaux "cachés"). Ces résultats sont obtenus en définissant des concepts de dépendance et de dépendance causale, et en s'appuyant sur une représentation du temps.

Je me suis impliqué dans l'utilisation de ces travaux pour concevoir des architectures de sécurité distribuées, ce qui a donné les projets suivants, qui seront détaillés dans les

prochains paragraphes :

- la définition et la mise en œuvre d'une plate-forme d'expérimentation et de validation de la sécurité multi-niveau,
- la sécurité et les nouveaux systèmes d'exploitation répartis (en particulier le système Chorus),
- le projet SAIDA,
- le projet PEDALO.

Les deux derniers projets (au moins) proposent des applications civiles de la sécurité multi-niveau.

3.3.1 Plate-forme de sécurité multi-niveau [93-94, 96-00]

3.3.1.1 Contexte

La démarche de conception de systèmes sûrs a été appliquée d'abord pour une machine puis pour une architecture de réseau local. Cette dernière s'organise autour d'interfaces de communication au médium dotées d'un ensemble de dispositifs matériels et logiciels constituant un sous-système de sécurité local. Ce SSS local assure la mise en œuvre des deux fonctions suivantes :

- contrôle des lectures et écritures sur le médium,
- émission et réception de données de sécurité sur le médium.

La sécurité de l'ensemble du réseau repose sur un fonctionnement cohérent et synchrone des SSS locaux. Un poste centralisé de sécurité l'assure par la gestion et le pilotage global de ces interfaces à l'aide d'un protocole. Ce protocole, et par extension le réseau global, a été baptisé SMAC pour Secure Medium Access Control.

Dans le cas d'une politique de sécurité multi-niveau, le médium de sécurité sera ainsi multiplexé temporellement selon les différents niveaux de sécurité, des machines hôtes d'un niveau de sécurité donné ne seront connectées au réseau que lorsque le médium prendra le dit niveau. Les communications utilisateurs se feront en utilisant le protocole de communication standard Ethernet.

Le protocole de communication sûr a été spécifié puis formellement validé. Mon travail a consisté à le valider quantitativement, à définir une plate-forme d'expérimentation et de validation, à réaliser une maquette et des applications représentatives.

3.3.1.2 Evaluation quantitative du protocole SMAC

Le protocole SMAC a fait l'objet d'une campagne de simulation. Le simulateur à événements discrets NETSIM (MIT) a été employé et une nouvelle modélisation des protocoles effectuée. Pour simplifier, de nouveaux automates étendent ceux proposés pour les protocoles standards (Ethernet, IP, TCP), tout en autorisant une comparaison. La simulation, qui prend en compte différents niveaux de protocole, est ainsi proche des préoccupations de l'utilisateur.

Les résultats sont encourageants, des mesures sont même favorables au protocole SMAC dans des cas de forte charge. J'ai ainsi monté que la sécurité et les performances ne sont pas toujours deux termes antagonistes.

3.3.1.3 Plate-forme de démonstration et d'expérimentation

Puisque le protocole SMAC a été validé, nous avons décidé de développer une plate-forme de démonstration et d'expérimentation en sécurité multi-niveau.

Démonstration, car il faut montrer pour convaincre de la validité des idées originales et innovantes développées au plan théorique, de la faisabilité à coût raisonnable de réalisations concrètes.

Validation, car le développement de la plate-forme permettra de proposer des solutions s'appuyant sur l'existant, en minimisant les nécessaires adaptations des produits sur étagère qui ne disposent pas de mécanismes de sécurité. Cette démarche comprend également l'évaluation de la compatibilité avec des applications réparties existantes.

Validation encore, car la plate-forme permettra d'obtenir des évaluations qualitatives et quantitatives de performances, tant du point de vue de la sécurité (résistance aux attaques, force des mécanismes), que d'un point de vue général système (perte de rapidité, confort d'utilisation).

Le schéma suivant décrit la plate-forme réalisée :

- les machines sont du type PC, d'une part pour des raisons de coût, d'autre part pour le besoin d'améliorer leur sécurité (système Windows, applications existantes ou écrites dans un langage de programmation général utilisant la bibliothèque des sockets). Ce qui sera développé ne nécessitera pas de modifications si on ajoute des stations de travail. On considérera le plus souvent qu'un niveau de sécurité, par exemple haut ou bas, est affecté statiquement à chaque poste.

- le réseau est du type Ethernet, les protocoles de plus haut niveau TCP/IP,

- l'ajout principal est un composant matériel minimal, dénommé interface de communication sécurisée, entre le poste de travail (sa carte réseau) et le médium. Ce composant joue le rôle d'"interrupteur", poste et médium son physiquement

déconnectés lorsque la politique de sécurité le demande.

- une alternative est un sous-réseau de sécurité pour synchroniser ces interfaces par des dialogues entre l'intelligence minimale de ces cartes et un poste central de sécurité,

- d'autres dispositifs multi-niveau sont nécessaires, comme par exemple une machine pouvant changer dynamiquement son niveau de sécurité et pouvant jouer le rôle d'ascenseur (transfert d'information entre le bas et le haut) ou de déclassifieur (l'inverse après contrôle ou altération des informations). Poste central de sécurité et ascenseur sont des ressources protégées physiquement non manipulables par l'usager.

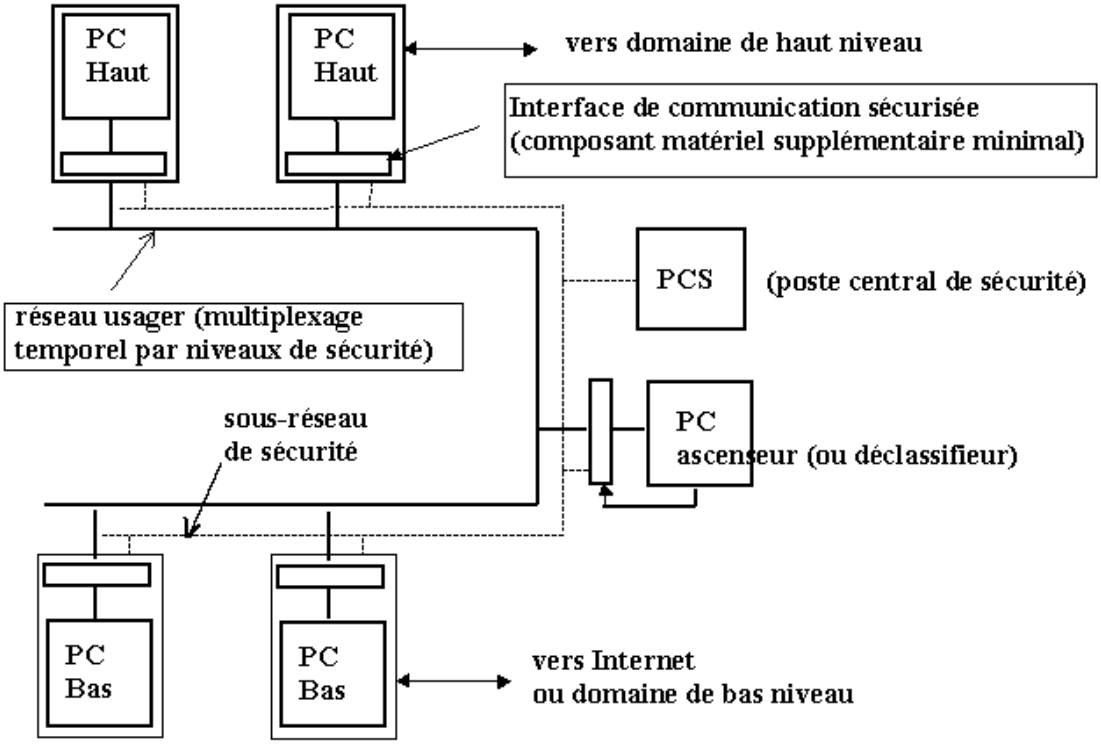


Plate-forme de sécurité multi-niveau

Quelques difficultés rencontrées ont été :

- la réalisation de l'interface de communication sécurisée, et essentiellement trouver les bons circuits pour interrompre la connexion entre carte Ethernet et médium,

- tromper pendant la déconnexion par une astuce (génération d'un signal de collision) les cartes réseau des postes de travail,

- synchroniser les interfaces.

Une étude d'industrialisation de ce dispositif a été réalisée.

3.3.1.4 Exemple d'applications

Des applications ont permis l'évaluation de cette plate-forme :

- système de messagerie multi-niveau,
- base de données dupliquées multi-niveau,
- connexion sûre à Internet. Une machine double port du niveau bas est connectée au réseau sûr et au réseau extérieur, un programme spécifique sert de relais sur la machine ascenseur.

Il a été montré que, pendant l'exécution de ces applications, une machine de bas niveau peut être laissée à un attaquant, qui dispose par exemple d'un programme "sniffer".

Rapports : [Sir94], [AuSi94], [AuBiCuEiSiYa97], [LlSi98]

Articles : [SiAu94]

Stages : [Das97], [Piq99], [Gau99], [Cor00]

3.3.2 Sécurité et nouveaux systèmes d'exploitation répartis [94-96]

3.3.2.1 Sécurité et algorithmes répartis

Une étude a été effectuée sur le lien entre les algorithmes répartis et la sécurité, en particulier la notion de dépendance causale. Parmi les algorithmes répartis de base, ce sont les algorithmes de gestion d'horloges logiques qui ont été choisis. Ces algorithmes ont de nombreuses implications, soit au niveau système, par exemple pour mettre en oeuvre une gestion de mémoire partagée distribuée, soit au niveau applicatif, par exemple pour une authentification distribuée. L'application à la simulation distribuée n'était pas alors à l'ordre du jour.

3.3.2.2 Sécurité et systèmes d'exploitation

Le niveau système d'exploitation peut être le niveau où ajouter des mécanismes de sécurité. Cependant ajouter des mécanismes forts à un système existant comme Unix est une tâche mal aisée et du ressort des constructeurs (ce raisonnement date d'avant le succès des logiciels libres).

Aussi j'ai préféré suivre l'évolution des nouveaux systèmes d'exploitation répartis :

- adaptés aux architectures réparties que nous devons rendre sûres,

- avec des noyaux de systèmes fraîchement repensés auxquels il est plus facile d'ajouter des fonctions de sécurité.

3.3.2.3 Etude des micro-noyaux

D'une manière caricaturale, l'objet principal dans un système comme Unix est le fichier (disques, périphériques, mémoire centrale sont manipulables par l'intermédiaire de descripteurs de fichiers). Ceci est bien adapté pour le système d'exploitation des machines isolées (éventuellement de gros serveurs). Puis les architectures parallèles et distribuées sont apparues et les extensions nécessaires ont été ajoutées au noyau de base avec deux conséquences : taille démesurée du noyau monolithique, moindre optimisation des communications.

Les nouveaux systèmes d'exploitation répartis (Amoeba, Chorus, Mach, etc.) sont une réponse à ces deux problèmes. Ils utilisent la technologie dite des micro-noyaux : le cœur du système est repensé pour qu'une abstraction de base soit la communication, il doit être minimal pour être facilement évaluable et portable. Sur ce cœur seront reconstruits les services des systèmes précédents (facilités d'Unix ou de Windows).

Ceci peut-être illustré (voir la figure) en énumérant quelques objets de base du système Chorus :

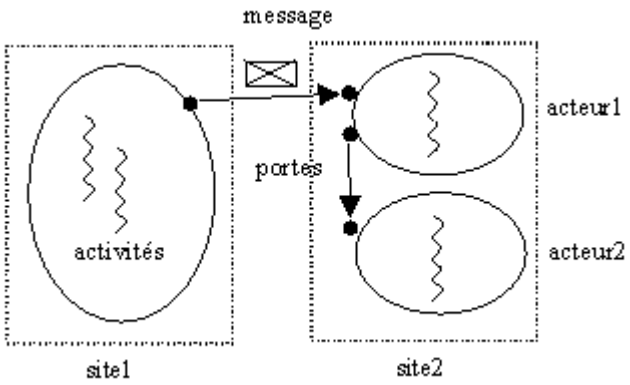
- un acteur constitue un environnement d'exécution et regroupe des objets comme les portes et les activités,

- une activité est l'unité d'exécution dans le système. Toutes les activités d'un acteur partagent ses ressources et sont ordonnancées individuellement par le noyau.

- un message est l'unité de communication entre deux activités, il n'est pas typé. Un message est adressé à une porte.

- une porte est une adresse logique avec une file d'attente de messages. Elle est attachée à un acteur mais peut migrer en cours d'application. Les communications sont soit asynchrones (mode IPC) soit synchrones (mode RPC). La répartition des acteurs en sites est transparente.

- un groupe de porte permet d'attribuer un nom logique à un ensemble de portes.



Quelques entités du système Chorus

3.3.2.4 Sécurité du système Chorus

Bruno d'Ausbourg et Christel Calas ont étudié comment adapter le micro-noyau Mach à la plate-forme multi-niveau présentée précédemment. J'ai également envisagé la proposition Trusted Mach de la société américaine TIS. Mon travail a plutôt concerné des propositions d'extensions de sécurité pour le système Chorus (proposé par une société française, avec un soutien européen et porté sur les architectures à base de Transputers, appartenant désormais à Sun Microsystems).

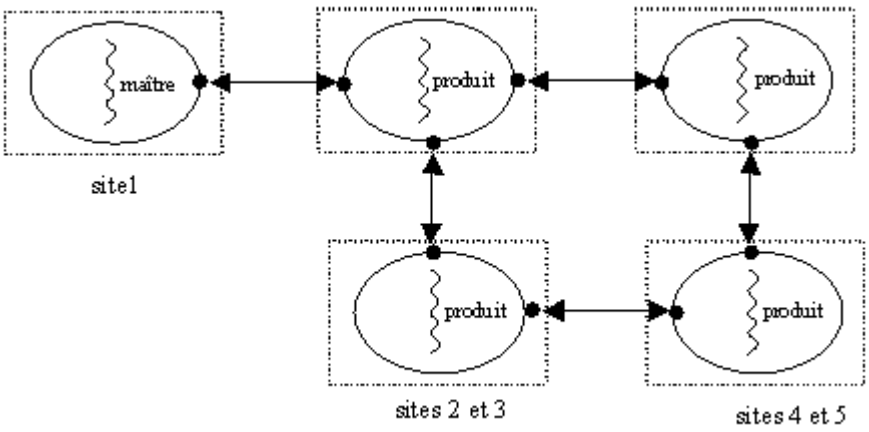
Une approche pragmatique a été suivie. Système d'exploitation et plate-forme matérielle ne sont pas modifiés. En accord avec la philosophie des micro-noyaux, j'ai ajouté un système de filtre et de nouveaux serveurs de sécurité ont été mis en œuvre. Le filtre intègre des contrôles d'accès multi-niveau pour les communications de base (accès aux portes).

La plate-forme proposée permet également de développer avec plus de facilités des applications distribuées. Ces applications sont décrites en terme de phases (se référer au projet Modulor), à partir des besoins de communication de chaque phase la mise en place des activités avec les portes nécessaires est automatiquement effectuée. Des mécanismes de sûreté de fonctionnement analogues à ceux développés dans le projet Modulor sont compatibles.

3.3.2.5 Exemple d'applications

Des applications multi-niveau ont été développées permettant de valider cette plate-forme logicielle. Il s'agit d'un calcul distribué et de son monitoring, et du squelette d'une base de données dupliquées.

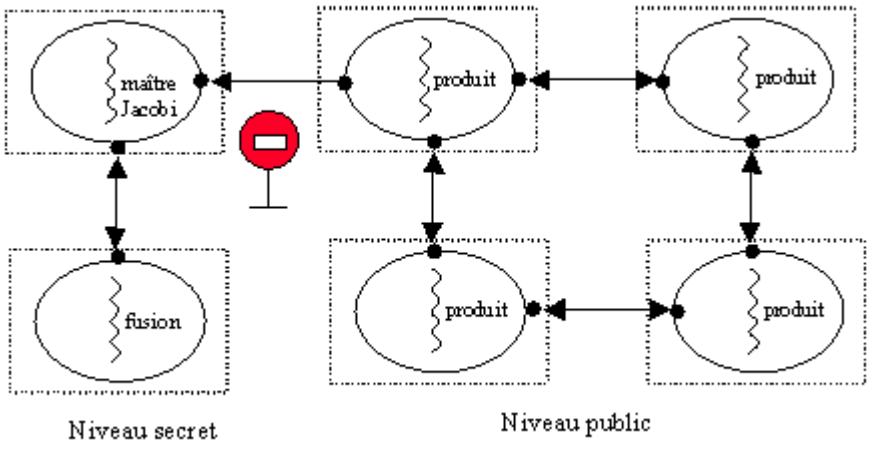
L'application de calcul distribué correspond à l'application Jacobi. La version sans sécurité s'inspire fortement de la mise en œuvre de Jacobi spécifiée pour Milord puis testée sur Modulor. Le schéma suivant représente le graphe d'activités.



Version simple de Jacobi

Chaque acteur comporte une seule activité et peut être alloué sur le même site ou sur des sites différents. Cette activité correspond principalement au produit d'une matrice et d'un vecteur. Des portes de communication mettent en œuvre une grille de communication et masquent les protocoles de bas niveau (et les liens de communications physiques dans le cas de Modulor). Enfin, le site 1 est le site maître qui collecte les résultats.

Dans ce programme de test, la version avec sécurité est obtenue en décrétant les acteurs récupérant les résultats numériques comme du niveau secret. Cela pourrait être typiquement une application de renseignement, avec diverses collectes d'informations de niveau public, et une synthèse d'informations du niveau secret (synthèse obtenue par fusion de données publiques et secrètes).



Version avec sécurité

Rapports : [AuCaSi94], [Sir95], [Sir95b], [SiDeFa96]

Articles : [Sir95]

Stages : [Mir95], [Fab95], [Fab96]

3.3.3 Le projet SAIDA [97-00]

3.3.3.1 Objectifs

Le projet SAIDA (Security Assurance In Distributed Application) a été mené en collaboration avec la DERA Malvern (nos homologues anglais). Il s'agissait de proposer et d'expérimenter divers moyens permettant d'assurer que des applications réparties, construites à base de composants sur étagères (COTS), respectent des propriétés de confidentialité et d'intégrité.

3.3.3.2 Approche

Les contrôles de sécurité proposés s'opèrent à trois niveaux distincts et font appel à des techniques spécifiques :

- les composants de l'application sont isolés dans des capsules logicielles dont le code est contrôlé par analyse des flux d'information,
- le code résultant est exécuté sur des architectures adaptées à la sécurité (logicielles, systèmes, voire matérielles),
- enfin, les problèmes spécifiques posés par la distribution sont résolus par la mise en œuvre de protocoles de sécurité vérifiés et prouvés.

3.3.3.3 Application

Au cours des trois années, une étude de cas sérieuse a été utilisée : un support de simulation distribuée implantant le protocole standard HLA.

Ici se pose un problème de présentation insoluble car la simulation distribuée fait l'objet de mon dernier thème de recherches, je reviendrai plus en détail sur l'architecture de sécurité définie quand j'évoquerai ces travaux.

A la fin de l'étude, une architecture est disponible pour faire de la simulation distribuée qui, d'une part répond à un standard, d'autre part est la seule qui intègre des fonctions de sécurité.

Cette architecture est opérationnelle, et j'ai étudié l'impact des différentes fonctions de sécurité (le contrôle d'accès, le cryptage, les signatures digitales) sur les performances temps réel de HLA.

Cette étude s'est terminée par une synthèse méthodologique et l'établissement d'une liste de recommandations. Il est possible de considérer l'étude suivante (PEDALO) comme une application de cette méthode.

Rapports : [BiCaHaSi97], [BiCaSi99], [BiCaSiWi99], [BiCaSi00]

Articles : [Sir98a], [Sir98b], [BiCaSiZa98], [Sir99], [BiSi99], [BiSi00]

Stages :[Har98]

3.3.4 Le projet PEDALO [98-00]

3.3.4.1 Objectifs

Le projet PEDALO (Protection des Agents Logiciels) a été mené pour France-Télécom, Caen dans le cadre d'une CTI (Consultation Thématique Informelle). Le but était de définir des fonctions de protection dans les systèmes à agents fondés sur CORBA.

Pour illustrer ce travail, une étude de cas a été proposée par France-Télécom. Pour rendre sûre cette étude de cas, il a fallu d'abord l'analyser, puis choisie et définir une politique de sécurité basée sur les rôles, puis définir une architecture de sécurité et implanter cette architecture pour des démonstrations.

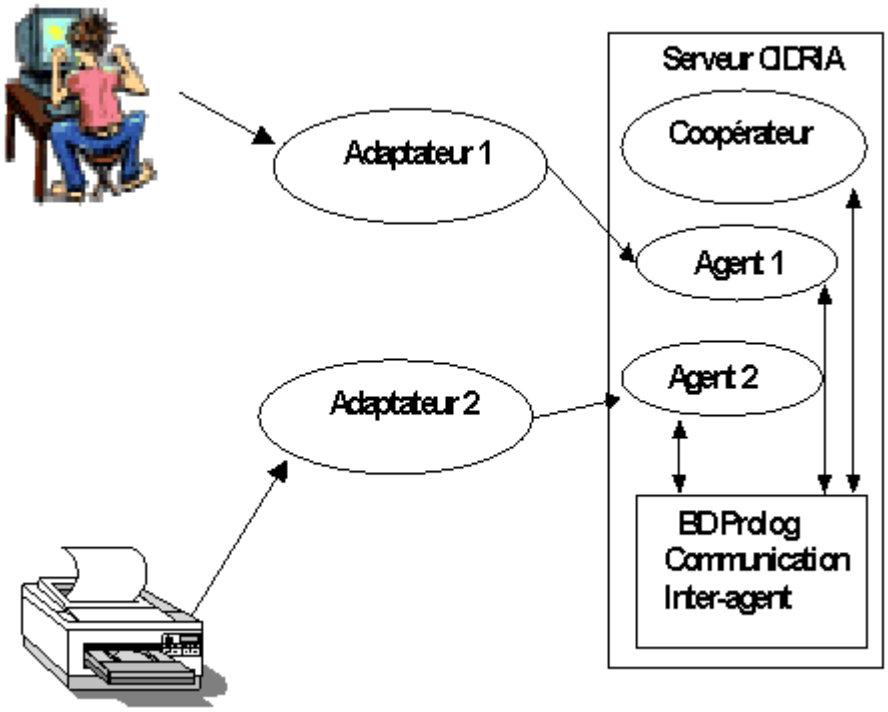
Dans une seconde phase, la même approche a été appliquée à une extension conséquente de l'étude de cas qui a été proposée, extension qui permet de connecter plusieurs serveurs et qui permet donc de gérer un grand nombre de clients.

A partir de cette étude de cas, des solutions théoriques puis pratiques ont été proposées qui pourront être réutilisées pour d'autres applications réparties.

3.3.4.2 Application

L'application choisie est une application de travail collaboratif distribué, dénommée CIDRIA. Cette application fait appel à un serveur CORBA qui assure l'interface entre les ressources et les utilisateurs. Ces ressources et ces utilisateurs sont représentés dans le système par des clients CORBA du serveur, clients appelés adaptateurs.

L'architecture de base est mieux expliquée par un schéma.



Architecture de base de CIDRIA

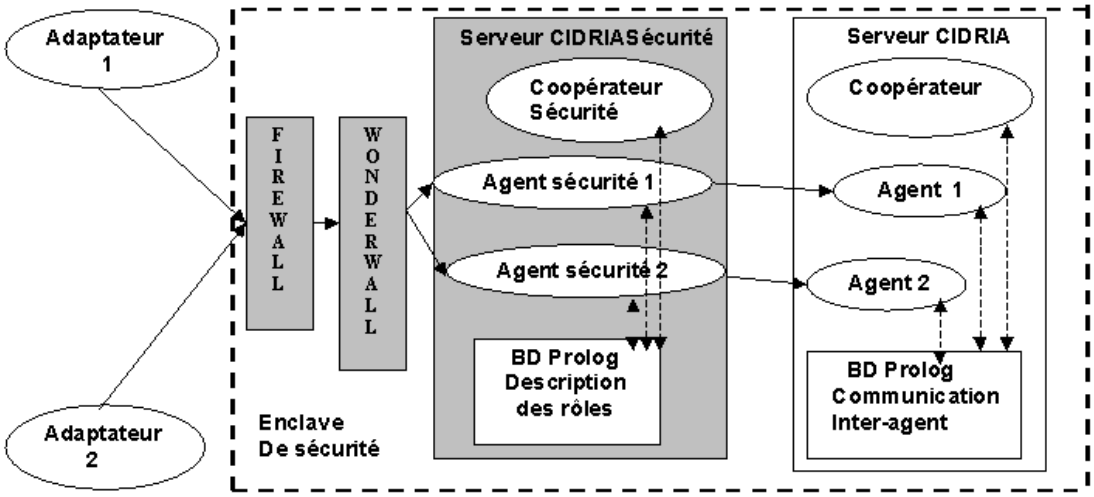
Dans le serveur CIDRIA, les ressources et leurs adaptateurs sont à leur tour représentés par des objets dénommés agents. Un objet particulier, dénommé coopérateur, gère l'activation des agents et cette connexion. Des méthodes types pour un agent représentant une ressource sont les méthodes add_competence, reply_request, une méthode type pour un agent représentant un utilisateur est la méthode add_request. Les communications entre les agents se font par l'intermédiaire d'une base de données commune Prolog.

3.3.4.3 Sécurité d'un serveur CIDRIA

Une politique de sécurité basée sur les rôles a été choisie. Succinctement, des rôles de base ont été définis comme ressource ou utilisateur, qui permettent soit d'ajouter des requêtes soit d'y répondre. Des rôles dérivés permettent de décrire les types de requêtes autorisées, par exemple, un utilisateur sera autorisé à imprimer. Enfin un contexte peut être pris en compte, comme l'appartenance de l'utilisateur à une certaine entreprise.

Ces rôles alimenteront un mécanisme de contrôle d'accès à mettre en place. Ce contrôle d'accès sera plus fin d'une part, leur mise en œuvre plus portable d'autre part, que certains mécanismes de sécurité définis pour CORBA, où le contrôle porte sur le nom des méthodes (add_competence) mais pas les paramètres utilisés (imprimer). Une mise en œuvre transparente a été recherchée, qui ne nécessite pas de modifications pour les clients et serveurs existants.

L'architecture de sécurité suivante a été définie puis testée :



Architecture de sécurité de CIDRIA

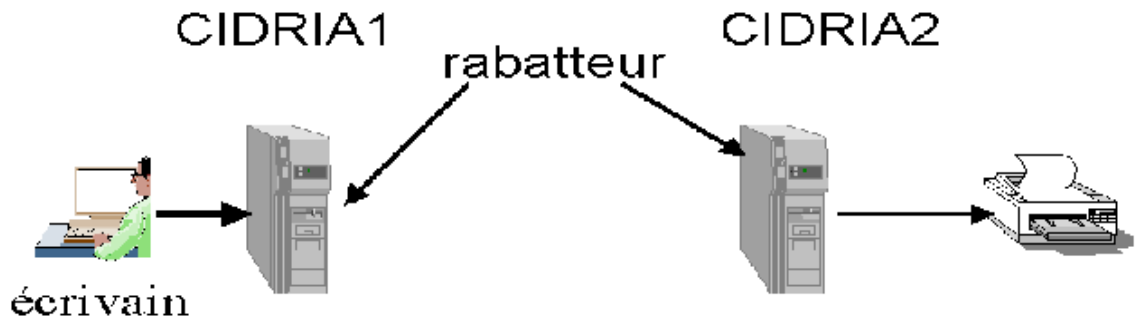
Un serveur de sécurité, utilisant la même technologie CORBA, est inséré entre les clients et le serveur existants. Des objets, utilisant les mêmes noms de méthodes, réalisent les contrôles d'accès nécessaires puis assurent les relais. Ces contrôles sont décrits par l'administrateur de sécurité dans une autre base de données Prolog.

Un autre point dur est d'assurer l'incontournabilité de ces contrôles. Serveur CIDRIA et serveur de sécurité sont protégés dans une enclave de sécurité. L'accès à cette enclave se fait par l'intermédiaire d'un garde-barrière. Ce qui est le plus simple est de rediriger les requêtes vers un processus interne sur un port TCP connu (les requêtes CORBA suivent le protocole standard IIOP). A cet endroit, un composant de sécurité du commerce, Wonderwall de IONA Technologies, redirige les requêtes vers les bons objets du serveur de sécurité. Ce composant est apte à réaliser les contrôles sur les noms de méthodes à invoquer, cette possibilité est également utilisée pour des raisons de performance.

3.3.4.4 Interconnexion sûre de systèmes distribués

Pour traiter un grand nombre de clients d'une part, pour tenir compte des serveurs existants dans différentes entreprises d'autre part, il est naturel de se poser la question de la coopération entre plusieurs serveurs CIDRIA.

Une solution pour cette interconnexion a été proposée, qui est basée sur l'introduction de clients dénommés rabatteurs, qui sont connectés simultanément à plusieurs serveurs. Dans l'exemple ci-dessous, le rabatteur joue le rôle d'une ressource imprimante pour CIDRIA1, cette ressource n'existe pas dans le domaine de CIDRIA1 mais le rabatteur transmet la requête vers le serveur CIDRIA2 en jouant le rôle d'un utilisateur.



Interconnexion de deux CIDRIA

Puis le problème de la sécurité de cette interconnexion a été abordé. La proposition est simplement d'interconnecter plusieurs CIDRIA sûrs, c'est-à-dire de réutiliser et de dupliquer l'architecture de sécurité définie.

Le problème est de ne pas ajouter de nouvelles autorisations dans le système global obtenu (ce qui était interdit avant interconnexion doit le rester), inversement il ne faut pas ajouter de nouvelles contraintes (ce qui était possible doit le rester). Pour cela, les rôles de sécurité sont automatiquement générés que prendra le rabatteur à partir d'un contrat d'interconnexion établi en accord avec les différents responsables de sécurité. Ces rôles seront minimaux et ne permettront le transfert que d'un certain type de requête.

3.3.4.5 Conclusion

Malheureusement la sécurité n'est pas souvent prise en compte dans la conception des systèmes et ceci devient un frein à leur déploiement. Au cours de ce projet, une méthodologie a été appliquée pour ajouter de la sécurité à des systèmes répartis existants.

Bien que appliqué à une application de travail coopératif, la proposition d'une architecture de sécurité et d'une politique à base de rôles concerne généralement la sécurité des systèmes à objets.

Je pense également que la solution proposée pour l'interconnexion de politiques de sécurité pourra être réutilisée. Le besoin est identifié pour l'interconnexion sûre de simulations distribuées (voir chapitre suivant).

Rapports : [BiSi98], [BiSi99], [BiRaSi00]

Articles : [BiSi00], [BiRaSi00]

Stages : [Pes98], [Lou99], [Rau00]

3.4 Simulation distribuée [96-…]

Des besoins en nouvelles techniques de simulation ont été identifiés, en particulier pour les applications de la défense : étude de systèmes d'information complexes, études technico-opérationnelles, simulations distribuées interactives pour l'entraînement, etc. Ces besoins se retrouvent de la même manière dans le monde civil.

La solution est recherchée dans la définition de nouvelles techniques de simulation distribuée.

Etant d'une part spécialiste des systèmes distribués, ayant été plusieurs fois réalisateur de simulation, ce sujet est devenu mon thème de recherches actuel.

Je présenterai successivement :

- le standard HLA,
- la réalisation d'une infrastructure d'exécution pour la simulation distribuée (CERTI),
- des exemples d'applications,
- le problème de la sécurité des simulations distribuées,
- le problème des performances des simulations distribuées.

3.4.1 Le standard HLA

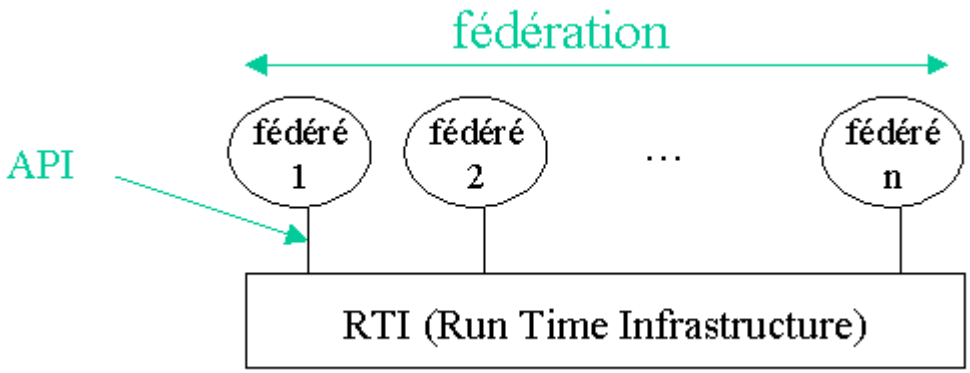
3.4.1.1 Simulation distribuée à événements discrets

La modélisation et simulation à événements discrets est une aide depuis la conception des systèmes complexes (systèmes de défense, systèmes aéronautiques) jusqu'à leur utilisation. Par la suite, j'emploierai le terme de simulation à la place du couple modélisation et simulation. Cette technique, abondamment utilisée, se tourne maintenant vers la distribution car cette distribution permet de multiplier les performance de calcul, elle permet également de rendre modulaire la simulation soit selon les objets simulés, soit selon les métiers ou entreprises des utilisateurs. L'argument clé est qu'il sera plus facile de faire interopérer des modules existants, de les réutiliser, au sein d'une architecture distribuée qu'au sein d'un programme monolithique. Mais, pour cela, il faut également disposer de standards.

3.4.1.2 Le nouveau standard

Le sigle HLA signifie High Level Architecture, c'est un standard d'architecture défini pour la simulation distribuée à événements discrets. Ce standard a d'abord été un standard de fait dans le monde de la défense américaine (l'ONERA a étrangement participé à sa naissance), puis il est devenu un standard civil (IEEE, OMG).

Dans ce standard, le terme fédéré désigne un simulateur individuel, le terme fédération désigne une simulation distribuée.



Architecture HLA

Ce standard spécifie :

- un ensemble de règles qui précisent les rôles et devoirs des fédérés et des fédérations,
- une description objet et papier des objets manipulés (modélisés puis simulés) par les fédérés et les fédérations,
- l'interface de programmation (API) utilisable par les fédérés lors de leur dialogue avec le RTI ou Run Time Infrastructure.

Le RTI est l'élément critique de l'architecture. Les familles de service les plus importantes concernent :

- la gestion de la fédération ; créer, joindre, quitter, détruire, ...
- la gestion des déclarations : un fédéré peut publier une classe d'objets et un autre fédéré peut souscrire à cette classe,
- la gestion des objets : un fédéré peut mettre à jour des attributs d'un objet (après avoir publié la classe et créé une instance), les fédérés qui en ont fait la demande recevront les avis de mise à jour (reflect),

- la gestion du temps : cette famille distingue HLA d'autres "middlewares" ou systèmes répartis. Chaque fédéré gère une horloge locale et confie la coordination de cette horloge avec les horloges des autres fédérés au RTI (time advance request). Chaque mise à jour d'objet peut être estampillée, et le RTI assure, dans lemode conservatif, que la livraison des mises à jour se fera en accord avec l'horloge du fédéré et par dates croissantes.

3.4.2 CERTI

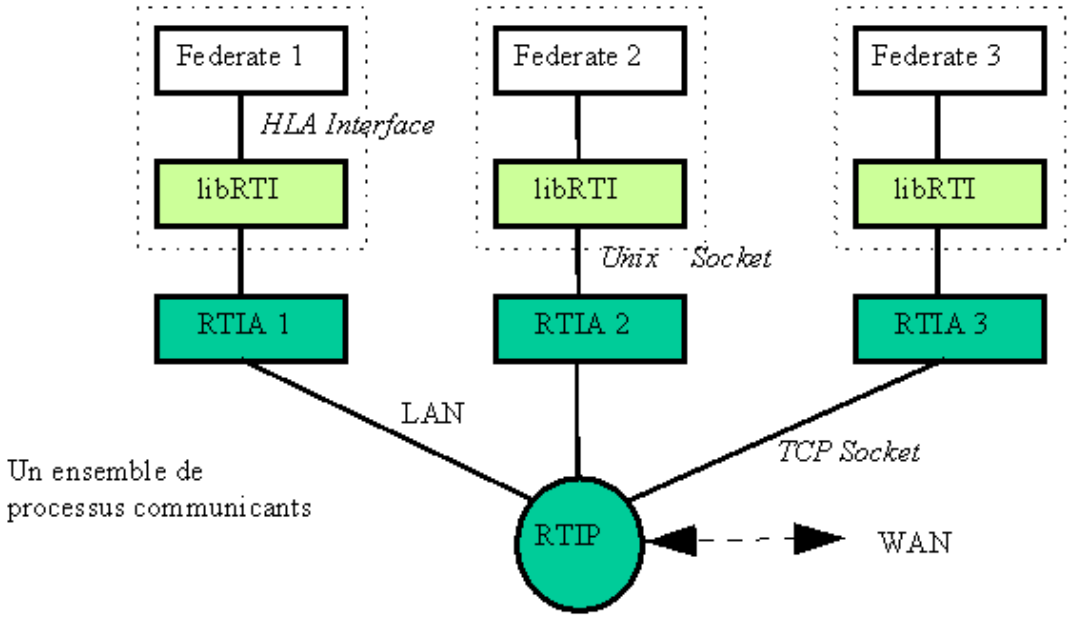
3.4.2.1 Objectifs

Connaissant les spécifications de HLA, il a été décidé de se munir d'une plate-forme d'expérimentation de cette architecture. En particulier, connaissant les spécifications de l'API, il a été décidé de concevoir et d'implanter un prototype de RTI au CERT, de nom CERTI. Les objectifs étaient multiples :

- mieux comprendre l'architecture HLA,
- étudier en détail certaines techniques de base et initialiser les recherches nécessaires pour avoir une mise en œuvre efficace,
- fournir une version utilisable pour les concepteurs de simulation,
- mieux comprendre l'impact de l'utilisation de HLA sur les simulations.

3.4.2.2 Architecture

La version que j'ai développée depuis 1997 est fondée sur une architecture de processus communicants (voir figure), développés en C++ et utilisant le protocole de communication standard TCP/IP. Elle fournit les principaux services de HLA (dont la gestion du temps).



Architecture de CERTI

Cette architecture s'est révélée efficace :

- la bibliothèque libRTI ne fait que peu de traitement, elle met en forme un message contenant le type d'appel et les paramètres et l'envoie, elle fait le traitement inverse lorsqu'il faut subir les "callbacks" du RTI,

- le processus RTIA (Ambassadeur) est un processus très réactif, qui gère les communications soit avec le fédéré, soit avec le processus RTIP, ainsi que des filesd'attente. Il est apte à répondre directement au fédéré pour certains services et il exécute une partie des algorithmes répartis (gestion du temps notamment),

- le processus RTIP (Passerelle) sert :

    - d'une part de relais pour les communications entre les ambassadeurs,

    - d'autre part de point centralisateur pour la gestion des listes de publication et d'abonnement des objets. Plusieurs processus RTIP sont prévus pour factoriser les communications et optimiser les communications à longue distance, mais cette optimisation n'a pas encore été testée.

3.4.2.3 Avancement

CERTI a été un des premiers RTI réalisés au monde et cette plate-forme a répondu aux objectifs initiaux : des applications HLA ont pu être développées sans attendre l'accès à la version du DMSO, accès pour l'instant autorisé, et de nouvelles recherches ont été

entreprises. Ceci est évoqué dans les pages suivantes.

Des versions successives ont été développées pour :

- optimiser les communications (travail sur le format des messages et une utilisation adéquate de TCP),
- permettre des communications longue distance,
- ajouter de la sécurité,
- mettre en œuvre le service de gestion de la propriété,
- mettre en œuvre le service de gestion de région de publication et d'abonnement pour réduire les communications (en cours),
- suivre les évolutions de la norme (actuellement version 1.3 des spécifications).

CERTI a pu être diffusé à quelques partenaires (SUPAERO, LAAS, IRIT, CENA) et la question du moment est de le transformer en un logiciel libre.

3.4.3 Exemple d'applications

Faute de temps, je n'ai pas mis en œuvre l'application de Jacobi sur la plate-forme HLA. Ce travail reste à faire si on veut évaluer la pertinence de cet environnement pour du calcul distribué.

La spécification de cette mise en œuvre serait simple : la matrice initiale serait comme précédemment découpée en sous-matrices allouées à des fédérés dont la tâche principale serait d'effectuer le produit d'une sous-matrice par un sous-vecteur.

Chaque fédéré s'abonnerait à la classe sous-vecteur pour recevoir le bon opérande depuis un fédéré maître et publierait une classe sous-vecteur résultat.

La synchronisation des traitements utiliserait le temps coordonné ou plus simplement suivrait le flot de données.

3.4.3.1 Application test : billard

Cette application est représentative d'une simulation de déplacement de mobiles qui interagissent et que l'on observe.

Chaque fédéré simule le comportement d'une boule de billard et détermine sa position à des instants successifs en tenant compte des rebonds avec les bords et des collisions avec d'autres boules, collisions gérées à l'aide des interactions de HLA. Chaque fédéré publie sa

position et s'abonne à la position des autres (en fait la classe). A chaque pas de simulation, chaque fédéré communique sa nouvelle position au RTI et reçoit les avis de position des autres fédérés. Ces informations servent en particulier à gérer un écran graphique où l'on voit la connaissance de la situation globale de chaque fédéré.

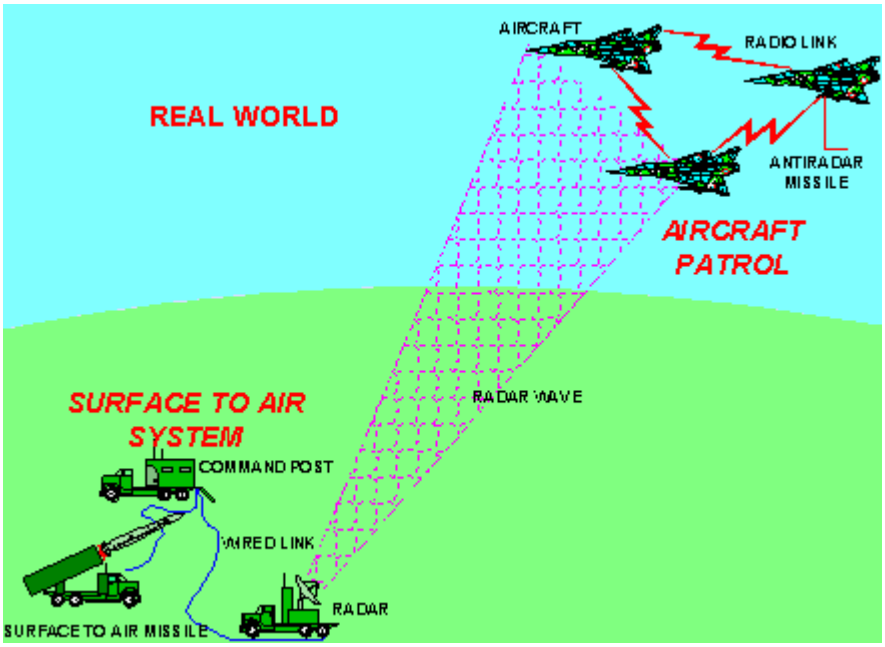
Bien que simple, cette application présente l'avantage de mettre en œuvre la majorité des mécanismes de base de HLA et constitue donc un excellent cas d'école (utilisé par exemple pour la SAE pour le stage modélisation et simulation de systèmes complexes dont je suis le co-animateur). Sont testés ainsi :

- la constitution d'une fédération,
- la synchronisation initiale des fédérés,
- les mécanismes de publication et abonnement et les mécanismes d'échange de données ou d'interaction,
- la boucle principale de toute simulation HLA,
- la gestion du temps coordonné (chaque fédéré avance au même rythme, ceci est en particulier vérifiable par :)
- la collecte d'information et la visualisation graphique d'une situation.

Cette application sert également pour les tests de performance (benchmark) de HLA. Ce test est exigeant car la charge de calcul de chaque fédéré est faible et la vitesse de simulation est déterminée par l'accomplissement des services de HLA.

3.4.3.2 Simulation d'une défense anti-aérienne

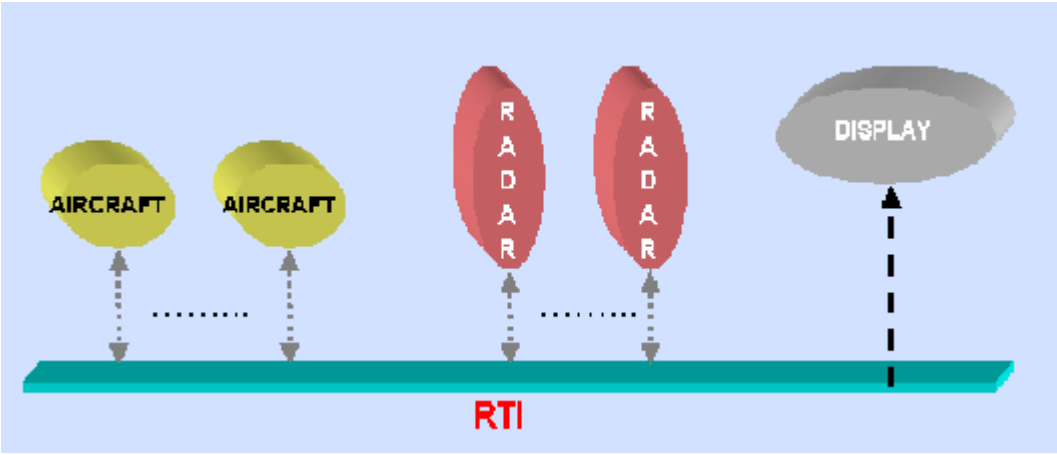
Il s'agit d'une application de défense anti-aérienne qui simule l'attaque de postes radar au sol par une patrouille d'avions armés de missiles air-sol, le comportement de chaque avion étant simulé individuellement. Les postes radar sont défendus par un réseau de missiles antiaériens. L'objectif de la mission est de détruire les postes radar, alors que la défense doit conserver intacts les postes radar en interceptant tous les objets détectés non identifiés. Cette application est inspirée de celle utilisée par le CAD pour illustrer la méthode ESCADRE pour les simulations technico-opérationnelles. Elle est illustrée par la figure suivante :



Défense anti-aérienne

La simulation d'un engagement entre avions et postes radar au sol est effectuée par l'exécution d'une fédération HLA comportant plusieurs fédérés selon le nombre d'unités engagées.

Chaque avion et chaque radar au sol est simulé par un fédéré différent. Les missiles sont au contraire des objets manipulés par les avions et les radars, qui les guident jusqu'à leur cible. Le modèle de comportement d'un missile est donc inclus dans le fédéré qui le guide jusqu'à sa cible. Un script permet de lancer autant de fédérés Avion et Radar que l'on souhaite. Un dernier fédéré, le fédéré Moniteur assure la visualisation de l'engagement entre les avions et les radars.



Fédération correspondante

Des interactions HLA sont échangées entre les fédérés Avion et Radar pour imiter ce qui se passe physiquement, c'est à dire des émissions d'onde et des réflexions selon la position de la cible.

J'ai donc mis en œuvre sur CERTI cette application plus réaliste, avec des modèles d'avions, de radars et de missiles qui peuvent être encore raffinés. Mais d'un point de vue technique de programmation HLA, il ne s'agit qu'une extension de l'application billard évoquée précédemment.

3.4.3.3 Multi-résolution

J'ai étudié (avec Martin Adelantado) des mécanismes permettant de conduire des simulations à résolution variable. Une approche expérimentale a été préférée à une approche théorique, compte tenu de la diversité des solutions envisageables selon le cadre applicatif retenu. Le cadre expérimental était offert par l'application de défense anti-aérienne.

Le principe général de la multi-résolution consiste à simuler des instances d'objets "Escadrille" composées d'un ensemble de n avions. Chaque escadrille est simulée par un fédéré en s'appuyant sur un modèle de comportement proche de celui utilisé pour les fédérés "Avion". La classe Escadrille possède les mêmes attributs que la classe Avion, auxquels on rajoute un certain nombre d'attributs décrivant l'organisation des avions au sein de l'Escadrille, et l'état courant de l'entité. Ces attributs seront utilisés pour désagréger l'Escadrille en un ensemble de composantes Avion. La position de l'Escadrille pourra être le centre de gravité des avions, ou bien tout simplement, la position de l'avion leader.

Identifier les situations de déclenchement des transitions entre état agrégé et état désagrégé, constitue un des points les plus délicats de cette application. Une stratégie par sphère d'influence a été choisie, permettant d'opérer ces transitions de manière automatique selon la situation de l'engagement. Ainsi, tant que l'escadrille n'est pas détectée par un radar, les avions sont agrégés sous forme de patrouille.

Lorsque les avions ne sont plus en situation d'engagement, on revient à un modèle agrégé de plus faible résolution. Le critère de sortie de la situation d'engagement peut être l'instant où tous les avions appartenant à une même patrouille sont hors de portée des postes radar au sol.

La possibilité de gérer la multi-résolution avec HLA a été validée, différentes alternatives de mise en œuvre ont été étudiées, puis la spécification d'un nouveau service de HLA a été proposée facilitant cette mise en œuvre. Il ne s'agit pas d'une entorse aux spécifications de HLA puisque ce nouveau service est construit sur les services de base existants.

3.4.4 Sécurité des simulations distribuées

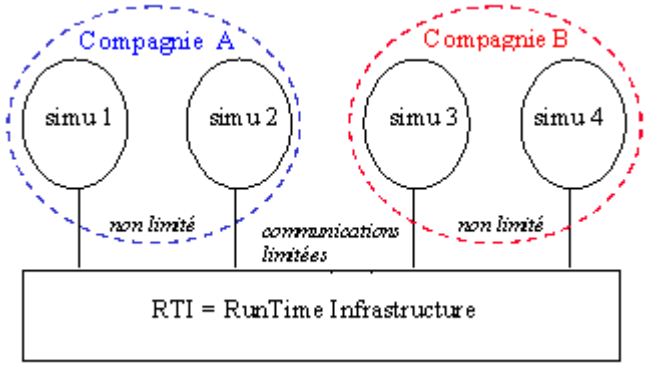
3.4.4.1 Maîtrise d'un RTI

Un premier aspect dans la prise en compte de la sécurité des simulations distribuées est la maîtrise de l'infrastructure d'exécution (RTI). J'en ai écrit le code et je connais les canaux de communication à mettre en œuvre entre les différents composants. Ceci permet :

- de réduire les possibilités de chevaux de Troie (je n'en ai pas introduit dans le code et le code source demeure accessible),
- de faire des expérimentations de simulations distribuées longue distance (Internet), expérimentations compatibles avec l'infrastructure de sécurité existante, qui intègre en particulier des gardes-barrières. J'ai obtenu l'accord pour l'installation des relais de connexion TCP qu'il fallait (à une machine déterminée et un port déterminé depuis une machine déterminée dans une session de travail définie en mode usager simple). Des démonstrations ont été réalisées, d'une part entre SUPAERO et le CERT, d'autre part, dans le cadre de FéRIA, entre l'IRIT, le LAAS et le CERT.

3.4.4.2 Sécurité d'une simulation inter-entreprises

Ces travaux ont été menés dans le cadre du projet SAIDA (Security Assurance In Distributed Application) déjà évoqué. Il s'agit en particulier d'apporter une réponse aux problèmes d'une simulation inter-intreprises, où un échange illimité d'information doit avoir lieu via le RTI entre les simulateurs de la même entreprise, et où des restrictions doivent exister entre les simulateurs de deux entreprises différentes pour conserver la confidentialité de leur savoir-faire. C'est une application des études sur la sécurité multi-niveau.



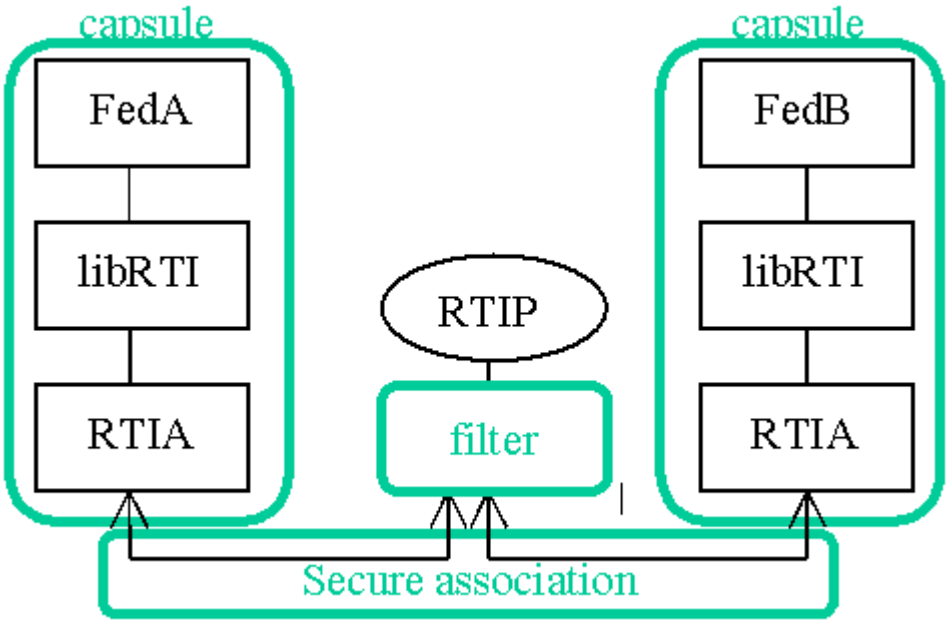
Simulation distribuée inter-entreprises

L'architecture de sécurité, que j'ai définie avec Pierre Bieber, est présentée sur la figure suivante. On y retrouve diverses fonctions de sécurité :

- association de sécurité : il s'agit de protéger les communications entre processus RTIA et RTIP. La mise en œuvre utilise l'interface standard GSS-API, qui cache l'utilisation des différents protocoles cryptographiques possibles.

- contrôle d'accès : il permet de contrôler l'accès aux classes d'objets entre les différents fédérés. Ce contrôle n'est nécessaire que dans les phases de publication et d'abonnement, le processus RTIP ne répercutant les mises à jour des objets que vers les fédérés qui ont pu s'abonner avec succès. Le coût de la sécurité est par conséquent réduit. La mise en œuvre implique l'ajout de nouvelles fonctions dans le code du RTIP ou un filtre.

- capsule de sécurité : il s'agit de contraindre le fédéré, pour communiquer, à n'utiliser que les services du RTI et à les utiliser correctement. La mise en œuvre repose sur des outils d'analyse de code, ou, plus simplement, sur la confiance accordée par chaque entreprise dans la partie de la simulation qu'elle a développée.



Architecture de sécurité pour HLA

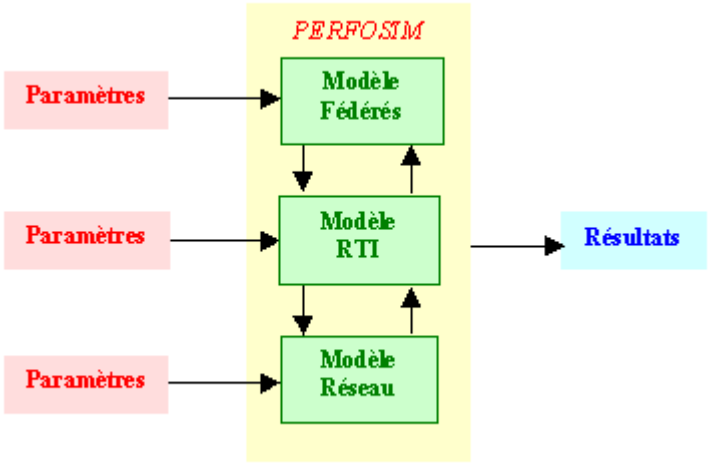
3.4.5 Performance des simulations distribuées

Disposer du support technique indispensable à la mise en œuvre de simulations distribuées est nécessaire, mais pas suffisant. Il faut également disposer du moyen de s'assurer qu'une architecture de simulation distribuée peut satisfaire les exigences de performances. Réciproquement, il faut aussi disposer du moyen, étant donné un niveau de performances

attendu pour une simulation, d'exhiber les caractéristiques des composants d'une architecture apte à satisfaire ces exigences.

3.4.5.1 Réalisation d'un outil

C'est la raison pour laquelle j'ai développé, avec Bruno d'Ausbourg et Jean-Loup Bussenot, dans le cadre du projet PERFOSIM, un outil d'évaluation prédictive des performances de simulations distribuées HLA. Cet outil prend en compte trois types de composants d'architecture : les différents modèles (fédérés) de simulation constituant l'application de simulation envisagée (fédération), l'infrastructure d'exécution (RTI) et ses variantes ainsi que l'infrastructure de communication réseau.



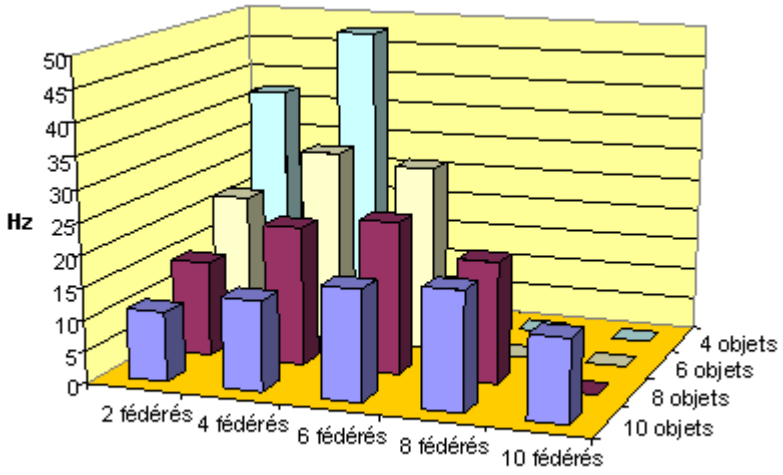
Architecture de PERFOSIM

L'outil réalise en fait une simulation de la simulation distribuée dont on cherche à évaluer les performances et implante lui-même la technologie HLA. La simulation PERFOSIM est une fédération faisant interopérer trois fédérés, interchangeables, chacun bâti sur un modèle des Fédérés, du RTI et du Réseau, comme indiqué par la figure. Les valeurs affectées aux paramètres des différents modèles permettent de décrire et de modifier la structure et le fonctionnement de chacun des composants de la simulation distribuée simulée.

Le choix et l'élaboration de ces modèles, chargés de décrire des entités de niveaux différents et devant cependant communiquer, se sont avérés constituer une opération délicate, dont la réalisation s'est appuyée sur les compétences préalablement acquises au travers de la réalisation d'un RTI (CERTI) et de l'écriture de premières fédérations pilotes de simulateurs, ainsi que sur les compétences du département en simulation et systèmes distribués.

3.4.5.2 Exemple d'application

Devant effectuer la simulation du comportement d'un nombre donné d'objets, le problème est dans cet exemple de pouvoir évaluer combien de fédérés et de machines sont nécessaires (un fédéré pouvant simuler un ou plusieurs objets) à cette simulation. Le critère de performance retenu est la vitesse de simulation, soit le nombre de pas de simulations effectués par seconde. Les courbes obtenues (voir figure) adoptent un profil attendu. Avec un nombre trop faible de fédérés la simulation ne profite pas du gain apporté par le parallélisme, tandis qu'en plaçant un objet par fédéré la charge de communication obtenue est maximale. Ainsi, pour dix objets, la répartition optimale se jouerait sur six ou huit fédérés. Ce résultat peut être pondéré en considérant des critères économiques tels que le coût des machines ou du développement des fédérés.



Exemple d'évaluation de performances

Sur ce thème, j'ai organisé un séminaire ONERA, qui a regroupé lÉtats Majors, la DGA et des industriels. Il a permis, outre la communication nécessaire sur le sujet, d'améliorer et de valider l'approche PERFOSIM.

Les références suivantes concernent les différents paragraphes sur HLA.

Rapports : [BiCaHaSi97], [SiDe99], [AdSi99], [BiCaSi99], [AuBuSi01], [AuBuSi01b]

Articles : [Sir98a], [Sir98b], [BiCaSiZa98], [Sir99], [BiSi99], [AdSi00], [BiSi00], [AdBoSi00], [AdSi01], [Sir01]

Stages : [Des97], [Har98], [Lem00], [TLe00], [Ven00], [Pru01], [Dou01], [Fay01]

Thèse en cours: Benoit Bréholée, Simulation distribuée et domaines

3.5 Conclusion, perspectives

3.5.1 Architectures de systèmes distribués

Les recherches présentées précédemment ont de nombreux points communs, sur lesquels je peux revenir en conclusion.

Architectures ...

Il s'agit la plupart du temps d'un travail sur l'architecture.

Selon le Petit Larousse, il s'agit de l'art de bâtir et d'orner les édifices. C'est une définition flatteuse puisqu'il est question d'art, mais les études d'architectures informatiques concernent bien le problème de bâtir de nouveaux calculateurs avec les matériaux à notre disposition. Cela a été le cas des projets multi-AP, Milord, Modulor ou du réseau local SMN.

Il ne s'agit pas que d'un problème industriel de construction de machines. J'ai été bien au-delà de la définition de nouvelles architectures en me préoccupant constamment de leur utilisation : langage ou environnement de programmation, outils de partitionnement ou de placement qui contraignent à leur tour les spécifications de l'architecture. J'ai également développé des applications pour valider le couple architecture - logiciel dédié. C'est ainsi que j'ai étudié des applications reconfigurables pour profiter des apports de l'optique. Ce genre de travaux est réutilisable ou transposable dans bien d'autres contexte, il est abandonné aujourd'hui mais ce n'est pas loin, par exemple, de la définition et de l'utilisation de clusters de PC Linux.

... de systèmes ...

Je peux dire que plus que des recherches sur les architectures de calculateurs, j'ai mené des recherches sur les architectures de systèmes.

C'est notamment la cas des systèmes complexes où il s'agit de bâtir des assemblages de fonctions logicielles s'exécutant dans des processus sur des processeurs existants.

Sans surprise, le terme architecture se retrouve dans le A de CORBA (voir l'étude de la sécurité d'applications utilisant CORBA) ou de HLA (voir la réalisation d'une plate-forme de simulation distribuée).

... distribués

L'autre point commun de mes différentes études est que l'objet a toujours été un système (au sens large) distribué.

Il est difficile de mettre en valeur la façon de penser que cela implique (modularité, parallélisme des événements, etc.), mais je peux témoigner de la grande difficulté de mes

étudiants à définir une architecture distribuée, à mettre au point un algorithme puis un programme réparti.

perspectives des ...

Comme perspectives de recherche, je vois encore les domaines qui auraient besoin d'architectures distribuées, à utiliser ou à définir. Le développement des réseaux est bien la tendance actuelle, et, à défaut de proposer un cinquième thème de recherches totalement novateur, je peux donner quelques perspectives dans le domaine de la simulation distribuée dans les pages suivantes.

3.5.2 Projet de Recherche Fédérateur SICODIS [01-…]

Un Projet de Recherche Fédérateur regroupe autour d'un même thème des équipes de plusieurs départements de l'ONERA et de différentes branches.

Ce projet trouve sa motivation dans quelques constatations simples : la simulation fait partie des moyens expérimentaux utilisés dans de nombreux départements de l'ONERA ; les simulateurs sont trop souvent développés pour les besoins d'une expérimentation spécifique, sans réel souci, a priori, de réutilisation ; tout simulateur a besoin de disposer d'un environnement qui peut relever de plusieurs disciplines, cet environnement, plutôt que de faire l'objet d'un développement approximatif ad hoc, peut être fourni par d'autres simulateurs.

L'objectif global de SICODIS (SImulation COopérative DIStribuée) est de montrer que l'on peut améliorer de manière significative l'interopérabilité, la réutilisation et la pérennité de simulateurs en faisant reposer leur développement sur des standards spécifiques, en l'occurrence sur le standard de simulation HLA.

Cette démonstration se fera au travers d'une application dimensionnante mise en œuvre sur la plate forme de simulation CERTI.

Les objectifs peuvent être présentés selon les trois axes : application, support et méthodologie.

Pour l'application, il s'agit de :

- développer une application pilote de détectabilité des cibles par un radar passif (DEMR),

- en association avec cette application, développer dans d'autres départements plusieurs applications tests (DPRS, DCSD) afin de diffuser les nouvelles technologies de simulation distribuée d'une part, afin de préparer la collaboration de ces différents acteurs à l'application pilote d'autre part,

- spécifier et mettre en œuvre une application pilote de la simulation distribuée

coopérative, l'aspect coopératif concernant dans le cadre de cette étude diverses équipes et départements, l'aspect distribué différents sites de l'Office.

Au niveau du support, ce qui est recherché est de :

- maintenir l'avance en simulation distribuée en approfondissant les techniques nécessaires et en prenant en compte les aspects du standard non encore supportés à ce jour, ni par les outils développés en interne, ni par les rares produits existant sur le marché,

- maintenir la maîtrise d'un support de simulation distribuée essentielle pour tenir compte des besoins de sécurité informatique et pour profiter des évolutions des technologies de communication, offrir un ou plusieurs supports aux départements de l'ONERA ainsi que les initiations, formations ou conseils nécessaires,

L'axe méthodologie implique :

- étudier les problèmes liés à la conception et à la validation de simulations distribuées. Ceci suppose en amont de concevoir les moyens d'expliciter des scénarios de simulation, et en aval de disposer d'outils pour observer le déroulement de la simulation et les comparer aux évolutions souhaitées.

- enfin, la structure de simulation est en principe une structure ouverte sur laquelle l'utilisateur a le choix d'implanter n simulateurs sur m machines connectées par un réseau. Un dernier objectif sera de fournir les moyens de dimensionner et de configurer cette structure en fonction de critères non fonctionnels tels que les performances en temps et en espace, l'isolation de certaines données, la sûreté de fonctionnement.

Un objectif supplémentaire serait de comprendre les liens entre la simulation à événements discrets et les autres formes de simulation utilisées à l'ONERA (simulation numérique pour simplifier), progresser vers une architecture et une méthodologie plus intégrée est un objectif à long terme.

3.5.3 Projet FéRIA SCD [01-…]

3.5.3.1 Présentation du groupe de travail

La recherche en sciences et techniques de l'information à Toulouse est essentiellement développée autour de trois grands laboratoires : l'ONERA, l'Institut de Recherche en Informatique de Toulouse (IRIT), et le Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS).

Ces laboratoires ont estimé nécessaire de faire face ensemble à un certain nombre de problèmes posés par l'évolution scientifique et technologique, d'où la constitution d'une Fédération de Recherche en Informatique et Automatique (CNRS). Le projet SCD (Simulation Coopérative Distribuée) est l'un des 7 projets ou groupes de travail identifiés.

3.5.3.2 Objectifs du projet SCD

L'objectif du projet que j'anime est d'offrir aux équipe fédérées la possibilité d'étudier en commun les problèmes ouverts du domaine de la simulation distribuée, et d'évaluer les solutions qu'elles pourraient proposer dans un contexte expérimental réaliste.

On distinguera deux phases dans le déroulement du projet qui seront :

- le déploiement sur les trois sites, puis l'interconnexion, de noyaux support de simulation commune, et le partage et/ou la distribution d'applications,

- l'étude des problèmes techniques non encore résolus en simulation distribuée, et leur expérimentation sur la plate-forme multi-sites.

3.5.3.3 Plate-forme multi-sites et applications

La plate-forme support qui est déployée sur les trois sites est construite à partir du RTI développé par le DTIM (CERTI). Sur un plan administratif et technique les possibilités de connexion au travers du réseau Internet ont été étudiées. Des premières expérimentations ont été réalisées en 2001.

Les applications qui seront implantées en premier lieu seront des applications du type déplacement et interaction de mobiles. Elles seront retenues parmi les applications disponibles dans les 3 laboratoires (jeu de billard dont le portage a été effectué en 2001, simulation de défense antiaérienne, simulation distribuée de plusieurs véhicules (chars) pilotés au clavier par les utilisateurs dont le portage est prévu en 2002).

Ces applications initiales ont été retenues en raison de leur disponibilité d'une part, et des problèmes ouverts qu'elles permettent d'aborder d'autre part. Elles sont appelées à évoluer et à être complétées pendant la durée du projet.

3.5.3.4 Problèmes ouverts

Les problèmes ouverts sont nombreux et seront abordés soit à travers des réunions mensuelles de travail (présentations), soit à travers de co-encadrement de stagiaires. On peut espérer travailler sur :

Extension des services offerts par la plate-forme

Les simulations que l'on met en œuvre se déroulent sur plusieurs sites, chacun d'entre eux permettant de gérer une partie de la simulation, avec des options qui lui sont propres, et devant se synchroniser avec les autres sites. On s'attachera dans les travaux reliés à cet axe à :

- prendre en compte différentes alternatives pour la gestion du temps (temps réel, temps coordonné, approches optimistes ou pessimistes de la synchronisation...),
- gérer la diffusion optimale des informations (globales, par sous-domaines...),
- développer des applications de test liées à l'utilisation du service ddm (data distribution management).

Réseau et Qualité de Service

L'objectif serait ici d'étudier l'adaptation des résultats obtenus sur la qualité de service pour les simulations DIS et les réseau et protocoles TCP/IP nouvelle génération au contexte des simulations HLA :

- étude de l'adéquation de HLA à la prise en compte des caractéristiques et des besoins applicatifs dans un environnement Internet de nouvelle génération tel que l'Internet à services différenciés,
- proposition d'une architecture de communication de bout en bout (incluant ou non HLA) permettant la prise en compte et la satisfaction des besoins applicatifs au regard des critères définis.

Interopérabilité de systèmes de simulation

Le problème d'inclure dans une simulation HLA un simulateur qui n'a pas été développé suivant la norme pourrait se voir comme un problème d'encapsulation dans lequel on mettrait autour du simulateur à intégrer une capsule qui respecterait l'API de HLA. Le problème le plus aigu qui se pose alors est celui de la gestion du temps simulé vu par le nouveau simulateur et vu par le reste de la simulation.

Un problème analogue se pose quand on veut intégrer dans une simulation fonctionnant en temps coordonné un module ayant un comportement temps réel.

Des réflexions de base sur la gestion du temps dans les modèles, sur les différentes forme de temps coordonné et de temps réel, sur les interactions entre les activités de modélisations et les contraintes liées à HLA sont à reprendre.

Il n'existe pas, probablement, de solution générale au problème, le module temps réel n'ayant pas la possibilité "d'attendre", ou de "revenir en arrière" comme le demanderait la gestion du temps coordonné, éventuellement optimiste. Il existe néanmoins des compromis acceptables permettant de marier les deux types de simulation, et que l'on s'efforcera de

déterminer et d'expérimenter.

Performances des simulations

En complément de l'étude PERFOSIM, il s'agirait de :

- de définir et de mettre en place les principes d'étalonnage adaptés aux nouvelles technologies de réseaux de types LAN et WAN,
- de développer ou de réutiliser des modèles pour ces réseaux,
- de les valider en comparant les données qu'ils fournissent à celles qui sont attendues ou observées sur un système réel.

Simulation distribuée sur clusters

Soit en coopération avec le projet PAR (Programmation d'Applications Réparties) de FéRIA, soit en coopération avec le laboratoire de SUPAERO, il s'agirait de se donner les moyens d'effectuer efficacement des simulations distribuées sur des clusters. Un chemin est d'adapter CERTI à ce nouvel environnement d'exécution.

Les clusters étant adaptés au calcul parallèle, un autre aspect serait de reprendre le lien entre calcul parallèle et simulation distribuée :

- soit en établissant des ponts entre différents applications ou en ré-injectant la notion de temps dans des simulations numériques souvent "statiques",
- soit en utilisant HLA comme une bibliothèque logicielle avec laquelle on peut écrire des applications parallèles (point de vue de l'utilisateur),
- soit en prenant le point de vue du concepteur de RTI en recherchant le point commun dans la mise en œuvre d'applications et de simulations parallèlesefficaces. Ce point commun pourrait résider dans une couche de communication intermédiaire que l'on pourrait faire utiliser à CERTI.

3.5.4 Simulation distribuée et domaines [01-…]

Les études menées ont déjà permis de répondre à de nombreuses questions concernant HLA, notamment dans le cas de fédérations de taille moyenne sur un réseau local. Elles soulèvent de nombreuses questions, en particulier sur le plan des performances, de l'interopérabilité et de la sécurité informatique, pour les fédérations à grand nombre de fédérés et sur un réseau étendu (WAN).

L'objectif de cette thèse en cours (Benoit Bréholée) est d'étendre le travail entrepris sur

CERTI en se basant sur l'introduction de la notion de domaines. Cette notion "classique" devrait permettre :

- une hiérarchisation des communications dans la simulation distribuée. L'idée est de minimiser les communications inter-domaines en les factorisant. Il serait ainsi possible d'apporter une réponse au vaste problème du multicast fiable, réponse qui ne reposerait pas sur l'utilisation de routeurs spécifiques et qui s'apparenterait aux propositions de réseaux "intelligents".

- d'aborder le problème de l'interopérabilité qui recouvre différentes facettes dont :

    - la possibilité de connecter deux fédérations différentes par un fédéré pont. Ceci permettrait de relier deux fédérations utilisant un RTI de fournisseur différent.

    - la possibilité, par ce biais, d'intégrer des fédérés écrits dans des langages comme Java ou Ada à CERTI,

    - d'envisager la sécurité d'une simulation distribuée par un couple pont-garde,

- enfin la notion de domaine se rapproche de la proposition du service d'optimisation de gestion des données de HLA basée sur la notion de régions d'abonnement et de publication. Ce service est encore rarement implanté et on pourrait proposer une mise en œuvre avec une architecture de CERTI adaptée (connexion de processus RTIP) ou en émulant cette architecture par la connexion de différentes fédérations. Tout comme pour le problème du multicast fiable, on ressent le besoin de tisser des liens entre l'expression des besoins au niveau logiciel et la mise en œuvre.

3.5.5 Validation et simulation distribuée

La simulation est utilisée au cours du développement d'un système depuis les phases de conception (est-ce que les performances temporelles seront satisfaisantes ?) jusqu'aux phases finales (par exemple, comment s'entraîner avec un système ?). C'est dire l'importance que l'on attache aux résultats de simulation et à leur qualité.

3.5.5.1 La simulation comme un moyen d'essai

D'une façon encore plus exigeante, elle sert à la validation des systèmes. Des essais peuvent être effectués sur le simulateur d'un avion au sol par exemple, essais qui, d'une part, compteront dans le processus de certification de cet avion, et d'autre part, qui éviteront des prises de risques aux pilotes d'essais.

Mais alors se pose le problème de validation de la simulation, avec plusieurs aspects :

- validation de l'utilisation de la simulation,

- vérification et validation des modèles.

3.5.5.2 Utilisation de la simulation

Il s'agit d'abord de ne pas sortir du cadre expérimental pour lequel les modèles ont été conçus. La solution est souvent d'ordre méthodologique, dans la définition d'un processus de développement et d'utilisation des simulations.

Le cadre expérimental étant respecté, il faut faire un nombre "suffisant" de tests. Outre l'aspect statistique souvent envisagé dans les études de simulation, il s'agit du problème général de couverture des tests.

Il faudrait étudier, d'une part comment la distribution introduit de nouveaux problèmes dans la génération des jeux de test, d'autre part les problèmes spécifiques de mise en œuvre de ces tests dans un contexte distribué (synchronisation des fédérés, recueil distribué d'un état, etc.).

3.5.5.3 Validation des modèles

Ce problème est déjà souvent maltraité pour les simulations monolithiques car très dur à résoudre. Il faudrait que les résultats de simulation soient identiques aux résultats fournis par le système réel (s'il existait et s'il pouvait être soumis aux mêmes conditions). Le plus souvent, on essaie de définir un modèle reprenant les composants du monde réel et leurs interactions. Dans le cas de simulation pour l'entraînement, il faudrait également définir ce qu'est un entraînement réussi.

3.5.5.4 Utilisation de méthodes formelles

Simulation et utilisation de méthodes formelles sont souvent utilisées pour la validation des systèmes. Il s'agirait d'approfondir les liens entre ces deux techniques :

- le premier lien est que l'on construit toujours des modèles avant, soit de les simuler, soit de les valider formellement. Une description commune des modèles serait souhaitable, c'est déjà le cas dans certains outils commerciaux, cela n'a pas encore été envisagé pour HLA.

- l'utilisation de méthodes formelles peut servir à la génération de jeux de tests et donc à la simulation. De même on pourrait injecter les propriétés formelles à vérifier dans l'outil de simulation pour s'en assurer lors des diverses exécutions (avec les difficultés déjà évoquées de recherche de propriétés dans un système réparti).

- un utilisateur de méthodes formelles a besoin, pour des problèmes de mise au point, de voir vivre ses modèles, c'est-à-dire de les simuler.

Les points évoqués précédemment correspondent à une utilisation pragmatique des méthodes formelles, une utilisation plus théorique serait la génération automatique de modèles valides, puis la génération automatique de fédérés et de fédérations.

4 Activités d'enseignement

4.1 A l'ENSAE (SUPAERO)

Le CERT et L'ENSAE entretiennent des relations privilégiées, aussi j'ai de préférence exercé une activité de maître de conférence (vacataire depuis 1982) puis professeur (vacataire) dans cette grande école. La majorité des cours suivants ont été réalisés sous la responsabilité de Bernard Lécussan, Professeur d'informatique de l'ENSAE. Qu'il soit ici une nouvelle fois remercié de sa confiance.

Première année

L'intérêt, pour moi, d'enseigner en première année est, à partir de matières simples, de faire découvrir l'informatique aux étudiants. Il s'agit donc plus d'exercer de la pédagogie que d'aborder des disciplines pointues. C'est pour cette raison que je ne me suis jamais lassé de ces modules.

Module "accès au système"

Ce module est le premier module d'informatique dispensé , il a pour objectif de permettre aux étudiants de maîtriser les outils de base disponibles sur le réseau de stations de travail du centre informatique de l'école. Il a du évoluer considérablement au fil du temps, aujourd'hui sont enseignées l'utilisation du système Unix et l'utilisation des réseaux (rlogin, mail, ftp, etc.).

Module "architecture des calculateurs"

Ce module présentait l'architecture des calculateurs, les notions de représentation de l'information et de code assembleur. Il n'est plus enseigné.

Module "algorithmique et programmation"

Ce module vise à donner aux étudiants les compétences nécessaires pour développer correctement des programmes informatiques, en insistant sur les techniques de modularité et d'abstraction.

Nous avons utilisé successivement différents langages :

- le langage Pascal, bien adapté à l'enseignement avec quelques contraintes et des vérifications systématiques,

- le langage Fortran, facile à apprendre et qui permet une bonne introduction à l'algorithmique. Ce langage pouvait être utilisé ultérieurement dans l'enseignement du calcul scientifique.

- le langage C, actuellement enseigné. Il demande une bonne discipline de

programmation et servira de base à l'apprentissage des langages objets en deuxième année.

Deuxième année

Module "calcul scientifique"

Ce module comprenait d'une part une initiation au langage Fortran, d'autre part une initiation au calcul scientifique à l'aide de bureaux d'études, comme :

- résolution d'une équation différentielle par la méthode des différences finies (résolution d'un système tridiagonal, méthodes itératives de Jacobi, de Gauss-Seidel, etc.),
- approximation d'une fonction par la méthode des moindres carrés (utilisation de la méthode de Cholesky).

Troisième année, Spécialisation informatique, DEA

On aborde des enseignements qui sont de plus en plus proches de mon activité de recherche. Ceci est une deuxième raison expliquant mon intérêt pour l'enseignement et les sollicitations reçues. On trouvera naturellement des contributions dans les cours de systèmes répartis et de sécurité.

Module "compléments systèmes"

Il a pour objectif de rappeler les notions d'interface d'un système d'exploitation, présentées très rapidement en première année, et de les compléter par la description des principaux mécanismes de base sous-jacents. Ceci était indispensable pour comprendre les cours suivants. Mon intervention a, en particulier, concerné :

- la gestion des mémoires (mémoire principale, fichiers Unix, etc.),
- la mémoire virtuelle,
- les fonctions systèmes de C (gestion de processus, communications inter-processus, etc.).

Module "réseaux informatiques"

Dans ce module, un travail personnel était demandé nécessitant un développement sur plusieurs stations de travail. Mon intervention concernait la définition de ce travail et la présentation préalable des mécanismes de communication entre processus distants via les

sockets Unix.

Module "construction de systèmes répartis"

Ma contribution dans ce module a porté sur l'enseignement de :

- quelques algorithmes répartis,
- le modèle CSP, le langage Occam, les Transputers,
- les nouveaux systèmes répartis (les micro-noyaux Mach, Chorus, Amoeba, la mémoire virtuelle distribuée),
- le langage Ada et la distribution.

On illustrait ainsi différentes alternatives dans la construction de systèmes répartis en se reposant sur des aides matérielles, du système d'exploitation, du langage.

Ce module était un également un cours de DEA. Il est partiellement repris dans le module "systèmes informatiques répartis et liaisons embarquées".

Module "sécurité des systèmes informatiques"

Il s'agit d'aborder le problème de la sécurité des systèmes répartis en s'intéressant plus particulièrement à la technologie des gardes-barrières (définition, architectures, évaluation). Le problème de la portabilité d'applications réparties est également considéré.

Ce cours fait également partie d'un module de DEA. Il est aussi enseigné aux étudiants de troisième année de l'ENSEEIHT.

Module "validation de systèmes"

Dans ce cours sont considérées deux approches pour la validation des systèmes : l'utilisation de méthodes formelles, l'utilisation de techniques de simulation. Je reprends un sous-ensemble du stage présenté dans le paragraphe suivant.

Ce cours compte pour le DEA.

Liaison entre les modules

Les modules précédents se sont souvent enchaînés lors de la réforme de la Spécialisation informatique de SUPAERO. En tant que participant aux différents modules, j'ai pu établir quelques liens entre les enseignants et les enseignements. Citons par exemple :

- la programmation d'applications multi-processus locaux (système Unix), puis sur un réseau (sockets d'une part, avec un système d'exploitation réparti d'autre part),

- l'illustration de problèmes de sécurité dans des applications multi-processus, l'expérimentation de codes mobiles,

- la modélisation et simulation de protocoles de communication (CSMA/CD, TCP, etc.).

4.2 A la Société des Amis de SUPAERO et de l'ENSTA (SAE)

Cette société s'occupe de la formation permanente. Outre des interventions ponctuelles dans les stages de "langage C" et de "Réseaux d'ordinateurs et systèmes répartis", je suis principalement co-animateur et intervenant dans un nouveau stage dont le titre est "Modélisation et Simulation de Systèmes Complexes".

Le programme de stage comporte :

- une introduction à la modélisation et à la simulation à événements discrets,

- une présentation d'outils et langages (Netsim, QNAP, ...),

- des exemples de simulation (réseaux informatiques, productique),

- des démonstrations de HLA (simulation technico-opérationnelle d'une défense anti-aérienne).

C'est un des premiers stages consacrés en France à HLA.

5 Activités d'animation et d'encadrement

5.1 Responsabilité scientifique de contrats de recherche

Etant donné le statut d'EPIC (Etablissement Public à Caractère Industriel et Commercial) de l'ONERA, la quasi totalité de mes travaux a été menée dans le cadre de contrats de recherche. Avec les risques d'oubli habituels, les principaux contractants ont été :

- la DGA (Direction Générale de l'Armement) avec ses différents services :
    - la DRET (Direction des Recherches, Etudes et Techniques) puis le STTC (Service des Technologies et Techniques Communes) ont soutenu les activités menées dans le cadre du programme "Grand Calculateur", les projets Milord et Modulor, une partie des activités des projets du thème Sécurité et Simulation distribuée,
    - le CELAR (Centre d'Essais de l'Armement) a soutenu des études en Sécurité,
    - le SPAé (Service de Programmes de l'Aéronautique) a soutenu le maquettage de HLA,
- le Ministère de la Recherche a apporté son soutien au projet Modulor via le PRC (Programme de Recherche Coordonnée) ANM (Architecture Nouvelle de Machine) et le plan Etat-Région,
- la Région Midi-Pyrénées a soutenu les travaux de développement de la maquette Modulor,
- la Communauté Européenne a également apporté son soutien au projet Modulor via le programme ESPRIT PCA (Parallel Computing Action),
- le CNES (Centre National d'Etudes Spatiales) a financé des travaux dans le domaine des architectures de traitement bord distribuées,
- France Télécom a supporté de projet Pedalo.

Mentionnons enfin L'ONERA pour des travaux menés au profit d'autres départements (maquette de multiprocesseur du centre de Chatillon) ou sur des ressources propres (première maquette de HLA par exemple).

Chaque action de soutien a donné lieu à un contrat dont j'ai assuré la responsabilité scientifique (ou dans certains cas la co-responsabilité). Une partie non négligeable de temps (pris sur le temps de la recherche) a été passée à rechercher et négocier les nouveaux contrats avec les contractants, à discuter des orientations scientifiques à soutenir, à rédiger des propositions ou à répondre à des appels d'offres. Ceci est le corollaire de recherches

appliquées.

La liste des principaux rapports de recherches produits (voir plus loin) donne un aperçu de cette charge contractuelle.

5.2 Encadrement de travaux de recherche

Au cours de ces différents contrats de recherche, j'ai assuré l'encadrement (ou le co-encadrement) de plusieurs Appelés Scientifiques du contingent affectés par la DRET au CERT en soutien de nos projets de recherche : J.-M. Labouyrie en 1987-1988, F. Boniol en 1988-1989, V. David en 1989-1990, P. Nortier en 1996-1997, Th. Hautesserres en 1997-1998.

J'ai également encadré différents travaux de stagiaires de troisième année d'école d'ingénieur ou de DESS ou de DEA. Je n'énumère pas ici ces stagiaires qui ont pour une bonne part contribué à nos réalisations, ce sont les auteurs des rapports de stage dont la liste figure plus loin.

Le département étant laboratoire d'accueil dans le cadre de la formation doctorale de l'ENSAE, j'ai participé, d'une façon officieuse, aux travaux de thèse de X. Thibault et V. David. J'encadre actuellement le travail de thèse de B. Bréholée dont le sujet est "Simulation distribuée et domaines".

6 Principaux rapports techniques

6.1 Base de données réparties et dupliquées

[Sir80] P. Siron
Réalisation d'outils logiciels pour la gestion d'une base de données répartie et dupliquée.
Rapport de fin d'études ENSEEIHT, Juin 1980.

6.2 Les architectures parallèles

6.2.1 Le projet Marianne

[AdCoSi82]M. Adelantado, D. Comte, P. Siron
Simulateur d'architectures multiprocesseur de type MIMD à mémoires locales.
Rapport no 2/3191/DERI, Décembre 1982.

[AdCoSi82]M. Adelantado, D. Comte, S. Donnet, P. Siron
Evaluation de performances d'architectures MIMD. Langage XANADU et rémanence des données.
Rapport final no 3/3210/DERI, Janvier 1984.

[Sir84] P. Siron
Étude des problèmes d'accès aux données dans une architecture multiprocesseur MIMD à mémoires locales.
Thèse de docteur ingénieur de l'ENSAE, Janvier 1984.

6.2.2 Le projet Multi-AP

[CoFrHiSi84]D. Comte, Ch. Fraboul, N. Hifdi, P. Siron
Maquette multi AP120B : spécification des outils de contrôle de l'architecture MIMD couplage lâche et de la chaîne de production de programmes associés.
Rapport int. no 1/3233/DERI, Juin 1984.

[AdFrHiSi86]M. Adelantado, Ch. Fraboul, N. Hifdi, P. Siron
Superviseur MIMD couplage lâche de la configuration multi-AP et chaîne de production de programmes parallèles associée.
Rapport final no 4/3246/DERI, Février 1986.

[Sir86] P. Siron
Superviseur MIMD couplage lâche de la configuration multi-AP : le module interpréteur.

Rapport final no 2/3250/DERI, Janvier 1986.

[FrSi86] P. Siron
Superviseur MIMD couplage lâche de la configuration multi-AP : le module manager.
Rapport final no 3/3250/DERI, Février 1986.

6.2.3 Le projet Isis

[DiRoSi86] B. Dupont de Dinechin, J.-Y. Rousselot, P. Siron
Etude du parallélisme sur un processeur ISIS.
Rapport final no 5/3246/DERI, Janvier 1986.

[RoSiRa86] J.-Y. Rousselot, P. Siron, J.-M. Labouyrie
Etude du parallélisme sur un processeur ISIS (réalisation de la bibliothèque de "multitasking").
Rapport final no 6/3268/DERI, Décembre 1986.

[RoSi87] J.-Y. Rousselot, P. Siron
Etude du parallélisme sur un processeur ISIS.
Rapport final no 7/3289/DERI, Décembre 1987.

[BoRoSi89] F. Boniol, J.-Y. Rousselot, P. Siron
Etude du parallélisme sur un processeur ISIS.
Rapport final no 6/3325/DERI, Janvier 1989.

6.3 Les architectures distribuées

6.3.1 Le projet Milord

[CoSiTh87] P. Churoux, D. Comte, M. Fracès, M. Laug, P. Siron, X. Thibault
Fonctions optiques dans l'ordinateur, impact des technologies optiques dans les architectures parallèles.
Rapport final no 1/3272/DERI, Janvier 1987.

[CoSiTh88] D. Comte, P. Siron, X. Thibault, J.-P. Bouzinac, P. Churoux, M. Fracès, J. Isbert, M. Laug, D. Chauvé, Ph. Rolland, B. Duchenne
Multiprocesseur Interconnecté par Liaisons Optiques Reconfigurables Dynamiquement (MILORD).
Rapport final no 2/3295/DERI, Mars 1988.

[CoSiTh89] D. Comte, P. Siron, X. Thibault, J.-P. Bouzinac, P. Churoux, M. Fracès, J.-C.

Sanudo, B. Duchenne, Ph. Rolland, J. Duffaut, M. Bodilis
Multiprocesseur Interconnecté par Liaisons Optiques Reconfigurables Dynamiquement (MILORD).
Rapport final no 2/3314/DERI, Mai 1989.

[CoSi90] D. Comte, P. Siron, J.-P. Bouzinac, M. Fracès, E. Bodin, Ph. Rolland
Multiprocesseur Interconnecté par Liaisons Optiques Reconfigurables Dynamiquement (MILORD).
Rapport final no 2/3352/DERI, Mai 1990.

6.3.2 Le projet MODULOR

[FrRoSi89] Ch. Fraboul, J.-Y. Rousselot, P. Siron
Etude d'architectures reconfigurables (Projet MODULOR).
Rapport final no 8/3325/DERI, Janvier 1989.

[BoFrRoSi90] F. Boniol, Ch. Fraboul, J.-Y. Rousselot, P. Siron
Etude d'architectures reconfigurables (Projet MODULOR).
Rapport final no 3/3344/DERI, Février 1990.

[DaFrRoSi91] V. David, Ch. Fraboul, J.-Y. Rousselot, P. Siron
Etude et réalisation d'une architecture modulaire et reconfigurable : projet MODULOR.
Rapport final no 1/3364/DERI, Mars 1991.

[FrSi92] Ch. Fraboul, P. Siron
Etude d'une architecture parallèle reconfigurable tolérante aux pannes.
Rapport final no 2/3420/DERI, Novembre 1992.

[FrLaSiFuPaSz93] Ch. Fraboul, M. Lanne, P. Siron, I. Futo, I. Papp, J. Szeredy
Programmation logique dans un environnement distribué et temps réel.
Rapport final no 1/3427/DERI, Janvier 1993.

[FrMaSi93] Ch. Fraboul, L. Maillet, P. Siron
Architecture temps réel distribuée tolérante aux pannes.
Rapport final no 1/3451/DERI, Juillet 1993.

[FrSiFuPaSzNo93] Ch. Fraboul, P. Siron, I. Futo, I. Papp, J. Szeredy, F. Noailles
Mise en œuvre de modèles de programmation concurrent sur une architecture parallèle à mémoire distribuée.
Région Midi-Pyrénées. Rapport no 9200225/DERI, Août 1993.

6.4 Sécurité des systèmes répartis

[Sir94] P. Siron
Etude de l'évaluation des performances du protocole SMAC. Modèle CSMA/CD et SMAC.
Rapport intermédiaire no 1/3502/DERI, Février 1994.

[AuSi94] B. d'Ausbourg, P. Siron
Etude de l'évaluation des performances du protocole SMAC. Résultats des mesures - interprétation des résultats - Conclusion.
Rapport final no 2/3502/DERI, Mai 1994.

[AuCaSi94] B. d'Ausbourg, Ch. Calas, P. Siron
Sécurité informatique sur systèmes répartis.
Rapport final no 4/3482/DERI, Novembre 1994.

[Sir95] P. Siron
Systèmes distribués : évaluation de Chorus.
Rapport no 3697/DERI, Janvier 1995.

[Sir95b] P. Siron
Sécurité informatique, systèmes opératoires répartis et algorithmes répartis.
Rapport final no 2/3530/DERI, Octobre 1995.

[SiDeFa96] P. Siron, D. Demulière, J.-F. Fabre
Sécurité informatique et systèmes répartis.
Rapport final no 2/3562/DERI, Septembre 1996.

[AuBiCuEiSiYa97] B. D'Ausbourg, P. Bieber, F. Cuppens, G. Eizenberg, P. Siron, K. Yazdanian
Synthèse des études en sécurité multi-niveau : application aux bases de données réparties.
Rapport final no 1/3726/DERI, Octobre 1997.

[BiCaHaSi97] P. Bieber, J. Cazin, Th. Hautesserres, P. Siron
SAIDA - case-study overview, security analysis and preliminary work.
Rapport final no 2/3702/DERI, Décembre 1997.

[BiSi98] P. Bieber, P. Siron
Définition des fonctions de protection des agents logiciels.
Rapport intermédiaire no 1/3738/DTIM, Juin 1998.

[LlSi98] J.-H. Llareus, P. Siron
Plate-forme de démonstration et d'expérimentation en sécurité multi-niveau.
Rapport final no 3600.59/DTIM, Juillet 1998.

[BiCaSi99] P. Bieber, J. Cazin, P. Siron
SAIDA - Security architecture design, validation and implementation.
Rapport final no 2/3741/DTIM, Janvier 1999.

[BiSi99] P. Bieber, P. Siron
Architecture de sécurité pour CIDRIA.
Rapport intermédiaire no 3/3738/DTIM, Juin 1999.

[BiCaSiWi99] P. Bieber, J. Cazin, P. Siron, V. Wiels
SAIDA - security architecture evaluation.
Rapport technique no 1/3774/DTIM, Décembre 1999.

[BiCaSi00] P. Bieber, J. Cazin, P. Siron
SAIDA - security architecture synthesis and recommendation.
Rapport final no 0093901F/DTIM, Septembre 2000.

[BiRaSi00] P. Bieber, D. Raujol, P. Siron
Pédalo : contrôle des rôles pour des architectures à bases d'objets répartis.
Rapport final no 00937/DTIM, Décembre 2000.

6.5 Simulation distribuée

[SiDe99] P. Siron, P. Desseaux
Support de simulation interactive distribuée.
Rapport final no 1/3601/DTIM, Janvier 1999.

[AdSi99] M. Adelantado, P. Siron
Maquette HLA.
Rapport final no 3/3776/DTIM, Septembre 1999.

[AuBuSi01] B. d'Ausbourg, J.-L. Bussenot, P. Siron

Evaluation de performances d'architectures de simulation distribuée (PERFOSIM).
Rapport intermédiaire no 2/05018/DTIM, Janvier 2001.

[AuBuSi01b] B. d'Ausbourg, J.-L. Bussenot, P. Siron
Evaluation de performances d'architectures de simulation distribuée (PERFOSIM).
Rapports de réalisation no 3 et 4/05018/DTIM, Septembre 2001.

[AdSi01] M. Adelantado, P. Siron
Projet de Recherche Fédérateur SICODIS. Support de simulation coopérative distribuée.
Rapport technique no 3/05723/DTIM, Décembre 2001.

[BrSi02] B. Bréholée, P. Siron
Projet de Recherche Fédérateur SICODIS. Simulation distribuée et domaines.
Rapport technique no 1/Q0195/DTIM, Février 2002.

7 Liste d'articles

Les articles sont présentés ici par thèmes. Cette liste pourrait être également triée pour distinguer :

- les conférences invitées et articles dans les revues : [AdBeCoCrFrHiSiSy84], [CoSiThChFrLa87], [BoCoFrRoSi89], [FrRoSi91]

- les conférences et séminaires internationaux avec comités de lectures : [AdCoSiSy83], [AdBeCoSi84], [AdCrFrHiSi85], [AdSi85], [AdFrHiSiBeDa88], [RoSi88], [ChFrLaCoSiTh87], [FrBoChLaCoSiTh88], [ThCoSi88], [CoSiThFrBo89], [FrBoBoCoSi90], [FrRoSi89], [FrRoSi89b], [DaFrRoSi92c], [DaFrRoSi92e], [FrSi93], [PaSeSiWi95], [Sir98a], [AdSi00], [AdBoSi00], [AdSi01], [SiAu94], [BiCaSiZa98], [BiSi99], [BiSi00], [BiRaSi00], [BrSi02], [AuBuSi02b]

- les conférences et séminaires nationaux avec comités de lectures : [CoSiTh87], [DaFrRoSi91], [DaFrRoSi92], [FrSi93b]

- les autres communications : [RoSiBo89], [ChCoFrLaSiTh87], [FrRoSi89c], [FrRoSi90], [FrRoSi91b], [FrSi91], [DaFrRoSi92b], [DaFrRoSi92d], [FrSi94], [Sir99], [Sir01], [Sir95], [Sir98b], [AuBuSi02a]

7.1 Les architectures parallèles

7.1.1 Le projet Marianne

[AdCoSiSy83]M. Adelantado, D. Comte, P. Siron, J.C. Syre
A MIMD supercomputer system for large scale numerical applications.
IFIP, Paris, 1983.

[AdBeCoCrFrHiSiSy84]M. Adelantado, Ph. Berger, D. Comte, P. Cros, Ch. Fraboul, N. Hifdi, P. Siron, J.C. Syre
A MIMD supercomputer system.
Conférence invitée.
Proc. Experiences in applying parallel processors to scientific computation, Oregon, 1984.

[AdBeCoSi84]M. Adelantado, Ph. Berger, D. Comte, P. Siron
Expression of concurrency and parallelism in an MIMD environment.
Computer Physics Communications 37, 1985.
Vector and Parallel Processors in Computational Science II, Oxford, Août 1984.

[AdCrFrHiSi85]M. Adelantado, P. Cros, Ch. Fraboul, N. Hifdi, P. Siron
An experimental MIMD system, parallel languages and supervision functions.

Parallel Computing, Berlin, Septembre 1985.

[AdSi85]M. Adelantado, P. Siron
Xanadu : a task sequencing language for MIMD architectures.
SCS85, First International Conference on Supercomputing Systems, St Petersburg, FL, Décembre 1985.

7.1.2 Le projet Multi-AP

[AdFrHiSiBeDa88]M. Adelantado, Ch. Fraboul, N. Hifdi, P. Siron, Ph. Berger, M. Dayde
Xanadu: an experimental approach to MIMD computing.
Proc. CONPAR88, Manchester, Septembre 1988.

7.1.3 Le projet Isis

[RoSi88]J.Y. Rousselot, P. Siron
MIMD tools for a supercomputer under Unix system.
CONPAR88, Manchester, Septembre 1988.

[RoSiBo89]J.Y. Rousselot, P. Siron, F. Boniol
Outils d'expression du parallélisme MIMD pour le supercalculateur Isis.
Journées Algorithmes parallèles et architectures nouvelles, Toulouse, Mars 1989.

7.2 Les architectures distribuées

7.2.1 Le projet Milord

[CoSiTh87]D. Comte, P. Siron, X. Thibault
De l'optoélectronique à l'optoinformatique.
Congrès Télémat - Horizons de l'optique, Marseille, Juin 1987.

[CoSiThChFrLa87]D. Comte, P. Siron, X. Thibault, P. Churoux, M. Fracès, M. Laug
Communication par crossbar optique dans une architecture informatique massivement parallèle.
Revue physique appliquée, no 22, Octobre 1987.

[ChCoFrLaSiTh87]P. Churoux, D. Comte, M. Fracès, M. Laug, P. Siron, X. Thibault
Multiprocesseur à communications optiques : le projet Milord.
Séminaire Algorithmes et architectures à haut degré de parallélisme, La Briantais, Novembre 1987.

[ChFrLaCoSiTh87]P. Churoux, M. Fracès, M. Laug, D. Comte, P. Siron, X. Thibault

Optical crossbar network analysis.
Int. symposium on the technologies for optoelectronics, Cannes, Novembre 1987.

[FrBoChLaCoSiTh88]M. Fracès, J.P. Bouzinac, P. Churoux, M. Laug, D. Comte, P. Siron, X. Thibault.
A multiprocessor based on an optical crossbar network: the MILORD project.
Proc. SPIE Optical Computing Conference, Août 1988.

[ThCoSi88]X. Thibault, D. Comte, P. Siron
A reconfigurable optical interconnection network for highly parallel architecture.
Proc. Frontiers of Massively Parallel Computation, Fairfax, Octobre 1988.

[CoSiThFrBo89]D. Comte, P. Siron, X. Thibault, M. Fracès, J.P. Bouzinac
Communications optiques dans une architecture multiprocesseur.
Proc. colloque int. Fonctions optiques dans l'ordinateur, Toulouse, Octobre 1989.

[FrBoBoCoSi90]M. Fracès, E. Bodin, J.P. Bouzinac, D. Comte, P. Siron
The optical crossbar network MILORD machine: last developments and results.
Proc. Int. congress on optical science and engineering, The Hague, Mars 1990.

7.2.2 Le projet Modulor

[BoCoFrRoSi89]F. Boniol, D. Comte, Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR : la chaîne de développement de programmes parallèles reconfigurables.
Proc. Journées AFCET-GROPLAN, Développement de programmes pour machines parallèles, BIGRE no 66, Chamonix, Mai 1989.

[FrRoSi89]Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR: a modular reconfigurable highly parallel architecture.
Proc. of the 1st European Workshop on hypercube and distributed computers, Rennes, Octobre 1989.

[FrRoSi89b]Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR: software tools for developping programs on a reconfigurable highly parallel architecture.
Proc. Parallel Computing 89, Leiden, Septembre 1989.

[FrRoSi89c]Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR : la chaîne de développement de programmes parallèles reconfigurable sur un module.
SITEF 1989.

[FrRoSi90]Ch. Fraboul, J.Y. Rousselot, P. Siron
Software tools for developing programs on a reconfigurable parallel architecture.
Proc. Workshop PCA, Southampton, Juillet 1990.

[FrRoSi91]Ch. Fraboul, J.Y. Rousselot, P. Siron
Software tools for developing programs on a reconfigurable parallel architecture.
Conférence invitée. EUROCOURSE. Computing with Parallel Architectures. Ispra, Décembre 1990.
Ouvrage collectif édité chez Kluwer Academic Publishers.

[FrRoSi91b]Ch. Fraboul, J.Y. Rousselot, P. Siron
Designing and programming a reconfigurable parallel architecture.
Proc. Workshop PCA, Bonn, Mai 1991.

[DaFrRoSi91]V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
Définition d'une architecture modulaire et reconfigurable.
Recueil des communications 3ème symposium sur les Architectures nouvelles de machines, Palaiseau, Juin 1991.

[FrSi91]Ch. Fraboul, P. Siron
Conception et programmation d'une architecture parallèle reconfigurable.
SITEF 1991.

[DaFrRoSi92]V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR : présentation de l'architecture et de la chaîne de développement.
Proc. 4èmes rencontres du parallélisme, Villeneuve d'Ascq, Mars 1992.

[DaFrRoSi92b]V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR: exploiting reconfiguration capabilities of a modular parallel architecture.
Proc. PCA Workshop EWPC92, Sitjes, Mars 1992.

[DaFrRoSi92c]V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
Designing and programming a reconfigurable parallel architecture.
Proc. Transputing'92, Arc et Senans, Mai 1992.

[DaFrRoSi92d]V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
MODULOR : étude et réalisation d'une architecture parallèle modulaire et reconfigurable.
Rencontres Recherche en Informatique et Industrie (journées PRC), Toulouse, Juillet 1992.

[DaFrRoSi92e] V. David, Ch. Fraboul, J.Y. Rousselot, P. Siron
Partitioning and mapping communication graphs on a modular reconfigurable parallel architecture.
Proc. CONPAR'92, LNCS 634, Septembre 1992.

7.2.3 Architecture de systèmes distribués, temps réel et tolérants aux pannes

[FrSi93] Ch. Fraboul, P. Siron
Un environnement de programmation d'applications distribuées et tolérantes aux pannes sur une architecture parallèle reconfigurable.
AGARD Avionics Panel Meeting, Mai 1993.

[FrSi93b]Ch. Fraboul, P. Siron
Architecture parallèle reconfigurable tolérante aux pannes.
Journées modèles s'exécution et architectures paralléles - Nouvelles technologies pour l'architecture - PRC GDR AMN, Rennes, Décembre 1993.

[FrSi94]Ch. Fraboul, P. Siron
Techniques de reconfiguration dynamique appliquées aux systèmes embarqués.
Journée thématique Temps réel et sûreté de fonctionnement dans les applications de la Défense, Mars 1994.

[PaSeSiWi95] F. Pagani, C. Seguin, P. Siron, V. Wiels
Verification experiments on a large fault-tolerant distributed system.
Proc. 2nd AMAST Workshop on real-time systems,Bordeaux, Juin 1995.

7.3 Sécurité des systèmes répartis

[SiAu94] P. Siron, B. D'Ausbourg
A Secure Medium Access Control protocol: security versus performances.
Proc. ESORICS 94, LNCS 875, Brighton, Novembre 1994.

[Sir95] P. Siron
Une plate-forme pour le développement d'applications distribuées et sûres avec le système Chorus.
Atelier CNES - Techniques et technologies des segments sols informatiques, Toulouse, Décembre 1995.

[Sir98b] P. Siron
Simulation distribuée et sécurité.
Journées CELAR/CASSI "Internet", Rennes, 29-30 Janvier 1998.

[BiCaSiZa98] P. Bieber, J. Cazin, P. Siron, G. Zanon
Security Extensions to ONERA HLA RTI Prototype.
1998 Fall Simulation Interoperability Workshop, Orlando, September 13-18, 1998.

[BiSi99] P. Bieber, P. Siron
Design and implementation of a distributed interactive simulation security architecture.
Proc. of the 3rd IEEE Int. Workshop on Distributed Interactive Simulation and Real-Time Applications, Washington, October 22-23, 1999.

[BiSi00] P. Bieber, P. Siron
Security Architectures for COTS based Distributed Systems.
Proc. OTAN-IST Panel Symposium on New information processing techniques for military systems. Istanbul, 9-10 Octobre 2000.

[BiRaSi00] P. Bieber, D. Raujol, P. Siron
Security Architectures for Federated Cooperative Information Systems.
Proc. ACSAC'2000. 16th Annual Computer Security Applications Conference. New Orleans, December 11-15, 2000.

7.4 Simulation distribuée

[Sir98a] P. Siron
Design and Implementation of a HLA RTI Prototype at ONERA.
1998 Fall Simulation Interoperability Workshop, Orlando, September 13-18, 1998.

[Sir99] P. Siron
Implementation of an HLA Distributed Simulation Prototype.
Scientific and Technical Activities, ONERA, 1998.

[AdSi00] M. Adelantado, P. Siron
Air-Ground Combat Simulation through the ONERA HLA Run-Time Infrastructure.
Advanced Simulation Technologies Conference. Military, Government & Aerospace Simulation Symposium. Washington, April 16-20, 2000.

[AdBoSi00] M. Adelantado, S. Bonnet, P. Siron
Multiresolution Modeling and Simulation with the High Level Architecture.
Proc. ESS'2000. 12th European Simulation Symposium. Hamburg, September 28-30, 2000.

[AdSi01] M. Adelantado, P. Siron
Multiresolution Modeling and Simulation of an Air-Ground Combat Application.
2001 Spring Simulation Interoperability Workshop, Orlando, March 25-30, 2001.
3 présentations aux forums VWS (Vehicle/Weapon Systems), IMPL (Federation Implementers) et RTI&C (Run-Time Infrastructure & Communications).

[Sir01] P. Siron
Simulation Distribuée et Standard HLA : Infrastructures d'exécution, Applications et Problèmes.
Présentation invitée Journées FAC'2001. Formalisation des Activités Concurrentes.
Toulouse, 25-26 Avril 2001.

[AuBuSi02a] B. d'Ausbourg, J.-L. Bussenot, P. Siron
Improving Distributed Simulation Performance.
Scientifical & Technical Activities 2001, ONERA, June 2002.

[BrSi02] B. Bréholée, P. Siron
CERTI: Evolutions of the ONERA RTI Prototype.
2002 Fall Simulation Interoperability Workshop, Orlando, September 8-13, 2002.

[AuBuSi02b] B. d'Ausbourg, J.-L. Bussenot, P. Siron
PERFOSIM: a Performance Evaluation Tool for HLA Distributed Simulations.
Proc. of the 6th IEEE Int. Workshop on Distributed Interactive Simulation and Real-Time Applications, Fort Worth, October 11-13, 2002.

8 Stages encadrés

8.1 Les architectures parallèles

8.1.1 Le projet Multi-AP

[Bou83]S. Bouvier
Simulation et évaluation d'architectures multi-"array processors".
Spécialisation Informatique, ENSAE, Juillet 1983.

8.1.2 Le projet Isis

[Bon88]F. Boniol
Études d'outils d'aide à la mise au point, de mesure et de suivi de programme dans un environnement paralléle.
Spécialisation Informatique et DEA, ENSAE, Juin 1988.

8.2 Les architectures distribuées

8.2.1 Le projet Milord

[Thi89]X. Thibault
Etude et évaluation de systèmes de communication optique dans les multiprocesseurs à haut degré de parallélisme.
Thèse de docteur de l'ENSAE, Octobre 1989.

8.2.2 Le projet Modulor

[Que88] P. Quemard
Programmation parallèle en Occam.
UPS Toulouse, Juillet 1988.

[Fou89]M. Fourneaux
Développement d'applications sur une architecture dynamiquement reconfigurable à base de Transputers.
ENSEEIHT, Juin 1989.

[Rak89]G. Rakotondrafara
Étude des problèmes de communication dans une architecture reconfigurable dynamiquement.
INSA et DEA Toulouse, Septembre 1989.

[AdRe90]F. Adam, J.C. Reculeau
Parallélisation, sur une architecture reconfigurable, d'algorithmes d'optimisation de systèmes de transport.
Mastère Informatique ENSAE, Juin 1990.

[Ona90]J. Onandia
Développement et évaluation d'algorithmes sur une architecture parallèle reconfigurable.
UPS Toulouse, DESS, Juin 1990.

[Bai90]F.X. Baillet
Outils de gestion de la reconfiguration pour machine Supernode.
DEA Informatique, Juin 1990.

[NoRo91]J.-F. Nouveau, L. Rodet
Hash-code et architecture parallèle.
Projet d'Initiation à la Recherche, ENSAE, Mars 1991.

[RiBo91]D. Ribayrol, D. Boumard
Projet MODULOR, conception d'un prototype.
INSA Toulouse, Juin 1991.

[Hen91]A. Hennemann
Évaluation d'une architecture parallèle reconfigurable dans un système informatique embarqué.
Mastère Informatique ENSAE, Juin 1991.

[Goi91]S. Goiffon
Système d'exploitation distribué pour réseau de processeurs.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1991.

[Cho92]G. Chopard
Réalisation d'un interface graphique pour le développement d'applications parallèles reconfigurables.
CNAM-IIE, Septembre 1992.

[Lan92]M. Lanne
Parallélisation d'algorithmes de renforcement de cohérence d'arc pour des problèmes de satisfaction de contraintes, sur une architecture parallèle reconfigurable à mémoire distribuée.

Mastère Informatique ENSAE, Septembre 1992.

[Fui93]E. Fuilla-Weishaupt
Programmation logique sur une architecture parallèle à mémoire distribuée.
Spécialisation Informatique ENSAE, Juin 1993.

[Cho93]G. Chopard
Analyse, maquettage et validation d'une chaîne de développement de programmes parallèles reconfigurables.
CNAM-IIE, Septembre 1993.

[Dav93]V. David
Algorithme parallèle sur les arbres de décision et raisonnement en temps contraint. Etude et application au minimax.
Thèse de docteur de l'ENSAE, Novembre 1993.

8.2.3 Architecture de systèmes distribués, temps réel et tolérants aux pannes

[Bas92]B. Bassil
Étude d'une architecture parallèle reconfigurable tolérante aux pannes.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1992.

8.3 Sécurité des systèmes répartis

[Mir95]E. Miravète
Évaluation de nouveaux systèmes d'exploitation distribués : sécurité et fiabilité.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1995.

[Fab95]J.F. Fabre
Étude en vue d'une sécurisation du système Chorus.
ENSAE, Juin 1995.

[Fab96]J.F. Fabre
Sécurité informatique et coopération de systèmes : état de l'art et synthèse.
DEA Informatique ENSAE, Juin 1996.

[Das97]J.S. Dasse
Développement d'une plate-forme de démonstration et d'expérimentation en sécurité multi-niveau.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1997.

[Pes98]Ch. Pesard
Services de protection pour des applications distribuées sur Internet.
ENSAE, Septembre 1998.

[Piq99]S. Piquereau
Développement d'une plate-forme de démonstration et d'expérimentation en sécurité multi-niveau (version 2).
ENSEEIHT, Juin 1999.

[Lou99]F. Loudet
Services de protection pour des applications agents distribuées.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1999.

[Gau99]T. Gaudemet
Propositions pour un réseau local Multi Level Security.
PIR, ENSAE, Juin 1999.

[Cor00]K. Corollou
Développement d'une plate-forme de démonstration et d'expérimentation en sécurité multi-niveau.
INSA-GII, Toulouse, Juin 2000.

[Rau00]D. Raujol
Architecture de sécurité pour objets distribués.
Diplome d'ingénieur CNAM, Décembre 2000.

8.4 Simulation distribuée

[Des97]P. Desseaux
Prototypage d'un système de simulation distribuée compatible avec HLA.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1997.

[Har98]A. Harzi
Simulation distribuée, High Level Architecture, Optimisations et applications.
UPS Toulouse, DESS Informatique Fondamentale, Juin 1998.

[Lem00]S. Lemanceau
Implementation and evaluation of a simulator interconnection based on HLA standard.

ENSEEIHT, Juin 2000.

[TLe00]Th. Lemaire
Extension d'une application de simulation distribuée.
Stage ouvrier ENSAE, Été 2000.

[Ven00]S. Venturi
Evaluation d'une plate-forme de simulation distribuée.
INSA-GII, Toulouse, Juin 2000.
DEA Systèmes Informatiques, Toulouse, Septembre 2000.

[Pru01]S. Prunet
Extension d'une plate-forme de simulation distribuée.
UPS Toulouse, DESS Informatique Fondamentale, Juin 2001.

[Dou01]R. Douyère
Contribution à l'évaluation des performances de simulations distribuées.
Rapport de stage technique, ESIEA, Juillet 2001.

[Fay01]F. Fayet
Extension d'une plate-forme de simulation distribuée.
ENSEEIHT, Mastère d'informatique, Septembre 2001.

[Cra02]M. Cramoisan
Simulation distribuée. Connexion d'une simulation HLA avec l'environnement Ptolemy.
DESS CAMSI, UPS Toulouse, Juin 2002.

[Sal02]Ch. El Salloum
Distributed Simulations with HLA. Implementation of a real time simulation.
Stage Erasmus, SUPAERO, Université de Vienne, Juillet 2002.

Glossaire

ANM Architectures Nouvelles de Machines

AP Array Processor

API Application Programming Interface

CAD Centre d'Analyse de la Défense

CERT Centre d'Etudes et de Recherches de Toulouse

CERTI CERT RTI

CNES Centre National d'Etudes Spatiales

CORBA Common Object Request Broker Architecture

COTS Component Off-The-Shelf

CSP Communicating Sequential Processses

CSP Constraint Satisfaction Problem

CVSI Conception et Validation de Systèmes Informatiques

DCSD Département Commande des Systèmes et Dynamique du vol (DERA)

DEMR Département ElectroMagnétique et Radar

DERA Département d'Etudes et de Recherches en Automatique

DERI Département d'Etudes et de Recherches en Informatique

DERO Département d'Etudes et de Recherches en Optique

DGA Délégation Générale de l'Armement

DMSO Defense Modeling & Simulation Office (DoD)

DRET Direction des Recherches et Etudes Techniques (DGA)

DTIM Département de Traitement de l'Information et Modélisation (DERI)

ENSAE Ecole Nationale Supérieure de l'Aéronautique et de l'Espace

ENSEEIHT Ecole Nationale Supérieure d'Electronique, d'Electrotechnique, d'Informatique et d'Hydraulique de Toulouse

EPIC Etablissement Public à caractère Industriel et Commercial

FéRIA Fédération de Recherches en Informatique et Automatique

GSS Generic Security Services

HLA High Level Architecture (pour la simulation distribuée à événements discrets)

INSA Institut National de Sciences Appliquées

IIOP Internet Inter-ORB Protocol

IP Internet Protocol

IPC Inter Processus Communication

IRIT Institut de Recherche en Informatique de Toulouse

LAAS Laboratoire d'Analyse et d'Architecture des Systèmes

MIMD Multiple Instruction Multiple Data stream

MILORD Multiprocesseur Interconnecté par Liaisons Optiques Reconfigurables Dynamiquement

MRT Ministère de la Recherche et de la Technologie

NETSIM Network Simulator

OMG Object Management Group (CORBA, HLA, ...)

ONERA Office National d'Etudes et de Recherches Aérospatiales

ORB Object Request Broker (CORBA)

PC Personal Computer

PCA Parallel Computing Action

PEDALO Protection des agents logiciels

PERFOSIM Performance des Simulations distribuées

PRC Programme de Recherches Coordonnées

PRF Projet de Recherche Fédérateur (ONERA)

RTI Run Time Infrastructure (HLA)

SAE Société des Amis de l'ENSAE et de l'ENSTA

SAIDA Security Assurance In Distributed Applications

SCD Simulation Coopérative Distribuée (FéRIA)

SICODIS Simulation Coopérative Distribuée (PRF)

SIMD Single Instruction Multiple Data stream

SLM Spatial Light Modulator

SMAC Secure Medium Access Control

SMN Sécurité Multi-Niveau

SSI Sécurité des Systèmes d'Information

SSS Sous-Système de Sécurité

STTC Service des Techniques et Technologies Communes (DGA)

TCP Transport Control Protocol

Recueil d’articles

Le premier article reproduit dans ce document "A MIMD supercomputer system for large scale numerical applications" est également le premier article que j'ai écrit et présenté à la conférence IFIP en 1983. Il traite des architectures parallèles de type MIMD et de leur étude par simulation.

Parallèle, simulation sont deux mots clés qui m'ont longtemps poursuivi.

Il existe également une version de cet article est en français.

L'article "Communication par crossbar optique dans une architecture informatique massivement parallèle" est consacré au projet Milord. Il est paru dans la revue de physique appliquée d'octobre 1987.

On imagine la difficulté qu'il y a eu de parler d'architectures informatiques aux spécialistes d'optique, et, inversement, de vulgariser quelques notions d'optique pour les informaticiens, voir les numériciens.

L'article "MODULOR : la chaîne de développement de programmes parallèles reconfigurables" est un des premiers articles sur le projet Modulor. Je l'ai présenté aux Journées AFCET-GROPLAN consacrées au Développement de programmes pour machines parallèles de Chamonix, en mai 1989. Il a été édité dans la revue BIGRE n°66.

Il est représentatif des objectifs que l'on peut viser au début d'un gros projet et je suis satisfait de ne pas avoir beaucoup dévié de la ligne fixée.

L'article "Partitioning and mapping communication graphs on a modular reconfigurable parallel architecture" est un des derniers articles consacrés au projet Modulor. Je l'ai présenté à la conférence internationale CONPAR'91, qui a eu lieu en septembre 1992 à Lyon. Il est paru dans LNCS 694.

L'architecture de Modulor est nettement plus évoluée. Les contraintes de partitionnement et de placement d'une application distribuée à grand nombre de processus sont prises en compte. Le système a été formellement validé d'une part, réalisé d'autre part, et la maquette a permis de tester de nombreuses applications.

L'article "Security Architectures for Federated Cooperative Information Systems" est un exemple de travaux menés dans le domaine de la sécurité des systèmes répartis. Je l'ai présenté à la conférence internationale ACSAC (Annual Computer Security Applications Conference) à la Nouvelle Orléans en décembre 2000.

Une architecture de sécurité y est en particulier présentée, qui repose sur la distribution et répond aux problèmes de la distribution.

L'article "Multiresolution Modeling and Simulation of an Air-Ground Combat Application" est représentatif des travaux menés dans le domaine de la simulation distribuée. Je l'ai présenté à la conférence internationale Simulation Interoperability Workshop, à Orlando en mars 2001, consécutivement dans 3 forums : VWS (Vehicle/Weapon Systems), IMPL (Federation Implementers) et RTI&C (Run-Time Infrastructure & Communications).

Ces présentations multiples sont caractéristiques du fait qu'on étudie simultanément les aspects développement d'applications de simulation distribuée HLA et développement d'infrastructures d'exécution pour la simulation.

A mes parents

A mon épouse

Remerciements

Je voudrais d'abord remercier sincèrement tous les membres du jury : les professeurs William Jalby, Thierry Priol et Bernard Toursel qui ont accepté d'être les rapporteurs de ce travail et la contrainte d'un déplacement supplémentaire, le professeur Patrick Sallé, pour sa confiance et son activité de correspondant auprès de l'INP, les professeurs Christian Fraboul et Bernard Lécussan, également collègues de travail ancien ou en cours, qui m'ont tant encouragé pour cet exercice.

Je tiens à remercier tous mes collègues qui m'ont accompagné au cours de mes études.

Risquer une liste de noms est bien imprudent car je vais oublier certainement des personnes importantes. Que ces personnes me pardonnent.

Martin Adelantado, Bruno d'Ausbourg, Philippe Berger, Pierre Bieber, Frédéric Boniol, Benoit Bréholée, Josette Brial, Jean-Loup Bussenot, Jacques Cazin, Pascal Churoux, Dominique Comte, Patrice Cros, Frédéric Cuppens, Vincent David, Noëlle Desblancs, Gérard Eizenberg, Jack Foisseau, Christian Fraboul, Michel Fracès, Nourredine Hifdi †, René Jacquart, Bernard Lécussan, Jean-Henri Llareus, Christiane Massoutié, Pierre Michel, Christiane Payrau, Jean-Yves Rousselot, Claire Saurel, Christel Seguin, Jean-Claude Syre, Xavier Thibault, Virginie Wiels

Ma fierté est d'avoir travaillé avec un grand nombre de personnes (les étudiants et les stagiaires n'ont pas été cités ci-dessus) dans un bon état d'esprit, et de pouvoir retravailler demain avec eux et avec plaisir au gré des contrats. Malheureusement, ma rédaction n'a pas été assez rapide et des personnes chères nous ont quitté.

Avertissement

Les travaux qui sont présentés par la suite sont des travaux que j'ai dirigés (ou co-dirigés) et réalisés (ou co-réalisés).

La partie direction correspond d'une part à la détermination des travaux scientifiques à réaliser, d'autre part aux tâches administratives d'écritures de propositions de contrat, de négociation et de suivi de ces contrats.

La partie réalisation se manifeste par une production de rapports (production contractuelle) et d'articles, dont les principales références sont mentionnées pour chaque projet, et parfois d'outils ou de maquettes.

Rédiger ces pages présente un piège : s'attribuer une partie des responsabilités ou des travaux d'autres personnes. J'insiste sur le travail en équipe qui a été, le plus souvent, réalisé. Les personnes ayant participé d'une manière ou d'une autre aux projets mentionnés sont les signataires des rapports et des articles correspondants, ainsi que les stagiaires qui ont été intégrés aux études.

Je revendique néanmoins une part primordiale dans tout ce qui suit et je prie mes collègues de m'excuser si je n'ai pas mis dans cet exercice suffisamment en valeur leurs contributions.