

# DOCUMENT DE PRESENTATION DE TRAVAUX

pour obtenir

## **l'Habilitation à diriger des recherches en Informatique**

présenté à

**l'Université Joseph Fourier (Grenoble 1)**

par

**Andrzej Duda**

## **Modélisation, conception et réalisation de systèmes répartis**

Jean-Pierre Verjus	Professeur à l'INPG, Président,
Claude Kaiser	Professeur au CNAM, Rapporteur,
Sacha Krakowiak	Professeur à l'Université Joseph Fourier, Directeur,
Jacques Mossière	Professeur à l'INPG, Examineur,
Brigitte Plateau	Professeur à l'INPG, Rapporteur,
Michel Raynal	Professeur à l'Université de Rennes 1, Rapporteur,



*Je remercie :*

*Jean–Pierre Verjus, pour m’ avoir fait l’honneur de présider le jury,*

*Sacha Krakowiak et Roland Balter, pour m’ avoir accueilli au sein du projet Guide, pour m’ avoir guidé dans mes choix et pour l’aide précieuse qu’ ils m’ ont apporté dans mon travail,*

*Claude Kaiser, pour avoir accepté la tâche ingrate de rapporteur et pour ses commentaires critiques sur le manuscrit,*

*Jacques Mossière, pour ses remarques critiques et pour avoir accepté de participer au jury,*

*Brigitte Plateau, pour avoir accepté de juger mon travail et pour son aide et son encouragement constant,*

*Michel Raynal, qui a aussi accepté d’ être rapporteur,*

*Dave Gifford, qui m’ a accueilli au sein de son équipe et m’ a encouragé pendant mon séjour au MIT,*

*Fabienne Boyer, Emmanuel Lenormand, Stéphane Perret et Daniel Veillard pour avoir apporté des remarques et des corrections au manuscrit,*

*toutes celles et ceux pour leur collaboration sans laquelle ce travail n’ aurait pu aboutir, en particulier, les chercheurs de l’équipe d’ Epsilon : Guy Bernard, Yoram Haddad, Gilbert Harrus et Georges Hebrail ; de l’équipe Guide : Dominique Decouchant, André Freyssinet, Michel Riveill, Cécile Roisin, Xavier Rousset de Pina ; et de l’équipe PSRG : Mark Sheldon, Jim O’Toole, Ron Weiss,*

*tous les membres de l’ Unité Mixte Bull–IMAG Systèmes qui ont contribué à ce que le laboratoire soit un cadre de travail agréable et efficace, en particulier Béatrice Claudio et Joëlle Macé, qui avec tant de gentillesse nous libèrent des problèmes administratifs,*

*tous les membres de l’équipe Opéra pour leur patience et le support de Grif.*

## Résumé :

Ce document présente une synthèse des activités autour d'un thème central : informatique répartie.

Dans la première partie sont présentés les travaux dans le domaine de l'évaluation de performances et les mesures de systèmes répartis. Dans un premier temps, nous étudions des primitives de synchronisation de type *fork-join*. Par la suite, nous analysons les performances des programmes parallèles et nous comparons des stratégies de duplication d'objets. Enfin, nous présentons une méthode d'estimation du temps global pour la mesure de performances des systèmes répartis.

La deuxième partie concerne la conception et la réalisation de systèmes répartis. Nous commençons par spécifier les primitives de communication du projet *Epsilon* et nous poursuivons en présentant le travail effectué dans le cadre du projet *Guide* : la gestion de l'exécution répartie, les mécanismes d'invocation d'objets à distance, les protocoles de communication, la gestion des groupes d'objets et l'administration de sites.

La troisième partie est consacrée aux aspects inhérents aux autoroutes de l'information du futur : la recherche de l'information sur grands réseaux et la représentation des données multimédia.

En conclusion, nous présentons un projet de recherche s'inscrivant dans le domaine du multimédia réparti qui concerne le développement de l'infrastructure de l'information pour les réseaux futurs. Ce projet a pour l'objectif la conception et le développement des services communs et des outils pour faciliter la mise en œuvre des applications multimédia distribuées.

# Introduction

Ce mémoire présente une synthèse de mes activités scientifiques depuis la soutenance de ma Thèse de Docteur Ingénieur en octobre 1984.

Cette thèse concluait une étape de recherches dans le domaine de l'évaluation des performances et portait sur l'analyse des états transitoires dans les systèmes informatiques. Elle a été préparée sous la direction du Professeur Erol Gelenbe et soutenue à l'Université de Paris-Sud à Orsay.

Le présent mémoire couvre la période d'octobre 1984 à janvier 1994 et décrit mes activités au sein de trois équipes de recherche :

- *Epsilon* qui faisait partie du laboratoire ISEM à l'Université de Paris-Sud à Orsay où j'ai travaillé sur l'évaluation de performances et les mesures de systèmes répartis,
- *Guide* qui faisait partie du laboratoire Bull-IMAG Systèmes à Grenoble où j'ai travaillé sur la conception et la réalisation d'un système réparti à objets,
- *Programming Systems Research* qui fait partie du Laboratory for Computer Science au MIT à Cambridge où j'ai travaillé sur deux aspects des autoroutes de l'information : la recherche de l'information sur réseaux et la représentation des données multimédia.

Une notice individuelle est jointe à ce mémoire. Elle contient un curriculum vitae, une présentation de mes activités de recherche, d'enseignement et d'encadrement ainsi qu'une liste de publications.

Mes activités de recherche depuis 1984 se sont concentrées autour du thème des systèmes répartis. La difficulté de présentation de ces activités vient du fait que j'ai travaillé sur ce thème commun, mais du point de vue de trois domaines différents : d'abord l'évaluation des performances, ensuite la conception des systèmes à objets et enfin l'accès à l'information répartie. Donc, malgré un dénominateur commun, mes activités de recherches se situent dans ces trois domaines qui possèdent leurs propres méthodologies et techniques de recherches.

Les systèmes répartis ont commencé à voir le jour dans les années 80 avec l'apparition des premiers réseaux locaux et des micro-ordinateurs. Les réseaux locaux ont offert un moyen de communication rapide (par exemple le réseau *Ethernet* à 10M-bits/s) qui facilitait la décentralisation de systèmes informatiques. Il est devenu possible d'accéder aux fichiers distants situés sur une autre machine, de lancer l'exécution d'un programme à distance et de communiquer avec d'autres utilisateurs. La communication rapide entre des stations de travail a également rendu possible un traitement en parallèle où des parties d'un même programme s'exécutent sur des machines différentes.

A cette époque, la construction de systèmes répartis s'est heurtée à deux grands problèmes : la complexité et les performances. Par rapport aux systèmes centralisés, un système réparti présente un niveau de complexité plus élevé. En outre, à cause de la décentralisation de leurs éléments, les systèmes répartis avaient des caractéristiques nouvelles comme par exemple l'absence de mémoire commune, l'absence d'un état global et des délais importants de communication. La problématique des systèmes répartis a suscité un intérêt de recherche énorme qui a donné lieu à un nouveau domaine : celui de l'algorithmique distribuée. Ce domaine a pour but de trouver des algorithmes et des mécanismes de base qui permettent la construction méthodique des systèmes et des applications répartis malgré leur complexité et leurs nouvelles caractéristiques.

Quant aux performances des systèmes répartis, celles-ci dépendent de multiples facteurs et au début elles ont été décevantes. La boutade de Tanenbaum illustre bien la difficulté de construire un système réparti performant :

*"Building a 16-node distributed system that has total computing power about equal to a single-node system is surprisingly easy"*

Pour améliorer les performances d'un système, il est nécessaire de pouvoir évaluer celles-ci. L'évaluation de performances d'un système en cours de conception permet de prédire son fonctionnement futur, tandis que l'évaluation de performances d'un système existant permet de déterminer les goulots d'étranglement et, une fois les modifications faites sur les points faibles, de quantifier le gain obtenu. Pour évaluer

les performances d'un système, on utilise différents outils : la modélisation, la simulation et les mesures. Ces outils n'ont pas le même domaine d'application. La modélisation ne peut être mise en œuvre que pour un système dont on sait établir un modèle de comportement et lorsque le modèle est soluble analytiquement (ou par des méthodes numériques). La simulation peut s'appliquer si l'on est capable de décrire le fonctionnement du système, et les mesures ne peuvent être faites que sur des systèmes existants. Lorsque plusieurs de ces méthodes peuvent être utilisées, elles sont bien sûr complémentaires : les mesures permettent de valider un modèle de simulation ou un modèle analytique, et la simulation permet de valider un modèle analytique. De plus, les différentes méthodes se nourrissent mutuellement : ainsi, un modèle analytique grossier pourra mettre en évidence les quantités critiques à mesurer, et les résultats de mesure pourront permettre de raffiner le modèle.

Pour aborder les problèmes de performances des systèmes répartis nous nous sommes intéressés aux modèles quantitatifs du parallélisme et de la répartition ainsi qu'aux techniques de mesures. Les modèles quantitatifs du parallélisme présentés dans le Chapitre I permettent d'analyser le comportement des systèmes répartis et de prévoir l'influence de la synchronisation sur les performances. La simplification inhérente à la modélisation peut parfois provoquer une discordance entre les résultats et les valeurs observées. Il est donc nécessaire de compléter la modélisation par la validation et par les mesures. Mesurer un système réparti est une tâche extrêmement difficile à cause de la décentralisation de ressources et de l'absence d'une horloge globale. Nous présentons dans le Chapitre I une technique originale d'estimation du temps global qui permet d'effectuer des mesures dans un système réparti. Pour expérimenter cette technique et faciliter la programmation des applications réparties, nous avons mis en place un projet de construction d'une maquette de système réparti – *Epsilon*. Cette étape d'expérimentation vient après les études théoriques des modèles de performances qui permettent de comprendre la nature et le comportement de systèmes répartis. La mise en œuvre de prototypes et l'expérimentation est l'ultime étape dans le développement des systèmes, la seule qui valide les choix de conception et les stratégies employées en vraie grandeur.

La conception et la réalisation d'un prototype de système relève plus d'un art que de la science. Il s'agit de concevoir les abstractions qui reflètent au mieux les besoins des utilisateurs et qui en même temps permettent de fournir les fonctionnalités demandées avec des performances satisfaisantes. Une fois les choix de conception arrêtés, vient ensuite la phase de réalisation qui doit implémenter les abstractions de la manière la plus efficace. Enfin, la phase d'évaluation permet de vérifier la validité des solutions. Dans le projet *Epsilon*, nous nous sommes posé le problème de la conception de primitives de programmation d'applications réparties sur des systèmes hétérogènes comme Unix ou *MS/DOS*. Unix dans sa version *BSD 4.2* a offert un ensemble de primitives de communication basées sur le concept de *socket* et a

permet de programmer des applications en termes de processus communicants par messages. Néanmoins, la programmation était fastidieuse et difficile pour un programmeur d'application. En outre, si l'on voulait utiliser des systèmes hétérogènes, il fallait écrire des programmes spécifiques pour chaque système. *Epsilon* a proposé des primitives de haut niveau qui facilitait la programmation des échanges de messages entre processus d'application et rendait les programmes indépendants du support système. Les primitives sont présentées dans le Chapitre II.

Le projet *Epsilon* n'a été qu'un début de travail de conception et de construction de systèmes répartis qui s'est ensuite prolongé dans le projet *Guide*. Ce projet a pris une direction nouvelle en adoptant un modèle à objets où toutes les abstractions visibles sont des objets. Un objet est une entité autonome qui encapsule ces données et le code qui manipule les données. La notion d'objet est une façon élégante et rigoureuse de constituer une unité de nommage, de répartition et de stockage dans un système réparti. L'encapsulation et les possibilités de désignation uniforme des objets facilitent la gestion de l'information dans le système et offrent une vision simple et la fois puissante au programmeur d'applications. Probablement, l'idée la plus importante du modèle à objets est l'invisibilité de la répartition – la répartition reste cachée du programmeur qui ne sait pas où les objets se trouvent ni où une exécution a lieu. Paradoxalement, ce système se voulant réparti cache la répartition et n'offre que la visibilité du parallélisme, notion déjà familière aux programmeurs d'applications grâce aux systèmes de multiprogrammation.

Dans le projet *Guide*, j'ai travaillé sur les problèmes d'exécution répartie et de communication, ainsi que sur des protocoles de diffusion fiable, des groupes d'objets et sur l'administration des sites. La communication dans *Guide* se faisait à l'aide de l'*invocation d'objet à distance*. C'est un mécanisme de base qui s'apparente à l'appel de procédure à distance (*Remote Procedure Call* – RPC), mais qui ne présente pas les importants problèmes de sémantique et de réalisation dont souffre le RPC. Nous discutons ces problèmes en détail dans le Chapitre II. Notre intérêt pour la communication et l'exécution s'est également tourné vers la définition de protocoles de diffusion et d'invocation multiple d'objets. L'idée est d'étendre la sémantique de l'invocation d'objet à un groupe d'objets qui sont gérés et invoqués comme un tout. Cette invocation multiple a besoin de protocoles de diffusion pour être efficace. Nous avons spécifié un protocole de diffusion pour le support de l'invocation multiple d'objets. La première utilisation de ce concept est l'administration des sites dans une cellule *Guide*. Des objets qui contiennent de l'information sur la configuration de la cellule sont gérés comme des groupes d'objets.

Le projet *Guide* est un exemple d'un système réparti fortement couplé, c'est à dire, un système où les sites sont connectés à un réseau local impliquant des communications rapides. Actuellement, on commence à voir le développement d'un nouveau

type de systèmes répartis, à savoir les systèmes à grande échelle. Il s'agit de réseaux rapides qui connectent de nombreuses machines sur de longues distances. Le facteur d'échelle est important – ces nouveaux réseaux peuvent relier des millions de machines comme c'est le cas actuellement d'Internet. Une autre caractéristique importante de ces réseaux est l'apparition de nouveaux types de données qui intègrent différents média comme des images, du son ou de la vidéo (l'intégration de ces données est désignée sous le terme très médiatisé aujourd'hui de *multimédia*). Jusqu'à maintenant, la plupart des informations traitées par ordinateur étaient textuelles, mais les progrès technologiques apportent des possibilités nouvelles permettant de traiter aussi bien les données multimédia. Dans le futur, une convergence des réseaux de distribution de télévision, des réseaux téléphoniques et des réseaux de transmission de données se dessine en donnant ce qu'on appelle les *autoroutes de l'information*. Ces autoroutes vont véhiculer des données multimedia et permettront de supporter des applications nouvelles comme la vidéo à la demande, des vidéo-conférences, le télé-enseignement, le commerce ou les bibliothèques électroniques. Mon travail au sein de l'équipe *Programming Systems Research* au MIT concernait deux aspects des autoroutes de l'information du futur, à savoir la recherche de l'information sur grands réseaux et la représentation des données multimédia. Ce travail fait l'objet du Chapitre III. Les propositions des activités futures et les conclusions terminent ce document.



Chapitre I < e v p >

Chapitre II < s r >

Chapitre III < c r >

Chapitre IV < c o n c l u s i o n s >