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

L'ENFANT BANNI

Un PC sans PC

par Hervé Le Pezennec, DE-LEMA

En attendant ma livraison de PC tout neufs et avant de pouvoir les installer, les configurer, les tester et relater tout cela dans un prochain article, je me suis penché sur les possibilité de substitution: les émulateurs pour PC. «Qui tente à égaler ou à surpasser quelque chose», comme dit le petit Larousse. Egaler cela serait déjà pas mal, non?

UN ÉMULATEUR POUR QUOI FAIRE?

Comme je l'ai déjà dit au début de cette série d'articles, nous avons un problème aujourd'hui de plates-formes et de systèmes hétérogènes à gérer. Les budgets limités ainsi que la place disponible sur nos bureaux ne permettent pas d'avoir deux ou trois écrans par personne. Le parc actuel étant principalement composé de Macintoshs pour la bureautique et de stations Unix pour les développements, il est intéressant alors d'émuler un PC sur un matériel existant. Le PC permettant comme on l'a déjà vu de remplir au mieux les deux fonctions précitées auparavant.

Une émulation permet d'avoir à l'écran un système PC (Windows 3.11, Windows 95, entre autres) et de partager un certain nombre de ressources matérielles ou logicielles comme le réseau, les périphériques, la mémoire ou les disques.

Si le but de l'émulateur paraît attirant au niveau des fonctionnalités, cela ne doit pas être au détriment du coût et de la complexité de gestion des deux systèmes. Afin de tirer des conlusions les plus objectives possible, il faut faire un bilan entre la qualité des fonctions émulées et le surcoût (financier, travail, apprentissage…) de l'opération.

Dans la suite de l'article je passe en revue les trois types d'émulations que j'ai pu tester.

L'ÉMULATION LOGICIELLE

L'émulation logicielle est la plus connue et la plus simple en principe: il s'agit de concevoir un programme qui émule une interface PC dans une fenêtre (matérielle et logicielle) dans un environnement appartenant à votre machine.

Le plus gros problème que l'on voit directement sur le dessin ci-

Suite en page 7

SOMMAIRE

PROCHAINES PARUTIONS

parution FI		délai FI
5	20.05.97	01.05.97
6	17.06.97	29.05.97
SP	02.09.97	26.06.97
7	16.09.97	28.08.97
8	21.10.97	02.10.97
9	18.11.97	30.10.97
10	19.12.97	25.11.97

SIC-INFO

SECTION EXPLOITATION

NOUVEAUX COLLABORATEURS

Dans le cadre du regroupement des fonctions de support des serveurs centraux liés à des projets, principalement dans le domaine du parallèlisme, deux collaborateurs qui ont travaillé dans le cadre du projet PATP (groupe de support) ont rejoint récemment la SE. Il s'agit de :

- M. Christopher Potter, dont les fonctions sont principalement le support d'applications parallèles, en particulier par PVM et MPI, modèles de programmation et méthodes algorithmiques, optimisation et aide au *debugging*, ainsi que le support de librairies et l'évaluation de nouveaux produits.
- M. Steve Williams, qui offre ses compétences pour le développement et le support de modules graphiques liés aux applications pour serveur parallèle: conseils pour la visualisation, modules de communication, tests de nouveaux produits et packages graphiques. Il peut aussi aider à la préparation de documents comportant des séquences graphiques.

Tout utilisateur peut prendre contact avec l'un ou l'autre de ces collaborateurs en fonction de ses besoins :

M. Potter, tél. 693 45 52, potter@sic.epfl.ch
M. Williams, tél. 693 22 86, williams@sic.epfl.ch

Nous souhaitons la bienvenue à MM. Potter et Williams et espérons qu'ils pourront vous apporter l'aide nécessaire au développement de vos projets et applications sur les serveurs centraux de l'EPFL.

Michel Jaunin, SIC

RAPPEL

ARRÊT DE NESTOR ET DISTRIBUTION DE LOGICIELS

L'arrêt de Nestor, qui assure aujourd'hui la fonction de serveurs de fichiers et de distribution de logiciels, est prévu pour fin mai. La page Web

http://sewww.epfl.ch/SIC/SE/servcentraux/arret_nestor.html

donne les informations les plus récentes à ce sujet. L'application ASIS reprend la distribution des logiciels. Pour toute information concernant ASIS consultez la page

http://castor.epfl.ch/asis/

Anne Possoz, SIC

SERVEURS CENTRAUX

LOGICIELS SUR LE CRAY-T3D

Afin d'utiliser les nouvelles versions des logiciels pour le T3D, une nouvelle procédure a ete mise en place par CRAY. Vous devez **impérativement** rajouter:

pour le .profile

```
if [[ -f /opt/modules/modules/init/ksh ]]
then
      . /opt/modules/modules/init/ksh
      module load modules PrgEnv
fi
```

pour le .cshrc

```
if (-f /opt/modules/modules/init/csh) then
    source /opt/modules/modules/init/csh
    module load modules PrgEnv
endif
```

Si vous ne le faites pas, vous utilisez des anciens logiciels qui ne sont plus supportés.

Ce programme *module* permet en plus de sélectionner la version du logiciel souhaitée (si par exemple vous voulez utiliser un ancienne version d'un compilateur).

Vous pouvez également taper *module avail* pour voir les versions disponibles.

A noter enfin, que le compilateur f90 complet est maintenant disponible sur le T3D.

Christopher Potter, SIC

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, Patrick Lachaize, Hervé Le Pezennec, François Roulet, Christian Simm & 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 9 771420 719001

LE TEST DE LOGICIELS À OBJETS

par Stéphane Barbey, Laboratoire de Génie Logiciel, EPFL-DI
e-mail: stephane.barbey @ di.epfl.ch, URL: http://lglwww.epfl.ch/~barbey

INTRODUCTION

Alors que l'utilisation des méthodes par objets s'est étendue, voire imposée pour certains types d'applications, aux premières étapes du cycle de vie du logiciel (analyse, conception et implémentation), la phase de test n'a pour l'instant été que partiellement étudiée dans l'optique d'une approche par objets. Cette lacune est d'autant plus grave que, les méthodes par objets favorisant le développement de composants réutilisables, les conséquences de fautes non détectées peuvent se retrouver dans plusieurs applications, d'où la nécessité d'accorder à cette phase une attention particulière.

Dans cet article nous présentons quelques-uns des problèmes et des solutions propres au test de logiciels à objets. Nous commençons par définir ce qu'est le test et comment il est pratiqué de façon générale. Ensuite nous exposons les problèmes liés au test de logiciels à objets et nous décrivons certaines façons de résoudre ces difficultés. Comme ces solutions font l'objet de recherche au Laboratoire de génie logiciel, nous concluons par la description de nos axes de recherche dans ce domaine.

Le lecteur intéressé à approfondir ce sujet trouvera également des pointeurs vers notre recherche et celles d'autres chercheurs à travers une courte bibliographie et une liste de liens vers des pages WWW.

QU'EST-CE QUE LE TEST?

Le test du logiciel est une activité de vérification: elle se caractérise par la comparaison des propriétés attendues d'un programme – sa spécification – et celles que le programme réalise effectivement – son implémentation. Son but est d'établir que l'implémentation vérifie les propriétés exigées par la spécification, ou de détecter des différences entre elles. Il existe plusieurs types de test, qui correspondent à diverses propriétés qu'on peut attendre d'un programme. Dans cet article, nous nous intéresserons au test fonctionnel, soit à la vérification de son comportement. D'autres types de test pratiqués sont le test de performance ou de robustesse. Le test se distingue d'autres activités de vérification telles que la revue de code ou la preuve mathématique en ce qu'il est une activité dynamique: le test exige que le programme soit exécuté, pour que ses résultats puissent être collectés et comparés à la spécification.

Le test fonctionnel est le processus qui vise à trouver des erreurs dans le comportement d'un logiciel ou d'un composant logiciel. On attend de ce type de test qu'il montre non seulement qu'un logiciel fait ce qu'on attend de lui, mais aussi qu'il ne fait pas ce qu'on en n'attend pas. Ainsi, le but du test n'est pas de montrer qu'un logiciel ne comporte pas d'erreur, ni de corriger les erreurs.

En soit, le test fonctionnel pourrait être simple: il suffirait de tester exhaustivement tous les comportements d'un programme et de vérifier que chaque comportement satisfait la spécification. Malheureusement, procéder à un test exhaustif est généralement hors de question: le nombre de cas est souvent trop grand, voire infini, pour qu'on puisse tous les tester dans un temps raisonnable. Le problème du test est donc un problème d'échantillonnage ou de couverture: il faut sélectionner parmi les comportements possibles ceux qui assurent une meilleure couverture du programme, c'est à dire qui sont représentatifs du comportement général du programme, sans oublier les cas particuliers (les tests aux bornes), susceptibles de révéler des erreurs. La qualité d'un ensemble de tests se mesure donc à la qualité de la couverture qu'il offre.

Il existe deux grandes familles de critères de sélection: les tests en boîte blanche et les tests en boîte noire. Les tests en boîte blanche sont basés sur un examen du texte du programme: les tests sont sélectionnés de manière à remplir certaines exigences de couverture de l'implémentation (toutes les instructions, tous les chemins exécutables,…). S'ils sont faciles à mettre en œuvre, les tests en boîte blanche souffrent cependant d'un défaut rédhibitoire: cette stratégie ne permet de tester que ce qui figure dans le programme, mais ne permet pas de mettre à jour des oublis du programme par rapport à la spécification. Les tests en boîte noire ne prennent pas en compte la façon dont le programme est écrit, mais sont issus directement de sa spécification: il s'agit d'examiner la description du comportement attendu – les axiomes de la spécification – pour sélectionner des données en entrée pour les programmes testés, et calculer les résultats escomptés. Si cette description est écrite dans un langage formel, manipulable par un ordinateur, cette procédure peut être automatisée ou semi-automatisée. Les tests sélectionnés par cette stratégie garantissent une bonne couverture des domaines d'entrée du programme, et des oublis par rapport à la spécification peuvent être détectés. De plus, lors de modifications du programme ne remettant pas en cause la spécification, il est possible de conserver les tests précédemment sélectionnés pour vérifier le programme modifié.

Le processus de test s'effectue donc en trois étapes. Le testeur va d'abord sélectionner des tests assurant un

suite

bon échantillonnage des comportements possibles. Ensuite, il va exécuter ces tests sur le programme et observer son comportement. Finalement, il va comparer les comportements observés avec les comportements attendus. Cette dernière étape est réalisée au moyen d'un oracle, dont le rôle est de décider si le test est un succès, c'est-à-dire que le programme satisfait la spécification, ou si c'est un échec, c'est-à-dire qu'une erreur a été détectée. Il faut noter que pour le testeur, un échec est un succès! Plus il détecte d'erreurs, meilleurs sont ses tests. La réalisation d'un oracle n'est pas dépourvue de difficultés: il n'est pas toujours facile d'observer le comportement d'un programme. En conséquence, lors de la sélection de tests, il faut ajouter au critère de couverture celui d'observabilité: sélectionner des tests desquels l'oracle pourra décider du succès ou de l'échec.

LE TEST DE LOGICIELS À OBJETS

Le test de logiciels à objets peut être pratiqué selon les stratégies décrites ci-dessus, mais certaines de ses caractéristiques nécessitent des méthodes spécifiques. Rappelons brièvement qu'un système à objets consiste en un ensemble d'objets, instanciés à partir de classes, et désignés par des références. Ces objets possèdent chacun un état encapsulé, inaccessible de l'extérieur de l'objet, et communiquent entre eux par l'envoi de messages, c'est-à-dire en invoquant des méthodes sur les références. Les classes d'où sont instanciés les objets peuvent être organisées hiérarchiquement par des relations d'héritage, ce qui permet de créer des sous-classes par dérivation de classes dites parentes, et de redéfinir les méthodes héritées pour en spécialiser le comportement. Lorsqu'une référence peut désigner des objets appartenant à une classe ou à ses sous-classes, on parle de polymorphisme. Lors de l'invocation d'une méthode sur une telle référence, la méthode effectivement appliquée est choisie de façon dynamique à l'exécution du programme entre la méthode originale et ses spécialisations. Ce processus s'appelle la liaison dynamique.

LES PROBLÈMES…

Le premier problème auquel est confronté le testeur est de trouver une unité indépendante de test. Si, dans les systèmes traditionnels construits selon une approche structurée, chaque procédure ou fonction peut être testée indépendamment et constitue l'unité de test, ceci n'est pas possible avec les méthodes (les messages) des objets:

- une méthode n'existe que par rapport à la classe à laquelle elle est attachée, on ne peut pas tester de méthode sans l'appliquer à un objet;
- chaque objet possède un état; le contexte dans lequel une méthode est exécutée n'est pas seulement défini par ses paramètres, mais (principalement) par l'objet auquel elle est appliquée.

Le deuxième problème est causé par l'encapsulation: le comportement d'une méthode ne s'observe généralement pas en examinant les données en sortie de l'opération, mais en examinant l'état de l'objet à l'issue de l'invocation. Or cet état est encapsulé, c'est-à-dire inobservable: on ne peut y accéder que par l'intermédiaire de méthodes. On ne peut donc pas construire des oracles simples basés sur la comparaison des objets.

L'héritage en lui-même ne cause pas de difficulté: lors du test d'une sous-classe, on peut la considérer comme une unité indépendante qui regrouperait les propriétés héritées et les propriétés redéfinies. En revanche, des raisons économiques poussent à réutiliser les tests sélectionnés pour tester une classe parente lors du test d'une sous-classe afin de ne retester dans la sous-classe que ce en quoi elle diffère de sa classe parente. C'est cette sélection qui pose problème: contrairement à ce que suggère l'intuition, les méthodes héritées sans être spécialisées doivent parfois être retestées dans le contexte de la sous-classe car leur comportement peut être influencé par celui des méthodes spécialisées. Il est dès lors difficile de savoir parmi les tests sélectionnés pour la classe parente ceux qui peuvent être éliminés et ceux qui doivent être rejoués.

Finalement, la liaison dynamique comporte son lot de problèmes: lorsqu'une référence polymorphe apparaît, par exemple dans les paramètres d'une méthode, elle peut désigner des objets appartenant à plusieurs classes et ayant des comportements spécialisés. Le choix des données en entrée impose alors de sélectionner des tests qui couvrent tous ces comportements.

… ET LEURS SOLUTIONS

La solution au premier problème est évidemment de considérer la classe comme unité de test. Comme les méthodes ne peuvent être testées individuellement, elles sont testées en interaction, c'est-à-dire qu'on teste des séquences de méthodes appliquées à un objet. Un test prend la forme d'un couple <formule, résultat>, où formule est une combinaison de méthodes, et résultat une valeur booléenne indiquant que la formule décrit un comportement du programme admissible ou non. De cette façon, comme on l'a dit plus haut, on peut écrire des tests non seulement pour vérifier que le programme fait ce qu'on en attend (un comportement admissible), mais aussi qu'il ne fait pas ce qu'on en attend pas (un comportement inadmissible). Ainsi, si l'on voulait tester le système de cartes à puce qui donne accès aux bâtiments de l'EPFL, on aurait des tests pour vérifier que l'accès est autorisé quand l'utilisateur entre le bon numéro, mais aussi des tests pour vérifier que l'accès est refusé dans le cas contraire. Ainsi, pour une classe Porte avec les méthodes insérer (carte), entrer_code (code) et ouvrir, et un objet de cette classe désigné par la référence porte, on pourrait avoir les tests:

```
<<porte.insérer (carte_avec_pin_140789)>
<porte.entrer_code (140789)><porte.ouvrir>,
vrai>
```

suite

et

```
<<porte.insérer (carte_avec_pin_140789)>
<porte.entrer_code (010891)><porte.ouvrir>,
faux>
```

Cependant, comme l'ordre dans lequel les méthodes peuvent être invoquées n'est généralement pas spécifié, le nombre de combinaisons possibles est souvent infini. Il est donc nécessaire de trouver une stratégie pour réduire le nombre de tests à un ensemble fini tout en maintenant une qualité satisfaisante. C'est pourquoi nous avons adapté au test de logiciels à objets une stratégie développée au LRI (Université de Paris-Sud, Orsay) pour le test de spécifications algébriques. Alors que les testeurs partent généralement d'un ensemble de tests vide et y ajoutent les cas jugés intéressants, au risque d'en oublier et surtout sans pouvoir en mesurer la qualité, cette stratégie part de l'ensemble exhaustif des tests et, en appliquant des hypothèses — appelées hypothèses de réduction — sur le programme, réduit cet ensemble infini à un ensemble fini. Ces hypothèses sont construites de manière à garantir la préservation de la pertinence du jeu de test exhaustif. Ainsi, au lieu de définir ce qu'il va vérifier, le testeur définit ce qu'il ne va pas tester. Dès lors, la qualité de l'ensemble de tests se mesure à celle des hypothèses appliquées.

Ces hypothèses de réduction correspondent à une formalisation de pratiques courantes chez le testeur. Elles se divisent en deux catégories: les hypothèses de régularité et les hypothèses d'uniformité. Les hypothèses de régularité admettent que si un objet se comporte correctement pour un cas d'une complexité donnée, il se comportera également correctement dans les cas d'une complexité supérieure. Dans l'exemple du système de cartes à puce, un exemple de complexité serait la répétition d'une combinaison de méthodes: si, par exemple, le système d'insertion de cartes est testé en insérant une carte vingt fois, on peut émettre l'hypothèse qu'il fonctionnera également correctement pour un plus grand nombre d'insertions. Dès lors, on renoncera à tester l'insertion vingt et une fois, vingt-deux fois,… D'autres exemples d'hypothèses de régularité sont la forme de la combinaison (par exemple que certaines méthodes doivent en précéder d'autres) ou l'absence d'interactions entre certaines méthodes (parce qu'elles ne modifient pas la même partie de l'état).

Les hypothèses d'uniformité concernent les paramètres des méthodes: si une méthode se comporte correctement pour une certaine valeur d'un paramètre, elle se comportera correctement pour toutes les autres valeurs. Ainsi, dans l'exemple de l'insertion de cartes à puce, on peut émettre l'hypothèse qu'il n'est pas nécessaire de tester le système avec toutes les valeurs possibles du code d'identification (ce qui pour six chiffres donne un million de cas de tests), mais qu'une valeur choisie aléatoirement suffira. Cette hypothèse d'uniformité n'est évidemment pas applicable à tous les cas: si l'on veut tester les cartes de photocopieuses, on ne se contentera pas de tester le système avec une carte contenant cent unités, car on oublierait le cas intéressant où la carte est vide (et où le système ne doit pas autoriser l'utilisateur à se servir de la photocopieuse.) Plutôt que de pratiquer les hypothèses d'uniformité avec une valeur unique, le testeur va donc plutôt décomposer l'ensemble des valeurs qu'un paramètre peut prendre en sous-ensembles, et appliquer une hypothèse d'uniformité (sélectionner une valeur unique) pour chacun de ces sous-ensembles. Dans le cas de la carte de photocopieuse, les sous-ensembles intéressants sont la carte vide et les cartes non-vides. Cette décomposition se fait par une analyse des cas précisés dans les axiomes de la spécification.

Le deuxième problème, celui de l'oracle, qui est lié à l'encapsulation, est partiellement résolu par la forme des tests. Pour chaque test sélectionné, l'oracle va générer un pilote de test, qui va exécuter la combinaison et récolter les résultats. Si l'oracle n'a pas pu exécuter toute la combinaison, le résultat du test est *faux*. Ainsi, dans le cas du test

```
<<porte.insérer (carte_avec_pin_140789)>
<porte.entrer_code(140789)><porte.ouvrir>,
vrai>
```

si la porte ne s'ouvre pas après avoir entré le code (c'est-à-dire si le programme rejette l'appel à la méthode ouvrir), une erreur a été détectée. Mais cela n'est parfois pas suffisant, il convient également de vérifier la valeur des paramètres en sortie du test, et de les comparer avec les résultats attendus. Comme ces résultats pourraient ne pas être observables (à cause de l'encapsulation), il convient également de raffiner les tests par la construction de contextes observables, ce qui se traduit par la néces-

suite

LA BIBLIOTHÈQUE DU TESTEUR

Bien qu'écrit à la fin des années septante, l'ouvrage de référence en matière de test reste le livre de G. Myers, «The Art of Software Testing» (John Wiley & Sons, 1979). Ce livre décrit non seulement des techniques de tests, mais aussi la composante humaine de l'activité de test. Le livre de B. Beizer, «Software Testing Techniques» (Van Nostrand Reinhold, 1990) mérite également de figurer sur les rayons de vos bibliothèques. Plus récent, le livre de M. Roper, «Software Testing» (McGraw-Hill, 1994) donne également un bon aperçu des diverses techniques de test.

Aucun ouvrage spécifique au test par objets n'est actuellement disponible, même si quelques ouvrages sont en préparation. Cependant, certains livres déjà parus contiennent des chapitres consacrés au test par objets, comme le livre de M. Roper, mentionné ci-dessus, ou celui de B. Marick, «The Craft of Software Testing: subsystem testing including object-based and object-oriented testing» (Prentice Hall, 1994).

Le cybernaute trouvera aussi aux croisées du réseau des réseaux des renseignements sur le test par objets, à commencer par une page du Laboratoire de génie logiciel

http://lglwww.epfl.ch/Research/OO_Research/toos

d'où il pourra télécharger nos principales publications et trouver des liens vers d'autres sites intéressants. Des bibliographies du test de logiciels à objets sont disponibles à

http://liinwww.ira.uka.de/bibliography/Object/ootest.html

et

http://www.ksccary.com/biblio.htm

Des listes de liens se trouvent également à l'URL

http://www.rhein-neckar.de/~cetus/oo_testing.html

sité d'enrichir les tests avec des méthodes (ou des fonctions) sur les éléments qui ne sont pas observables, et ceci récursivement jusqu'à ce qu'ils soient rendus observables. Cette adjonction de méthodes se fait déjà lors de la sélection des tests.

Les problèmes liés à l'héritage et au polymorphisme sont d'abord des problèmes de spécification. L'héritage tel qu'on le trouve dans les langages de programmation par objets (Ada 95, C++, Smalltalk,…) est une relation syntaxique: il exige des méthodes des sous-classes qu'elles respectent le profil des méthodes de leur classe parente, mais il n'impose pas de respecter leur comportement. Dès lors, tous les débordements sont permis… et la réutilisation des tests de la classe parente pour tester une sous-classe est compromise. Néanmoins, le testeur peut en tirer partie si la relation d'héritage est forte, soit qu'elle inclut également la sémantique des opérations (leur comportement). On parle alors de relation de sous-typage fort. On peut distinguer trois cas de méthodes dans les sous-classes: les méthodes héritées telles quelles (sans spécialisation), les méthodes spécialisées et les méthodes ajoutées (c'est-à-dire ne figurant pas dans la classe parente.) Pour ces dernières méthodes, il est évidemment nécessaire de rajouter de nouveaux tests. Pour les méthodes spécialisées, les tests du parent doivent être réappliqués pour s'assurer que le sous-typage fort est respecté. Des nouveaux cas de tests peuvent être ajoutés pour traiter les nouveaux cas résultant de la spécialisation. Dès lors, si les méthodes spécialisées se comportent comme celles de leur parent, on peut émettre l'hypothèse, dite d'incrémentalité, que les méthodes héritées mais pas spécialisées n'ont pas besoin d'être retestées. Dans le cas contraire (si l'héritage est uniquement syntaxique), il est nécessaire de procéder à une analyse du code du programme pour distinguer les méthodes héritées sans spécialisation qui n'ont pas besoin d'être retestées.

Le polymorphisme est un cas particulier d'hypothèse d'uniformité, et de décomposition en sous-ensembles: en plus de devoir tenir compte des valeurs possibles des paramètres, il faut également tenir compte de leur classe. A nouveau, ce problème de sélection est simplifié si les classes possibles sont unies par une relation de sous-typage fort: les sous-classes ayant le même comportement que leur parent, on peut émettre une hypothèse d'uniformité sur la classe parente. Dans le cas contraire, le testeur devra ajuster l'hypothèse d'uniformité en fonction de la dépendance de la méthode testée envers ses paramètres et en fonction des relations de spécialisation qui existent entre les sous-classes concernées.

LES ACTIVITÉS DE TEST AU LABORATOIRE DE GÉNIE LOGICIEL

Les recherches en cours au Laboratoire de génie logiciel concernant le test de logiciels à objets s'orientent dans plusieurs directions:

- la spécification de systèmes d'objets. Pour pouvoir générer des tests, il est nécessaire de disposer de spécifications complètes et non ambiguës. Ceci est difficile à partir de spécifications informelles, écrites en langage naturel. Un axe de recherche important est le développement d'un langage de spécification formel, nommé CO-OPN/2 (Concurrent Object-Oriented Petri Net/2), adapté au développement de systèmes à objets concurrents et répartis. Ce langage est basé sur les spécifications algébriques et les réseaux de Pétri, et intègre une notion de sous-typage fort;
- une théorie du test, qui formalise les idées exposées dans cet article;
- l'intégration de CO-OPN/2 et de notre méthode de test dans le cycle de vie du logiciel, et principalement dans la méthode de développement par objets Fusion;
- des méthodes opérationnelles et des outils de test, qui permettent d'assister le testeur dans la sélection des tests à partir de spécifications CO-OPN/2. Un premier prototype a été réalisé.

Cette recherche est effectuée en collaboration avec des partenaires européens tels qu'Eurecom (Sophia-Antipolis, Nice) ou les membres du projet Esprit DeVa (Design for Validation). Nous sommes intéressés à collaborer avec d'autres partenaires pour valider ces méthodes.

CONCLUSION

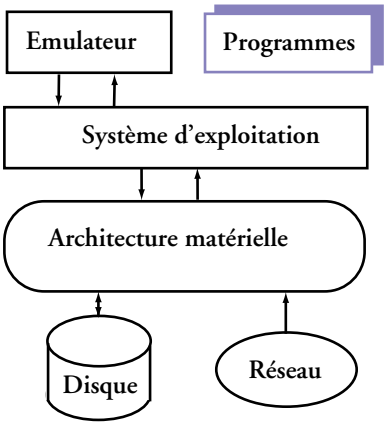
L'utilisation des approches par objets n'a pas résolu le problème du test. Même si ces approches accroissent la qualité du logiciel, elles n'empêchent pas le programmeur — un être humain — de commettre des erreurs. Il est dès lors nécessaire de disposer de méthodes de test qui soient adaptées à ce genre de développement.

Cet article présente une vue des problèmes et des solutions au test de logiciels à objets. Notre méthode de test a l'avantage de reposer sur des bases formelles solides, qui garantissent sa cohérence. Elle contraste avec les approches empiriques qui ne fournissent que des solutions ad hoc adaptées à des situations et des langages particuliers. Cette approche permet également, par l'utilisation des hypothèses de réduction, de mesurer la qualité du test en exprimant sa couverture par la différence entre le test exhaustif et les cas que l'on renonce à tester, rendant explicite la prise de risque due l'échantillonnage. Son adéquation à la génération automatique de tests en fait un outil dont le potentiel économique en milieu industriel est non négligeable.

La rigueur de cette méthode de test permet d'obtenir une confiance dans les jeux de tests sélectionnés, et, en conséquence, dans les composants testés. Leur réutilisation est ainsi favorisée, ce qui permet de développer du logiciel de qualité en diminuant les coûts de maintenance.■

Suite de la première page

après est que l'émulateur doit passer aux travers de plusieurs couches afin d'accéder au CPU ou aux périphériques. Cela se ressent au niveau de la performance (avec un tout beau PowerMacintosh on arrive tout juste à la puissance d'un 486) et au niveau de la fiabilité (surtout quand le système d'exploitation n'est pas très robuste et je n'ai pas dit MacOS !).



TEST1: SOFTWINDOWS SUR MAC

J'ai pu tester un émulateur PC parmi les plus connus en environnement Macintosh et Unix: SoftWindows d'Insignia *(www.insignia.com)*. La version pour Mac est disponible sur le serveur du SIC et il suffit d'un clic pour l'installer. Il faut prévoir assez de place car l'émulateur crée un fichier de plusieurs dizaines de MégaBytes comme disque C: pour votre pseudo-PC. Après quelques bombes au démarrage, je me suis renseigné au SIC pour apprendre qu'il faut un minimum de 400 MB réservé sur votre disque et 64 MB de mémoire afin de faire fonctionner correctement les deux environnements en parallèle. Ce n'était pas mon cas et j'ai eu toutes les peines du monde à juste démarrer mon émulateur. La gestion des périphériques est bonne ce qui n'est pas le cas pour le réseau. On ne peut voir que les autres Macintosh et non les PC. Ceci est plutot génant car on ne peut pas se connecter au serveur PC du SIC afin de charger les applications souhaitées. Nous sommes condamnés à utiliser les disquettes ou alors faire des copies sur le disque Macintosh et les transférer du côté PC. Rien qu'à l'expliquer on voit déjà la difficulté du truc…

La version que j'ai testée correspondait au système Windows 3.11 qui pour moi n'offre aucun intérêt. Il existe maintenant une émulation de Windows95 mais qui n'est pas disponible sur le site de l'Ecole, le SIC ayant décidé de ne pas poursuivre dans cette voie.

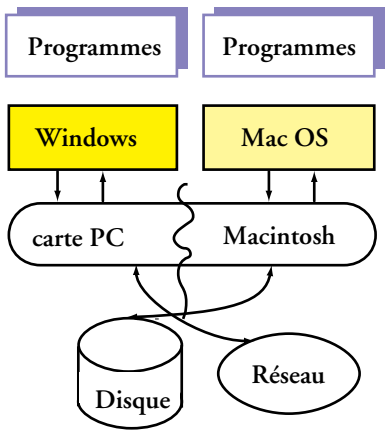
TEST 2: SOFTWINDOWS SUR SILICON GRAPHICS

J'ai pu tester également la version Windows 3.11 sur Silicon Graphics qui était livrée avec les stations Indy. Je crois qu'aujourd'hui la version Windows95 est livrée avec les nouvelles stations O2 qui les remplacent. Je n'ai malheureusement pas pu tester cette version et les e-mails envoyés à cette fin chez Insignia sont restés sans réponse…

Sur SG l'installation est sans problème et le partage des disques et de la mémoire vive est plus souple et plus transparent. Encore mieux si vous avez la chance d'avoir un lecteur de disquettes, il suffit de cliquer sur un fichier texte pour PC et l'émulateur se met en route et l'ouvre directement dans WordPad. Malheureusement les problèmes de réseau sont les mêmes qu'avec l'émulation sur Macintosh: impossible de voir un autre PC. Il aurait fallu tester cela sur la version Windows 95. Si une personne a pu y arriver, merci de me le faire savoir et je transmettrai.

L'ÉMULATION MATÉRIELLE

Afin de gagner en performance, nos chers vendeurs d'émulateurs ont eu l'idée de proposer une carte à enficher dans votre Macintosh avec un véritable PC à l'intérieur (CPU+RAM+drivers). Les autres périphériques étant toujours partagés.

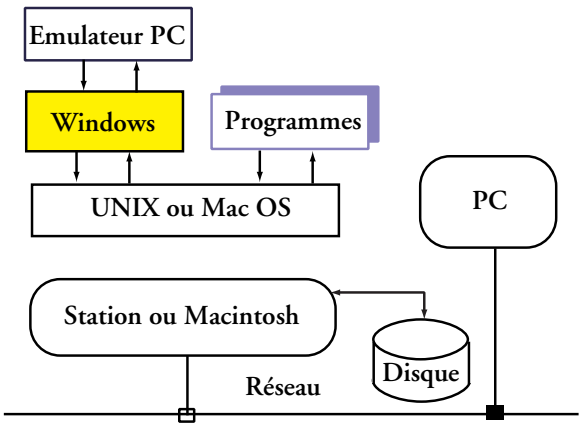


Rappellez vous de la série Centris 610 ou PowerMac 6100 avec carte DOS. Et bien vous pourrez vous vanter maintenant que vous avez connu quelqu'un qui a un tel Macintosh sur son bureau. Je vois tous ces airs envieux et ces regards avides d'impatience qui brûlent de savoir… Et bien oui, pour seulement un surcoût de frs 700.- vous avez les mêmes problèmes qu'avec un PC sans en avoir les avantages. Ah je vous le dis nous regretterons le temps où le marketing d'Apple arrivait à nous vendre ses fabuleux produits.

Le but était louable et la réalisation plaisante. En appuyant sur quelques touches du clavier on basculait dans l'environnement Windows en trois secondes de fondu enchaîné. Les deux machines jumelles étaient indépendantes car chacune avait son CPU et vous aviez la possibilité de faire tourner un programme en même temps dans chaque monde. Les performances correspondaient exactement à la puissance du CPU que vous aviez sur votre carte, qui était un 486 pour mon cas. Par contre, vous vous en doutez, toujours ce même problème de réseau. Je pense qu'il aurait suffit de rajouter à la carte PC sa propre interface et moyennant deux cables de connexion nous aurions pu résoudre tous les problèmes. Aujourd'hui les cartes PC sont quasi introuvables et je pense qu'Apple a abandonné cette direction.

suite

L'ÉMULATION VIRTUELLE



Je ne savais pas quel titre donner à cette troisième possibilité d'émulation qui n'en est pas une ! Je m'explique. Pour éviter tous les problèmes de performances de l'émulateur *soft* qui utilise toutes les ressources de votre machine et les problèmes de configuration des émulateurs *hard*, il existe la solution d'utiliser un vrai PC distant qui communique avec votre plateforme et qui s'affiche sur votre écran. C'est pour cela que je l'ai appelé émulateur virtuel.

Le principe consiste à ajouter à un PC une couche de communication qui existe également sur votre Mac, terminal X ou station Unix. Cette couche ne gère que l'affichage et les périphériques d'entrées/sorties tels que souris, clavier et autres. Sur votre écran vous avez alors une fenêtre représentant un PC avec lequel vous pouvez travailler. Comme c'est un vrai PC qui est derrière, vous avez l'assurance d'une compatibilité parfaite, même pour le réseau…

J'ai pu tester la solution de NCD (Wincenter) par l'intermédiare du DMA qui l'a installé pour ses besoins internes ***(www.ncd.com/pwin/pwinds.html)***. Jean-Claude Berney a bien voulu me créer un compte afin de pouvoir me connecter et utiliser un bout de leur serveur PC. Et cela marche du premier coup…

Chaque utilisateur a son propre espace disque et a l'impression d'être le seul à utiliser le PC. Le réseau est parfaitement visible pour le partage de fichiers, les imprimantes et les services Internet comme la messagerie. Les périphériques de votre station comme ceux du serveur PC sont également disponibles: disquettes, cartes sons, et tous les ports standards. Les accès réseau générés par cet émulateur sont raisonnables, et ne créent pas plus de trafic qu'une autre application Xwindows.

Un gros atout pour cette solution est la possibilité de gestion et de configuration de tous les comptes sur le serveur, ce qui simplifie beaucoup le travail et est également une preuve de fiabilité du système.

Insignia, roi des rois en matière d'émulateur PC propose également sa solution équivalente à NCD: *NTRIGUE* ***(www.insignia.com/ntarea.html)***. On peut penser que l'avenir est plutôt de ce côté là.

CONCLUSION

Pour faire un bilan des différentes solutions exposées, il est intéressant de calculer les coûts:

- *Emulation* soft*: même si le logiciel ne vaut que quelques centaines de francs, il faut rajouter à cela une extension mémoire, voire disque. Le résultat a une performance moyenne et n'est pas très stable surtout dans un environnement Macintosh.*
- *Emulation* hard*: le coût concerne essentiellement le prix d'une carte PC sur laquelle il faut souvent rajouter de la mémoire. Les performances sont bonnes mais le partage de certaines ressources comme le réseau n'est pas aisé.*
- *Emulation virtuelle: il faut faire un gros investissement pour l'achat du serveur (PC + disque + périphérique pour env. 10Kf) auquel il faut rajouter l'achat du logiciel qui permet ce partage en réseau (env. 10Kf). On voit alors que l'investissement est rentable pour un groupe d'utilisateurs d'au moins 10 personnes. Par contre cette solution est très intéressante par des économies indirectes grâce à sa gestion centralisée qui minimise l'administration et son utilisation sur de vieilles stations pouvant être utilisées comme terminaux X.* ■

FI spécial été 97 le techno-quotidien

APPEL AUX ARTICLES

Il est temps d'y penser, d'aiguiser vos neurones et de huiler vos mulots, vous êtes tous invités à participer à ce numéro spécial, qui comme chaque année sera centré sur un thème. Nous avons choisi pour cet été, d'y raconter comment l'informatique et surtout la société de l'information dans laquelle nous évoluons modifient nos comportements.

Quelques axes de réflexion

- les nouveaux comportements de consommateur (porte-monnaie électronique, magasins virtuels, le *data mining*…)
- les changements dans les métiers (*office sharing*, médecine, imprimerie, enseignement…)
- les nouveaux médias (musique numérique, journaux électroniques, l'information à la demande…)
- l'aide aux handicapés (au niveau de l'audition, de la vue, de l'intégration sociale…)
- la nouvelle société (justice, police, trafic routier, le village global…).

N'hésitez pas à nous contacter pour nous soumettre d'autres idées ou transmettre les coordonnées de personnes (même extérieures à l'école) susceptibles d'écrire un article. Le délai pour la soumission des articles est fin juin, mais un contact préliminaire est souhaitable!

Jacqueline Dousson, SIC, dousson@sic.epfl.ch

Le Coin des Curieux

LE WORD NOUVEAU EST ARRIVÉ…

par Jacqueline Frey, arobasque

Installation de MS-Office 97

Rien à signaler de particulier si ce n'est que Windows 95 et 8 MO de RAM au minimum sont requis. 110 MO d'espace disque sont demandés pour l'installation standard de la version comprenant Word, Excel, PowerPoint et Outlook SANS Access. Prévoir allègrement 180 MO d'espace disque pour être à l'aise.

Lancement de Word

Passés la surprise et le choc de la nouvelle interface, on s'aperçoit que les changements ne sont pas aussi traumatisants que supposés (en tout cas pour l'instant…, faudra voir à l'usage). Je propose de passer en revue les différents menus de Word afin de mettre la main sur les commandes que nous avions l'habitude d'utiliser. En effet, certaines d'entre-elles ont été renommées, déplacées, voire supprimées et de nouvelles commandes sont venues se rajouter.

1ère remarque

Les boutons sont jolis, les petites animations de menu sont mignonnes comme tout, les retraits sur la règle sont enfin un peu plus gros mais je soupçonne mon vieux 486 d'être encore plus poussif qu'avant (si, si, c'est possible) et mon écran d'être décidément trop petit. Le menu Macro que j'avais rajouté dans la version Word 7 est toujours là, les barres d'outils personnalisées, les insertions automatiques et corrections automatiques, les modèles de lettres «spécial arobasque» sont également là. C'est bien sûr la moindre des choses mais je tenais à le confirmer, nous voilà donc tous rassurés.

2ème remarque

Où sont donc passées les commandes Modèles du menu Fichier, les Annotations et les Champs de formulaire du menu Insertion, la Numérotation des titres du menu Format, les Synonymes et Coupure de mots auxquelles - je l'avoue - je m'étais sentimentalement attachée. Sans parler de l'aide intuitive à laquelle je faisais parfois appel, mais ne le dites à personne.

Afin de faciliter un peu la tâche des utilisateurs qui comme moi font le passage de Word 7 à Word 97, voici listés, par menu, les changements intervenus dans la nouvelle version.

MENU FICHIER

La commande: Modèles

Elle a été renommée **Modèles et compléments**. Elle est transférée dans le menu **Outils**. Permet d'attacher un modèle.

Commande: Envoyer

Elle a été renommée **Destinataire du message** (dans la commande **Envoyer vers**).

Commande: Routage

Elle a été renommée **Destinataire du routage** (dans la commande **Envoyer vers**).

Nouvelle commande: Enregistrer au format HTML

Elle n'est disponible que si vous avez installé l'outil de création de pages Web. Elle vous permet de convertir un document Word en une page Web.

Nouvelle commande: Versions

Elle vous permet d'enregistrer plusieurs versions du document actif dans un seul document, et afficher un résumé des versions pour le document actif.

MENU EDITION

Commande: Insertion automatique

Elle a été transférée dans le menu **Insertion** (c'est logique) auquel on a rajouté un sous-menu classant les insertions par catégories.

Commande: Signet

Elle a été transférée dans le menu **Insertion**.

Nouvelle commande: Coller comme lien hypertexte

Elle vous permet de coller et de mettre en forme le contenu du presse-papiers en tant que lien hypertexte. Vous devez ensuite modifier le lien hypertexte avec Lien hypertexte (menu **Insertion**), afin d'indiquer un fichier ou un URL auquel associer le texte sélectionné.

suite

MENU AFFICHAGE

Commande: Annotation

Elle a été renommée **Commentaire** et transférée dans le menu **Insertion**.

Nouvelle commande: Lecture à l'écran

Ce mode optimise l'affichage de façon à faciliter les déplacements dans un document grâce au plan situé dans la zone de gauche. Extra!!

MENU INSERTION

Commande: Annotation

Elle a été renommée **Commentaire**. Utilisez cette commande pour insérer une marque de commentaire au point d'insertion ou à la sélection et afficher un volet dans lequel vous pouvez saisir vos commentaires (comme vous aviez l'habitude de le faire avant…)

Commande: Cadre

Elle a été supprimée. Pour insérer des zones de texte, utilisez la commande **Zone de texte**. Pour insérer des images, utilisez le sous-menu de la commande **Image**. Notez la commande permettant la création de formes automatiques (comme dans PowerPoint).

Commande: Champ de formulaire

Elle a été supprimée. Utilisez la commande **Barre d'outils** du menu **Affichage**, sous-menu **Formulaires** pour insérer des champs de formulaires.

Nouvelle commande: Lien hypertexte

Elle vous permet d'insérer un lien à un fichier Microsoft Office, HTML ou autre. Ce fichier peut être situé sur un site Web interne ou externe quelconque ou sur tout serveur.

MENU FORMAT

Commande: Numérotation des titres

Elle a été transférée sous la commande **Puces et numéros** du menu **Format** sous l'onglet Hiérarchisation.

Les commandes: Cadre et image

Elle ont été supprimées et remplacées par la commande **Zone de texte**, **Image** ou **Options de Forme** selon l'objet qui est sélectionné.

Nouvelle commande: Orientation du texte

Elle vous permet de changer l'orientation du texte dans des cellules de tableau, des zones de texte et des formes automatiques. Extra!! Résultats visibles en mode Page uniquement.

Nouvelle commande: Arrière-plan

Elle vous permet de changer la couleur et la texture de l'arrière-plan de votre document pour une visualisation en ligne. Résultats visibles en mode Lecture à l'écran uniquement.

MENU OUTILS

Les commandes: Orthographe et Grammaire

Elles sont maintenant fusionnées dans la commande **Grammaire et orthographe**.

Commande: Synonymes et Coupure de mots

Elle a été transférée dans la commande **Langue**.

Commande: Révisions

Elle a été renommée **Afficher les modifications** dans la commande **Suivi des modifications**. La fonctionnalité Fusionner révisions est renommée Fusion de documents et transférée dans le menu **Outils**.

Nouvelle commande: Synthèse automatique

Elle vour permet de créer automatiquement une synthèse du document actif.

Nouvelle commande: Rechercher la référence

Elle vous permet de trouver une expression ou un mot sélectionnés dans le Bibliorom Larousse. (Cette commande n'est disponible que si le Bibliorom Larousse est installé.)

Nouvelle commande: Assistant courrier

Elle facilite et prend en charge la création, la modification et la mise en forme de lettres.

Commande: Visual Basic éditor

Nouvelle option de la commande **Macro**. Passe à l'environnement Visual Basic, qui permet la création et la modification de code Visual Basic.

MENU TABLEAU

Nouvelle commande: Dessiner un tableau

Elle vous permet de dessiner des tableaux et des cellules avec le pointeur de la souris, qui prend alors l'aspect d'un crayon ou d'une gomme. Extra!!

suite

Nouvelle commande: Uniformiser la hauteur des lignes

Elle vous permet d'appliquer une même hauteur aux lignes ou aux cellules sélectionnées.

Nouvelle commande: Uniformiser la largeur des colonnes

Elle vous permet d'appliquer une même largeur aux colonnes ou aux cellules sélectionnées.

MENU AIDE

Commande: Rubriques d'aide Word

Elle a été renommée **Aide sur Microsoft Word**. Affiche alors le Compagnon Office, qui permet de visualiser l'aide de Word. Sympa mais peut-être un peu lourd!!

Commande: Aide Intuitive

Elle a été supprimée. L'accès à l'aide intuitive se fait par le biais du Compagnon Office… toujours lui.

Nouvelle commande: Qu'est-ce que c'est?

Elle remplace le bouton d'Aide de la barre d'outils Standard de Word 7.

Commande: Microsoft sur le Web

Elle permet de se connecter à la page d'accueil de Microsoft sur le World Wide Web (nécessite un accès à Internet).

Les informations contenues dans cet article ne suffisant pas – j'en suis parfaitement consciente – je saisis l'occasion pour vous informer que des cours de transition Word 7 – Word 97 ont bien entendu été prévus par le SIC.

A bientôt pour de nouvelles aventures…■

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
& 021 693 2244 - Fax: 021 693 2220
E-mail: scalfo@sic.adm.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)**
N° 4128 A (7 demi-jours)
20, 22, 27, 29.05, 03, 05 & 10.06.97 13h30 - 17h15
Introduction au Macintosh, à Internet (Présentation d'Internet, Intranet • Netscape et navigation sur le Web • Recherche et moteurs de recherche • Bookmarks), ClarisDraw 1.0, Word 6.0, Excel 5.0, FileMaker Pro 3.0.

N° 4128 B (6 demi-jours)
12, 17, 19, 24, 26.06 & 01.07.97 13h30 - 17h15
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.

BASES DE DONNÉES

FileMaker Pro 3.0 **avancé** **(5 demi-jours)**
N° 4130 03, 05, 10, 12 & 17.06.97 08h15 - 12h00

METTEUR EN PAGE

PageMaker 6.0 **(3 demi-jours)**
N° 4143 16, 18 & 25.06.97 13h30 - 17h15
Prérequis: connaissances approfondies de Word et d'un logiciel de dessin!

PRÉSENTATION

PowerPoint 4.0 **avancé, niv. 1** **(1 jour)**
N° 4139 09.06.97 08h15 - 17h15

PowerPoint 4.0 **avancé, niv. 2** **(1 jour)**
N° 4140 23.06.97 08h15 - 17h15
Prérequis: connaissances de PowerPoint !

TABLEUR

Excel 5.0 **avancé, niv. 1** **(3 demi-jours)**
N° 4138 16, 18 & 20.06.97 08h15 - 12h00

Excel 5.0 **avancé, niv. 2** **(1 jour)**
N° 4136 13.05.97 08h15 - 17h15

suite

Excel 5.0 **macros** **(2 demi-jours)**
N° 4137 30.05 & 06.06.97 08h15 - 12h00

TRAITEMENT DE TEXTE

FrameMaker 5.1 **introduction** **(3 demi-jours)**
N° 4127 22, 27 & 29.05.97 08h15 - 12h00

Mailing (Word - FileMaker) **(1 demi-jour)**
N° 4134 03.07.97 08h15 - 12h00
Prérequis: connaissances de base de Word et FileMaker Pro!

Word 6.0 **avancé** **(5 demi-jours)**
N° 4145 30.04, 01, 14, 15 & 21.05.97 13h30 - 17h15

Word 6.0 **formulaires** **(1 demi-jour)**
N° 4133 02.06.97 08h15 - 12h00

Word 6.0 **longs documents** **(2 demi-jours)**
N° 4132 26 & 28.05.97 08h15 - 12h00

Word 6.0 **trucs + astuces** **(1 demi-jour)**
N° 4135 19.06.97 08h15 - 12h00

TRAITEMENT D'IMAGE

PhotoShop 3.05 **(3 demi-jours)**
N° 4142 11.06.97 08h15 - 17h15
&13.06.97 08h15 - 12h00

Director **débutant** **(3 demi-jours)**
N° 4141 20, 21 & 23.05.97 08h15 - 12h00

WEB

Editeur HTML (Web-Weaver) **(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 (modèle client-serveur, Internet, hypertexte, URL) • les commandes HTML les plus importantes • les différentes méthodes pour créer/récupérer des textes pour un serveur
N° 4131 26, 28.05 & 02.06.97 13h30 - 17h15
Prérequis: avoir déjà utilisé un logiciel (Mosaic ou Netscape)

COURS SUR PC - WINDOWS'95

Cycle de base complet A + B **(13 demi-jours)**
N° 2609 A (7 demi-jours)
21, 26, 28.05, 02, 04, 09 & 11.06.97 13h30 - 17h15
Introduction à Windows'95, Internet (Présentation d'Internet, Intranet • Netscape et navigation sur le Web • Recherche et moteurs de recherche • Bookmarks), PowerPoint 97, Word 97, Excel 97, FileMaker Pro 3.0.

N° 2609 B (6 demi-jours)
16, 18, 23, 25, 30.06 & 02.07.97 13h30 - 17h15
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 de Windows 95.

BASES DE DONNÉES

Access'97 **introduction** **(2 demi-jours)**
N° 2625 23 & 30.05.97 08h15 - 12h00

Access'97 **avancé** **(2 jours)**
N° 2626 24 & 26.06.97 08h15 - 17h15

Access Basic **(2 demi-jours)**
N° 2627 09 & 13.06.97 08h15 - 12h00

DESSIN

Designer 6.0 **(2 demi-jours)**
N° 2612 23 & 25.06.97 08h15 - 12h00

LANGAGE DE PROGRAMMATION

VisualBasic 4.0 **intro 1** **(2 demi-jours)**
N° 2619 05 & 07.05.97 08h15 - 12h00

VisualBasic 4.0 **intro 2** **(4 demi-jours)**
N° 2628 03, 05, 10 & 17.06.97 13h30 - 17h15

METTEUR EN PAGE

MS-Publisher **(3 demi-jours)**
N° 2608 21, 26 & 28.05.97 08h15 - 12h00
Vous, utilisateurs Windows, vous n'avez pas la chance d'avoir des cours FrameMaker ou PageMaker, en revanche nous vous proposons MS-PUBLISHER. Merveilleux programme de mise en page, facile à mettre en oeuvre, MS-PUBLISHER vous permettra de réaliser de superbes documents.
Prérequis : connaissances de base d'un traitement de texte

PRÉSENTATION

PowerPoint'97 **avancé, niv.1** **(2 demi-jours)**
N° 2623 27 & 29.05.97 13h30 - 17h15

TABLEUR

Excel'97 **avancé, niv. 1** **(3 demi-jours)**
N° 2622 02, 04 & 06.06.97 08h15 - 12h00

Excel'97 **avancé, niv. 2** **(1 jour)**
N° 2620 12.05.97 08h15 - 17h15

Excel'97 **macros** **(2 demi-jours)**
N° 2621 20 & 22.05.97 13h30 - 17h15

TRAITEMENT DE TEXTE

Word'97 **avancé** **(5 demi-jours)**
N° 2610 27, 29.05, 03, 05 & 10.06.97 08h15 - 12h00

Word'97 **formulaires** **(1 demi-jour)**
N° 2616 12.06.97 08h15 - 12h00

Word'97 **longs documents** **(2 demi-jours)**
N° 2613 17 & 19.06.97 08h15 - 12h00

Word'97 **mailing** **(1 demi-jour)**
N° 2614 02.07.97 08h15 - 12h00
Prérequis: connaissances de base de Word.

suite

Word **transition 7.0 à'97** **(1 demi-jour)**
N° 2617 13.05.97 13h30 - 17h15

Word'97 **trucs + astuces** **(1 demi-jour)**
N° 2615 30.06.97 08h15 - 12h00
Prérequis: connaissances de base de Word.

WEB

FrontPage **(3 demi-jours)**
N° 2607 28, 30.04 & 05.05.97 13h30 - 17h15
N° 2611 16, 18 & 20.06.97 08h15 - 12h00

COURS SUR STATIONS DE TRAVAIL

Unix **introduction** **(1 demi-jour)**
N° 3158 20.05.97 08h15-12h00
Prérequis: débutant = aucune connaissance de Unix.

JAVA, programmation avancée **(2 jours)**
N° 3149 01 et 02.05.97 09h00-17h30
Objectifs du cours:
Manipulations graphiques • Interface graphique utilisateur • Applications indépendantes • Threads et Multithreading • Entrees-sorties fichiers • Programmation réseau Java • Utilisation de méthodes natives
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.

Langage C **(5 jours)**
N° 3156 09 au 13.06.97 09h00 - 17h30
Le langage C est un langage de programmation à usage général, de la famille des langages algorithmiques impératifs. Il a été créé en 1970 par Denis Ritchie des Belle laboratories dans le but d'utiliser ce langage pour réécrire UNIX. Dès son origine le langage C est intimement lié à UNIX et le succès de ce système d'exploitation a fortement contribué à la popularité de ce langage.
Prérequis: Expérience de la programmation. Des connaissances de l'environnement Unix seraient un plus.
Objectifs et contenu:
• Historique et présentation • Évaluation • Aspect général d'un programme • Les délimiteurs • Les identificateurs • Les constantes • Les variables • Déclaration des variables • Liste de variables • Les variables constantes • Les variables volatiles • Les types de base • La fonction PRINTF • Les instructions • Les instructions conditionnelles • Instructions d'aiguillage • Instructions répétitives • Instructions associées aux boucles • Les fonctions • Les classes d'allocation des objets • Initialisation des variables • Les tableaux • Les pointeurs • Les structures • Champs de bits • Les unions • Les énumérations • Taille des types et variables • Les opérateurs • Arguments d'un programme C • Combinaison de types • Définition de type • Conversion de type • Directives de compilation • L'environnement de programmation C • La librairie standard LIBC • Les entrées/sorties • Manipulation de chaîne • Allocation mémoire • Librairie mathématique.

Programmation sur Origin 2000 **(5 jours)**
N° 3155 26 au 30.05.97 09h00-17h30
Les sujets suivants seront traités:
Architecture Origin 2000
Logiciels
Environnement de programmation parallèle
Optimisation mono-processeur
Outils de developpement
Prérequis:
Avoir une très bonne expérience dans l'un de langages suivants: C, C++ ou Fortran. Connaissances de PVM et de la programmation parallèle exigée.
Attention: ce cours est réservé au personnel de l'EPFL.

TCL / TK **(3 demi-jours + 1 jour)**
N° 3159 du 20 au 22.05.97 14h00-17h30
et 23.05.97 09h00-17h30
Description:
Tcl/Tk (développé par J. Ousterhout) permet d'écrire rapidement des applications avec interface graphique (GUI) pour l'environnement Xwindows.
Tcl est un langage de script, interprété. Tk est la boîte à outils pour la partie graphique X. Les applications écrites en Tcl/Tk ont l'aspect Motif tout en étant créées très rapidement grâce à l'interface de haut niveau et à la nature interprétée du langage.
Connaissances préalables exigées:
Aucune en particulier, mais une connaissance de Unix, d'un langage de scripts comme sh, csh... et une vision globale de Xwindows seront des atouts précieux.
Objectifs et contenu:
A l'issue du cours, les participants seront capables d'écrire la plupart des applications en Tcl/Tk sans se référer trop souvent à la documentation et de construire des interfaces graphiques agréables à utiliser.
Pour plus de renseignements, voir le serveur:
http://slwww.epfl.ch/SIC/SL/logiciels/TclTk/TclTk_announce.html

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.

Formulaire d'inscription en page 16

OÙ VONT LES VIEUX ORDINATEURS POUR MOURIR?

par Jacqueline Dousson, SIC

Attention: *cet article ne concerne naturellement que vos appareils privés, pour les appareils professionnels, votre employeur pourvoit certainement au recyclage des appareils, comme c'est le cas pour l'EPFL.*

Votre PC (ou votre télévision, ou votre fax…) a, au moins, 5 (ou 6 ou10) ans de service derrière lui. Son état ne vous permet pas de le revendre à un amateur, vous aimeriez donc lui assurer une fin tranquille et conforme à vos idéaux de citoyen du monde, respectueux de la planète bleue. Evitez alors les *objets encombrants* ou autres dépôts de ferraille, car tout appareil électronique comporte des composants fortement polluants (piles, condensateurs, plastiques ignifugés). Il existe une filière *officielle* de retraitement pour les appareils de bureautique usagés, mise en place par la SWICO (Association suisse des fabricants et importateurs de matériel électronique de bureau) à laquelle 66 sociétés adhèrent aujourd'hui: les sociétés signataires imputent sur le prix de vente une taxe de recyclage, qui permet un retraitement approprié du matériel usagé que les signataires s'engagent à reprendre. Afin de rendre obligatoire cette taxe d'élimination une révision de la LPE (Loi fédérale sur la protection de l'environnement) a été acceptée par le Parlement. La nouvelle ordonnance qui en découle et qui est en consultation prévoit donc que tout fournisseur de matériel électronique ou électroménager s'engage à reprendre gratuitement les appareils hors d'usage et à garantir un circuit de retraitement approprié.

Des filières de post-traitement spécialisé se sont donc mises en place. Près de Lausanne, BIRD associé à l'atelier POLYVAL de St-Sulpice a traité 110 tonnes de matériel électronique en 1996. La Fondation POLYVAL (atelier pour handicapés) démonte le matériel et le sépare en une quarantaine de catégories destinées à des traitements différents (condensateurs, verre, piles, revêtements luminescents des tubes cathodiques, plastiques ignifugés, métaux recyclables, etc.). Chaque catégorie est ensuite envoyée par benne vers le poste de retraitement adapté.

Et vous, qu'est-ce-que vous devez faire de votre PC? si vous le portez directement au centre régional de collecte SWICO (Cargo Domicile, route de Genève 97 à Lausanne), il sera repris gratuitement.

Si vous voulez en savoir plus sur le système mis en place par la SWICO, sur l'évolution de la législation ou sur les prestations du centre BIRD-POLYVAL, passez à Computer'97 sur le stand 2808 (du 22 au 25 avril). ■

ETRE VU ET CONNU SUR LE WEB

par Jacqueline Dousson, SIC

Cet article reprend les informations envoyées le 11 mars dernier aux délégués informatiques des départements par le comité éditorial Web de l'EPFL.

Un des buts d'une page Web institutionnelle est d'aider à la diffusion de l'image de cette institution, notamment auprès de ceux qui ne la connaissent pas. En effet les utilisateurs qui connaissent l'EPFL en trouveront facilement l'adresse dans les différents annuaires (par pays, par thèmes) qui existent. Par contre, là où il faut être particulièrement attentif, c'est par exemple fournir à un néo-zélandais qui cherche des renseignements sur les cellules solaires à effet photovoltaïquene l'adresse du *laboratoire de photonique et interfaces* du DC.

Qu'utilisera ce néo-zélandais?

- les annuaires spécialisés, et là, les responsables de serveurs d'unités doivent veiller à ce que leur serveur y soit présent. Un passage par l'arbre fourni par Yahoo (www.yahoo.com) peut vous conduire à ces annuaires spécialisés si vous ne les connaissez pas (dans le cas précédent: Science/Energy/Solar Power/ Photovoltaic Power Information Resources ...)
- les outils de recherche, comme AltaVista. C'est dans le but d'être repéré et en bonne position par les outils tels AltaVista qu'ont été rédigées les cinq règles suivantes.

suite

Il faut savoir qu'AltaVista par exemple utilise un algorithme qui donne le plus de poids à une page si les mots cherchés se trouvent dans le titre, dans les mots-clés (règle 4), dans la description (règle 2) et apparaissent souvent (sans être consécutifs). Des mots dans une figure ou un annuaire ne sont pas reconnus.

RÈGLE NO 1:
UTILISER UN TITRE EXPLICITE

Le titre défini par **<title>** n'est pas affiché dans votre page, mais apparaît comme titre donné par les engins de recherche et est utilisé comme signet (bookmark). Ce titre doit être explicite. Comme vous avez 50 caractères pour afficher votre titre sur AltaVista, utilisez-les avec une mise en ligne attractive et le plus de mots-clés possible.

RÈGLE NO 2:
AJOUTEZ UN CHAMP «DESCRIPTION»

Si ce champ n'existe pas, les 150 premiers caractères de votre page sont traités comme un *welcome text* qui sera affiché par AltaVista. L'effet est en général assez désastreux. Vous pouvez écrire votre page pour que ces 3 lignes soient lisibles comme un résumé contenant le maximum d'information, mais ce qui est prévu par HTML est que ce texte figure dans un champ

<meta name = "description" content = "Votre en-tête pour le résumé AltaVista">

Vous êtes alors libre de faire votre mise en page sans choisir les mots du début.

RÈGLE NO 3:
RÉPÉTEZ LES MOTS IMPORTANTS ET UTILISEZ DES SYNONYMES

N'hésitez pas à répéter dans le texte les mots qui caractérisent le document et leurs variantes. Par exemple, *Swiss Metro* ne donne rien de significatif, mais *SwissMetro* conduit aux bons liens. Ne réutilisez pas tel quel vos résumés de recherche prévus pour des documents imprimés. Répétez les mots importants et ajoutez des synonymes et équivalents en langues étrangères. Et naturellement ajoutez les liens aux pages décrivant plus précisément les différents travaux de votre unité. Cette page deviendra une page aiguillage bien positionnée par les outils de recherche. Elle amènera chez vous tous les spécialistes de votre domaine, qui peuvent avoir une vision et un vocabulaire différent.

RÈGLE NO 4:
AJOUTEZ DES MOTS-CLÉS

Cette règle est très importante. Définissez 5 à 20 mots-clés, synonymes, équivalents étrangers dans le champ:

<meta name = "keywords" content = "vos mots-clés séparés par des virgules">.

Le champ des mots-clés est bien valorisé par la plupart des outils de recherche.

RÈGLE NO 5:
DÉCLAREZ LES NOUVEAUX SERVEURS

Un nouveau serveur ou une page d'accueil qui n'est pas référencée par un serveur doit être annoncée; l'indexation des pages référencées se fait alors automatiquement dans la semaine. AltaVista dispose d'une case *Add URL*.

En se déclarant à l'adresse

http://free.submit-it.com

l'information est transmise aux principaux outils de recherche à la mode (AltaVista, Yahoo, Excite, etc.).

En résumé, une page HTML a la forme suivante:

```
<html>
  <head>
  <title>Titre explicite </title>
  <meta name = "description" content =
     "description 150 caractères">
  <meta name = "keywords" content =
     "5 à 20 mots-clés">
  </head>
  <body>
  Votre page avec ses ordres de
    formatage et ses liens
  ......
  </body>
</html>
```

Vous pouvez en savoir plus sur l'utilisation des «meta» en relation avec les outils de recherche en consultant

http://www.enterprise.net/stars/Location/Meta/Tag.html. ■

OFFRE D'EMPLOI

OFFRE D'EMPLOI

Développeur Windows 95 pour portage logiciel

Recherchons un développeur Windows 95 pour portage d'un logiciel développé initialement sur Macintosh en Pascal Objet.
Contrat à durée déterminée (environ 6 mois)

Compétences requises:

- parfaite connaissance de l'environnement PC
- connaissance de Toolbox Macintosh
- connaissance du bâtiment et des sciences de la construction souhaitée.

Employeur:

Ecole Polytechnique Fédérale de Lausanne, département d'architecture, Institut de technique du bâtiment (ITB).

Contact:

François Iselin – ☎ 021/693 32 23

CALENDRIER

AVRIL 97

Mardi 22	14h30	Salle Conférences SIC	**MacLine**— Groupe des utilisateurs Mactintosh François Roulet, ☎ 693.4590, 🖃 roulet@sic.adm.epfl.ch
du Mardi 22 au Samedi 26		Palais de Beaulieu	Computer'97 (lire FI-3/97 en page 21)
Jeudi 24	15h00	Salle Conférences SIC	**CI** — Commission Informatique M. le prof. Germond, ☎ 693.2662, 🖃Alain.Germond@lre.de.epfl.ch
Mardi 29	14h15	Salle Conférences SIC	**CTI** — Commission Technique Informatique M. Reymond, ☎ 693.2210, 🖃 reymond@sic.adm.epfl.ch
Mercredi 30	14h00	Salle Conférences SIC	**Présentation** de produits BORLAND **S'inscrire chez Marielle Flütsch**, ☎ 693.4587, 🖃 flutsch@sic.adm.epfl.ch

MAI 97

Mardi 13	08h30	Salle polyvalente du SIC	**Comité de rédaction du FI** J. Dousson, ☎ 693.2246, 🖃 dousson@sic.adm.epfl.ch
Jeudi 15	14h15	Salle Conférence SIC	**PolyPC** — Groupe des utilisateurs d'IBM PC et compatibles Ch. Zufferey, ☎ 693.4598, 🖃 zufferey@sic.adm.epfl.ch

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:

.....

.....