ECOLE POLYTECHNIQUE
FEDERALE DE LAUSANNE

flash informatique

p/a EPFL - SERVICE INFORMATIQUE CENTRAL – CP 121 – CH 1015 LAUSANNE – TEL. +41-21 693 2211 – FAX +41-21 693 2220

OBJET ET RÉPARTITION
LE GRAND MARIAGE

Rachid Guerraoui, Laboratoire de Systèmes d'Exploitation, Département d'Informatique

L'union entre les concepts d'objet et de répartition semble être l'une des combinaisons gagnantes de la fin des années 1990, comme le fut à la fin des années 1980 la combinaison entre les interfaces graphiques et les bases de données relationnelles qui conduisit au fameux modèle client-serveur. Avec les objets répartis, un pas de plus est franchi: il ne s'agit plus de systèmes locaux et fermés dans lesquels les rôles de client et de serveur sont établis à l'avance, mais de systèmes ouverts dans lesquels les objets, à la fois clients et serveurs, coopèrent dans un monde démocratique à travers Internet. Les efforts déployés par l'industrie pour affiner les technologies orientées objet et les rendre économiquement et techniquement viables sont considérables, et le danger pour des équipes de recherche universitaires est de se retrouver paradoxalement à la traîne. Malheureusement pour les utilisateurs des objets répartis, et pour les sociétés qui en font des bénéfices, mais fort heureusement pour les chercheurs dans le domaine, l'union des concepts d'objet et de répartition, bien que consommée, est semée d'embûches. Des problèmes fondamentaux sont encore à résoudre, loin des batailles stratégiques que se livrent DCOM, CORBA, Java RMI et ONE.

DES MOTS MAGIQUES

Java RMI, ONE, CORBA, et DCOM sont des acronymes dont vous avez probablement entendu parler, qui vous ont peut-être fait peur, ou du moins vous ont un peu dépassé. Vous avez peut-être fait semblant d'en comprendre certains, ou même pire, vous les avez utilisés pour avoir l'air dans le coup.

Ces mots magiques ont deux points communs. Le premier est qu'ils sont tous considérés comme très branchés, et par conséquent sont particulièrement appréciés dans les réunions au sommet sur les nouvelles technologies informatiques. Je me souviens d'un professeur d'informatique qui, au sujet de la manière d'être un bon ingénieur commercial, nous conseillait d'utiliser beaucoup d'acronymes, et de souvent répéter ceux que nos interlocuteurs semblaient ne pas comprendre. Cela, disait-il, permet d'établir un rapport de force en notre faveur, et de faciliter les négociations par la suite.

Le deuxième point commun de ces acronymes est que tous sont les fruits du même mariage, d'un grand

suite en page 6

SOMMAIRE

PROCHAINES PARUTIONS

parution FI	délai FI
9 18.11.97	30.10.97
10 19.12.97	25.11.97

RACCORDEMENT D'ORDINATEURS PRIVÉS SUR EPNET

Yves Despond, SIC

PRINCIPE

Le raccordement direct, sur le site de l'EPFL, au réseau EPNET d'ordinateurs privés d'employés ou d'hôtes temporaires de l'EPFL est possible s'il est réalisé pour des besoins de services, ratifié par le responsable de l'unité et formellement autorisé par le SIC.

Cet article décrit en détail les conditions à respecter et la marche à suivre pour obtenir un tel raccordement.

CONDITIONS

- Le raccordement doit être utilisé exclusivement pour la mission définie par l'EPFL, dans les rapports de services pour les employés ou dans l'invitation pour les hôtes.
- Les ordinateurs privés raccordés à EPNET n'ont, sauf exception, pas le droit d'utiliser des logiciels sous licence appartenant à l'EPFL. Les licences financées par l'OCFIM doivent en effet être utilisées exclusivement sur des ordinateurs acquis par l'intermédiaire de l'OCFIM. Les exceptions sont définies par le contrat qui lie l'EPFL au fournisseur de chaque logiciel et sont documentées sur le serveur Web du SIC ainsi que dans le logiciel lui-même. Les directives édictées par la Direction de l'Ecole le 9 décembre 1996 définissent les responsabilités des collaborateurs de l'EPFL pour ce type d'utilisation illégale.
- Le raccordement sera supprimé et l'autorisation révoquée en cas de non respect des présentes conditions.
- L'autorisation de raccorder un ordinateur privé au réseau EPNET n'implique pas d'office la mise à disposition d'une prise d'accès pour cet équipement. Une prise sera mise à disposition gratuitement si cela est possible sans frais ou travaux importants. Dans le cas contraire, le SIC pourra, à la demande de l'unité concernée, effectuer les installations nécessaires. Les frais seront mis à la charge de l'unité qui peut demander une contribution financière au bénéficiaire.
- Le nom IP de la machine doit contenir le texte *priv* afin de permettre aux membres du service informatique central et de ceux des départements d'identifier ces ordinateurs et d'adapter leurs prestations en conséquence. Le nom se composera de l'acronyme de l'unité suivi de *priv* et terminé par un numéro d'ordre. Soit, par exemple, *sicpriv1*.
- L'adresse IP de la machine sera attribuée dans le *subnet* de l'unité concernée.
- Le trafic réseau avec Switch sera attribué à l'unité qui a fait la demande de raccordement.
- Le SIC fournit un câble RJ45 permettant le raccordement de l'ordinateur à une prise du réseau EPNET. Aucun autre matériel (interface réseau…) ou logiciel n'est fourni. Les informations nécessaires à la configuration des ordinateurs supportés à l'EPFL sont disponibles sur le Web. Le SIC ne fournit pas d'aide pour la configuration.

MARCHE À SUIVRE

- Le demandeur et le responsable de l'unité remplissent le formulaire: *Autorisation de raccordement d'ordinateurs privés au réseau sur le site de l'EPFL* et l'envoient au SIC, Section Téléinformatique. Ce formulaire est disponible à la réception du SIC.
- Le SIC étudie la demande et autorise le raccordement si les conditions sont remplies. Un exemplaire signé du formulaire est retourné au demandeur pour l'informer de l'acceptation de sa demande. Un nom et une adresse IP sont attribués. Une prise est mise à disposition si cela est possible sans frais ou travaux importants.
- Si des frais ou travaux importants sont nécessaires, le SIC en informe le demandeur et lui indique leur coût.
- L'unité peut demander au SIC de mettre à disposition une prise. Elle accepte par ce fait d'en assumer les coûts

Pour plus de renseignements, prenez contact avec Yves Despond, tél. 4591, e-mail: Yves.Despond@epfl.ch ou consultez la page Web: stiwww.epfl.ch.■

FLASH INFORMATIQUE

Les articles de ce journal ne reflètent que l'opinion de leurs auteurs. Toute reproduction, même partielle, n'est autorisée qu'avec l'accord de la rédaction et des auteurs.

Rédacteur en chef: Jacqueline Dousson, fi@sic.adm.epfl.ch
Comité de rédaction: Jean-Daniel Bonjour, Jean-Michel Chenais, Milan Crvcanin, Laurent Desimone, Jean-Jacques Dumont, Pierre-André Haldy, Catherine Jean-Pousin, Hervé Le Pezennec, François Roulet, Christophe Salzmann & Jacques Virchaux
Mise en page et graphisme: Appoline Raposo de Barbosa
Impression: Atelier de Reprographie EPFL
Tirage: 4000 exemplaires
Adresse Web: http://sawww.epfl.ch/SIC/SA/publications/
Adresse: SIC-SA EPFL 1015 - Lausanne
Téléphone: 021/693 22 46 & 22 47

ISSN 1420-7192

SUS AUX SPAMMERS

Pierre Collinet, SIC, postmaster@epfl.ch

Internet et les moyens de communication électronique sont des outils sans égal pour celui qui sait bien les utiliser. Bien sûr, c'est avec cet objectif que nous construisons les services du SIC.

Malgré tous nos efforts, si vous utilisez les News ou l'Email, vous avez probablement été surpris(e), voire scandalisé(e), par des messages vous invitant à visualiser *gratuitement* des images pornographiques ou à participer à une opération d'envoi en chaîne pour amasser des milliers de dollars par semaine, et cela sans effort, cela va de soi... Ces messages sont appelés des *spams*.

Les spammers sont ces personnes qui n'hésitent pas à envoyer des messages publicitaires, à des immenses listes d'adresses email. Ces listes d'adresses email (par millions d'adresses) sont constituées illicitement et sont vendues par des sociétés ou des indépendants à des prix cassés parce que le gain n'est pas tant dans le *service* d'envoi que dans les ventes ultérieures.

PROBLÈME 1

En conséquence de cela, les utilisateurs sont assaillis de spams venant de tous les coins du monde. Nous recevons beaucoup de plaintes de personnes exacerbées, à juste titre, par ces vagues de messages publicitaires.

De tels messages sont toujours envoyés de telle façon qu'il est impossible de déterminer avec exactitude qui est l'expéditeur. Dans un premier temps, ils ont caché leur identité en utilisant des programmes d'envoi qui mettent des informations fausses dans les messages, c'est ce qu'on appelle en anglais *forged email addresses*, des adresses email falsifiées.

SOLUTION 1

Notre machine sicmail est à présent configurée pour refuser le trafic de *transit*, c'est-à-dire tout message provenant de l'extérieur du site et en destination d'adresses hors site.

Nous encourageons tous les managers qui n'ont pas encore fait ce pas, de configurer les services sendmail de façon à refuser le trafic de transit. Ils trouveront ci-dessous les références utiles.

PROBLÈME 2

Comme tous les managers du monde se sont mis à essayer de coincer les spammers, qui risquent gros à présent aux USA devant les tribunaux américains, il était plus facile d'utiliser une adresse bien correcte et légitime d'un autre site plutôt que de perdre du temps à essayer de se cacher par tous les noms.

Et c'est ainsi que sicmail a été utilisée comme intermédiaire (serveur SMTP) –et que d'autres machines le sont encore– par ces expéditeurs peu scrupuleux. Les messages parvenaient aux destinataires dans le monde entier, de façon à donner l'impression qu'ils étaient émis de la machine sicmail de l'EPFL. La couverture était idéale.

Bien entendu, nous avons reçu de très nombreuses plaintes provenant du monde entier et des promesses de procès, des injures,... puisque c'était facile de nous identifier. Nous avions servi, tout d'abord sans le savoir, de transmetteur de ces spams et nous étions donc complices sans être coupables.

SOLUTION 2

Nous ne pouvons plus nous permettre de laisser simplement entrer les messages vers les boîtes internes. Il faut à présent traiter le trafic entrant, c'est-à-dire filtrer ces spams en destination des utilisateurs de l'Ecole. C'est une opération des plus délicates car, comme vous l'aurez compris, les adresses d'expéditeur de ces messages changent facilement puisqu'elles sont rarement valides et, quand elles le sont, l'adresse prétendue est celle d'un site innocent. Par conséquent, il n'est quasiment pas possible de tenir à jour des listes d'adresses de spammers. Pour ce faire, nous avons dû observer avec beaucoup de soin les messages de spams et découvrir des critères qui permettent de séparer les messages falsifiés des messages légitimes.

Nous avons construit un filtre spécial, rassemblant les critères déterminés jusqu'à présent. Il a été mis de toute urgence en exploitation, sans nous permettre de faire de réels tests *100% bug free* (100% sans erreur), mais nous l'adaptons tous les jours en fonction de nos observations et des remarques des utilisateurs. Ces critères sont sujets à changement sans préavis puisque certains d'entre eux peuvent être corrigés ou modifiés par les expéditeurs à tout moment.

LA SITUATION D'AUJOURD'HUI

Les opérations de filtrage ont déjà eu des effets non désirés, comme de gêner le transfert de messages avec une annexe en format PostScript et générer des retards d'acheminement assez conséquents (pouvant aller jusqu'à plusieurs heures). Ces inconvénients sont à présent réparés.

Afin d'éviter des pics de trafic et des débordement de capacité de nos machines, dus aux spams nocturnes et du week-end, nous avons parfois dû temporairement couper tout trafic avec des sites bien légitimes, mais cachant de nombreux pollueurs. En exemple, citons *aol.com*, *hotmail.com* et *quantcom.com*.

Avant de conclure par une liste de conseils pratiques, je voudrais profiter de l'occasion pour vous exprimer ma gratitude pour vos informations et suggestions et surtout pour votre compréhension et votre patience pendant cette période difficile. Je tiens à remercier tout spécialement mon collègue Jacques Virchaux pour son aide précieuse, autant morale que technique, dans ces moments difficiles qui semblent parfois se mettre bout à bout.

CONSEILS PRATIQUES

Ils sont pour la plupart des rappels, bien qu'ils nous ont récemment occasionné des soucis de taille:

- Il faut éviter de répondre à ces messages de spams, non seulement afin d'éviter de favoriser ce type de commerce, mais aussi parce que cela ne sert à rien: les adresses d'expéditeur sont toujours volontairement fausses et vos réponses ne les gênent en rien puisqu'ils ne les reçoivent pas, mais occupent inutilement les responsables de services de messagerie.
- Surtout, ne faites PAS de petit programme pour renvoyer toutes les 10 secondes, croyant leur *faire comprendre quelque chose*… ce sont nos machines qui en souffrent et le Postmaster aussi... et donc vous-mêmes aussi puisque le service se dégrade.
- Les pubs par email ne continuent de pleuvoir QUE parce que suffisamment de personnes répondent favorablement à ces pratiques et commandent du matériel ou des services…

CONSEILS AUX ADMINISTRATEURS SYSTÈME

- Si vous utilisez la dernière version de *sendmail*, version 8.8.x, alors, vous pouvez:
  - éviter de recevoir des messages publicitaires non désirés en configurant la règle *check_mail*
  - éviter que votre machine ne soit utilisée par des tiers pour le transit en configurant la règle *check_relay*.

  Vous trouverez plus de détails sous:
  - <*http://www.sendmail.org/*>
  - <*http://www.sendmail.org/antispam.html*>
  - <*http://www.informatik.uni-kiel.de/%7Eca/email/check.html*>
  - <*http://spam.abuse.net/*>

Merci à Claude Diderich pour ces informations dans les News de l'Ecole.

- Profitez de vérifier que votre machine ait au moins un serveur POP de version 2.2. La dernière version (2.3) se trouve sur le serveur de votre ligne de produit habituelle.
- Rassemblez vos utilisateurs autant que possible sur une machine serveur de laboratoire ou de département, surtout s'ils utilisent le protocole POP pour relever leur boîte électronique.
- En conséquence, si vous n'utilisez pas ou plus la fonction email d'une machine, supprimez le processus email en le retirant des opérations de démarrage et en tuant le processus actif. L'idéal serait de demander à l'équipe réseau du SIC la déviation des messages éventuels de cette machine vers une autre machine (par exemple, le serveur du département, qui doit être configuré pour accepter les messages adressés à la précédente machine) ou simplement la suppression du record Mx du DNS. N'oubliez pas de renommer également le programme 'mail' de ces machines afin d'éviter des confusions aux utilisateurs.

ETAPES À VENIR :

Les autres grands sites du monde (universités, grandes entreprises,...) souffrent du même problème que nous. De nombreuses discussions sont ouvertes sur les moyens de régler le trafic publicitaire, hormis les considérations juridiques en cours dans bien des pays.

Afin de concentrer nos efforts et de réagir de façon commune dans les sites suisses, SWITCH propose de servir de point de rencontre pour un travail d'équipe. Une première réunion réservée à ce thème précis aura lieu au début novembre.

L'IETF, l'organisation régissant les standards d'Internet, a constitué un groupe de travail pour standardiser le *secure email*, sur la base de PGP (Pretty Good Privacy Inc.). Les choses pourraient bouger aussi de ce côté là dans les mois à venir.■

LE COIN DES CURIEUX

LES AUTOMATISATIONS DE WORD 97

par Jacqueline Frey, arobasque

Chaque utilisateur de Word a – une fois au moins – pesté contre les petits trucs qui s'activent tout seuls ou par défaut et qui nous imposent du coup une mise en forme ou des options dont nous n'avons absolument pas envie ni même besoin.

Pour plus de clarté, j'ai pris soin de classer scientifiquement ces automatisations en trois catégories distinctes:

- les insupportables
- les tolérables
- les indispensables

Comme d'habitude, les opinions et affirmations émises dans cet article n'engagent que son auteur. N'hésitez pas à faire parvenir vos remarques, expériences et suggestions à frey@arobasque.ch. Je me ferai un plaisir sinon un devoir d'en tenir compte dans mes prochains articles.

Ce mois-ci:

LES INSUPPORTABLES

Les fameux guillemets typographiques

Qui n'a pas envoyé au diable ces guillemets typographiques qui en plus d'être indésirables changent de commande et de boîte de dialogue à chaque nouvelle version de Word.

Solution

menu *Outils*, commande *Correction automatique*, onglet *Lors de la frappe*. Notez que la désactivation de l'option guillemets vous débarrassera par la même occasion des tirets se transformant en tirets cadratins ainsi que des apostrophes ' en ' et des espaces en espaces insécables.

Les majuscules automatiques en début de phrase

Solution pour s'en débarrasser à tout jamais

Menu *Outils*, commande *Correction automatique*, onglet *Correction automatique*.

Le dossier proposé par défaut dans les boîtes de dialogue Enregistrer sous et Ouvrir

Que celui qui sauvegarde ses documents dans le dossier *Mes documents* lève la main ? Merci.

Solution pour changer de dossier par défaut

Menu *Outils*, commande *Options*, onglet *Dossiers par défaut*. Indiquez ensuite sous la catégorie *Documents* quel est le dossier souhaité. Utilisez pour cela le bouton *Changer* et allez chercher votre dossier dans l'arborescence.

La saisie semi-automatique telle que Madame, Monsieur

Habituellement la saisie semi-automatique est pratique. Lorsque vous tapez un mot, Word regarde les 3 premières lettres saisies et propose dans une bulle d'aide le mot complet (comme pour les jours de la semaine ou les mois). Si cela vous convient, vous appuyez alors sur la touche Enter et Word tape le texte pour vous. Génial! Seulement voilà, il vous est certainement arrivé de saisir, en début de lettre, le mot Madame, généralement suivi d'un retour marge puisqu'on souhaite taper ensuite le 1er paragraphe de la lettre. Word place alors automatiquement à la suite de Madame, le mot Monsieur.

Solution pour désactiver la saisie semi-automatique

Menu *Outils*, commande *Correction automatique*, onglet *Insertion automatique*. Désactivez la case à cocher *Afficher le conseil de saisie*. Malheureusement la correction des mois par exemple ne se fera plus.

Si c'est une seule insertion automatique qui vous agace, supprimez-la plutôt de la liste.

Les listes à puces

Si vous tapez un tiret ou les signes ***** ou > ou encore la lettre **o** en début de phrase suivi soit d'un espace ou d'une tabulation, Word considère ceci comme un début de liste à puces et attribue un symbole en relation et une mise en retrait tout cela sous vos yeux émerveillés, exemple:

– word
> word
• word

Solution pour se débarrasser des puces automatiques

Menu *Outils*, commande *Correction automatique*, onglet *Lors de la frappe*, option *Liste à puces automatiques*.

La touche Tabulation créant un retrait au lieu d'une tabulation

Voilà une trouvaille ! Ca doit bien faire des lustres que la touche *Tabulation* sert à faire des tabulations, voilà que Word nous attribue à la place une mise en retrait.

Solution

Menu *Outils*, commande *Options*, onglet *Edition*, option *TAB et RETOUR ARRIERE*.

La sélection d'un mot en entier

Solution pour désactiver la sélection complète d'un mot

Menu *Outils*, commande *Options*, onglet *Edition*, option *Lors d'une sélection, sélectionner automatiquement le mot entier*.

Les animations des menus déroulants

Devinette: au début, je suis rigolo, ensuite je deviens lassant et à la longue je deviens franchement pénible; qui suis-je ? Réponse, les menus déroulants animés.

Solution pour s'en débarrasser

Menu *Outils*, commande *Personnaliser*, onglet *Options*, option *Animations de menus*.

L'enregistrement rapide

Bien sûr, ça part d'un bon sentiment, cette idée d'enregistrement rapide. Je rappelle pour les distraits à quoi ça sert en prenant comme base l'aide de Word: en enregistrement rapide, Word enregistre uniquement les modifications apportées au document. Cette opération est plus rapide qu'un enregistrement complet au cours duquel Word enregistre la totalité du document modifié. Activez la case à cocher *Autoriser les enregistrements rapides* lorsque vous travaillez sur un document très volumineux. Cependant, un enregistrement complet utilise moins d'espace disque qu'un enregistrement rapide. Si vous travaillez sur un document via un réseau, désactivez la case à cocher *Autoriser les enregistrements rapides*. Les enregistrements rapides ne peuvent pas être effectués via un réseau!!

Important: effectuez un enregistrement complet dans les cas suivants:

- avant de partager un document avec d'autres utilisateurs.
- lorsque vous avez terminé de modifier un document et que vous l'enregistrez pour la dernière fois. Et là, je pose une question: *qui pense à désactiver cette option quand il a fini de travailler?*
- avant de commencer une tâche exigeant une grande quantité de mémoire, par exemple la recherche de texte ou la compilation d'un index.
- avant de transférer le texte du document vers un autre programme.
- avant de convertir le document dans un autre format de fichier.

Vous ne pourrez pas dire qu'on ne vous avait pas averti?!?. Je résume simplement: l'enregistrement rapide c'est bien, mais ne l'utilisez pas sinon vous allez avoir plein d'embrouilles.

Solution pour désactiver l'enregistrement rapide

Menu *Outils*, commande *Personnaliser*, onglet *Enregistrement*, option *Autoriser les enregistrements rapides*.

Remarque: Si vous activez la case à cocher *Toujours créer une copie de sauvegarde* située dans le même onglet, Word désactive la case à cocher *Autoriser les enregistrements rapides*, car les copies de sauvegarde peuvent uniquement être créées à partir d'enregistrements complets. ■

suite de la première page

mariage, d'un mariage comme on en voit rarement. Il s'agit du mariage de deux technologies, presque aussi vieilles que l'informatique, mais chacune particulièrement en vogue ces derniers temps: *la programmation par objets* et *les systèmes répartis*. On peut se demander si un tel mariage n'est pas seulement un mariage de raison, permettant aux acteurs industriels développant chacune des deux technologies d'étendre un peu plus leur champ d'action. Un peu comme lorsque l'on apprend que Michael Jackson épouse Lisa Marie Presley, et que l'on se demande si le premier ne veut tout simplement pas se faire adopter par un public nostalgique du vrai rock et plus regardant sur certaines mœurs, et si la seconde n'aimerait pas simplement entrer par la grande porte dans le monde du show-business.

Il y a très probablement du vrai dans un scénario arrangé, mais au-delà du mariage de raison, il y a une complémentarité indéniable entre les deux partenaires, en tout cas en ce qui concerne les concepts d'*objet* et de *répartition*.

IL ÉTAIT UNE FOIS UN LANGAGE DE PROGRAMMATION

Le premier langage de programmation par objets, Simula, a été défini en 1965 par Ole-Johan Dahl et Kristen Nygaard en Norvège. Il avait pour objectif les applications de simulation, mais ses concepteurs réalisèrent que les concepts sous-jacents du langage pouvaient s'appliquer à d'autres types d'applications. Ils développèrent alors le successeur de Simula, Simula 67, comme une extension du langage Algol 60, en y rajoutant les concepts d'*encapsulation* et d'*héritage*, à travers les mécanismes de *classes* et de *sous-classes*. Le concept d'encapsulation permettant de réunir des données et des procédures dans une même entité antropomorphique, appelée objet, et de séparer les interfaces des mises en œuvre, s'avéra particulièrement utile pour concevoir les composants d'un programme de manière modulaire, et de s'abstraire des détails de leur mise en œuvre. Ce concept inspira C.A.R. Hoare en 1972 dans la construction de *types abstraits de données*, puis un peu plus tard en 1977, Barbara Liskov et Alain Snyder dans la conception du langage *CLU*. Par ailleurs, le concept d'héritage, permettant d'automatiser le procédé de réutilisation du code par spécialisation, s'avéra particulièrement utile pour faciliter le prototypage et rendre plus lisibles les programmes. La puissance de ce concept fut démontrée à travers le langage Smalltalk, développé par Alain Kay et Adèle Goldberg au début des années 80 chez Xerox, à travers la bibliothèque des composants de l'interface graphique de développement. Pour la petite histoire, on raconte que des ingénieurs de chez Apple, alors qu'ils développaient l'interface graphique du système Mac OS, rendirent souvent visite à leurs voisins de Xerox. On sait par la suite que le fameux Windows95 s'inspira quelque peu (beaucoup) de cette interface graphique.

La dénomination *langage de programmation orienté objets*, ou plus simplement *langages à objets*, fut attribuée aux langages, tels que Simula, Smalltalk et Eiffel, qui offraient des mécanismes de construction de classes et de sous-classes. Pendant longtemps, ces langages furent considérés comme des jouets pour les universitaires, ou au mieux comme des outils de composition de fenêtres en couleur. La raison invoquée était la lenteur d'exécution des programmes. Le succès industriel de C++, un peu plus pragmatique et plus proche des machines que ses prédécesseurs, mit fin à ce tabou. On connait la suite de l'histoire avec un fameux café[1].

LE GRAND MARIAGE

DÉJÀ À L'ÉPOQUE

Le premier langage à objets, Simula 67, permettait déjà de programmer des applications réparties (du moins logiquement). Grâce à la notion de *coroutine* de Simula, le programmeur pouvait écrire des activités indépendantes, et les faire exécuter de manière concurrente. Quoi de plus naturel en fait puisque, d'une part Simula était destiné à des applications de simulation, en particulier de processus industriels où certaines activités se déroulent en parallèle, et d'autre part, le concept d'objet était destiné à modéliser des entités du monde réel, souvent autonomes. Du fait probablement du poids culturel de la programmation séquentielle, et des limitations technologiques de la vitesse de communication en réseau, l'aspect *répartition* des langages à objets fut mis en veilleuse. Pas complétement néanmoins, car dans les années 80, au MIT par exemple, Carl Hewit et Gul Agha s'amusaient avec des objets actifs pour modéliser des pièces de théâtres (les fameux *acteurs*), et Barbara Liskov proposait une extension répartie du langage CLU, appelée ARGUS, pour permettre à des objets de communiquer à travers des machines distantes (les fameux *objets gardiens*). Un peu plus à l'Ouest, à l'Université de Washington, Andrew Black proposait un système, baptisé Emerald, qui permettait de rendre mobile des objets en les faisant voyager d'une machine à une autre.

TOUT S'EST ENSUITE PASSÉ TRÈS VITE

Depuis, la vitesse de communication entre machines est devenue de plus en plus rapide grâce aux progrès technologiques des réseaux. Par ailleurs, l'intérêt de la répartition est devenu fondamental. Des applications réparties, telles que le travail coopératif et le commerce électronique, sont devenues à la mode. De plus, on s'est rendu compte que la répartition permet de tolérer les défaillances, de répartir les charges et de partager les ressources –l'un des programmes répartis les plus utilisés est celui qui permet le partage d'une imprimante par plusieurs utilisateurs–. Bref, on s'est rendu compte que la répartition permet de laver encore plus blanc.

La première phase dans la création d'architectures réparties a consisté à permettre à des machines, appelées *serveurs*, d'accomplir des tâches spécifiques pour le compte d'autres machines, aux capacités plus restreintes, appelées *clients*. Ce type d'architecture était particulièrement adapté à des réseaux locaux, regroupant un nombre limité de machines. La seconde phase, plus ambitieuse, a consisté à permettre à des machines de coopérer dans un monde plus démocratique. On parle encore de clients et de serveurs, mais un serveur

[1] Pour ceux qui ne le savent pas (encore), **Java** signifie **café** en argot américain

peut se retrouver lui-même client et vice-versa. L'objectif est ici de concevoir des applications mettant en œuvre plusieurs réseaux locaux, dans une structure ouverte, où des machines peuvent se connecter pendant l'exécution d'une application.

Pour d'une part mieux structurer et faciliter la maintenance des programmes répartis, considérés à juste titre comme particulièrement complexes, et d'autre part mieux modéliser l'autonomie et la communication entre composants répartis, l'union entre les concepts d'*objet* et de *répartition* était devenue inéluctable. Il y a eu deux mariages: un mariage suivant une approche *appliquée*, et un deuxième suivant une approche *intégrée*.

LE MARIAGE PAR L'APPLICATION

Cette approche a consisté à *appliquer* les concepts de la programmation par objets, en tant que tels, pour structurer les systèmes informatiques répartis. Les différents composants de ces systèmes, tels que les processus, les sémaphores, les transactions, etc., sont représentés par des classes spécifiques d'objets. On parle alors de *services répartis*. La programmation reste séquentielle et on procède par extension des *bibliothèques* séquentielles, en modélisant les aspects relatifs à la répartition avec de nouvelles classes. L'objectif est en particulier d'apporter une certaine généricité aux architectures. Le programmeur peut alors appliquer le concept d'encapsulation, pour modifier certains composants du système de manière modulaire, en fonction d'une application ou d'une machine particulière, puis appliquer le concept d'héritage, pour spécialiser certains composants à travers des nouvelles classes. La figure 1 représente des classes modélisant des concepts de répartition tels que les transactions, les transactions emboîtées, les processus, les sémaphores, les verrous ou les moniteurs.

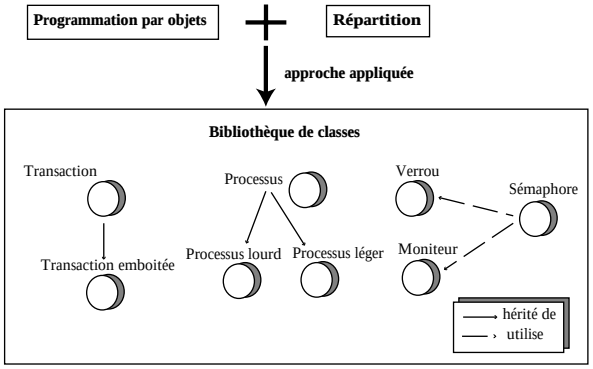


Figure 1 – Approche appliquée

L'assemblage des différentes classes est soit laissé au bon soin du programmeur, soit préalablement réalisé par les concepteurs des classes. Dans le premier cas, on parle de boîte à outils, et le programme contrôle les connexions entre les classes, i.e., les communications entre les objets. Dans le second cas, le programmeur trouve une *charpente* préalablement construite, suivant un *motif* bien rodé (on parle respectivement de *framework* et de *pattern*). Sa tâche se réduit dans ce cas à mettre en œuvre certaines opérations appelées à son insu lors de l'exécution d'un programme réparti (on parle de *call back*), suivant le principe de Hollywood, où l'on répond aux jeunes qui se présentent pour un premier rôle «don't call us, we will call you !».

En résumé, le mariage par application consiste à utiliser un langage et une méthodologie de programmation à objets pour structurer un système réparti. En plus des objets séquentiels d'une application, on trouve des objets permettant de représenter les concepts de la répartition. Cela permet de bien structurer les mécanismes de répartition, mais laisse le programmeur en charge de deux tâches distinctes: d'une part la programmation des objets séquentiels et centralisés de l'application, et d'autre part la gestion de la répartition, également exprimée à l'aide d'objets, mais *pas les mêmes* !

LE MARIAGE PAR L'INTÉGRATION

Au lieu de laisser ces dimensions relativement orthogonales, l'approche *intégrée* vise justement à les fusionner, et à étendre les concepts fondateurs des objets en y intégrant les concepts sous-jacents de la répartition. Plusieurs niveaux d'intégration complémentaires peuvent être considérés.

Invocation d'objets à distance

Du fait qu'un objet contient ses propres données et opérations, il constitue une unité indépendante d'exécution et de répartition (fig. 2).

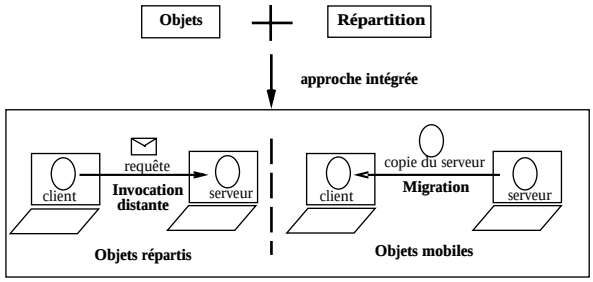


Figure 2 – Approche intégrée

Cela permet de considérer une application répartie comme un ensemble d'objets, chacun situé sur une machine distincte. La métaphore de la communication par «envoi de message», dans Smalltalk par exemple, prend ainsi tout son sens lorsqu'il s'agit d'objets situés sur des machines différentes. On parle d'*invocation d'objet à distance*, et de nombreux systèmes assurent la transparence d'une telle invocation: tout se passe comme si l'objet était local. Pour prendre en compte l'inaccessibilité des objets, le système Argus du MIT dans les années 80 permet par exemple d'associer des exceptions à chaque invocation. Si un objet est situé sur une machine qui est inaccessible, à cause d'une panne du réseau de communication ou du processeur de la machine, l'exception est déclenchée. Cela permet par exemple d'invoquer un autre objet à la place.

Objets mobiles

L'autonomie des objets, en tant que capsules de données et d'opérations associées, facilite la migration éventuelle. Lorsqu'un objet client désire invoquer un objet serveur distant, plutôt qu'une invocation distante, la migration permet de déplacer l'objet serveur (ou une copie de cet objet) chez le client (fig.2). Cela est particulièrement utile lorsque le client désire effectuer plusieurs opérations sur un même serveur.

Objets actifs

En intégrant le concept de processus aux objets, ces derniers deviennent actifs. L'objet est alors doté d'une ressource de calcul, c'est-à-dire doué d'activité propre. Le concept d'objet actif a eu beaucoup de succès dans le domaine de l'intelligence artificielle, car il constitue une fondation assez naturelle pour construire des systèmes multi-agents (les fameuses sociétés de fourmis en intelligence artificielle).

Objets synchronisés

L'association de la synchronisation à la transmission de messages, c'est-à-dire à l'invocation, offre l'avantage de traiter de manière transparente une bonne partie de la synchronisation nécessaire pour assurer une sémantique correcte d'un programme réparti. On peut distinguer trois niveaux de synchronisation correspondant respectivement à la concurrence interne d'un objet, à son interface, et à la coordination entre plusieurs objets. Dans le premier cas (synchronisation intra-objet), on exprime les restrictions en terme d'exclusions entre opérations. Ainsi, par exemple, des lecteurs peuvent lire simultanément un même livre. Par contre, l'accès en écriture par un écrivain exclut tous les autres (écrivains comme lecteurs). Au 2ème niveau (synchronisation comportementale), il se peut qu'un objet ne puisse temporairement traiter certains types de requêtes qui font pourtant partie de son interface. Par exemple, un tampon de taille bornée ne pourra accepter une requête d'insertion tant qu'il est plein. Plutôt que de signaler une erreur, il est en général plus judicieux de laisser en attente une telle requête tant que la condition n'est pas remplie. Enfin, au 3ème niveau (synchronisation inter-objets), on peut désirer assurer une cohérence, non plus seulement individuelle, mais globale entre de multiples objets. Prenons l'exemple d'un transfert de fond entre deux comptes bancaires. En l'occurence, on veut assurer l'atomicité, au sens transactionnel, du transfert, entre les deux objets comptes bancaires. La synchronisation intra-objet ou comportementale n'est alors plus suffisante. On introduit une synchronisation qui met en œuvre les deux objets représentant les comptes bancaires.

ILS EURENT BEAUCOUP D'ENFANTS

Le mariage des concepts d'*objet* et de *répartition* a donné naissance à un nombre impressionnant de langages, de systèmes et de bibliothèques d'objets répartis. A ma connaissance, tous les projets actuels d'architectures réparties sont basés sur le concept d'objet. D'une part, les chercheurs issus de la communauté langage de programmation à objets étendent les environnements de programmation vers des architectures réparties, et d'autre part, les chercheurs issus de la communauté système adoptent le modèle objet pour structurer les concepts de la répartition.

La notion d'encapsulation, fondamentale d'un point de vue conceptuel, devient tout aussi fondamentale d'un point de vue pratique. Si une société informatique vend à travers le réseau un service sous forme d'un objet invoquable (e.g., un serveur Web), il est important pour cette société que les utilisateurs ne puissent pas connaître la mise en œuvre du service (ils pourraient se rendre compte qu'ils ont payé trop cher), et que les fonctionnalités du service soient les mêmes que celles d'une société concurrente, même si les deux utilisent des mises en œuvre différentes, des langages de programmation différents et des systèmes d'exploitation différents.

Bien que certains développements universitaires associent les objets et la répartition suivant une approche particulière (intégrée ou appliquée), la grande majorité des systèmes à objets répartis, et en particulier les produits commerciaux, font une synthèse pragmatique des deux approches. Parmi les rejetons du couple *objet* et *répartition*, les plus connus ont pour noms CORBA, DCOM, Java (Java RMI) ou encore ONE.

CORBA

Depuis 1989, une association internationale appelée l'OMG *(Object Management Group)*, définit la spécification de l'architecture d'un système à objets répartis, appelée CORBA *(Common Object Request Broker Architecture)*. Près de 700 sociétés sont actuellement membres de l'OMG, aussi bien des sociétés informatiques comme HP, Digital, Sun, Microsoft, ou IBM, que des utilisateurs comme Boeing ou Alcatel. A l'inverse d'ODP *(Open Distributed Processing)*, un autre effort de standardisation d'architectures réparties, CORBA a été défini dans un but très pratique, et de nombreuses mises en œuvre de la spécification CORBA sont actuellement sur le marché, incluant SOM de IBM, ObjectBroker de Digital, Orbix de Iona, et Visibroker de Visigenic.

CORBA associe les concepts d'objet et de répartition dans une approche à la fois intégrée et appliquée. L'intégration se fait à travers la notion d'invocation à distance d'objet (l'objet est l'unité de répartition), et l'application se fait à travers les services de répartition qui sont organisés sous forme de bibliothèques de classes.

Les objets CORBA, dont les interfaces sont décrites dans un langage spécifique, nommé IDL *(Interface Description Language)*, sont portables sur différents systèmes d'exploitation, et des objets d'une même application peuvent être écrits dans différents langages de programmation. Avec la spécification CORBA 2.0, et l'adoption du standard de communication IIOP*(Internet Inter-ORB Protocol)*, ces objets peuvent communiquer à partir de différentes mises en œuvre de CORBA. Cela répond à l'un des soucis majeurs de l'informatique de cette décennie, à savoir l'intégration de composants informatiques hétérogènes. Dans CORBA, le modèle objet est appliqué à tous les niveaux d'une application. Et en particulier, tous les services de répartition, comme le nommage, les transactions ou la persistance, sont spécifiés et mis en œuvre sous forme de classes d'objets CORBA. La notion de service CORBA explique, en grande partie, le succès commercial de CORBA, car elle permet à divers développeurs de logiciels de contribuer à la mise en œuvre de CORBA et d'en tirer les bénéfices. Des sociétés comme Oracle ou Ingres peuvent par exemple vendre leur technologie bases de données sous forme d'un service de persistance CORBA, alors qu'une société comme Transarc peut vendre un moniteur transactionnel sous la forme d'un service transactionnel CORBA.

DCOM

Bien que membre de l'OMG, Microsoft a développé son propre standard d'objets répartis, appelé DCOM *(Distributed*

Component Object Model). Le terme standard revêt dans ce contexte un caractère particulier, car il signifie standard pour, et seulement pour, les produits Microsoft.

L'histoire a commencé en 1993 avec OLE 2.0 *(Object Linking and Embedding)*, l'ensemble des mécanismes permettant aux utilisateurs d'un produit Microsoft (e.g., PowerPoint) d'intégrer des composants construits avec un autre produit Microsoft (e.g., des images Graph). Le fondement de OLE était COM *(Component Object Model)*, un standard de compatibilité binaire entre objets, et qui unifie l'architecture des différents services OLE. Contrairement à CORBA, OLE n'a pas été créé à l'origine pour la programmation répartie. Depuis 1996 est apparu DCOM qui désigne les composants Windows qui peuvent être distribués sur différentes plates-formes (Windows) à travers Internet. Ces composants sont basés sur une extension du modèle objet COM aux environnements distribués, appelé DCOM. L'usage initial des composants DCOM était la mise en œuvre de documents composés. Ici, la notion de document remplace la notion d'application dans CORBA. Un document est composé de plusieurs composants répartis (tableaux, textes, graphiques).

On retrouve, comme dans CORBA, l'intégration des concepts d'objet et de répartition, à la fois suivant une approche intégrée et appliquée. L'approche intégrée se matérialise par le fait que les objets sont des unités de répartition qui communiquent en utilisant des invocations à distance, et l'approche appliquée par le fait de développer des services de répartition sous forme de bibliothèques de classes. Néanmoins, à la différence de CORBA, aucun mécanisme ne permet de mettre en œuvre l'héritage de classes. Ce mécanisme est remplacé par l'agrégation (la composition d'objets) et la délégation (un objet délègue à un autre objet ce qu'il ne sait pas faire).

JAVA, JAVA RMI ET JAVA BEANS

Après l'échec du projet initial OAK (premier nom du langage Java), visant à définir un langage de programmation pour la télévision interactive et les équipements de téléphonie, Sun Microsystems propulsa de nouveau Java sur le devant de la scène en automne 1995, et suscita l'enthousiasme qu'on connaît des utilisateurs et des programmeurs d'Internet en pleine expansion. Java élimine quelques aspects critiquables de C++ tels que les pointeurs et les *templates*, et introduit quelques aspects sympathiques de Smalltalk tels que le ramassage de miettes et une forme de manipulation (assez primitive) de classes en tant qu'objets. Les développements de Java et de ses bibliothèques sont désormais adoptés par tous les logiciels visant Internet, y compris par Microsoft qui créa Visual J++, une version de Java destinée aux environnements Windows, sans pour autant renoncer au développement de DCOM, ni à sa participation à l'OMG.

Java intègre les objets et la répartition à travers le mécanisme de migration. C'est le code qui se déplace: il n'y pas, du moins dans les premières versions des bibliothèques, d'invocation d'objet distant. Lorsqu'un fouineur comme Netscape charge une page Web et rencontre dans un code HTML *(Hyper Text Markup Language)* une étiquette désignant une *applet* Java (i.e., un programme Java qui peut se télécharger à travers le Web), il retrouve le code de l'applet et le fait migrer vers le site client (en établissant une connexion TCP/IP). La machine virtuelle Java du fouineur permet ensuite d'exécuter le code Java (en fait le byte code) migré chez le client. Le fouineur efface ensuite le code de la mémoire lorsqu'il quitte la page Web en question.

Plus récemment, JavaSoft, une filiale de Sun Microsystems, développa une bibliothèque de classes, appelée RMI (*Remote Method Invocation*), permettant la communication répartie, puis sous le nom de Java Beans (grains de café Java), un ensemble d'outils d'introspection, de visualisation de programmes répartis, de persistance, et de gestion d'événements. On retrouve, en plus de la possibilité de migration initiale des objets Java, la possibilité d'invocation à distance d'objets ainsi qu'un ensemble de services répartis comme dans CORBA. A la différence de CORBA, RMI présuppose que le client et le serveur sont développés en Java, et que tous deux sont exécutés sur une machine virtuelle Java.

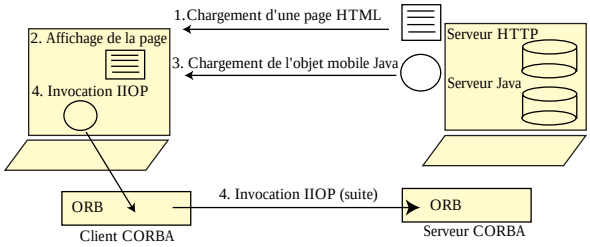


Figure 3 – Migration et invocation à distance sur le Web

ONE

Netscape communications a présenté en 1996 sa propre mise en œuvre de CORBA, en Java, baptisée ONE *(Open Network Environment)*. ONE offre la possibilité à une page Web téléchargée à travers le fouineur Netscape de dialoguer ensuite avec un serveur CORBA. Une page Web peut ainsi contenir des objets CORBA (avec des interfaces IDL) qui peuvent à leur tour utiliser des objets (et des services) CORBA. La figure 3 présente les différentes étapes mises en jeu. Tout d'abord, le navigateur télécharge une page Web (1), l'affiche (2), puis télécharge le code Java de l'applet (3). Ensuite, l'applet est localement exécutée et un objet CORBA est ensuite invoqué à distance (4). Cet objet peut lui même invoquer d'autres objets, etc.

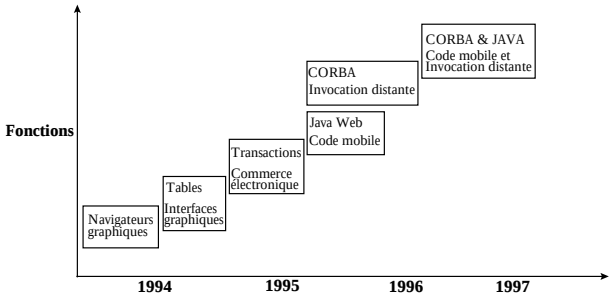


Figure 4 – Les objets sur le Web

L'intérêt de l'approche ONE est le subtil mélange entre les avantages de Java et de CORBA, qui se traduit par la possibilité à la fois de migration des objets, et d'invocation à

distance d'objets et de services (CORBA) hétérogènes. Une application peut ainsi avoir une partie applet Java téléchargeable à travers le Web, des parties serveurs, utilisant des bases de données (e.g., accédant à Oracle à travers CORBA), ainsi que des composants développés dans d'autres langages que Java (oui, il en existe encore !). De plus, le modèle de programmation des objets CORBA est très similaire à la programmation Java RMI. Le code IDL est directement généré à partir du code Java. Par ailleurs, les objets sont référencés par des adresses URL (comme n'importe quel document sur le Web). D'une certaine manière, cela traduit l'évolution des objets répartis sur le Web (fig. 4).

ET ILS VÉCURENT HEUREUX ? NON, ET HEUREUSEMENT D'AILLEURS ...

Maintenant que la messe est dite, que le mariage entre les concepts d'objet et de répartition est consommé, et que les fruits de ce mariage sont l'objet de toutes les convoitises par les grands industriels de l'informatique, on est en droit de se demander, en tant que chercheur universitaire, s'il est vraiment raisonnable d'entreprendre, ou même de continuer, des activités de recherche dans le domaine des objets répartis.

Une première réponse consisterait à dire «mais oui justement, cela permettrait d'avoir des contrats industriels plus facilement, et d'être au courant des nouvelles technologies à enseigner aux étudiants». D'une part, un laboratoire de recherche se retrouverait rapidement dans la situation d'intégrer des technologies sans les maîtriser vraiment, ou d'agir comme sous-contractant ponctuel, moins cher que des sociétés de services informatiques. J'ai du mal à imaginer la qualité des thèses qui s'en suivraient. D'autre part, et sans aller jusqu'à l'attitude extrême de ce grand professeur d'informatique américain qui aime à répéter «je n'enseigne jamais ce que l'industrie réclame, car ce qu'elle réclame, elle l'enseignera elle-même», il ne semble pas très académique de devoir modifier les manuels de cours au gré des changements de stratégies commerciales de Microsoft, IBM ou JavaSoft. Je viens d'apprendre par exemple que Microsoft vient de choisir comme nom pour son architecture répartie l'acronyme DNA (Distributed Inter-Net Applications): *l'ADN de Microsoft!* [1]

Heureusement, et même si beaucoup de pages Web semblent dire le contraire, le mariage entre *objet* et *répartition* n'en est qu'à ses débuts, et le chemin semble encore semé d'embûches. Aussi bien dans l'union par l'application que dans l'union par l'intégration, de nombreux problèmes techniques se profilent. Qui dit problèmes dit sujets potentiels de recherche, et des laboratoires de recherche universitaires peuvent continuer à décemment nourir l'industrie des objets répartis.

À LA RECHERCHE D'ABSTRACTIONS STANDARDS

Comme mentionné précédemment, la première manière d'unir les concepts d'objet et de répartition consiste, suivant une approche *appliquée*, à décomposer un système réparti en bibliothèques de classes.

Bien que des tentatives soient entreprises dans ce sens, il est encore beaucoup trop tôt pour considérer qu'il existe une bibliothèque de classes susceptible de devenir un standard pour la programmation répartie. Dans un contexte séquentiel, il existe effectivement des abstractions fondamentales, telles que le *tableau* ou l'*enregistrement*. Dans un contexte concurrent, le *sémaphore*, est lui aussi devenu une abstraction fondamentale qui fait désormais partie de la plupart des bibliothèques de programmation concurrente. Dans un contexte réparti, aucune abstraction fondamentale ne se dégage vraiment. Cela souligne le fait que le domaine de la programmation répartie est relativement nouveau et assez complexe. Cela est d'autant plus vrai lorsqu'on considère des problèmes de tolérance aux défaillances ou des contraintes temporelles.

Les difficultés d'un tel exercice (i.e., trouver et organiser des abstractions adéquates) résident d'une part dans une bonne compréhension des mécanismes minimaux de la programmation répartie, et d'autre part dans un consensus sur ces mécanismes. Un tel consensus devrait mettre en jeu différentes communautés de chercheurs: langages, répartition, et parfois bases de données (pour des mécanismes transactionnels par exemple).

L'INTÉGRATION NE SE FAIT PAS TOUJOURS SANS DÉGÂT

Certains mécanismes développés par la programmation par objets ont été fondés sur des hypothèses fortes d'informatique traditionnelle, c'est-à-dire séquentialité de l'exécution des programmes et espace mémoire unique. Ils peuvent ainsi atteindre leurs limites quand ils sont transposés directement dans l'univers de la programmation répartie, suivant l'approche applicative, où ils sont fusionnés avec des concepts de la répartition.

Particulièrement significatif est le cas du mécanisme d'héritage par exemple. La première limitation tient à la volonté d'appliquer le concept d'héritage à la spécialisation d'un nouvel aspect des objets: la synchronisation. Il se peut que dans certains cas la définition d'une sous-classe, rajoutant une seule méthode à la classe parente, impose la redéfinition de l'ensemble des spécifications de synchronisation, annulant ainsi le premier bénéfice de l'héritage: la réutilisabilité du code. Par ailleurs, la mise en œuvre de l'héritage dans un système réparti pose le problème de l'accès distant au code des classes parentes, à moins que celles-ci ne soient dupliquées sur toutes les machines, avec les coûts conséquents. De nombreux articles ont été écrits sur le sujet, qui reste néanmoins très ouvert.

La deuxième limitation tient cette fois aux hypothèses fortes des techniques de mise en œuvre des classes, et en particulier des variables de classe, ce qui limite leur transposition immédiate à un univers réparti. Il semble difficile, à moins d'introduire des mécanismes transactionnels compliqués, de garantir que la mise à jour d'une variable de classe soit immédiatement reflétée sur toutes les instances d'une classe, lorsque celles-ci sont situées sur plusieurs machines. Ce problème se pose pour des variables globales en général, mais est accentué dans les langages à objets car des variables de classes sont souvent implicitement utilisées par le programmeur, qui ne se rend pas toujours compte que ce sont des variables globales dont la mise en œuvre est très coû-

[1] Comme me l'a fait remarqué Benoît Garbinato au LSE: *Ça c'est du marketing!*

teuse dans un contexte réparti.

Enfin, la troisième limitation est relative à la notion de duplication, particulièrement utile pour tolérer les défaillances. Si elle est appliquée directement aux objets, la duplication provoque des incohérences d'exécution. Les protocoles de communication définis pour contrôler la duplication des services dans un système réparti considèrent un modèle client/serveur simple. L'application des mêmes protocoles aux objets pose le problème de la duplication des invocations. En effet, un objet agit généralement à la fois comme un client et un serveur. Autrement dit, s'il est dupliqué en tant que serveur, l'objet peut à son tour invoquer d'autres objets en tant que clients. Toutes les copies de l'objet invoqueront alors le même objet plusieurs fois. Le résultat est la duplication inutile des invocations. Cette duplication peut conduire au meilleur des cas à l'inefficacité du système, et au pire des cas à des incohérences. Une même opération (par exemple d'incrémentation) peut ainsi être exécutée plusieurs fois plutôt qu'une.

VERS UNE TROISIÈME APPROCHE COMPLÉMENTAIRE

L'approche appliquée permet d'offrir au programmeur de systèmes des mécanismes ou/et des concepts pour gérer la répartition. Les concepts de généricité et de classe permettent de structurer ces mécanismes et concepts en une véritable palette de mécanismes et de solutions. Cependant, plus encore qu'une palette de solutions, il est judicieux de permettre une extensibilité plus générale, c'est-à-dire d'offrir une méthodologie et des moyens lui permettant d'exprimer et de construire des solutions spécifiques si nécessaire, sans pour cela modifier l'application préalablement développée, c'est-à-dire de manière transparente à l'application.

L'approche intégrée offre cette transparence en fusionnant les concepts d'objet et de répartition. Néanmoins, les systèmes intégrés peuvent fixer trop tôt leurs modèles d'exécution et de communication. Le langage de programmation délègue alors la gestion des ressources, telles que le placement des objets et le séquencement des tâches sur chaque processeur, au système d'exécution sous-jacent. La sémantique du système est sinon pré-cablée, du moins non modifiable au niveau du langage lui-même. Même si le langage offre des facilités de changement des caractéristiques du modèle de calcul, et de gestion des ressources, le programme devient alors moins lisible et surtout moins réutilisable, du fait du mélange du texte du programme même, avec les portions spécifiant le contrôle.

Une troisième approche se dégage pour combler ces lacunes: l'approche *réflexive*. Elle a pour objectif de faire apparaître, au niveau du langage de programmation, diverses caractéristiques de représentation (statiques) et d'exécution (dynamiques) du programme. Ces caractéristiques sont décrites et modifiables par l'intermédiaire d'un *méta-programme*. La *réflexion* est une version de la méta-programmation, où le méta-programme est décrit dans le langage lui-même. Ceci offre ainsi l'avantage d'une approche plus homogène, le même langage étant employé, pour l'écriture des programmes, comme pour leur contrôle. L'approche réflexive est, en quelque sorte, un moyen terme entre l'approche intégrée et l'approche appliquée, car elle permet d'intégrer intimement les (méta-) bibliothèques avec le langage, tout en les séparant du programme. De nombreuses architectures réflexives sont actuellement proposées et évaluées, mais il est trop tôt pour dégager, et valider, une architecture réflexive générale pour la programmation répartie. Un des problèmes est la complexité éventuelle des architectures réflexives, qui est en partie lié aux possibilités accrues de paramétrisation qu'elles offrent. Un autre problème est celui de l'efficacité, du fait des indirections et interprétations supplémentaires que la réflexion peut entraîner. L'évaluation partielle est actuellement proposée comme une des techniques permettant de réduire ces surcoûts. Les notions de *filters* ou de *smart proxies* dans certaines mises en œuvre de CORBA peuvent être vues comme un premier pas d'une approche réflexive pragmatique dans l'industrie.

POUR EN SAVOIR PLUS

Sur les recherches dans le domaine des objets répartis, de bonnes sources d'informations sont les actes des conférences OOPSLA *(International Conference on Object Oriented Programming Systems, Languages, and Applications)*, publiés par l'ACM dans la collection Sigplan, et les actes de la conférence ECOOP *(European Conference on Object Oriented Programming)*, publiés par Springer Verlag dans la collection Lecture Notes in Computer Science. Pour en savoir plus sur les produits commerciaux, il suffit de demander à votre moteur de recherche préféré sur le Web de trouver Java RMI, ONE, CORBA, ou DCOM.■

UTILISATION DU TRACEUR HP A0 ET DE L'IMAGEUR AGFA

Le SIC est équipé d'un traceur HP A0 (http://slwww.epfl.ch/SIC/SL/servdist/HP755CM.html) permettant de réaliser des posters couleurs au format A0 (84 x 118.8 cm), accessible depuis les mondes Mac, PC ou Unix. Victime de son succès, il faut parfois attendre quelques heures avant d'obtenir son poster, et le premier essai n'est pas toujours le bon (choix des couleurs, des fontes…). Nous tenons donc à vous mettre en garde, ne prévoyez pas de faire vos posters la veille d'une conférence! Mettez toutes les chances de votre côté, votre niveau de stress (et le nôtre) y gagnera, en vous donnant quelques jours de délai. Le problème est encore plus crucial avec l'imageur (fabrication de diapos à partir de fichiers Postscript http://www.epfl.ch/SIC/SA/publications/FI95/fi-8-95/imageur.html) où l'utilisateur monopolise l'équipement pendant tout le temps d'impression de ses diapos. Rappel:quand vous mettez une pellicule dans l'imageur, pensez à laisser vos coordonnées, pour que le suivant puisse vous contacter.

Section Assistance

FORMATION

Les cours ci-dessous sont ouverts à tous, membres ou non de l'EPFL. Pour le personnel de l'EPFL, le SIC se charge des frais de cours.

Inscriptions et renseignements (matin uniquement):
Josiane Scalfo, SIC-EPFL, CP 121, 1015 Lausanne
tél.: 021 693 2244 - Fax: 021 693 2220
E-mail: josiane.scalfo@epfl.ch

Pour tout changement, consultez aussi les News, ou le serveur:
http://sawww.epfl.ch/SIC/SA/cours/cours.html

COURS SUR MACINTOSH

Cycle de base complet A + B **(13 demi-jours)**

Cycle de base A «logiciels standards»
Introduction au Macintosh, à ClarisDraw 1.0, Internet (Présentation d'Internet, Intranet • Netscape et navigation sur le Web • Recherche et moteurs de recherche • Bookmarks), Word 6.0, Excel 5.0, FileMaker Pro 3.0.
N° 4170 A (7 demi-jours)
03, 05, 10, 12, 17, 19 & 24.11.97 13h30 - 17h15

Cycle de base B «communication»
Introduction à l'utilisation des réseaux, Internet (Présentation d'Internet et d'Intranet plus poussée • Configuration du produit • FTP - transferts de fichiers • Netscape et navigation sur le Web • Présentation d'HTML), Messagerie & Astuces pratiques du système.
N° 4170 B (6 demi-jours)
26.11, 01, 03, 08, 10 & 15.12.12.97 13h30 - 17h15

BASES DE DONNEES

FileMaker Pro 3.0 avancé (5 demi-jours)
N° 4187 12, 17, 19, 24 & 26.11.97 08h15 - 12h00

METTEUR EN PAGE

PageMaker 6.5 (3 demi-jours)
N° 4201 01, 03 & 09.12.97 08h15 - 12h00
Prérequis: connaissances approfondies de Word et d'un logiciel de dessin!

PRESENTATION

PowerPoint 4.0 avancé, niv. 1 (2 demi-jours)
N° 4194 02 & 04.12.97 08h15 - 12h00

PowerPoint 4.0 avancé, niv. 2 (1 jour)
N° 4195 11.12.97 08h15 - 17h15
Prérequis: connaissances de PowerPoint !

TABLEUR

Excel 5.0 avancé, niv. 1 (3 demi-jours)
N° 4190 27.10.97 08h15 - 17h15
& 29.10.97 08h15 - 12h00

Excel 5.0 avancé, niv. 2 (2 demi-jours)
N° 4191 05.12.97 08h15 - 17h15

Excel 5.0 macros (2 demi-jours)
N° 4192 08 & 10.12.97 08h15 - 12h00

TRAITEMENT DE TEXTE

Mailing (Word - FileMaker) (1 demi-jour)
N° 4198 11.11.97 08h15 - 12h00
Prérequis: connaissances de base de Word et FileMaker Pro!

Word 6.0 avancé (5 demi-jours)
N° 4186 13, 18, 20, 25 & 27.11.97 08h15 - 12h00

Word 6.0 formulaires (1 demi-jour)
N° 4197 10.11.97 08h15 - 12h00

Word 6.0 longs documents (2 demi-jours)
N° 4196 03 & 05.11.97 08h15 - 12h00

Word 6.0 trucs + astuces (1 demi-jour)
N° 4199 07.11.97 08h15 - 12h00

TRAITEMENT D'IMAGE

PhotoShop 4.0 (4 demi-jours)
N° 4202 30, 31.10, 04 & 06.11.97 08h15 - 12h00

WEB

FrontPage (3 demi-jours)
Ce cours est destiné aux personnes qui devront mettre de l'information sur les serveurs de l'Ecole.
• les principes de base de WWW (modèle client-serveur, Internet, hypertexte, URL) • les commandes HTML les plus importantes • les différentes méthodes pour créer/récupérer des textes pour un serveur
N° 4189 14, 21 & 28.11.97 08h15 - 12h00
Prérequis: avoir déjà utilisé un navigateur Web

COURS SUR PC - WINDOWS'95

Cycle de base complet A + B **(13 demi-jours)**

Cycle de base A «logiciels standards»
Introduction à Windows 95, PowerPoint 97, Internet (Présentation d'Internet, Intranet • Browsers et navigation sur le Web • Recherche et moteurs de recherche • Bookmarks), Word 97, Excel 97, FileMaker Pro 3.0.
N° 2669 A (7 demi-jours)
04, 06, 11, 13, 18, 20 & 24.11.97 08h15 - 12h00

Cycle de base B «communication»
Introduction à l'utilisation des réseaux, Internet (Présentation d'Internet et d'Intranet plus poussée • Configuration du produit • FTP - transferts de fichiers • Browsers et navigation sur le Web • Présentation d'HTML), Messagerie & Astuces pratiques de Windows 95.
N° 2669 B (6 demi-jours)
25, 27.11, 02, 04, 09 & 11.12.97 08h15 - 12h00

BASES DE DONNEES

Access 97 introduction (2 demi-jours)
N° 2677 03.11.97 08h15 - 12h00
& 04.11.97 13h30 - 17h15

Access 97 avancé (4 demi-jours)
N° 2678 18, 20, 25 & 27.11.97 13h30 - 17h15

Access 97 programmation (2 demi-jours)
N° 2679 01 & 03.12.97 13h30 - 17h15

DESSIN

Designer 6.0 **(2 demi-jours)**
N° 2666 12.11.97 08h15 - 17h15

LANGAGE DE PROGRAMMATION

VisualBasic 5.0 intro niv. 1 (2 demi-jours)
N° 2680 23.10.97 08h15 - 17h15

VisualBasic 5.0 intro niv. 2 (2 jours)
N° 2681 07 & 14.11.97 08h15 - 17h15

PRESENTATION

PowerPoint 97 avancé, niv.1 (2 demi-jours)
N° 2676 30.10.97 08h15 - 17h15

SYSTEME

Transition de Macintosh à Windows 95 (1 demi-jour)
N° 2684 09.12.97 13h30 - 17h15

Transition de Windows 95 à NT 4.0 (1 demi-jour)
N° 6307 03.11.97 08h15 - 12h00
Voir descriptif ci-après (Cours sur PC - Windows NT 4.0)

TABLEUR

Excel 97 avancé, niv. 1 (3 demi-jours)
N° 2673 01, 03 & 05.12.97 08h15 - 12h00

Excel 97 avancé, niv. 2 (1 jour)
N° 2674 08.12.97 08h15 - 17h15

Excel 97 macros (2 demi-jours)
N° 2675 15.12.97 08h15 - 17h15

TRAITEMENT DE TEXTE

Word 97 avancé (5 demi-jours)
N° 2670 17, 19, 21, 26 & 28.11.97 08h15 - 12h00

Word 97 formulaires (1 demi-jour)
N° 2685 02.12.97 13h30 - 17h15

Word 97 longs documents (2 demi-jours)
N° 2682 11 & 13.11.97 13h30 - 17h15

Word 97 mailing (1 demi-jour)
N° 2687 10.11.97 08h15 - 12h00
Prérequis: connaissances de base de Word.

Word transition 7.0 à 97 (1 demi-jour)
N° 2686 28.10.97 08h15 - 12h00

WEB

FrontPage **(3 demi-jours)**
N° 2671 20, 22 & 27.10.97 08h15 - 12h00

COURS SUR PC - WINDOWS NT 4.0

Windows NT 4.0 Core Technologies (4 jours)
N° 6133 04, 05, 06 & 07.11.97 08h15 - 17h15
Objectifs principaux: Fournir les connaissances permettant d'installer, gérer, configurer, optimiser, intégrer et dépanner Windows NT 4.0 dans un environnement LAN et WAN de base. Les participants seront à même de corriger les problèmes courants de maintenance.
Prérequis: Ce cours s'adresse aux ingénieurs de support système, administrateurs de réseaux et candidats MCP, qui auront à concevoir, installer et gérer les réseaux sous NT Serveur et Workstation 4.0. Connaissance de Windows'95 indispensable. Avoir suivi les cours Notions Essentiels de Réseau et Administration de Windows NT 4.0.
Contenu du cours: • L'environnement de NT 4.0 • Installation de Windows NT • Configuration du système • Gestion des stratégies du système • Gestion du file-système • Gestion des partitions • La tolérance de panne • Le support d'applications • L'environnement de réseau • Configuration des protocoles • Les services de réseau • Service d'accès distant • Internet et Intranet • Les impressions • Implémentation des clients réseau • Synchronisation et duplicata • Le processus de démarrage • Dépannage général.

Windows NT Server 4.0, dépannage avancé (1 jour)
N° 6160 31.10.97 08h15 - 17h15
N° 6161 19.11.97 08h15 - 17h15
Objectifs principaux: Fournir aux professionnels du support les connaissances leur permettant de concevoir des stratégies de dépannage de Windows NT Server dans un réseau d'entreprise complexe. Résolution des problèmes par l'analyse et configuration des registres et préparation à l'utilisation de PSS (Product Services Support) si nécessaire.
Prérequis: Ce cours s'adresse aux ingénieurs système et de support, administrateurs de réseaux, qui auront à déboguer, gérer et optimiser les réseaux sous NT Server 4.0. Expérience de support et dépannage sur Windows NT indispensable. Avoir suivi le cours Administration Windows NT 4.0, support de cours Windows NT 4.0 Core Technologies et Windows NT 4.0 Services de répertoire ou connaissances équivalentes.
Contenu du cours: • Analyse de la structure des fichiers du système d'exploitation • Identification des fichiers utilisés au démarrage et ordre de chargement • Vérification de chargement des gestionnaires • Remplacement de fichiers endommagés ou manquants • Dépendance des fichiers • Aperçu et structure de la registry • Editeur de la registry • Dépannage en utilisant HKEY_LOCAL_MACHINE • Architecture de Windows NT 4.0 • Composants du mode privilégié de processeur • Composants réseaux du mode utilisateur • Isoler les problèmes du point de vue de l'architecture • Interprétation des écrans bleus • Travailler avec le Kernel Debugger • Utilisation du Crash Dump • Interprétation du DumpExam.

Windows NT 4.0 astuces pratiques (2 demi-jours)
N° 6308 20 & 21.11.97 13h30 - 17h15
Prérequis: Connaissances de l'interface Windows NT 4.0
Contenu du cours: • La création de multiples raccourcis et l'utilisation de leurs options • Les attributs de fichiers • La personnalisation de l'interface • La configuration de l'environnement • L'installation d'un logiciel depuis Olympe • Le module de recherche avec options avancées.

Windows 95 à NT 4.0 transition (1 demi-jour)
N° 6307 03.11.97 08h15 - 12h00
Prérequis: Bonnes connaissances de l'interface Windows 95
Contenu du cours: • Présentation de Windows NT 4.0 • Différences d'interface entre les deux systèmes• Le gestionnaire d'imprimante• Le gestionnaire d'utilisateurs • Les propriétés d'objets.

LABVIEW

LabView avancé (2 jours)
N° 6011 13 & 14.11.97 08h15 - 17h15

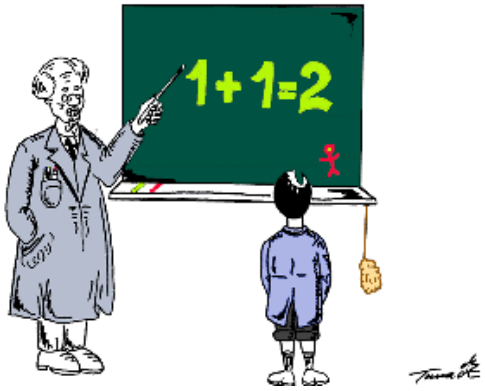
COURS SUR STATIONS DE TRAVAIL

Unix introduction (1 jour)
N° 3171 16.12.97 08h30 - 17h30
Pour débutant: aucune connaissance de Unix.

Administration avancée sous Solaris 2.x **(5 jours)**
N° 3170 01 au 05.12.97 09h00 - 17h30
Prérequis: Ce cours est destiné aux administrateurs de systèmes Unix et administrateurs réseau. Les connaissances du cours «Installation et administration des stations Sun sous Solaris 2.x (SunOS 5.x) sont un prérequis indispensable.
Objectifs: A l'issue de ce cours, les ingénieurs système seront capables d'exploiter au mieux un réseau local de stations et de serveurs Sun sous Solaris 2.x.
Administration avancée des réseaux IP.
Service de noms NIS+ d'ONC+.
Performances des services réseau et outils de surveillance.

JAVA, programmation avancée **(2 jours)**
N° 3168 13 et 14.11.97 09h00-17h30
Prérequis: Ce cours est réservé aux personnes ayant suivi le cours d'introduction à Java ou ayant une connaissance déjà bien établie d'un langage objet.
Objectifs du cours: • Manipulations graphiques • Interface graphique utilisateur • Applications indépendantes • Threads et Multithreading • Entrees-sorties fichiers • Programmation réseau Java • Utilisation de méthodes natives

Programmation C++ **(5 jours)**
N° 3169 24 au 28.11.97 09h00 - 17h30
Prérequis: Connaissances du langage de programmation C.
Description: Le langage C++ a été développé dans les laboratoires AT&T par Bjarne Stroustrup au début des années 80. Il représente une évolution du langage C dans trois directions principales: • la possibilité de créer et d'utiliser des types abstraits de données • la possibilité de faire de la programmation et de la conception orientée-objets • il fournit de nombreuses améliorations intéressantes aux structures existantes du langage C.



CONDITIONS D'INSCRIPTION

En cas d'empêchement à suivre le(s) cours, l'élève avertira le Service informatique central au minimum une semaine à l'avance (sauf cas exceptionnel), faute de quoi le SIC se réserve le droit de facturer à son unité les frais occasionnés pour le cours.
Une confirmation parviendra à l'élève environ deux semaines avant le cours. S'il est déjà complet, l'élève sera informé de suite et son nom placé en liste d'attente. Dès qu'un cours identique sera fixé, il recevra un nouveau formulaire d'inscription.
Le SIC se réserve le droit d'annuler un cours si le nombre minimum de 4 participants n'est pas atteint ou pour des raisons indépendantes de sa volonté. Aucune compensation ne sera due par le SIC.

INSCRIPTION POUR LES COURS ORGANISÉS PAR LE SIC

Remplir une inscription par type de cours (Mac, PC, Unix, …) et retourner à Josiane Scalfo, SIC-EPFL, 1015 Lausanne

Je, soussigné(e) Nom: ..... Prénom: .....

Tél.: ..... E-Mail: ..... Fonction: .....

Institut: ..... Dépt: ..... Adresse: .....

m'engage à suivre le(s) cours dans son (leur) intégralité et à respecter l'horaire selon les conditions d'inscription:

N° du cours Nom du cours N° cours de remplacement Date du cours

.....

.....

Date: ..... Signature: .....

Autorisation du chef hiérarchique (nom lisible et signature): .....

INTÉRÊT ET SOUHAIT POUR D'AUTRES COURS

Description ou titre des cours que je souhaite voir organiser par le SIC:

.....

.....

DU NOUVEAU DANS LES ANNUAIRES

DE L'ORIGINE TRÈS EXTRAORDINAIRE DU PROTOCOLE LDAP

Claude Lecommandeur, SIC

Au début était X.500 et son protocole d'interrogation DAP (Directory Access Protocol). À cette époque, les instituts de normalisation règnaient en maîtres dans nos contrées. À intervalles réguliers ces instituts accouchaient dans des douleurs infernales de monstres innommables, que malgré tout, ils nommaient. Les mises bas duraient des années, et les rejetons ainsi produits étaient le fruit de tant de compromis, voire de compromissions, que l'horreur était souvent au rendez-vous.

X.500 vit le jour vers cette époque, il était lourd, complexe, long à implémenter et difficile à gérer.

Quelques esprits éclairés virent cela et jugèrent que cela était mauvais. Ils décidèrent de créer une version simplifiée de DAP, ils la nommèrent LDAP, pour Lightweight DAP, car sa principale qualité était d'être légère, au moins en comparaison de DAP lui-même. Ces développements eurent lieu à l'Université du Michigan (UMICH). Au début, LDAP était juste une interface permettant d'accéder aux serveurs X.500, mais pourquoi s'arrêter en si bon chemin. Ces fiers esprits décidèrent de créer directement des serveurs qui pourraient gérer les données des annuaires et parler le protocole LDAP et ils virent que cela était bon.

Mais pendant ce temps, le monde avait changé, les instituts de normalisation avaient perdu beaucoup de leur pouvoir d'antan. D'autres grands prédateurs arpentaient la savane en dictant leurs lois. Les deux plus terribles, dont l'éclat des dents à lui seul suffisait à provoquer l'effroi, avaient pour noms Microsoft et Netscape.

Le plus petit des deux, mais peut-être le plus madré, était Netscape. Il avait à cette époque un grand besoin de protocole d'accès aux annuaires frais. Il promena son regard sur la savanne environnante, et il vit LDAP. Il pensa aussitôt qu'il pourrait faire du fric avec (Netscape, tout comme Microsoft était friand de cette nourriture). Il grogna fort, et annonça à tous qu'il allait utiliser LDAP et que donc c'était la nouvelle norme de fait. Ce qui prenait des années et des efforts considérables à la génération précédente de monstres, Netscape et Microsoft savaient le réaliser en quelques minutes en convoquant la presse.

LDAP qui avait jusqu'alors une diffusion confidentielle, se vit promu au rang fort envié de standard de fait.

Ainsi un annuaire basé sur LDAP peut être consulté directement par les principaux outils de courrier électronique, pour en extraire les adresses des destinataires. Il suffit de les configurer pour celà. *Netscape Communicator* (nom ridicule, mais c'est comme ça) et Internet Explorer entr'autres savent le faire.

Le SIC a mis à disposion les données du personnel et étudiants dans un tel annuaire. La machine serveur est ldap.epfl.ch, n'hésitez pas à l'utiliser.

AVEC NETSCAPE COMMUNICATOR 4

Menu Edit/Preferences/Mail and Groups/Directory: définir un nouvel annuaire (touche New)
Description: annuaire EPFL
LDAP Server: ldap.epfl.ch
Search Root: o=EPFL,c=CH
Port Number: 389
Maximum Number of Hits: 100

AVEC INTERNET EXPLORER 3

depuis Internet Mail, choisir New Message:
Menu File/Address Book/Directory Services: Add
Server: ldap.epfl.ch

sous Advanced:
Search Base: o=EPFL,c=CH

AVEC INTERNET EXPLORER 4

depuis l'outil Mail:
Tools/Accounts/Directory Services: Add
Server: ldap.epfl.ch

sous Advanced:
port: 389
Search Base: o=EPFL,c=CH ■

CALENDRIER

OCTOBRE 97

Mardi 28	14h15	Salle Conférences SIC	**CTI** — Commission Technique Informatique M. Reymond, ✆ 693.2210, ✉ *Michel.Reymond@epfl.ch*
Jeudi 30	16h00	Salle Conférences SIC	**CI** — Commission Informatique Alain Germond, ✆ 693.2262, ✉ *Alain.Germond@epfl.ch*

NOVEMBRE 97

Mercredi 5	10h00	Salle Conférences SIC	**HPLine** — Groupe des utilisateurs de stations HP Ion Cionca, ✆ 693.4586, ✉ *Ion.Cionca@epfl.ch* Info sur: *http://hpwww.epfl.ch/SIC/hpline.html*
Mardi 11	08h45	Salle polyvalente du SIC	**Comité de rédaction du FI** J. Dousson, ✆ 693.2246, ✉ *Jacqueline.Dousson@epfl.ch*
Jeudi 20	14h15	Salle Conférences SIC	**PolyPC** — Groupe des utilisateurs d'IBM PC et compatibles Ch. Zufferey, ✆ 693.4598, ✉ *Christian.Zufferey@epfl.ch* Info sur: *http://pcline.epfl.ch/pc/grp/home.htm*

NETD@YS 97

du 18 au 25 octobre 97

Jacqueline Dousson, SIC, e-mail: Jacqueline.Dousson@epfl.ch

Soutenu par Al Gore et Bill Clinton, le premier Netday a eu lieu aux USA le 9 mars 1996. Cet événement était centré sur les écoles de Californie, le but annoncé étant de rassembler volontaires et sponsors pour câbler les écoles de Californie (5 classes et une bibliothèque par école). Son succès fut tel (4000 écoles câblées) que le principe en fut repris en octobre 96 et mars 97. Le prochain Netday y aura lieu le 25 octobre et déjà 28000 volontaires et 3000 sponsors se sont annoncés. Les sites fleurissent avec conseils aux éducateurs, expériences concrètes, cours en ligne, allez y faire un tour (voir quelques points de départ en bas de l'article) et vous verrez à quel point cela va vite… Cette année, l'Europe s'y met aussi, la Commission Européenne a lancé l'initiative NetDays97, du 18 au 25 octobre. Tout en reprenant les principes de base (volontariat et stimulation des partenaires privés et publics), la manifestation européenne semble plus s'axer sur la sensibilisation, et le contenu que la simple connexion. En Suisse le Centre suisse des technologies de l'information dans l'enseignement (CTIE) coordonne l'action et invite tous les partenaires potentiels à participer, que ce soit sous la forme de présentations, de débats, de vidéoconférences… Retroussons nos manches! Il n'est plus temps de se demander si oui ou non Internet est utile pour l'école et nos enfants. Nous devons apprendre à l'utiliser, créer notre propre contenu, réfléchir aux enjeux sinon il sera bientôt trop tard pour nous lamenter! Microsoft s'occupera de l' éducation de nos enfants avec la même efficacité qu'il met à envahir nos ordinateurs professionneles et domestiques. Les moyens techniques sont là et accessibles à tous, maintenant la place est libre pour l'imagination. Ne laissons pas passer l'occasion.Soyez vigilants et durant cette semaine d'octobre vous verrez près de chez vous des initiatives se mettre en place. Apportez votre aide et vos connaisances techniques, amenez vos enfants, apprenez leur à maîtriser cet environnement, à moins que ce ne soit eux qui vous l'apprennent… Et espérons que les prochains Netdays européens se dérouleront en dehors des vacances scolaires vaudoises, afin que nos écoles s'impliquent de façon plus officielle!

QUELQUES URL

- *aux USA: http://www.netday96.com/ est un point d'entrée pour l'organisation des Netdays*
- *en Europe: http://netdays.eun.org/ et http://europa.eu.int/en/comm/dg22/netdays/homefr.html*
- *en Suisse: http://www.netdays97.ch/* ■

A l'EPFL: **les 24 heures du net** rassembleront des démonstrations, vidéoconférences, débats, du vendredi 24 octobre 17h au samedi 25 octobre 17h! Pour y participer en tant qu'acteurs, prenez contact.