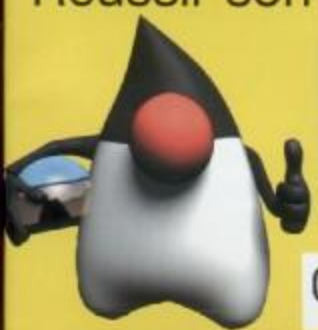
Programmez!
www.programmez.com
Mensuel - Mars 2007 - N°95
LE MAGAZINE DU DÉVELOPPEMENT

LES COULISSES DES PROJETS OPEN SOURCE

Comment les développeurs travaillent ? Qui les finance ?
Réussir son projet : témoignages



eclipse 2.1 THE ECLIPSE PROJECT

OpenOffice.org

ObjectWeb Open Source Middleware

MySQL

Mozilla

Microsoft .net

- Ma première application avec Windows Presentation Foundation
- Programmer Pocket Outlook

DÉVELOPPEMENT WEB

Cap sur Ruby on Rails, Rails et PHP

JAVA
- Des rapports faciles avec BIRT
- Comprendre le Design Pattern Décorateur
- Les algorithmes récursifs

TECHNIQUE
- Déboguer à distance avec Eclipse

WEB
- Découvrir Spring 2.0

STRATÉGIE
- Utiliser les tests unitaires automatisés

Richard Stallman

"Liberté, Égalité, Fraternité"

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

SOMMAIRE N°95

> Actus

L'actualité en bref 6
Agenda 6

> Evénements

L'Open Source s'expose à Paris 8
TechDays 2007 : le 1er événement informatique en France 22

> Interactivité

L'actualité programmrez.com 10

> Gourou

" Liberté, égalité, fraternité " interview de Richard Stallmann 12

> SGDB

En bref 16

> Projets

Xiti passe à .Net 18
Le test unitaire automatisé : une question de stratégie 19

> Dossier Open Source

Comment réussir son projet Open Source ? 24
Faut-il choisir l'Open Source 26
OpenOffice, l'autre grand de la bureautique 28
Mozilla : 240 versions à maintenir 30
Sun et l'Open Source / Objectweb et ses 120 projets 32
NetFilter : la couche pare-feu de Linux /MySQL et sa forge 33
JBoss et la communauté / La distribution Mandriva 34
Gnome : un des projets phares de GNU Linux 35
Wengo : de la téléphonie ouverte / Eclipse ouvert et fermé 36
DotNetNuke : projet ouvert.net 37
Témoignage d'un développeur 38
Logiciels Libres : des technologies reconnues et des business models qui rapportent 39
L'Open Source Valley 42
Comment héberger un projet libre 43

> Dossier : Ruby 2e partie

Démarrer avec Ruby On Rails, 2e partie 46
Ruby on Rails, AJAX et RJS 49
Cake : PHP aussi prend le train ! 53

> Développement Web

Les nouveautés de Spring 2.0 57

> Technique

Déboguer à distance avec Eclipse 59

> Code

PocketPC : échanger vos contacts grâce au Compact framework 2.0 61
Mes débuts en WPF 1re partie 65
BIRT : Eclipse devient un outil de création de report 68
Les algorithmes récursifs 71
Comprendre le Design Pattern Décorateur 74
DataBase2Code s'occupe du code et des données 77

> Temps libre

Ludique 80
Les livres du mois 82

CD-Rom 95 PROGRAMMEZ !

EasyEclipse for RubyOnRails
(Windows + Linux)
EasyEclipse est un projet visant à fournir des packages Eclipse pré-configurés pour différents usages, sous forme de "distributions". Chaque distribution se compose d'une archive contenant la plate-forme Eclipse et différents plugins. Distribution pour le développement Ruby et Ruby On Rails.

Ruby 1.8.5 (Windows + Linux)
Dernière version du langage de programmation Ruby

Ruby GNOME 2
Ruby-GNOME 2 est une suite d'extensions Ruby pour l'environnement de développement GNOME 2. Il s'agit de la nouvelle génération du projet Ruby-GNOME.

Rails 1.2.2
Rails est un framework, basé sur le langage Ruby, permettant le développement rapide d'applications web utilisant le modèle MVC (Modèle-Vue-Contrôleur).

RadRails 0.7.2 (Windows + Linux)
Environnement graphique de développement, RadRails inclut de base un éditeur à onglets avec coloration syntaxique, des aides à la saisie, un explorateur de fichiers du projet et différents modules pour accélérer le développement.

Eclipse SDK 3.3 (Windows)
Dernière version du SDK d'Eclipse

Looking Glass (Linux)
Interface graphique Java en 3D

VirtualBox 1.3.4 (Windows + Linux)
Logiciel de virtualisation. Libre et gratuit pour une utilisation personnelle.

FormFeedHours
Version développeur de FormFeedHours (Framework de mapping Bidirectionnel pour .Net 1.1) qui installe en même temps PopHours (générateur d'objets métier pour Access et SqlServer)
Edité par Wygwam

CodeCharge Studio 3.1
Pour le développement de sites Web interactifs ou des systèmes Internet et Intranet alimentés par des données.
Version limitée : 20 jours - Kapitec Software est le distributeur exclusif pour la France.

NOUVEAU : l'emploi

PROGRAMMEZ.COM CHANGE !

Programmez!
LE MAGAZINE DU DÉVELOPPEMENT

Rédaction : redaction@programmez.com
Directeur de la Rédaction : Jean Kaminsky
Rédacteur en Chef : François Tonic
Ont collaboré : J.M. Maman, F. Mazué, C. Padych, M. Vialette, N. Mérouze, R. Piacentini, M. Isvy, H. Darmet, J.B. Evain, F. Moreau, E. Leblond, G. Delamarre, A. Cogoluegnes, A. Fillatre, T. Clarenc, X. Vanneste, S. Dieudonné, S. Develay
Dessins : Michel Piédoue
Maquette : AJE Conseils
Publicité : régie publicitaire : ECC, Stéphane Kaminsky - Pour la publicité uniquement : Tél. : 01 49 76 06 42 - Fax : 01 49 76 07 35 publicite@programmez.com
Editeur : Go-02 sarl, 6 rue Bezout - 75014 Paris Coordination@programmez.com - Dépôt légal : à parution - Commission paritaire : 0707K78366 - ISSN : 1627-0908 - Imprimeur : ETC - 76198 Yvetot
Directeur de la publication : Jean-Claude Vaudecrane
Le numéro comporte un CD Rom et un encart STR pour les abonnés.

Abonnement : Programmez 22, rue René Boulanger, 75472 Paris Cedex 10 - abonnements.programmez@groupe-gli.com Tél. : 01 55 56 70 55 - Fax : 01 55 56 70 20 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30.
Tarifs abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 45 € - Etudiant : 39 € - CEE et Suisse : 51,83 € Algérie, Maroc, Tunisie : 55,95 € Canada : 64,33 € Tom : 79,61 € - Dom : 62,84 € Autres pays : nous consulter.
PDF : 35 € (Monde Entier) souscription en ligne.

Édito

A votre bon cœur !

"Un ordinateur sur chaque bureau et dans chaque foyer", prophétisait Bill Gates, au début de Microsoft. Plus précis, un premier ministre français, annonçait en novembre 2002 : " Un ordinateur pour chaque foyer en 2007 " ! La France était en retard, avec 42% des foyers équipés. Jean Pierre Raffarin a perdu son pari : le taux est de 55% en 2007. Certains ironiseront, en cette période préélectorale sur les promesses (ou les vœux pieux) des politiques !

Encore plus généreux et utopique : " un portable pour chaque enfant ! " One Laptop Per Child (OLPC), veut livrer aux gouvernements des pays émergents un portable à 100 dollars ! Un échantillon de cette machine représentait le clou du salon Linux Solutions, présenté par Red Hat. Open source et écologique, l'ordinateur est sous Linux ; sa mémoire Flash remplace le disque dur. Il dispose d'une webcam et d'une connexion sans fil haute vitesse, consomme 3 watts, avec une autonomie de 8 heures ! (*)

D'ailleurs, le 30 janvier, le parvis de la Défense voyait à la fois l'entrée du salon Linux Solutions, au CNIT et le feu d'artifice organisé par Microsoft devant la Grande Arche, pour célébrer le double lancement de Windows Vista et d'Office 2007, cibles emblématiques de l'open source. Le hasard de cette proximité hautement symbolique illustre l'enjeu important du logiciel libre et de l'open source.

Photo : Gaël Donat

En anglais, le terme de " free " a la redoutable ambiguïté de signifier à la fois " libre " et "gratuit". Le paradoxe reste économique. Une étude commanditée par Bruxelles se penche sur l'impact des logiciels libres sur l'industrie européenne du logiciel. Elle débute en rappelant que l'information et la communication représentent 10% du PNB des pays développés et plus de la moitié de leur croissance économique !

Open source : 12 milliards d'euros de codes sources

Le rapport estime la valeur des développements open source en Europe à l'équivalent de 12 milliards d'euros. Si les entreprises clientes avaient dû produire ce code, elles auraient financé 131 000 années/hommes de développement. Dans les faits, les dépenses effectives des entreprises ne se sont élevées, d'après le rapport, qu'à 10% de cette valeur, soit 1,2 milliards d'euros. J'avoue que ces calculs me laissent perplexe. Est-ce que cela veut dire que les développeurs sont les dindons de la farce, qu'ils auraient perdu 131 000 années de salaire ?

Mais les entreprises auraient-elles acquis les licences au prix fort ?
Par ailleurs, le rapport estime que développer du libre est aussi la meilleure formation professionnelle : les compétences engrangées en participant à des projets libres " aident les développeurs à trouver des emplois et peuvent aider à créer et entretenir des petites entreprises ". Pour encourager le développement du logiciel libre, le rapport propose des mesures fiscales : le don de code serait équivalent à une donation à une oeuvre de charité. Il ferait bénéficier particuliers et entreprises d'exonérations fiscales.

Serait-ce une idée pour Bill Gates, dont la fondation en fait un des premiers bienfaiteurs du globe ? Et si une partie des logiciels de Microsoft devenait généreusement libre ? Microsoft paierait moins d'impôts...
Boutade, bien sûr ! Ou... politique-commerciale-fiction ? Le sujet est grave et déchaîne les passions. Mais caricaturer les prophètes du logiciel, dans la presse, échappe encore aux tribunaux.

Jean KAMINSKY

** Le projet a été créé par une organisation sans but lucratif, à l'initiative du Media Lab, au MIT, et de son fondateur Nicolas Negroponte A ses côtés, deux autres pionniers de l'informatique : Alan Kay, l'inventeur de la programmation orientée objet et qui oeuvra chez Atari et Apple dans leurs débuts, et Seymour Papert créateur du langage Logo. Parmi les entreprises associées : AMD, Google, Red Hat.*

PROCHAIN NUMERO

N°96 - Avril 2007 -
Parution le 30 mars 2007

Virtualisation
Comment utiliser les environnements de virtualisation dans vos développements ? Quels outils choisir ?

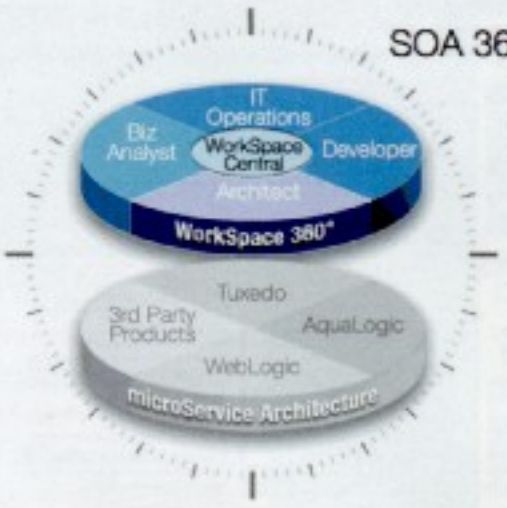
Futur du logiciel
Prospective : les changements à attendre dans les 5, 10 et 15 ans du côté des langages, outils et systèmes.

VIRTUALISATION

BEA mise sur le service et la virtualisation

Deux grands changements vont intervenir en 2007 pour BEA. Le premier concerne l'apparition du micro service. Dans le cadre de SOA 360, l'ensemble des outils et des fonctions de l'éditeur (gammes WebLogic, AquaLogic et Tuxedo) seront exposés sous forme de micro services que l'on pourrait consommer et inclure dans ses développements. Au lieu d'acheter un environnement complet, on pourra utiliser uniquement telle ou telle fonction, apportant une grande souplesse d'utilisation. L'agrégation des micro services se réalisera dans l'environnement de développement Workspace 360. On y trouvera une partie design pour l'interface, une partie développement, une partie métier, une partie architecte. Au cœur de l'outil, se trouve Workspace Central qui fournit les outils de collaboration, le cycle de vie SOA. Cette nouvelle offre est un pas important vers l'adoption de la SOA grâce à un environnement totalement tourné services. L'éditeur compte terminer cette "migration" d'ici le premier trimestre 2008. Restera à préciser le modèle économique : impossible de continuer à vendre des licences par processeur physique ! Notons que l'architecture micro serveur (ou mSA) est extensible. Cela signifie qu'un micro service tiers pourra être référencé dans l'environnement.

Il y a quelques semaines, BEA annonçait la virtualisation des applications. Dans ce but, l'éditeur a signé un accord avec WMWare pour pouvoir mettre en place des machines virtuelles Jrockit dans l'environnement de virtualisation. On ne virtualise pas l'application cliente mais plutôt l'environnement d'exécution, de production. Cela doit permettre de mieux répartir la charge entre différents serveurs même s'ils sont géographiquement distants. On optimise ainsi son serveur d'application. Côté administration, BEA devrait s'interfacer avec les outils d'hypervision du marché et non proposer son propre outil. Là aussi, la question du modèle économique se pose car la licence par processeur physique n'a plus de sens ! Cela devrait être disponible d'ici 2-3 mois.



Après avoir amélioré WebLogic Server et pris Eclipse pour Workshop (IDE de développement), BEA tente avec SOA 360 et la virtualisation de prendre de vitesse la concurrence, notamment les serveurs open source. À n'en pas douter, l'éditeur a marqué un point important sur la facilité de mise en place d'une architecture SOA et surtout dans sa conception. Le fait de proposer des services et non plus des outils monolithiques annonce un changement radical de la perception de l'outil de développement. Un pari peut-être audacieux, car un peu en avance de phase, mais qui pourrait bien se révéler gagnant dans les mois à venir.

Agenda

MARS

Le 13 Mars 2007
CNIT – Paris La Défense,
Vocal Expo
Toute l'information pour décider de l'avenir de votre relation client ! www.vocalexpo.com

Du 07au 08 mars 2007
Paris La Défense, CNIT,
- Documation 2007
Le rendez-vous annuel de la gestion de contenu et de document
www.documation.fr/

- **RTS embedded**
- **Display 2007**
- **Machine to Machine**
  www.birp.com/

Du 19 au 21 mars 2007
Paris La Défense, Espace Grande Arche
Sun Tech Days,
Http://fr.sun.com/sunnews/events/

22 mars – Paris, hôtel California Champs Elysées.
Tournée européenne
Value2 2007, Red Hat et JBoss
www.redhat.fr/promo/roadshow/index.php

Le 27 mars 2007
Paris La Défense, CNIT,
IBM - Tendances Logicielles
www-5.ibm.com/fr/events/innovationsoftware.html

MAI

XP Day France 2007
Paris, les 2 et 3 mai 2007
La conférence sur les méthodes agiles ! www.xpday.fr/

ETRANGER

Du 15 au 21 mars 2007
Allemagne, Hanovre, Parc des Expositions, **CeBIT 2007**
Le plus grand salon informatique d'Europe. www.cebit.de

ACCORD

Lyon et Microsoft France travaillent ensemble !

Bill Gates en personne était présent pour annoncer un partenariat avec Lyon, en présence de Gérard Collomb, sénateur-maire de la ville. M. Collomb débuta la conférence par une déclaration forte : " c'est un moment historique ". L'accord se décline en trois points : soutenir et dynamiser l'économie numérique de Lyon, développer l'internet au service du citoyen et enfin un important volet sur la coopération internationale en vue de " réduire la fracture numérique ! " Pour Lyon, selon le chiffre avancé par M. Collomb, l'économie de l'information / numérique représente environ 30 000 personnes. Le soutien à la filière jeux vidéo constitue sans doute le point central de l'action de Microsoft France à Lyon. Un concours récompensera le plus créatif, afin de l'aider à réaliser son jeu et à le publier, si tout va bien, sur Xbox Live ! Cette aide passera par un accès aux technologies et outils de développement de l'éditeur. Cela passera aussi par un soutien et une promotion envers les jeunes pousses et sociétés lyonnaises, une aide technologique, un support marketing et un soutien à l'internationalisation.

L'internet citoyen n'a pas été oublié. Il s'agit de rendre disponible au plus grand nombre Internet et les outils multimédia et particulièrement aux jeunes. Mais il s'agit aussi de mettre en place une eAdministration proactive facilitant la vie du citoyen et la communication entre l'administration et les habitants. Un volet éducatif fait partie de l'accord. Enfin sur la fracture numérique, Lyon souhaite aider à déployer la technologie et l'information numérique vers les pays défavorisés afin de réduire le fossé entre nous et les autres pays.

Bill Gates a pris la parole durant une dizaine de minutes. Il a rappelé l'importance de cette semaine avec Vista et Office. Il a considéré l'accord comme excitant et de bon sens et indiqué qu'il fallait saisir les opportunités (pour les sociétés). L'opportunité étant une voie de croissance économique, M. Collomb a rappelé que sur la eAdministration, Microsoft sera un guide stratégique mais ne se taillera pas pour autant la part du lion dans l'administration lyonnaise. Sur le volet financier, Bill Gates n'a pas voulu préciser les choses.

L'Open Source s'expose à Paris

Rendez-vous incontournable du monde de l'open source et GNU Linux en France, le salon Linux s'est tenu à Paris fin janvier. Plus animée que l'édition 2006, la version 2007 fut marquée par plusieurs tendances. La partie communautaire - associative a retrouvé son dynamisme d'antan.

Plus que jamais le côté professionnel et entreprise a marqué ce salon. Si en 2006, le matériel tenait une place importante, cette année, on constate une présence plutôt discrète du matériel (avec une forte présence réseau). Apple et Sun ne participaient pas au salon. HP, bien que sans stand était tout de même présent.

Les logiciels et les services se taillaient la part du lion sur les stands. La question n'est plus de savoir si je dois utiliser des outils ouverts mais qu'est-ce qu'ils peuvent m'apporter de plus ou de différent. La présence de solutions de ERP ou de CRM confirme l'orientation entreprise, tout comme les nombreuses sociétés de services dans le développement, mais surtout le conseil et la formation (qui connaît une forte demande). Car si l'entreprise est prête à passer à des outils libres / ouverts, encore faut-il disposer de compétences pour faire la migration éventuelle et surtout le support, l'assistance. Le support demande un point sensible. Les récentes annonces d'IBM ou de Sun sur ce domaine sont éloquentes quant aux besoins. L'utilisateur averti pouvait aussi trouver ce qu'il lui fallait. Par contre, comme l'an dernier, le grand public était quelque peu le parent pauvre. Très clairement, le salon s'adresse aux professionnels. Le contenu des conférences allait en ce sens.

Dynamisme à tous les niveaux

La partie associative, communautaire était particulièrement dynamique cette année avec notamment les projets KDE, Mozilla, OpenOffice. Pour le fun et les amateurs de robot, il y a le Tux Droid, une initiative ouverte originale. Au lieu d'avoir un chien ou un lapin, là, on bénéficie d'un pingouin équipé de capteurs, d'interface d'entrée / sortie, de processeurs. On programme le Tux Droid à partir d'un GNU Linux. Le langage utilisé est le Python (même si les couches basses sont conçues en C). Les premiers exemplaires seront vendus courant mars, uniquement sur internet. Le prix sera sans doute inférieur à 100 euros. Pour le moment, aucune version Windows n'est prévue. Pour en savoir plus : http://www.kysoh.com. Il y avait aussi hAckFest, un tournoi de développement, en équipe ou en individuel. Une soixantaine de personnes participèrent. Côté GNU Linux, les principaux acteurs systèmes étaient là, avec les plus grands stands : Mandriva, Red Hat et surtout Novell avec SuSe. On pouvait aussi noter le grand retour d'Ingres, une mythique base de données qui est depuis 2006 séparée de l'éditeur CA. Notons que si la virtualisation est un domaine hyper actif depuis quelques mois, le thème se révéla discret sur le salon hormis quelques stands comme celui de Microsoft.

Quelques solutions retenues :

- **ZaurusFR :** association pour la promotion d'une mobilité utilisant les logiciels ouverts et tournant sous GNU Linux ou BSD. Sur le marché mobile, GNU Linux et les solutions ouvertes demeurent encore en marge, surtout dans le modèle téléphone.
- **NovaForge :** une nouvelle solution conçue par Bull. Il s'agit de pouvoir gérer et stocker ces projets dans une forge logicielle. L'outil ne sera pas packagé ou téléchargeable. Bull le vendra notamment sous forme d'une solution complète avec installation, configuration. NovaForge est le résultat de l'agglomération de plusieurs dizaines de composants ouverts (versionning, outil de développement, build...).
- Cartographie ouverte : **Camptocamp** a présenté le framework de cartographie cartoweb, actuellement en version 3.1. Il s'utilise dans une architecture SOA et peut se développer en PHP. Il supporte Ajax. Le framework est disponible en licence GPL.
- **Unify – Gupta :** l'éditeur Gupta a été racheté par Unify. La version 5 de Gupta Team Developer a été dévoilée juste avant le salon Linux. Désormais, GNU Linux est pleinement supporté. L'éditeur Unify a surtout présenté la nouvelle structure issue du rachat de Gupta. L'éditeur a pu démontrer NXJ Developper (environnement de développement d'applications web).
- **Tobflow :** parmi les solutions ERP, CRM... on peut retenir Tobflow d'Audaxis, une solution de gestion de workflow et de formulaires en LGPL. Il utilise Java EE et supporte l'ensemble des serveurs d'application du marché.
- **Hermitage Solutions** présentait la gamme complète de Collax pour le travail collaboratif et la messagerie, avec une partie sécurité. Les serveurs sont disponibles en serveurs dédiés ou en boîte, prêts à installer.
- **Sogeti** montrait son framework telosys s'appuyant sur Ajax et fonctionnant en environnement Java EE. Telosys est adapté aux applications métiers. Il est hébergé chez ObjectWeb et est sous licence LGPL.

Un peu en aparté, le salon fut l'occasion de " subir " deux événements majeurs. Le premier étant l'incontournable lancement de Windows Vista et d'Office 2007 à deux pas du salon et surtout l'annonce spectaculaire du passage à GNU Linux (solutions Novell) de PSA de plus de 20 000 postes et serveurs.

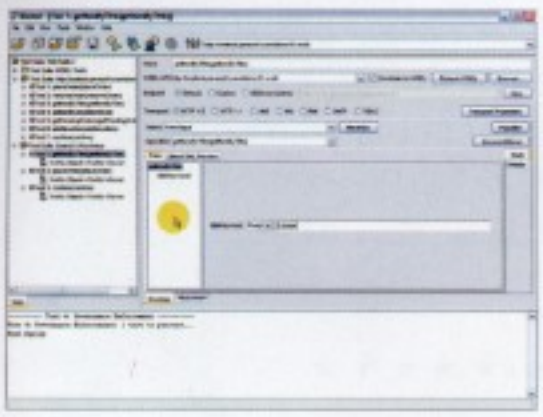
François Tonic

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

Actu

SOA

Parasoft renforce la qualité SOA



L'éditeur a dévoilé la version 5 de son outil SOAtest. Il s'agit d'un outil de test de non régression. Cette version doit renforcer la sécurité, la fiabilité et la conformité des architectures SOA, tout en simplifiant le travail de test des différentes couches. Une des grandes nouveautés est le module d'implémentation des politiques permettant de mettre en œuvre des politiques SOA définies. L'outil s'intègre maintenant avec le référentiel BEA Aqualogic Enterprise. " La nouvelle version de Parasoft SOAtest nous permet d'être plus productif lors de la création de cas de tests fonctionnels de base ou avancés ", commente Mark Carlson, architecte SOA pour la société de consulting Conneva .Inc. " L'interface utilisateur améliorée nous permet de créer des tests rapidement pour les opérations individuelles de service Web ou les conversations impliquant des opérations multiples. Des fonctionnalités supplémentaires des outils de BD nous permettent de créer des messages de tests à partir du contenu de tables plus facilement. "

DOCUMENT

Abbyy supporte le PDF/A

L'éditeur Abbyy propose un kit de développement de son OCR, FineReader Engine, supportant les langues thaï, européennes, l'hébreu, le cyrillique, etc. Surtout, il inclut l'export au format PDF/A (A pour Archive). Le SDK permet d'intégrer et de personnaliser FineReader. Les API sont utilisables en différents langages dont le C++, VB, Java. Vous pouvez distribuer vos applications dotées des fonctions d'OCR (licence runtime requise). Site : http://www.abbyy.com

OUTIL

Novalys lance Visual Expert 5.7

L'outil Visual Expert, permettant notamment la gestion du changement, arrive en version 5.7. Cette mouture inclut une recherche " full text ", le support étendu de PL/SQL, la possibilité d'explorer les références de procédures stockées dans un code PowerBuilder. L'outil prend aussi en charge Internet Explorer 7. De plus, l'éditeur annonce la sortie de Visual Expert pour Transact SQL. Site : http://www.visual-expert.com

RÈGLES

Ilog lance Jrules 6.5

L'éditeur français Ilog vient de lancer la version 6.5 de son outil de gestion de règles métiers, Jrules. Il permet maintenant de déployer des services de décision transparents (des services web basés sur des règles) dans des architectures SOA. On bénéficie aussi des requêtes sémantiques. L'utilisateur peut ainsi réaliser des requêtes sémantiques dans le référentiel de règles afin d'évaluer l'impact d'une modification de politique. Côté développeur, Eclipse 3.2 est pris en charge par le Rule Studio, une première dans la gestion de règle selon l'éditeur.
Site : http://www.ilog.com/products/jrules

@Sienerinformatique

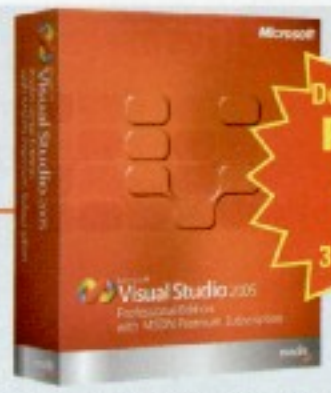
Fournisseur d'outils de développement depuis plus de 15 ans.

TOUT CE QU'IL FAUT POUR DÉVELOPPER :

DEVELOPPEMENT.NET

Visual Studio 2005 Professional

Des outils conçus pour relever les défis de tous vos développements



Derniers jours PROMO OPEN jusqu'au 31 Mars 2007

La gamme Visual Studio 2005 Professional vous permet de travailler dans un environnement de développement performant avec des outils de modification et de débogage ainsi que des concepteurs graphiques intuitifs qui vous permettent de bâtir rapidement des applications Windows, web et mobiles.

Avec abonnement MSDN Pro

Comprend Visual Studio Pro avec en plus SQL Server, Tous les principaux Système dont Vista, et la gestion 2 incidents.

Licence Open 2 ans à partir de, par an, 492 €HT

Derniers jours PROMO OPEN -> 31 Mars

Avec abonnement MSDN Premium

Derniers jours PROMO OPEN -> 31 Mars

Licence Open 2 ans à partir de, par an, 1115 €HT

Comprend Visual Studio Pro avec en plus Tous les produits Microsoft Server (Exchange, Windows, SQL), Tous les produits Office, dont Office 2007 Visual Source Safe, MBS, et la gestion 4 incidents.

Visual Studio 2005 Team System

Visual Team Tester, Développeur, ou Architect lic. Open à partir de 3899 €HT

- Visual Team Tester, Développeur, ou Architect
- Visual Team Foundation Server
- Visual Team Suite
- Visual Team Database

Nous consulter pour devis sur mesure

Microsoft CERTIFIED Partner

COMPOSANTS NET ASP JAVA

Infragistics

NetAdvantage for .NET 2006 Volume 3

à partir de 725 €HT

Support .net2 et Visual Studio, Contrôles Windows Presentation Foundation, interfaces stylées Office 2007...

NetAdvantage for JSF 2006 Volume 2

à partir de 575 €HT

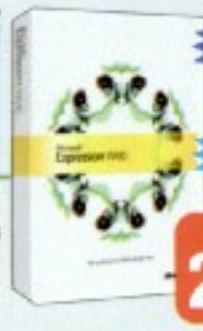
Composants serveurs Java type AJAX pour applications J2EE (grille, menu, arborescence, tableau, calendrier ect.)

DESIGN

Microsoft Expression Web

Créez un site Internet à votre image

Un outil professionnel de conception dédié à la création de sites modernes fondés sur des standards et proposant une qualité supérieure sur le Web.



Nouveau produit

Enfin disponible

à partir de 295 €HT

CONTACTEZ NOUS AU

01 41 83 75 20

www.sienerinformatique.com

L'actualité programmez.com

" Comme chaque mois maintenant, je vous invite à découvrir sur cette page les nouveautés et interactions autour du site programmez.com. Découvrez les résultats du sondage Vista de février, le nouvel espace emploi, le mini-site vista. Je vous attend sur les services interactifs du portail des développeurs "

NOUVEAU

Mini-site Vista

A l'occasion de la sortie de Windows Vista le 30 janvier dernier, Programmez.com a ouvert un mini-site, regroupant l'ensemble des actualités, articles, blogs et forums autour du nouveau système d'exploitation.
http://www.programmez.com/vista.php

• Forums

ASP : problème Selection Item dans DropDownList

D4rkTiger : " J'ai créé plusieurs DropDownlist dans un tableau et lorsque je choisis un employe les DropDownList doivent laisser apparaître la section de lEmploye aux heures prévues. Le Problème c'est que je n'arrive pas GRAPHIQUEMENT à le faire! "

Frederic Mazue : " Quand tu fais une sélection dans une DropDownList, un événement se déclenche. Si tu écris un gestionnaire pour l'événement et que tu mets dedans du code du genre de

```
AutreDropDownList.SelectedItem = "Heureprevue"
```

Tu dois pouvoir arriver à tes fins. "

http://www.programmez.com/forum/viewtopic.php?t=615

• Who's Who

Noham CHOULANT
Ingénieur en développement informatique
30 ans - " Ingénieur en développement informatique, je suis également vice président de GUWIV (groupe utilisateur de windows vista")

Jean-François JOLY
Demandeur d'emploi
26 ans - " Le développement c'est ma grande passion. Motivé, informé et volontaire j'aborde la vie avec sérénité. Positif mais réaliste "

Consultez l'annuaire Who's Who de Programmez :
http://www.programmez.com/whoswho.php

NOUVEL ESPACE EMPLOI

Riche en actualité et en articles dédiés à l'emploi, Programmez vous permet enfin de consulter les offres publiées et de postuler directement en ligne.

• Tips

Niveau : ●●● **Note des lecteurs :** ★ ★ ★ ★ ★

Auteur : **Frédéric Mazué**

Une mise à jour majeure de **PostgreSQL** doit être précédée d'une petite manipulation afin d'éviter toute mauvaise surprise lors du démarrage du nouveau démon. A ce titre, attention aux mises à jour automatiques des distributions Linux, en général assez brutales avec cette question. Une mise à jour PostgreSQL est majeure si l'on incrémente la première décimale: par exemple de 7.1 vers 7.2 mais pas de 7.1.1, vers 7.1.9.

Voici la marche à suivre

- sous user postgresql, dumper toutes les données:

```
pg_dumpall > SAVE
```

- arrêter le démon
- pour prendre une précaution supplémentaire, copier le répertoire des données de PostgreSQL. (Si vous avez assez d'espace disque bien sûr)

```
cp /var/lib/pgsql/data /var/lib/pgsql/data.old
```

- effacer les données:

```
rm -r /var/lib/pgsql/data/*
```

- procéder à la mise à jour:
- lancer le démon
- sous user postgresql, réinjecter les données sauvegardées:

```
psql template1 -f SAVE
```

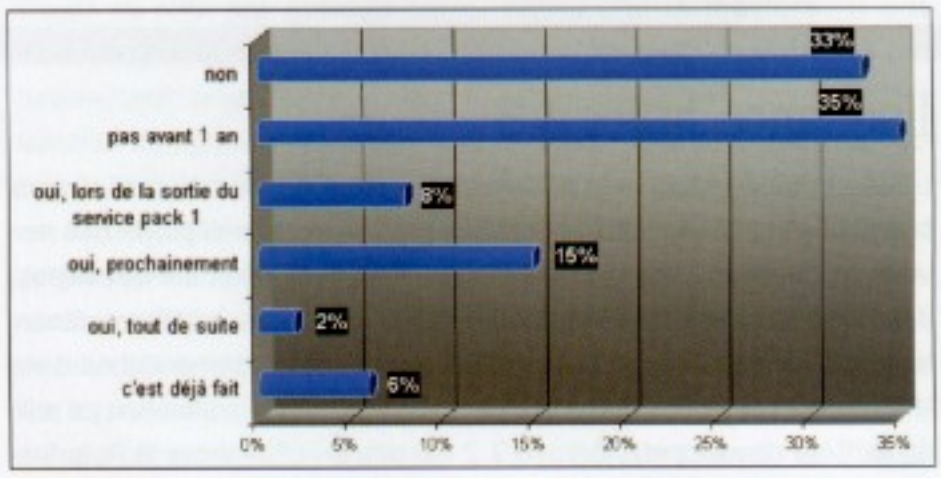
Si tout va bien /var/lib/pgsql/data.old pourra être supprimé. Si par extraordinaire tout allait mal, rétablir /var/lib/pgsql/data à partir de /var/lib/pgsql/data.old et revenir à l'ancienne version.

http://www.programmez.com/tips.php

• Sondage

Passerez vous à Windows Vista ?

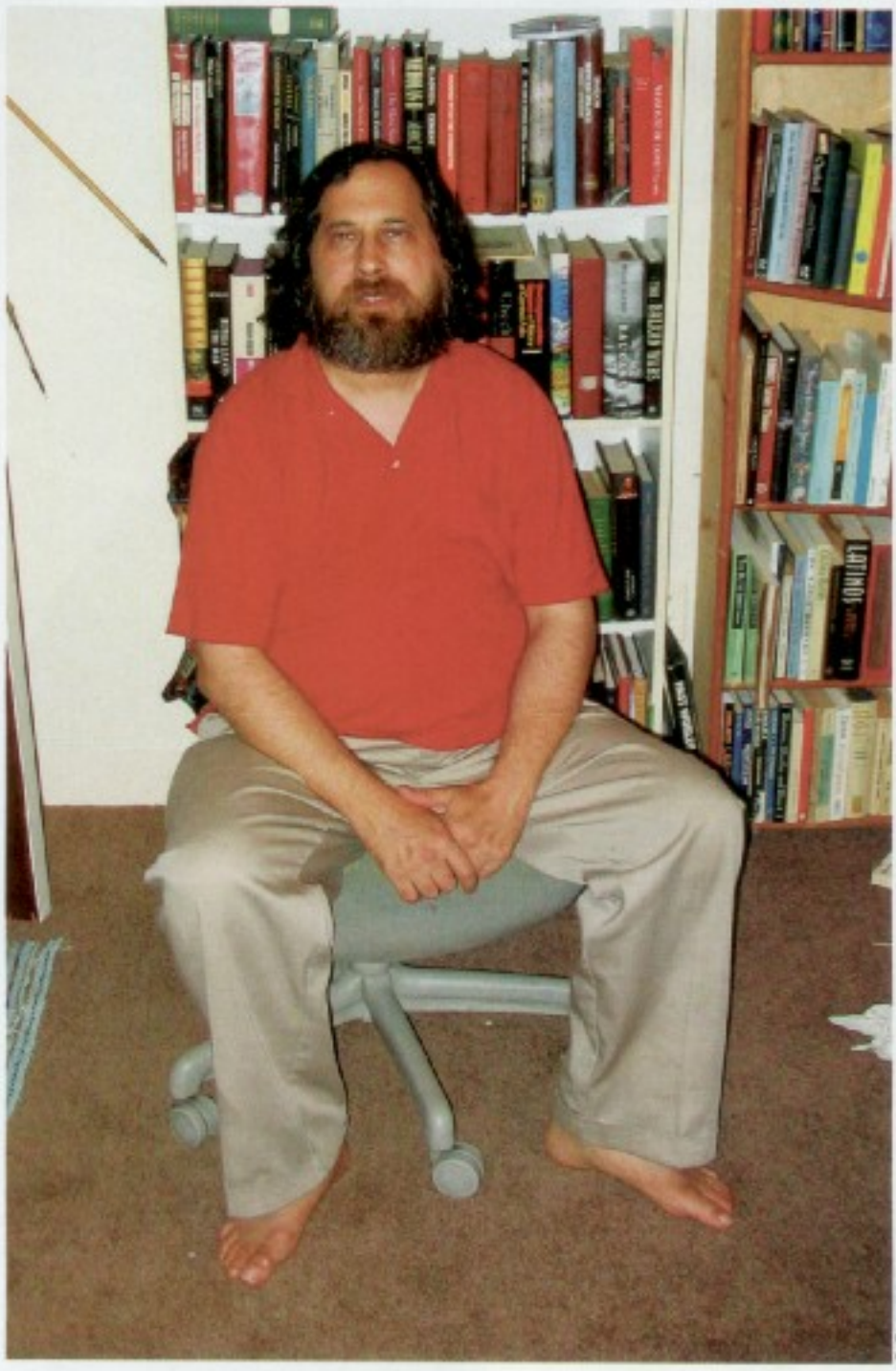
Ce sondage a mis en avant votre hésitation face au nouveau système d'exploitation de Microsoft. Vous êtes 25 % seulement à envisager la migration vers Windows Vista à court terme, ou à être déjà passé à Vista.



Sébastien DEVELAY
sebastien@sdgraphik.com - webmaster de programmez.com

" Liberté, égalité, fraternité "

Figure incontournable depuis des années du monde libre, Richard Stallman se pose en défenseur des libertés des utilisateurs. Programmez ! a voulu en savoir un peu plus sur sa vision, son travail et les valeurs qu'il défend.



D.R.

Programmez ! : Comment pourriez-vous définir le logiciel libre et sa philosophie ?

Richard Stallman : Je puis expliquer la base philosophique du logiciel libre en trois mots: liberté, égalité, fraternité. Liberté, parce que les utilisateurs sont libres. Egalité, parce qu'ils disposent tous des mêmes libertés. Fraternité, parce que nous encourageons chacun à coopérer dans la communauté. Plus spécifiquement, un programme est libre si l'utilisateur dispose des quatre libertés essentielles :

1. La liberté de faire tourner le programme, comme tu veux.
2. La liberté d'étudier le code source, et de le changer, de manière que le programme fasse ce que tu veux.
3. La liberté de diffuser des copies exactes du programme aux autres, quand tu veux.
4. La liberté de diffuser des copies de tes versions modifiées, quand tu veux.

Avec toutes ces libertés, le programme respecte les libertés des utilisateurs. Si une seule de ces libertés manque, le programme est " privateur " (pas libre). C'est-à-dire que son système social de développement et son utilisation sont un problème social.
Les utilisateurs méritent la liberté, et les libérer est notre mission. Nous voulons remplacer les programmes " privateurs " par des programmes libres. Nous avons commencé en 1984 le développement du système d'exploitation GNU, avec le but de rendre possible l'utilisation d'un ordinateur sans logiciel " privateur ".

P ! : On amalgame souvent logiciel libre et la notion d'open source, quel est votre avis sur cette confusion ? Et pourquoi en est-on arrivé à cette confusion ?

R. S. : Il y en a qui aiment le logiciel libre actuel, mais se sentent gênés par notre philosophie éthique, et préfèrent l'oublier. En 1998 ils ont formulé l'expression " open source " comme manière de parler de ces mêmes programmes libres, mais sans jamais prononcer le mot " libre ". Ils ont profité de leur expression alternative pour remplacer les valeurs éthiques par des valeurs strictement pratiques : à la place de Liberté, Egalité et Fraternité, ils mirent Rentabilité, Efficacité, et Fiabilité. Moi aussi j'apprécie ces valeurs pratiques, mais elles ne peuvent pas remplacer les Droits de l'Homme.

P ! : La notion de standard est aujourd'hui incontournable. Vous parlez de standard libre, des éditeurs comme IBM ou Sun parlent de standards ouverts. Pouvez-vous définir le standard libre ?

R. S. : Un standard est ouvert si les spécifications sont publiques. Pour être libre, il faut aussi qu'il n'y ait pas d'obstacles à l'implémentation du standard : par exemple, pas de brevet qui l'interdise.

P ! : Plusieurs grands projets sont supportés et surtout financés par les grands éditeurs (IBM, Sun, Novell, HP, Apple dans une moindre mesure, etc.). Sont-ils les seuls à pouvoir payer les recherches et le développement des grands projets comme Open Office, Eclipse, NetBeans, etc. ? Y voyez-vous un risque d'appropriation de ces éditeurs au détriment des communautés ?

R. S. : La participation des grandes entreprises dans le développement des logiciels libres apporte beaucoup au Monde Libre, mais il y a aussi des risques. Les mêmes entreprises font des logiciels " privateurs " ; par conséquent, il ne leur conviendrait pas que les clients valorisent la liberté et exigent le logiciel libre. Pour la plupart, elles ne parlent donc que d'" open source ".
Quant au risque d'appropriation, il existe, mais il ne dépend pas de la pratique de contribution. Ce risque existe pour les programmes libres porteurs des licences sans "gauche d'auteur" (ou copyleft) : c'est-à-dire, plus ou moins, quand un programme porte une licence qui n'est pas la GNU GPL ou similaire, n'importe quelle entreprise peut en faire une version

" privative ", que l'entreprise contribue au Monde Libre ou pas.

P ! : Vous parlez de GNU-Linux et non simplement de Linux. Quelle est la différence ? Qu'a apporté GNU-Linux dans le monde informatique ?

R. S. : Linux est le noyau que Linus Torvalds a écrit en 1991, et a libéré en 1992. En 1992, le système GNU était presque complet; seul le noyau manquait encore. Le noyau Linux a rempli la dernière case vide de GNU, et la combinaison, le système GNU-Linux, est le système, actuellement très utilisé, qui a offert aux utilisateurs la première opportunité d'utiliser un PC sans céder la liberté au logiciel " privateur ".

P ! : Vous êtes une personnalité incontournable du logiciel libre. Comment pourriez-vous définir votre rôle, votre travail ?

R. S. : Ma grande ambition est d'être le libérateur du cyber-espace. Je ne sais pas si ça se réalisera de mon vivant, mais je ferai mon possible pour que nous y arrivions un jour.

P ! : Vous rencontrez régulièrement les gouvernements et politiques. Comment ressentez-vous leur compréhension du logiciel libre ? De quelle manière présentez-vous cela ?

R. S. : Il y a parfois des législateurs qui apprécient les idées éthiques du logiciel libre. Le nouveau président de l'Équateur a compris en 15 minutes cela, et a annoncé son appui à la fin de la réunion. En France, Ségolène Royal nous a appuyé, ainsi que les Verts. Le plupart des législateurs actuels n'ont pas de vision éthique ni sociale, et ils obéissent au dictat de l'argent. Quand on leur parle, la tactique suggère naturellement de leur présenter le sujet en termes économiques: de proposer la migration pour faire des économies. Mais je ne veux pas le faire, parce que présenter le sujet sans mentionner les Droits de l'Homme c'est accepter qu'ils ne valent pas grand-chose. Même si cette approche réussit à encourager la migration, elle ne fait rien pour diffuser les idées du logiciel libre. Pour le mouvement du logiciel libre, ce n'est qu'une distraction.Je laisse donc cette méthode aux autres, y compris les entreprises qui peuvent attirer l'attention des mêmes législateurs beaucoup plus que moi. Moi, je promeus la liberté.

P ! : L'écologie, l'économie d'énergie sont désormais des préoccupations. Le logiciel libre peut-il y jouer un rôle important ?

R. S. : Pas directement. Indirectement le logiciel libre joue un rôle. L'origine de beaucoup des problèmes globaux est la domination des entreprises sur la démocratie. Les traités dits " de libre commerce " les aident à imposer leur contrôle. Le logiciel "privateur" est un des aspects de ce phénomène, et notre campagne pour le remplacer agit donc à l'encontre de la domination de la vie par les entreprises.

P ! : La licence la plus populaire est la GPL de GNU. Aujourd'hui, il existe des dizaines de licences plus ou moins "identiques". Sun, Apple, Microsoft proposent les leurs. Cette multiplication des licences pose-t-elle problème ? Que répondez-vous à ceux qui les critiquent pour leur manque de protection envers la privatisation?

D.R.

R. S. : Les licences libres ne sont pas "identiques". La Licence Publique Générale de GNU (GNU GPL) est une licence " gauche d'auteur " (copyleft) : elle exige que toute version du programme soit libre, c'est-à-dire, que tous les utilisateurs de toutes les versions aient les libertés essentielles pour l'ensemble du code de leur version. Il y a aussi deux autres grandes classes de licences libres: les licences permissives, qui ne défendent point la liberté des utilisateurs, et les licences mi-" gauche d'auteur " (mi-copyleft), qui défendent la liberté sur chaque fichier source mais n'interdisent pas l'adjonction de fichiers " privateurs ".

Il y a aussi des licences exceptionnelles. Par exemple, Microsoft a écrit une licence libre exprès pour interdire l'utilisation du code dans un programme sous la GPL.

Les licences qui ne pratiquent pas le copyleft ne défendent pas bien la liberté de chaque utilisateur, et permettent que le code s'utilise dans des programmes " privateurs " qui peuvent faire de la concurrence avec les versions libres. L'utilisation d'une licence faible est presque toujours un mauvais choix.

Néanmoins, l'existence d'un programme libre sous une telle licence est bien meilleure que l'absence de logiciel libre dans le domaine. Autrement dit, la liberté mal défendue est meilleure que l'absence de liberté.

■ Propos recueillis par **François Tonic**

Qui est Richard Matthew Stallman ?

D.R.

Né à Manhattan en 1953, il utilise son premier ordinateur en 1969. Au début des années 1970, il étudie à l'université d'Harvard et devient même hacker au MIT. C'est en 1983 qu'il initie le projet GNU (GNU is not Unix), un système d'exploitation libre. Il crée rapidement la Free Software Foundation puis l'année suivante, se consacre (presque) entièrement à ses projets. Il promeut rapidement la notion de copyleft. En 1989, il présente la version 1 de la fameuse licence GPL qui doit servir à son système d'exploitation toujours en développement. Quelques années plus tard, la naissance du noyau Linux incite Richard Stallman à utiliser ce noyau dans GNU, ainsi naît GNU/Linux.

Figure emblématique du monde libre, Richard mène son combat contre la gestion des droits numériques, les brevets logiciels. Il s'est opposé à la loi DADVSI, et n'hésite jamais à rencontrer les politiciens pour discuter des logiciels libres afin de les promouvoir. Il n'hésite pas non plus à prendre fermement position sur des sujets aussi variés que Harry Potter ou encore la guerre en Irak.

Site officiel : http://www.stallman.org/

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

SGBD

Tout MySQL pour 32 000 euros !

Vous cherchez une base de données en accès illimité sans limitation de processeurs ? MySQL propose depuis peu une licence entreprise illimitée pour 32 000 euros. Pour ce montant, annuel, une entreprise peut déployer autant de bases de données qu'elle souhaite, ce qui doit faciliter la gestion des licences et le déploiement. Cela couvre bien entendu le support. Il s'agit d'une assistance 24 / 7 par téléphone et support web, assurée par des développeurs MySQL. Cette licence illimitée se compose aussi de MySQL Enterprise Server et le service MySQL Network de surveillance et conseil. Bertrand Matthelié, VP marketing EMEA explique : " Nous changeons les règles du jeu : cela nous parait aberrant de facturer des millions d'euros ! Nous avons simplifié la vie des développeurs et des DBA. Aujourd'hui, nous simplifions la vie des DSI Les grandes entreprises ont déjà leur data wharehouse central, elles n'ont pas nécessairement besoin de les migrer. En revanche, elles développent des nouvelles applications, tous les jours, et de plus en plus d'applications en ligne, pour lesquelles, MySQL est un SGBD de référence ".

VIRTUALISATION

IBM renforce son offre

IBM a dévoilé une nouvelle technologie de virtualisation : sHype. Technologie dédiée aux centres de données, elle doit permettre une meilleure administration et sécurité des environnements virtuels, tout en garantissant une intégrité des données, le tout en environnement x86 et blade. sHype doit pouvoir fonctionner avec les hyperviseurs de marché. Une partie de sHype est disponible en open source et sert déjà dans le projet Xen. Pour mieux comprendre, sHype est une architecture d'hyperviseur sécurisée et peut donc être implémentée dans des solutions d'hyperviseurs. Il s'agit de fournir des services de sécurité dans l'environnement virtuel, notamment en isolant strictement les machines virtuelles.

VIRTUALISATION

Red Hat y croit en 2007

L'éditeur Red Hat confirme son intérêt pour la virtualisation. La distribution Red Hat Enterprise Linux 5 embarquera l'outil de virtualisation Xen, déjà éprouvé dans Fedora. Cette distribution Red Hat arrivera très rapidement. Jusqu'à présent, l'éditeur demeurait discret sur ses outils. Il sera possible d'installer des machines virtuelles dans la phase d'installation de la distribution. Il sera aussi possible de déplacer les machines virtuelles Xen d'un serveur à un autre.

PROCESSUS

WebMethod sort Fabric 7

L'éditeur a rendu disponible la 7e version de son environnement d'intégration et de gouvernance. La grande nouveauté de la version est l'ajout de fonctions de gestion des processus métier (composant Business Process Management Suite). L'outil utilise Eclipse pour la modélisation. La v7 inclut aussi des fonctions de gouvernance pour la SOA. WebMethod s'affirme ainsi comme un des leaders du marché des suites BPM. L'interface de Fabric et BPMS repose sur de l'AJAX et prend en compte les spécifications WSRP et JSR-168.

Site : www.webmethods.com

Perforce

Gestion *rapide* de configuration logicielle (GCL)

La fonctionalité Folder Diff,

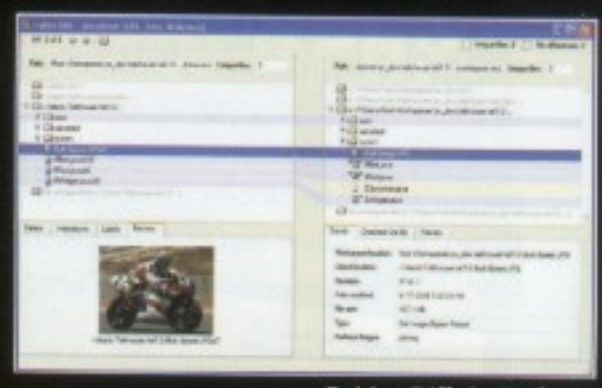
un atout de productivité du système de GCL Perforce.

Folder Diff est un outil interactif d'affichage en juxtaposition permettant de comparer l'état de deux groupes de fichiers.

À l'aide de Folder Diff, on peut rapidement déterminer les différences entre les fichiers situés dans des dossiers, des branches, des étiquettes ou sur votre disque local. Cette fonction est particulièrement utile lorsque vous devez réaliser des fusions de codes complexes.

De plus, si vous travaillez deconnecter, Folder Diff facilite la synchronisation des données avec celles du serveur Perforce lorsque vous vous connectez de nouveau au réseau.

Folder Diff n'est qu'un des nombreux atouts de productivité offerts par le système de GCL Perforce.



Folder Diff de Perforce

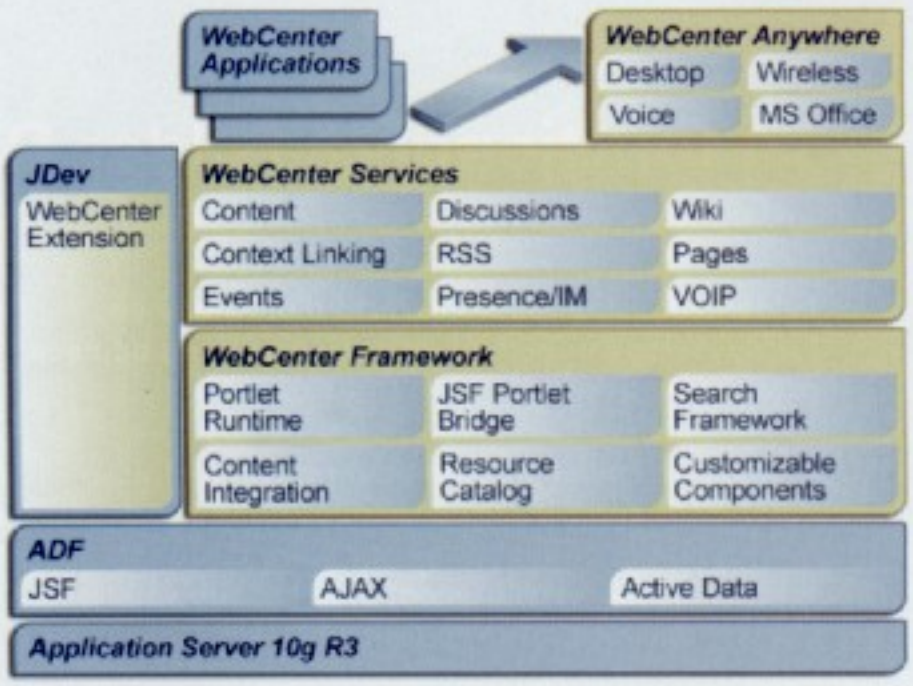
PERFORCE
SOFTWARE

Téléchargez sans conditions une copie gratuite de Perforce sur www.perforce.com. Vous bénéficiez d'une assistance technique gratuite pendant votre période d'évaluation.

MIDDLEWARE

Oracle fusionne SOA et Web 2

L'actualité d'Oracle a été très riche ces dernières semaines. Tout d'abord l'éditeur annonce la disponibilité d'Oracle Data Integrator, un outil d'intégration de données qui fait partie de la gamme Oracle Fusion Middleware et permet de réaliser de l'intégration, du chargement et de la transformation de données en milieu hétérogène. Il se base sur la technologie acquise avec le rachat de Sunopsis. Data Integrator a été optimisé pour les environnements Oracle tels que le SGBD ou le serveur d'application. L'éditeur a aussi dévoilé Oracle Secure Enterprise Search 10g Release 10.1.8. Ce moteur de recherche pour chercher l'information business ou critique permet maintenant de chercher la donnée des sources non-Oracle (ex. : Microsoft, Lotus Notes...).



D'autre part, Oracle a dévoilé Oracle WebCenter Suite. Il fait partie de la gamme Oracle Fusion Middleware. Il doit permettre d'utiliser au mieux les "avancées" du Web 2.0 dans les données et les process métiers. Il s'agit d'un environnement Java EE et il s'appuie sur les Java Server Faces. Il prend en compte le standard WSRP 2.0. Il s'agit en fait d'un environnement Web 2 mais aussi et surtout SOA. Il se présente sous la forme d'un framework. WebCenter Suite est l'offre complète comprenant le framework, une extension pour l'outil de développement Jdeveloper, une gestion de contenu. Le module Anywhere permet de supporter tout type de terminal mobile. Bref, il s'agit d'un environnement clé en main pour concevoir des applications d'entreprise intégrées à son système d'information tout en tirant parti de la SOA et du Web 2.
Site : http://www.oracle.com/technology/products/webcenter/index.html

ETL

JasperSoft lorgne sur ETL

L'éditeur JasperSoft vient de dévoiler son outil ETL en open source. Cette annonce fait suite à l'accord de partenariat avec Talent, spécialisé dans l'intégration de données, lui aussi open source. JasperETL, nom de l'outil de JasperSoft, étend l'offre de Business Intelligence de l'éditeur en proposant des fonctions d'intégration de données. Cette annonce renforce aussi la présence de l'open source dans un marché en pleine extension et concurrentiel. Le but est de démocratiser l'ETL. JasperETL comprend un designer graphique pour fournir une vue fonctionnelle des processus d'intégration, un éditeur de mapping, un éditeur pour afficher les flux de données ainsi qu'une trentaine de connecteurs ! Pour faciliter son utilisation, des assistants ont été intégrés. Une édition professionnelle sera disponible au printemps prochain. Elle intégrera un référentiel partagé de métadonnées. JasperETL est sous licence GPL.
Site : jasperforge.org

ÉCHANGE

Améliorer les échanges de données

Les échanges de données entre applications font partie des fonctions sensibles. Ce marché est aujourd'hui mature même si le coût de tels logiciels demeure stable. Avitis, éditeur de gestion de contenus d'entreprise, annonce la version 4.2 de sa solution d'échange XcMon.
Si les fonctions évoluent peu entre les différents éditeurs, Avitis joue sur les coûts, notamment en proposant aux utilisateurs de solutions concurrentes de migrer vers XcMon à un prix compétitif.
La version 4.2 ajoute de nouvelles fonctions de sécurité ainsi qu'une traçabilité des données échangées.

PORTAGE

iAnywhere sur Mac Intel

Sybase iAnywhere rend disponible maintenant sur MacOS X (tournant sur Mac Intel) SQL Anywhere 10.
L'édition développeur est elle aussi disponible en téléchargement et gratuitement sur le site de l'éditeur. MacOS X renforce ainsi son offre entreprise et notamment sur la partie SGBD et permettra de considérer MacOS X comme un système de production.

PROTECTION

CA prend en compte DB2 UDB

La nouvelle version de BrightStor Storage Resource Manager de CA est maintenant compatible avec Lotus Domino et DB2 UDB. L'objectif est de protéger les données des applications et des messageries, tout en assurant la disponibilité. L'outil doit permettre de simplifier la gestion de stockage. Cette nouvelle version étend aussi les fonctions de reporting.

MAJ

Nouveau service pack pour SQL Server 2005

Microsoft prépare son 2e service pack pour SQL Server. Disponible actuellement en pré-version (CTP), le SP2 doit permettre à la base de données de tourner correctement avec Vista et Office 2007. On disposera d'un nouveau format de stockage pour gagner en place (vardecimal) sur les données décimales et numériques. L'intégration avec Office 2007 est sur le menu notamment avec les fonctions de rapport, l'intégration avec Sharepoint 2007. Un nouveau viewer web de rapport sera disponible ainsi qu'une prise en compte plus large des sources de données. On pourra aussi gérer une base SQL Server Compact Edition à partir de SQL Server Management Studio Express. Bref beaucoup d'améliorations, de corrections de bugs et de nouveautés.

XiTi passe à .net

Célèbre pour ses outils de mesure d'audience des sites Internet, XiTi est un des leaders incontestés du marché. Alors que la version précédente était codée en ASP, XiTi a décidé une refonte totale avec .Net et ASP.Net

Démarrer un projet 100 % .Net sans connaître préalablement la technologie et le framework peut relever du défi. C'est pourtant ce qu'a réalisé l'éditeur Xiti pour ses outils Internet largement utilisés par les webmasters. " *Le projet fut assez long, plus d'un an ! Il a mobilisé 90 % de nos équipes et nous avons engagé parallèlement un gros recrutement* " commente Sébastien Carriot, CTO de XiTi. Cela n'allait pas forcément de soi, car il fallait à la fois reprendre les fonctionnalités de la version 6, insérer de nouvelles fonctions, tout en introduisant les modifications apportées à la version 6 qui devait continuer à évoluer, à côté du développement de la version 7... " Nous avions un cahier des charges mouvant " poursuit Sébastien Carriot. Même si l'éditeur utilisait la technologie ASP, il n'avait aucune expérience de .Net. " Cela a été un bouleversement " car pour pouvoir mener à bien le projet, il fallait former les développeurs en interne et recruter des profils purement .Net.

De ASP à ASP.Net : un changement en profondeur

L'utilisation de .Net n'a pas seulement modifié le langage de développement, il a marqué l'organisation des équipes. " *Nous avons intégré la vision de ce qu'est une grosse application, introduisant un travail plus modulaire, cependant, l'environnement de développement n'a guère posé de problème.* ".

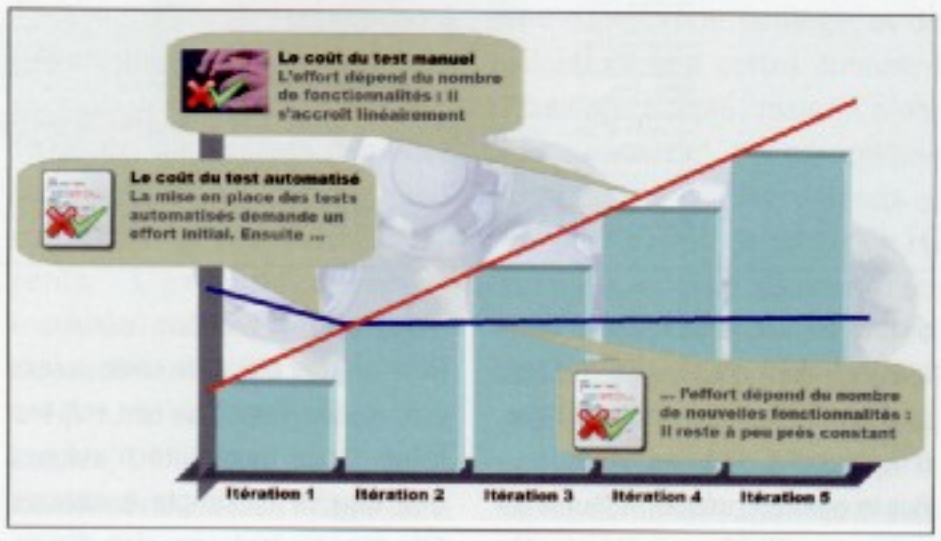
L'architecture d'une application ASP.Net ne ressemble plus à celle d'ASP qui mélange interface et code. Avec ASP.net, il faut intégrer une nouvelle rigueur, une nouvelle organisation dans les pages et comment se structure l'application. " *Nous avons dû intégrer la vision page et surtout définir une vision globale. Cela a été quelque peu perturbant. Nous avons pris conscience au cours du projet de l'importance du découplage du code et de l'interface. Il s'agit là d'une approche nouvelle pour nous* ". S'il était un peu tôt pour intégrer ce découplage dès le début, cette particularité d'ASP.Net sera sans aucun doute utilisée dans les futurs développements. Bouleversement dans le code mais aussi bouleversement dans l'organisation du projet et de l'outillage. " *On a mis en place une méthodologie collaborative, notamment sur l'utilisation du versionning. Cet aspect ne fut pas facile à gérer. Aujourd'hui, nous utilisons SourceSafe, nous testons Team System. On apprend.* ". La gestion humaine a été un autre souci. Car, pour pouvoir avoir une vue d'ensemble et mener un tel projet, il faut un architecte ou tout du moins, un responsable de haut niveau. " Il est clair que cela a modifié nos équipes. On avait un chef de projet. Désormais, nous avons un coordinateur pour prendre en charge la vision d'ensemble du projet. On commence à atteindre le niveau architecte même si ce n'est pas encore le cas. Nous n'avons pas un architecte mais trois personnes physiques. " Sur l'outillage, les développeurs sont restés avec les outils fournis par Microsoft à savoir principalement Visual Studio, même si quelques outils open source sont utilisés. *"On ne tient pas à avoir de l'administration, de modifications à réaliser sur les outils. On doit se concentrer sur le développement. Microsoft a apporté une sorte de standard même si on doit attendre les patchs, les correctifs.* ". .Net n'est pas venu d'un coup d'un seul. XiTi a suivi de près l'évolution de la plate-forme. " Nous avions testé la version 1 du framework. Il faut avouer que cela ne nous avait pas laissé une bonne impression. Aujourd'hui, c'est tout le contraire. On se demande comment on a pu s'en passer ! On commence même à regarder .Net 3.0. De plus, avec ASP, nous avions l'impression d'être tout seul, sans soutien. Avec .Net et ASP.net, c'est documenté, la technologie répond à quasiment toutes nos demandes, nos attentes (sur le développement web). C'est très positif ! Le temps de formation est à peu près le même qu'avant ". " *Le site bouge tous les mois. Il est très vivant. On arrive désormais à un bon niveau d'évolution même si cela a demandé du temps. Toute la partie gestion de version a été un peu problématique à mettre en place, pareillement pour la gestion des erreurs. On a intégré des composants Ajax. Ils posent quelques soucis car .Net 2.0 ne les supporte pas. On a testé le projet Atlas, pour le moment plus pour se faire plaisir que pour une vraie mise en œuvre. Nous préférons attendre avant de l'utiliser.*" conclut Sébastien Carriot.

■ **François Tonic**

Le test unitaire automatisé : *une question de stratégie*

Le test unitaire automatisé (en utilisant un framework de type XUnit le plus souvent) est un sujet à la mode. Tester automatiquement est de plus en plus souvent envisagé sur les projets informatique. Mais ceux qui s'essaient à l'exercice découvrent rapidement que les questions sous-jacentes à sa mise en œuvre abondent : qu'est-ce qui mérite d'être testé ? Qu'est-ce qui est testable ? Quand est-ce que l'on a suffisamment testé ? Qui va écrire les tests ? Qu'est-ce qu'un bon test ?

Trop d'équipes se lancent dans l'aventure sans une vision précise sur le rôle du test unitaire automatisé. Elles en espèrent, de façon assez abstraite souvent, un gain en qualité et en robustesse. Elles espèrent un déploiement de l'application sans drame et une maintenance facilitée. Ces équipes constatent, après quelques semaines ou mois d'efforts, que ces espérances restent aléatoires. Arrive alors la tentation : abandonner le test unitaire automatisé au nom du réalisme et des contraintes diverses du projet (moyens, délais).



Pourquoi tester automatiquement ?

Réponse : on teste automatiquement pour garder la maîtrise d'un développement. Maîtriser un développement, c'est maîtriser son coût, c'est maîtriser son délai. C'est maîtriser le bon fonctionnement de l'ensemble. C'est maîtriser la connaissance technique et fonctionnelle qui a été matérialisée sous forme de lignes de code. C'est entrer dans le cercle vertueux de la confiance : la confiance des utilisateurs qui voient arriver, version après version, une application stable et performante. La confiance du management qui constate que les chiffres et plannings annoncés sont globalement respectés. La confiance des développeurs, enfin, qui cessent de craindre l'écroulement général chaque fois qu'une fonctionnalité est remise en cause ou qu'un collègue s'éloigne de l'équipe.

La confiance des développeurs est le fait psychologique qui va permettre à la qualité d'un développement de rester stable à un degré élevé. Cette qualité va survivre aux multiples évolutions apportées au logiciel parce que le développeur ne va pas hésiter à revoir - en profondeur s'il le faut - ce qui existe déjà, pour l'adapter. Cela s'appelle faire du " refactoring ". Il préférera faire du refactoring plutôt que de contourner l'existant (la tristement célèbre "verrue" qui vient s'accumuler sur un " corps " logiciel de moins en moins sain) car il se sent protégé. Le test automatisé est le bouclier qui l'affranchit de la régression. Il sait que l'impact de ses interventions sera immédiatement rapporté par les tests automatisés. Il peut ainsi contrôler cet impact. Un développeur même brave – mais non téméraire – ne tentera pas de " refactoring " autre que superficiel, s'il ne dispose pas de ce bouclier.

De cette réponse à la question essentielle de l'intérêt du test, on peut déduire une réponse à la plupart des questions données en introduction. Qu'est-ce qui doit être maîtrisé dans un développement ? Réponse : tout. Donc, tout doit être testé. Si tout doit être testé, il faut que tout soit testable. Il faut donc faire le nécessaire pour que tout soit testable. Quand est-ce que l'on a suffisamment testé ? Quand nous avons testé chaque ligne (ou presque) de notre code. Qui peut savoir comment tester chaque ligne de code ? Réponse : le développeur qui les a écrites.

Il faut tester de façon automatique

Le test d'une application est une activité aussi vieille que l'informatique. Ce test est fait, encore aujourd'hui, dans la majorité des cas, par des opérateurs humains : les développeurs parfois, une équipe dédiée quelquefois, les utilisateurs finaux trop souvent.

L'avantage du test manuel est qu'il est facile à mettre en œuvre. Il ne réclame ni outil, ni véritable savoir-faire. Avoir le sens de l'organisation et du management suffit. L'inconvénient est que passer le test est cher. Or la valeur d'un test n'existe vraiment que tant qu'aucune évolution n'a été apportée au logiciel. De ce fait, le résultat d'une campagne de test perd rapidement toute signification. L'effort consenti est perdu. Il faut recommencer la campagne après quelques jours de développement. Même si ce développement ne concerne que la correction d'anomalies ... détectées pendant la campagne de tests précédente ! Répété très souvent, le test manuel finit par étouffer le développement par l'ampleur grandissante de l'effort qu'il réclame. Conséquence : le test manuel est de plus en plus partiel au fur et à mesure que l'application grossit. Il est aussi de plus en plus espacé dans le temps. La maîtrise du logiciel est graduellement perdue, et avec elle la confiance et la qualité.

Le test automatisé consiste donc à capitaliser sur l'effort de test, en acceptant d'augmenter le coût de l'activité qui n'est à réaliser qu'une fois – la définition du test – pour pouvoir réduire à presque rien l'activité qui doit être répétée

très souvent : l'exécution du test. L'effort est donc mis là ou l'investissement ne sera pas perdu.
Même si le coût d'écriture du test est plus élevé dans le cas du test automatisé, l'activité de test est infiniment moins chère dans le cas du test automatisé, à qualité égale. Du moins si l'objectif est de fournir une application au niveau de stabilité et de performance attendu par tout utilisateur.

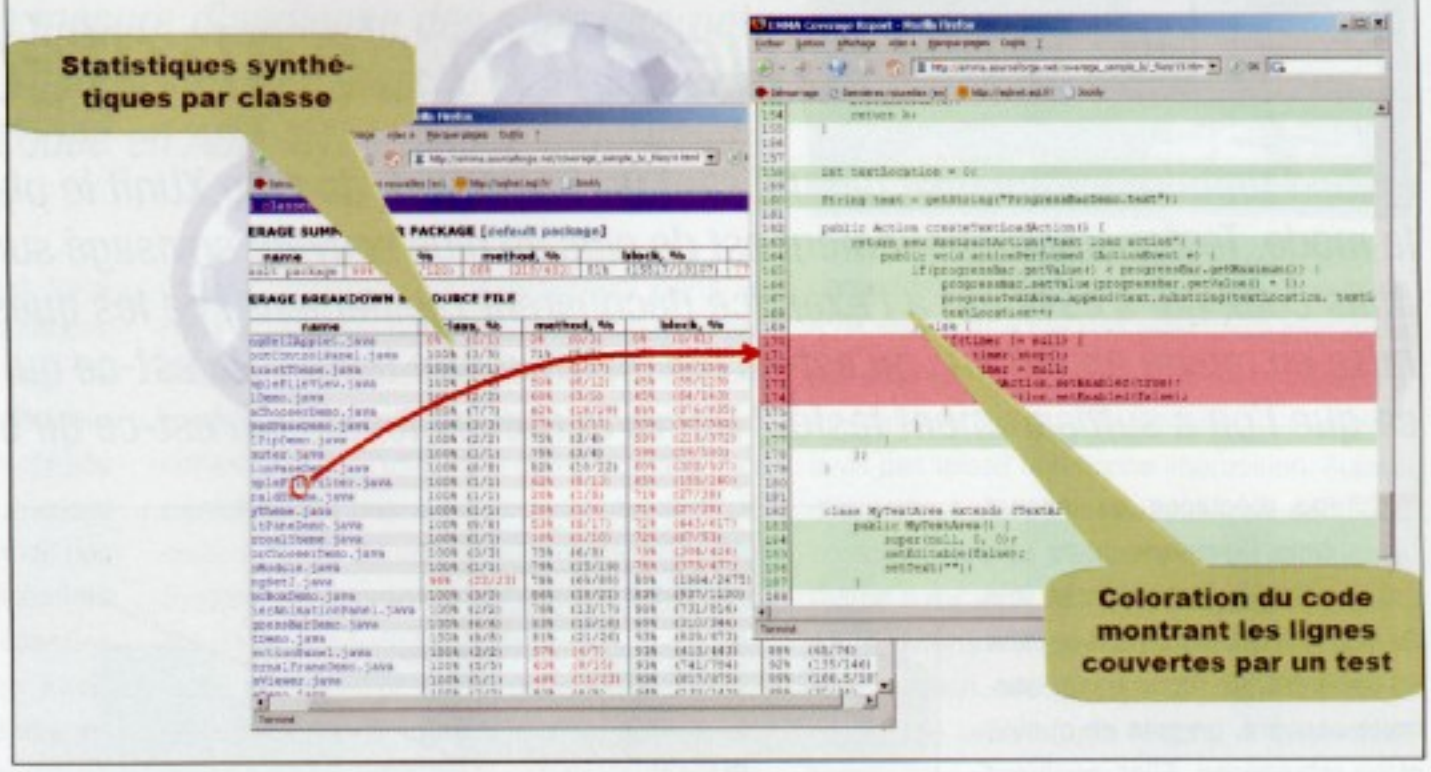
Tout doit être testé

On entend trop souvent des gens, pourtant chevronnés en développement, expliquer qu'il faut confiner le test aux couches basses ou l'appliquer uniquement sur la couche métier, recueil de l'intelligence de l'application. L'argument avancé est double :

- le test automatisé de certaines fonctionnalités (l'IHM notamment) est difficile,
- le test automatisé étant coûteux, il faut le réserver aux modules les plus précieux.

La faille de cet argumentaire est que la solution proposée échoue à assurer l'objectif majeur du test automatisé : garder la maîtrise du développement. La finalité du test automatisé étant perdue, l'équipe n'admet plus l'effort – même limité – demandé par l'activité de test automatisé. Considérons l'exemple récurrent de l'IHM. L'effort consenti sur la partie IHM d'une application représente de la moitié aux trois quarts de l'effort global. Ne pas tester unitairement cette partie revient à perdre le contrôle de la majeure partie de l'application. De plus, ne pas protéger les développements d'IHM sape la confiance, y compris sur la partie purement métier : comment s'assurer qu'une évolution de cette partie métier – même bien maîtrisée par du test – n'aura pas de conséquences imprévues sur l'IHM, puisqu'il n'existe pas de moyens de le vérifier ?
Il reste que tester une IHM a la réputation d'être difficile et fastidieux. En fait, tout dépend de la manière dont on s'y prend. C'est une question de technologie, d'outillage et de savoir-faire.
Sur le plan technique, il faut s'assurer que l'IHM peut être testée aisément. Il faut donc choisir une technologie qui permette le test automatisé. C'est le cas aujourd'hui, dans le monde Java, de Swing, de SWT ou de HTML/Javascript. C'est le cas des WebForms et des WinForms dans le monde .NET. Pour toutes ces technologies, il existe des extensions à JUnit, Nunit ou des API intégrées à VisualStudio qui remplissent cette fonction.
Sur le plan de la méthode, il faut préférer l'implémentation de miniscénarii au test unitaire exhaustif des écrans qui composent l'IHM. L'intérêt d'implémenter des miniscénarii est que le test est plus pertinent et moins fragile. Il est plus pertinent car il simule une manipulation réelle. Pour le développeur, le test devient alors intéressant. Il est aussi moins fragile car il ne s'occupe pas des éléments de l'écran qui ne le concernent pas. On peut donc espérer faire évoluer ces écrans sans pour autant invalider tous les tests qui les touchent.

EMMA : un outil OpenSource de calcul de la couverture de code



Tout doit être testable

Pour qu'une pièce de code puisse être testée, il faut qu'elle soit testable. Or la testabilité n'est pas une qualité si simple à obtenir. Elle n'a, en tout cas, rien d'automatique. Un amas de classes toutes dépendantes les unes des autres, sera impossible à tester. Des pages HTML mal formées, que seul IE est capable de comprendre, ne pourront pas être vérifiées avec des modules comme HttpUnit ou HtmlUnit. C'est la raison pour laquelle de nombreux projets qui veulent intégrer le test automatisé après coup, sont désemparés. Un composant logiciel qui n'a pas été écrit avec le souci du test, ne pourra plus être testé unitairement, à moins de procéder à une sérieuse refonte.
La testabilité doit être un élément essentiel dans le choix d'une technologie ou d'un outil, au même titre que la richesse fonctionnelle ou les performances. Ecartez systématiquement les composants externes que vous ne savez pas intégrer dans vos tests.
Rendre testable une pièce de code peut amener à introduire en son sein, des éléments qui sont surtout utiles aux tests, comme l'utilisation systématique de l'injection de dépendance – qui permet d'isoler facilement une pièce de code pour la tester – ou l'ajout d'identifiants dans les balises HTML – ce qui permet au programme de test de les retrouver facilement.

Mettre en oeuvre une stratégie de test

Un projet qui décide d'utiliser le test unitaire automatisé doit définir une stratégie lisible aux résultats évaluables :

- Pendant la phase de développement, on cherchera à obtenir la couverture totale de test. Et seulement elle. Cela consiste à s'assurer que chaque ligne de code est activée par au moins un test. On peut exclure de cette règle quelques traitements d'erreurs système difficiles à simuler. On n'ira pas plus loin : dès que cette couverture totale est atteinte, on arrête l'effort de test.
- Ensuite, quand une anomalie est découverte, elle doit être systématiquement reproduite par un test avant d'être corrigée. Ainsi, au fur et à mesure que des bogues sont remontés et résolus par l'équipe de développement, le jeu initial de tests va s'enrichir, jusqu'à devenir suffisant pour garantir la fiabilité de l'application.

Une manière efficace d'obtenir la couverture totale est de pratiquer le " Test Driven Development " (TDD). Le TDD consiste à mixer écriture de code et écriture du test, en faisant toujours immédiatement précéder l'écriture d'un bout de code par l'écriture du test qui le vérifie. Il ne s'agit pas d'écrire des cas de tests complets, puis d'écrire les classes qui les implémentent. Le TDD ne réclame pas d'anticipation. On procède par micro itérations. On commence par écrire une méthode de test et le code juste nécessaire pour que l'ensemble compile. On vérifie ensuite que cela ne s'exécute pas correctement. C'est la phase dite "rouge" car l'exécuteur XUnit utilisé affiche une barre rouge. Elle permet de vérifier que le test est pertinent, en s'assurant qu'il doit activer un code nouveau. Dans la phase suivante, on implémente le code métier nécessaire – juste nécessaire – pour que le test réussisse. C'est la phase "verte". Une phase de refactoring s'ensuit, protégée par tous les tests existants, le petit dernier y compris. Cette phase de refactoring vise à rétablir la qualité du logiciel. Et on recommence.

Pour éviter le problème de la page blanche, récurrent lorsque les équipes de développement n'ont pas d'expérience significative en matière de test automatisé, on fournira des exemples. Une façon efficace de procéder est que l'architecte logiciel livre une mini-application (en général une fraction minime de l'application à développer) contenant un exemple de codage des différents types de composants identifiés par l'architecture, avec, pour chacun d'eux, les cas de tests afférents. L'exemple reste la meilleure manière de transférer du savoir-faire.

L'écriture des tests doit être outillée. On utilisera pour cela un framework XUnit bien intégré dans un IDE. On ajoutera un outil de calcul de la couverture de code, lui aussi intégré dans l'IDE. Ce dispositif sur les postes de développement sera complété par l'utilisation d'un outil d'intégration continue. Cet outil rejoue régulièrement – au moins une fois par jour – tous les tests livrés et calcule la complétude de code. Le résultat de ces tests quotidiens doit être publié, sur un site Intranet dédié au projet.

Conclusion

Réussir à construire un jeu de tests unitaires automatisés pour un prix acceptable est avant tout une question de stratégie et de volonté. Il faut certes connaître l'outillage adapté, mais là n'est pas l'essentiel. Fournir uniquement des frameworks de tests et éventuellement une formation ne permet pas, généralement, d'obtenir le résultat escompté. Il faut avant tout expliquer pourquoi on doit tester automatiquement et comment on doit s'y prendre. Cette communication doit être pragmatique, concrète. N'invoquez pas le plan qualité ou le respect d'une démarche ISO ou CMMI. Montrez plutôt où sera le gain pratique dont le développeur va bénéficier. Le test automatisé est une des missions essentielles de l'architecte logiciel. Il doit apporter un support de proximité afin d'éviter le découragement. Et rester ferme devant les manifestations de scepticisme. Sachez qu'une expérience réussie suffit : une équipe qui a vécu un projet couvert par des tests automatisés sort convaincue. Ses membres seront ensuite vos meilleurs évangélisateurs. Ils seront les vecteurs de diffusion de ce nouveau savoir-faire.

Henri Darmet
Directeur Technique Objet Direct / Homsys Group

A propos de Homsys Group :
Créé en 1991, Homsys Group est spécialisé autour de la Business Intelligence (Homsys) et les technologies Objet et Internet (Objet Direct). Homsys Group est implanté à Paris, Marseille, Lyon, Toulouse, Bordeaux, Grenoble et Rennes.

www.homsysgroup.com
www.objetdirect.com

L'INFORMATION PERMANENTE
Programmez!
LE MAGAZINE DU DÉVELOPPEMENT
✓ Abonnez-vous
au format NUMERIQUE (PDF)
Tarif unique pour
LE MONDE ENTIER
35 €
(abonnement exclusivement en ligne www.programmez.com)

✓ 3 NUMEROS GRATUITS
Faites des économies, abonnez-vous !
Exemple : en France, vous gagnez 20 euros, soit plus de 3 numéros !
✓ ETUDIANTS
Tarif spécial 39€
(France métropolitaine exclusivement)
COUPON D'ABONNEMENT page 45 de ce numéro ou www.programmez.com

TechDays 2007 :
le 1er événement informatique en France

Microsoft avait vu grand pour les 3 jours de conférences sur les technologies maison. Les chiffres parlent d'eux-mêmes : plus de 13 000 inscrits, 200 sessions, 19 thèmes, 300 machines PC, 103 labs, plusieurs tonnes de clémentines ! .Net 3.0 et Office 2007 ont tenu la vedette, avec bien entendu Windows Vista. Une occasion aussi de voir pour la première fois Visual Studio Orcas, le futur Visual Studio et les nouveaux C# 3 et VB 9…

La sécurité aura été le thème le plus abordé durant ces trois jours avec pas moins de 46 sessions ! Puis Office 2007 et, petite surprise, le secteur public, se partagent la seconde marche avec 33 sessions. Les développeurs auront pu apprécier toute la diversité des thèmes de développement : Orcas, Linq, toutes les technologies .Net 3 et en particulier Windows Presentation Foundation, Ajax, et la gamme Expression. Pour Microsoft, les enjeux autour de .Net 3, de Vista et d'Office 2007 sont vitaux. Car il faut que les développeurs s'approprient ces technologies et logiciels le plus rapidement possible. Après un certain retard à l'allumage pour sensibiliser les codeurs, les TechDays ont pu démontrer concrètement toute la puissance de ce nouvel ensemble. Des sessions comme PHP sur Windows ou PowerShell rencontrèrent un vif succès. La taille réduite des salles ne permettait pas de faire rentrer tout le monde ! C'est un des principaux reproches que l'on puisse faire à l'événement. Parmi les thèmes manquants ou peu abordés : la virtualisation côté développeur et les tests.

Une véritable exposition

Contrairement aux DevDays du passé, les TechDays rassemblent une véritable exposition informatique avec pas moins de 74 stands avec tous les principaux acteurs du matériel, du logiciel, des services. Surtout, Microsoft France a beaucoup misé sur les communautés. Elles permettent de faire de l'évangélisation, de former, de sensibiliser aux technologies Microsoft. Nix était présent avec un stand Codes-Sources souvent plein ! Le stand de Wygwam était lui aussi bien rempli (Redo en a profité pour mettre en ligne de nombreux podcast vidéo).

Parmi les nombreux outils montrés, nous avons retenu CodeFluent de Soft Fluent. Il s'agit d'un environnement de développement entièrement piloté par modèles. On part d'un méta modèle écrit en XML. À partir de ce modèle, l'outil génère le composant (composants métiers sous forme d'assemblies). Le modèle peut aller très loin dans la description. L'éditeur estime que l'on peut générer environ 80 % de l'application. Surtout, il ne faut pas modifier le code généré car sinon, on casse l'intégrité de l'ensemble. L'outil n'est pas bidirectionnel. Par contre, l'outil à la génération sépare la partie personnalisable de la partie à ne pas modifier. Grâce à ce modèle XML, il est possible d'être indépendant de la plate-forme. Si aujourd'hui, CodeFluent fonctionne sous Windows / .Net, l'éditeur envisage de générer pour Oracle ou Java. Cependant, actuellement, le méta modèle doit être écrit à la main. L'éditeur travaille à la création d'un design autonome.

Micro Framework .Net : le premier .net bootable !

Nous vous en parlions il y a déjà quelques mois de cela. Aux TechDays, nous avons eu droit à une présentation complète du Micro Framework .Net, le nouveau framework .Net. Comme son nom l'indique, il se destine avant tout aux terminaux / appareils utilisant très peu d'énergie, comme une montre, ou ayant besoin d'un tout petit "système" comme une cafetière. Il est issu du projet Spot initié par Microsoft Research. Il n'a pas besoin de MMU et demande moins de 500 Ko de mémoire ! On code en C# et un template projet est déjà disponible (avec son émulateur) pour Visual Studio. On travaille dans un environnement entièrement managé, sans l'instruction unsafe, et sans temps réel. Surtout, pour la première fois, on dispose d'un framework capable de démarrer par lui-même, il n'a pas besoin de système d'exploitation. Il possède un bootstrap ! Loin d'être innocente, cette fonction pourrait bien être une évolution majeure des frameworks .Net. Car finalement, pourquoi ne pas se passer de Windows et de tout système ? Certes, la petitesse du Micro Framework et sa portabilité relativement aisée permettent cela, mais on peut aussi penser que l'on aura dans le futur un vrai framework .Net bootable ! Micro Framework sert aussi dans les modules sideshow des portables estampillés Windows Vista. On notera au passage que pour l'interface des applications, on dispose d'une librairieWPF allégée. Dans les mois à venir, on devrait voir apparaître de nouvelles librairies de .Net 3.

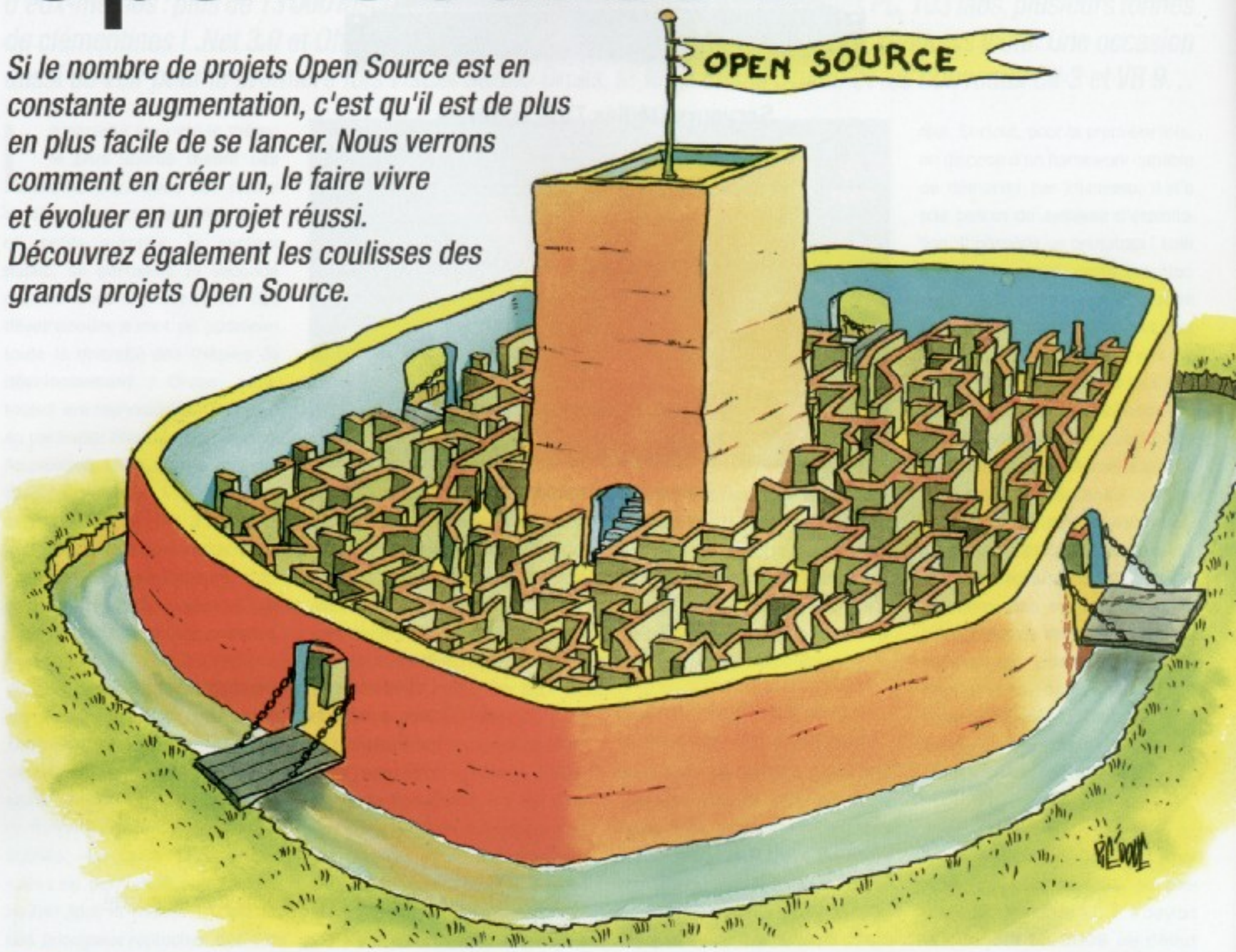
Vers Windows CE 7.0

À peine Windows CE 6.0 disponible, dans les couloirs, on évoquait déjà Windows CE 7. Il devrait voir le jour fin 2008 ou début 2009. Il n'y aura plus de versions intermédiaires (le passage de CE 4.x à CE 5.0 a laissé de mauvais souvenirs). Il y aura uniquement des " features packs " afin de rajouter de nouvelles fonctions. Parmi les pistes possibles de CE 7, l'apparition de .Net 3.x et des évolutions de langage de Visual Studio Orcas. La prochaine conférence sur la mobilité Microsoft se déroulera à Hanovre sans doute en juin prochain.

François Tonic

Comment réussir son projet Open Source ?

Si le nombre de projets Open Source est en constante augmentation, c'est qu'il est de plus en plus facile de se lancer. Nous verrons comment en créer un, le faire vivre et évoluer en un projet réussi. Découvrez également les coulisses des grands projets Open Source.



Cela peut paraître couler de source, mais il n'y a pas de projet de développement, Open Source ou pas, sans idée derrière. Que l'on travaille pour monter en compétence, pour s'amuser ou parce qu'on est payé pour le faire, un projet sans un but clair et précis est souvent voué à l'échec. Il faut aussi réfléchir à ce qui a été fait auparavant. Il est souvent plus facile pour les développeurs de commencer à zéro, et de construire ce qu'on avait imaginé.

Or, on se retrouve parfois avec plusieurs projets Open Source disponibles sur Internet, qui ont les mêmes buts, et qui ne divergent que sur certains points. Il est intéressant à la création de se poser la question, est ce que cela a déjà été fait ? Si oui, n'est-il pas plus intéressant de rejoindre un projet existant plutôt que de repartir de rien ? C'est parfois plus intéressant de contribuer plutôt que de tout reprendre. Le projet a alors beaucoup plus de chance de résister face au temps et à la chute de motivation qui sont les raisons de la disparition de la plupart des "idées".

Les outils

Un autre facteur de la multiplication des projets Open Source est qu'il est maintenant relativement aisé de créer une infrastructure pour gérer son projet. Des sites comme SourceForge [1], Google Code Hosting [2] ou Tigris [3] peuvent en quelques minutes créer tout un environnement de travail distribué. On utilisera particulièrement les mailing-lists pour communiquer ou les forums. Un Wiki en guise de site web pour faciliter l'édition et les modifications. Un "bug-tracker" pour gérer les bugs, les tâches, et l'avancée du projet. Un SCM (Source Control Management) comme Subversion pour gérer les sources. Tout cela devient accessible en quelques minutes seulement. SourceForge à l'avantage d'être là depuis long-

temps, et de fournir une grande gamme de services. Le prix à payer est parfois une certaine lenteur dans les services. Google Code Hosting à lui l'avantage de la jeunesse : peu de projets, une intégration avec les services Google, et la simplicité qui caractérise leurs produits. Pour les programmeurs du monde Microsoft, ce dernier a sorti il y a peu une plate-forme d'hébergements nommée CodePlex [3], qui non seulement héberge leurs efforts Open Source, comme IronPython, mais accueille le développement de projets tiers.
Parmi les outils utilisés par les programmeurs, on note aussi les blogs comme moyen de publication. Mais nous reviendrons plus tard sur la partie communication.

Le développement

Un projet Open Source, c'est avant tout une histoire de logiciel, de code. Quel que soit le langage utilisé, quel que soit le but du logiciel, si on choisit de le partager, c'est qu'on pense, ou qu'on espère, qu'il sera utile à quelqu'un d'autre.
Il convient donc de soigner son code. De le rendre lisible, pour que d'autres personnes puissent le relire, voire l'améliorer. Dès lors que plusieurs personnes travaillent sur le même code, il est important de se tenir à des conventions de code. Des règles d'indentation, de formatage. Quelle que soit la convention, l'important, c'est de s'y tenir, et que tout le monde s'y tienne. Cela favorise la relecture du code, et donc, particulièrement sa maintenance.
Un des formidables avantages de l'Open Source, est que pour la grande majorité des projets, les développeurs sont libres de faire ce qu'ils veulent. Cela peut, bien sûr, tourner en désavantage, mais du point de vue technique, c'est l'occasion de tester des technologies ou des concepts qui ne sont pas toujours implantés en entreprise. Cela va du plus simple, comme la gestion des sources centralisée, aux techniques les plus évoluées, comme la programmation orientée aspect, en passant par les tests unitaires, l'intégration continue, la génération avancée de code et ainsi de suite.
Au final, si l'on s'est fait plaisir, on s'est aussi formé sur des technologies ou des concepts qu'on n'aurait pas forcément eu la possibilité d'expérimenter dans une entreprise classique. C'est un des principaux avantages de l'Open Source. Internet raccourcit les distances, il est de plus en plus facile de travailler avec des experts dans leur domaine sur des projets qui n'auraient pas vu le jour sans cette ouverture sur le monde.
On retrouve dans la plupart des grands projets l'influence d'une personne sur une bonne partie du projet. La plupart du temps, c'est l'auteur original du projet. On le retrouve chez le plus connu des projets Open Source, Linux, avec Linus Thorvald. Chez Python, avec Guido Van Rossum, Larry Wall chez Perl. Même si ils ne touchent plus à l'intégralité du projet, ils sont suffisamment influents pour diriger le projet d'un point de vue technique, contrôler la direction dans laquelle il va, et les prochaines évolutions.
Mais si produire du code est essentiel, il faut aussi le tester, avant qu'il soit utilisé. C'est là un autre avantage direct de l'Open Source. Un adage dit "release early, release often". Littéralement, publier tôt, publier souvent. De cette manière, les personnes peuvent utiliser, tester le logiciel pour vous. Parce que vos tests ne couvrent pas forcément l'ensemble des possibilités de votre programme. Parce que d'autres personnes pensent autrement, elles vont pouvoir vous faire part des problèmes et des points à améliorer le plus tôt possible, avant que corriger le problème devienne un problème en soi. Un projet Open Source réussi, c'est un projet qui bénéficie d'une large communauté qui teste, conseille, et utilise un logiciel.

Le choix de la licence

Dans le monde du logiciel libre, et de l'Open Source, les licences sont importantes. En effet, si le logiciel ainsi que son code est mis à disposition sur Internet, les licences sont le moyen d'exprimer sous quels termes ils sont utilisables. Il existe un vaste choix de licences, et faire son choix n'est pas toujours aisé. N'étant pas un expert en licence, je ne m'attarderai pas sur le sujet. Faisons juste la distinction entre deux types de licences, celles qui forcent les personnes utilisant votre code à publier les modifications, comme la fameuse GPL, et celles qui au contraire, ne l'obligent pas, comme la BSD. Vous pouvez consulter le site de l'OSI [4] (Open Source Initiative) pour avoir accès aux licences les plus connues.

Faire vivre un projet

Faire vivre un projet passe avant tout par sa communauté. Une communauté active est signe de la bonne santé d'un projet. Il faut donc prendre le temps de communiquer, de partager. C'est une tendance que les personnes qui gravitent autour des projets Open Source ont. Une fois intégrés dans la communauté, ils vont la faire grandir inconsciemment, en répondant aux questions, en faisant des tests, en participant. C'est là encore une force majeure de ce mouvement, c'est l'esprit de partage qui règne en général.
Aussi, il est essentiel de bien communiquer rapidement. Communiquer en anglais est une bonne solution pour toucher le plus de monde possible. Utiliser un blog, les RSS, les mailing lists, les forums. Tout est bon pour communiquer. Mais il faut aussi contrôler. L'Open Source et le Logiciel Libre, c'est souvent une histoire de passion. De gens passionnés et convaincus. Mettez deux personnes passionnées avec des opinions différentes dans une même pièce, et vous courrez le risque d'un affrontement. Internet n'échappe pas à cette tendance, il faut donc vérifier et contrôler que les échanges dans la communauté restent courtois. Un nouveau venu recevant une réponse agressive suite à une question n'aura pas envie d'intégrer une communauté. Il est important de modérer les échanges, et de se concentrer sur les buts à atteindre, plutôt que sur les personnes.
La création d'un projet Open Source, pour qu'il soit réussi est donc un gros investissement (en temps surtout), pour définir tous les éléments vus précédemment, rédiger la documentation, etc. parce qu'un projet réussi c'est un projet qui, par son accessibilité a su attirer les contributeurs et les utilisateurs et il deviendra alors nécessaire d'avoir des objectifs clairs auquel tout le monde pourra se référer pour éviter les discordes.
J'aimerai ajouter qu'il faut un minimum d'ambition et de confiance en soi : les petites sources font les grandes rivières ; n'abandonnez pas votre idée en pensant que votre code n'est pas assez bon, publiez tôt, publiez souvent et la communauté vous aidera.

[1] http ://www.sourceforge.net/
[2] http://code.google.com/hosting/
[3] http://www.codeplex.org/
[4] http://www.opensource.org/

J.B. Evain
J.B. Evain est consultant indépendant, et Senior Engineer chez db4objects, Inc., où il participe au développement de la base de donnée Open-Source db4o.

Blog: http://evain.net/blog/

Faut-il choisir l'Open Source ?

Les avantages du modèle

De plus en plus de personnes participent à des projets Open Source. Ceci est dû à ce que les logiciels Open Source accueillent de nombreux avantages.

- **Qualité :** tout d'abord, il s'agit de la qualité en termes de stabilité et de performance. Ceci s'explique par la "loi de Linus" Torvalds (Créateur de Linux) : "Etant donné le nombre d'observateurs, tous les bogues sautent aux yeux".
- **Economie :** une autre des grandes préoccupations des entreprises et des particuliers, c'est le coût d'achat d'un programme. En effet, une distribution GNU/Linux coûte moins cher que son équivalent propriétaire. Il existe d'ailleurs de nombreuses distributions gratuites telles que Ubuntu, Debian, Fedora, OpenSuse, etc.
- **Réactivité :** les développeurs des communautés sont très réactifs. Relativisons cependant : la réactivité de Microsoft est à peu près équivalente, et d'autre part des mises à jour trop fréquentes peuvent augmenter les coûts d'une entreprise pour la maintenance de son parc informatique.
- **Pérennité :** au bout d'un certain temps, les éditeurs ont tendance à arrêter la maintenance d'un logiciel (ex : MS-DOS). C'est pourquoi, des équivalents libres existent. Ainsi, les logiciels libres participent à la pérennité des logiciels.
- **Liberté :** un autre gros avantage est la liberté de l'utilisateur. C'est-à-dire que ce dernier n'est pas tributaire des choix stratégiques de l'éditeur et chaque utilisateur peut adapter le programme à ses besoins.

Les inconvénients du modèle

- **Hétérogénéité :** la liberté que procurent les logiciels libres engendre malheureusement un grand nombre de projets différents et finalement assez proches fonctionnellement, il s'agit là d'une grande dispersion des ressources. D'ailleurs, suite à un désaccord, un projet peut 'forker', c'est à dire se diviser avec l'apparition d'un nouveau projet identique mais avec des choix stratégiques différents pour le futur.
- **Manque de finition :** même si c'est de moins en moins vrai, les applications ont tendance à manquer de finition (au niveau de l'interface graphique par exemple).
- **Manque de support :** il peut manquer pour certaines applications le support.

■ **Frédéric Moreau**

Le baromètre des tendances des logiciels libres 2006

	Linux	PHP	Java	MySQL	PostgreSQL	OpenOffice.org	The Gimp
Progression sur un an (2005 à 2006)	+30%	+20%	+5%	+50%	stable	+30%	Stable
Niveau de maturité des entreprises	4/5	4/5	4/5	3/5	2/5	2/5	1/5
Formation avancée	oui	oui	oui	oui	oui	oui	non

Source : Observatoire du logiciel libre/ Baromètre des tendances 2006 (édition de janvier 2007). Anaska/ Ib-Groupe Cegos

PostgreSQL attire un public plus restreint et spécialisé. Le nombre de personnes formées n'a pas beaucoup progressé et ib – groupe Cegos l'a retiré de son catalogue. Les entreprises qui utilisent PostgreSQL sont souvent de grandes structures qui recherchent surtout des formations très pointues comme celles sur PostGIS et MapServer proposées par Anaska. Pour ce qui est de GIMP/Scribus, Anaska n'a assuré que 4 formations en 2006 sur cet équivalent à Photoshop/PaintShop Pro.

Certaines entreprises se forment pour évaluer le potentiel de l'outil comme alternative. Anaska souligne qu' *" on peut cependant tabler sur une progression de l'utilisation de ces logiciels car si 2006 n'a pas été l'année de la concrétisation professionnelle, du côté universitaire et scolaire GIMP et Scribus sont enseignés. Les prochaines générations d'ingénieurs et d'informaticiens auront donc ces outils dans leurs bagages. "*

■ **Cyril Pierre-de-Geyer**

Décoder les différents types de licences

L'Open Source regroupe un ensemble de licences qui respecte certains principes fondamentaux :

- la libre redistribution
- le code source disponible
- la possibilité de créer des travaux dérivés

Parmi les nombreuses licences Open Source existantes, voici les principales, avec leurs caractéristiques respectives.

GPL (General Public Licence)

C'est certainement la plus connue, car elle met en place la structure juridique des logiciels libres du projet GNU. L'objectif de cette licence est de pouvoir garantir les quatre libertés fondamentales suivantes :

- la liberté d'exécuter le logiciel, pour n'importe quel usage ;
- la liberté de modifier le programme, ce qui passe par l'accès aux codes sources ;
- la liberté de redistribuer le programme ;
- la liberté d'améliorer le programme et de rendre publiques les modifications afin que l'ensemble de la communauté en bénéficie.

Cette licence impose dont la redistribution complète du programme dès la présence d'un fichier sous GPL. Elle possède donc un mode de propagation viral, ce qui n'est pas très adapté pour les éditeurs de logiciels.

Exemples : Linux, MySQL, GNOME, KDE, Ubuntu, Debian, PostFix.

BSD (Berkeley Software Distribution)

Cette licence, beaucoup plus permissive, permet de réutiliser tout ou partie du programme sans restriction, ce qui fait que MacOSX utilise des composants de la fameuse distribution BSD.

MPL (Mozilla Public Licence)

A mi-chemin entre la licence GPL et BSD, cette dernière permet d'inclure des fichiers sous d'autres licences (licences propriétaires par exemple) au sein du programme.

Exemples : Firefox, ThunderBird. ■ **F.M.**

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

Quel modèle économique choisir ?

Clairement, la seule vente d'un logiciel sous GPL est risquée car un concurrent direct n'aurait plus qu'à récupérer le code et se l'approprier. Comment les éditeurs s'adaptent et comment les utilisateurs réagissent ?
Les éditeurs disposent de plusieurs modèles économiques :

• **Vente de services :** le produit est là pour faire la renommée de l'entreprise. La rémunération est faite par le biais de services dédiés que propose cette dernière. Ainsi, la société est basée sur la barrière commerciale élevée. Exemple : JBoss est un serveur J2EE sous GPL, la société JBoss propose des services J2EE.

• **Intégration de composants :** l'entreprise, pour se démarquer peut également proposer de l'intégration de composants. Exemple : Distribution Linux commerciale comme RedHat, Suse.

• **Double licence :** une autre stratégie de produit peut être de mettre en place le programme sous deux licences, une libre et l'autre propriétaire. Ainsi la rémunération se fait sur la vente de licences propriétaires. Pour bien comprendre, prenons l'exemple de la licence de la bibliothèque QT, commercialisée par TrollTech. Un éditeur qui crée une application basée sur la bibliothèque QT, en licence GPL, doit normalement laisser un accès aux sources de par le caractère viral de la licence GPL. Pour que cette même entreprise puisse créer une application commerciale, l'entreprise doit alors se procurer QT sous licence commerciale.

• **Le principe de dynamitage** est une autre approche qui consiste à ouvrir le code à des fins de diffusion. Exemple : Netscape pour contrer Internet Explorer.
Il s'agit là d'ouvrir des composants non stratégiques, c'est-à-dire, pas générateurs de revenus. Exemple : Pilotes d'imprimantes.

• **Libérer le logiciel**, mais vendre du contenu. Exemple : un site d'information tel que AOL qui ouvre le code de son client AIM.

• **Produits dérivés :** de nombreuses entreprises utilisent le marché de l'Open Source pour vendre les produit dérivés. Exemple : l'Editeur O'Reilly vend des livres sur des applications Open Source.

• **Limite dans le temps :** pour terminer, citons les licences chrono-dégradables, c'est-à-dire certains logiciels sont sous licences fermées durant un certain temps, puis à une date limite, on ouvre le code. Exemple : Aladdin Enterprises rend le programme Gostscript GPL à peu près un an après sa sortie en licence propriétaire.

Cependant, un logiciel Open Source n'est pas forcément libre, tout dépend de la licence. Ainsi, il existe des licences Open Source restreignant l'utilisation des sources d'un logiciel. Exemple : Microsoft Shared Source, cette licence autorise la consultation de code pour permettre la collaboration avec certains partenaires.
Il existe ainsi tout un tas de licences plus au moins adaptées aux éditeurs, mais qui ont le mérite d'être une alternative à des produits 100% propriétaires.

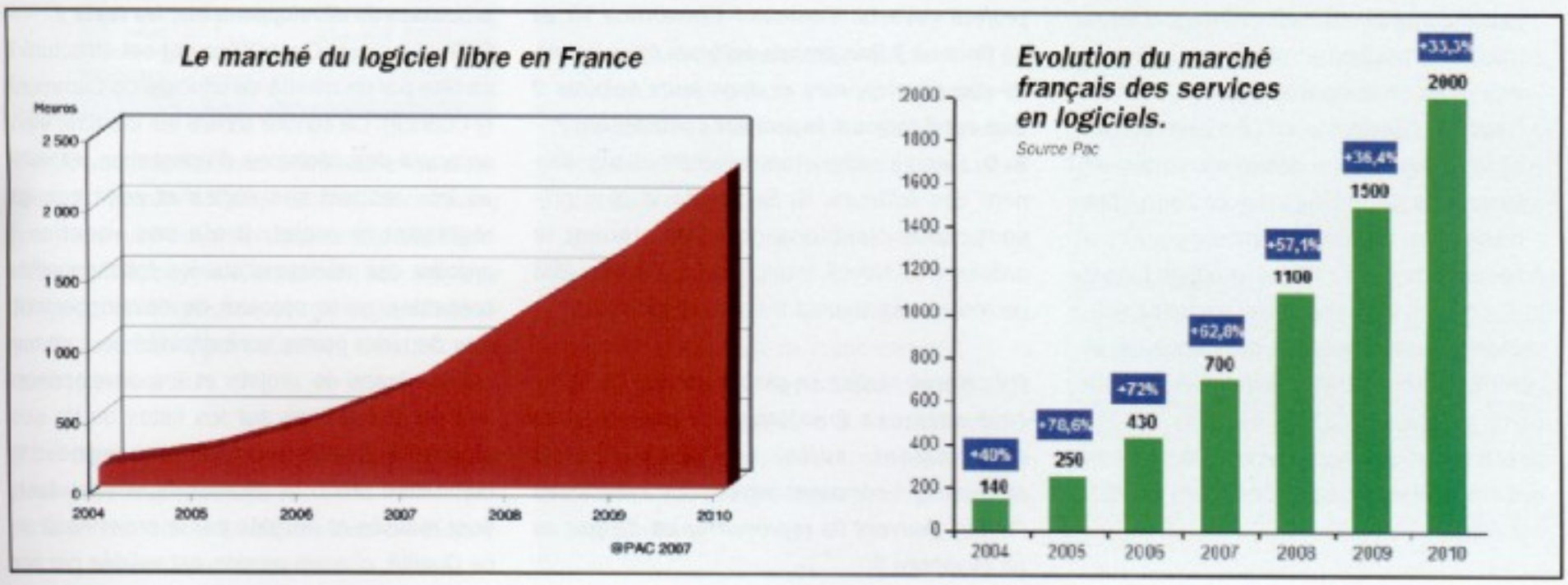
Quel scénario pour l'entreprise cliente ?

Une étude de Jean-Paul Smets-Solanes, Docteur en sciences informatiques a mis en évidence trois scénarios possible dans le cadre du déploiement d'un logiciel :

- Achat d'un logiciel commercial et adaptation du logiciel par des ingénieurs.
- Acquisition gratuite d'un logiciel libre et adaptation de ce dernier par des ingénieurs.
- Développement du logiciel par l'entreprise.

Ces trois scénarios nous conduisent à penser que développer peut devenir rentable à partir du moment où l'éditeur met du temps à corriger les bogues et/ou le coût des licences est trop élevé. Ainsi, quand l'offre de logiciel existant est trop pauvre ou trop chère, le développement devient rentable.
Par ailleurs, La dépendance vis-à-vis d'un éditeur peut être un frein si ce dernier tarde à corriger les bogues, refuse de divulguer certaines informations de l'application. Ainsi, l'utilisateur se voit contraint d'acheter les sources, ce qui peut être extrêmement coûteux.
Par ailleurs, les membres d'une communauté autour d'un logiciel libre ont tendance à corriger chacun, les bogues qui les gênent le plus. Ainsi, le délai de réaction sur chaque bogue est réduit si chacun se répartit le travail.
L'absence de coût pour la licence permet à un utilisateur d'investir plus d'argent dans la maintenance, l'adaptation et l'évolution de l'application.
Cependant, la migration d'un système peut générer un coût très important. Le coût d'achat des licences peut diminuer suivant le nombre acheté. Pour terminer, il reste tout de même aujourd'hui des applications propriétaires plus performantes que leur équivalent libre.

Frédéric MOREAU
Formateur Solaris & Linux
Elève ingénieur SUPINFO



LES COULISSES DES PROJETS
1RE PARTIE

Comment se structure un projet ?

LE RÔLE DES DÉVELOPPEURS

Il y a tout d'abord l'idée de l'application. À partir de là, le ou les développeurs ouvrent un site sur une forge et mettent en place les Wikis, la documentation, voire une première version très expérimentale.
L'enjeu est double : il faut développer l'application et se constituer une communauté qui pourra participer au codage. Sans communauté active et nombreuse, le projet aura bien peu de chance de vivre et surtout d'évoluer.
S'il n'existe pas de gouvernance standard, voici une organisation basique d'un projet de bonne taille :

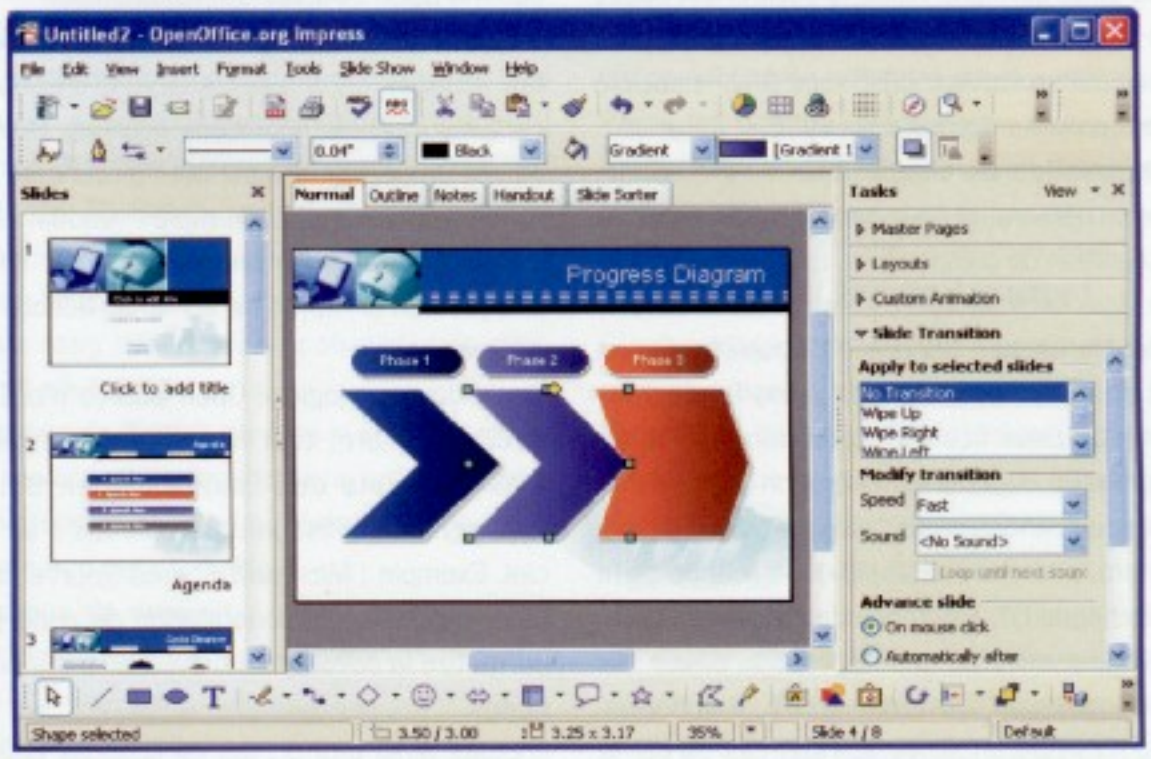
- à la base : la communauté des utilisateurs et des développeurs
- les contributeurs communautaires : dans la communauté, des développeurs peuvent participer aux développements du code, à la correction de bugs.
- les " commiters " : ce sont souvent des développeurs issus de la communauté et promus dans la structure du projet. Cette promotion se réalise par la méritocratie et/ou par élection.
- le " Lead " / " Project lead " : dans les projets importants ou des fondations, chaque module ou sous-projet est mis sous la houlette d'un responsable que l'on nomme "lead" ou " project lead ". Il a la responsabilité de son module, de son sous-projet en terme de planning, de gestion des ressources, du développement.

Au-dessus du Lead / Project lead, on trouve la direction à laquelle le lead participe activement. C'est là, avec les développeurs, la communauté que se décident les évolutions, le planning. Dans certains cas, la direction est un board, un comité rassemblant des développeurs, des leads ou des sociétés.

OpenOffice, l'autre grand de la bureautique

OpenOffice.org 2

OpenOffice.org est le projet phare du monde libre. Nous avons posé quelques questions à Sophie Gautier responsable francophone du projet OpenOffice.org – LINAGORA et à Sébastien Bahloul, directeur technique, pôle OSSA – LINAGORA.



Programmez ! : Que représente aujourd'hui OpenOffice en terme de projets, de personnes, de communautés (de développeurs et d'utilisateurs) ?
Sophie Gautier : On peut estimer à environ 150 à 200 les développeurs qui travaillent à temps complet sur OpenOffice.org.

P ! : on parle beaucoup de financement des projets ouverts. Comment OpenOffice vit et se finance ? Des grands éditeurs fournissent-ils des développeurs et donc leurs salaires ? Sun est-il toujours le premier contributeur ?
S. G. : les plus importantes contributions viennent des éditeurs. Si Sun reste le plus présent, nous comptons donc maintenant la présence de Novell, Intel, RedHat, Google. IBM commence également à nous rejoindre.

P ! : Novell réalise sa propre version en rajoutant quelques fonctions supplémentaires (notamment suite à l'accord avec Microsoft) : comment voyez-vous cela ? Les "forks" peuvent ils représenter un danger ou un avantage ?
S. G. : Novell est un contributeur important du projet OOo, qu'il alimente sa propre version ne représente pas un danger. Plus la suite est répandue et plus cela justifie sa qualité et sa fiabilité, le " proof of concept " ;-)

P ! : comment se structure un projet OpenOffice type ? Comment est-il dirigé de la tête à la base ? Qui prend les décisions sur les fonctions, les processus de développements, les tests ?
S. G. : Le projet OpenOffice.org est structuré à sa tête par un comité de pilotage (le Community Council). Ce comité arbitre les conflits, vient en appui des décisions d'orientation, et veille au bon respect des règles et principes qui régissent le projet. Il n'a pas vocation à prendre des décisions sur les fonctionnalités présentes ou le process de développement. Ces derniers points sont décidés en commun par les leads de projets et les développeurs lors de discussions sur les listes ou de sessions IRC dédiées, un release engineering committee prend la décision finale. Les tests sont réalisés et décidés par le projet Assurance Qualité, chaque version est validée par son

responsable de projet en langue native, pour le français, je suis responsable de la version.

P ! : Comment est-il possible de contribuer ?

S. G. : N'importe qui peut contribuer, qu'il soit codeur ou non. La première chose est de se présenter et d'informer la communauté de ce sur quoi l'on souhaite contribuer, afin que les membres présents puissent guider ses premiers pas. Il faut ensuite accepter la licence sous laquelle la contribution va être placée : LGPL pour le code, PDL pour la documentation. Enfin, un *join copyright assignment* doit être signé avec Sun (partage du copyright comme on en trouve dans divers autres grands projets, ndlr).

Sébastien Bahloul : La majorité des communautés dispose maintenant d'un guide du développeur qui permet d'identifier simplement le processus technique et juridique permettant de contribuer.

P ! : Quels outils de développements mettez-vous en oeuvre pour vos projets ? Qui les administre ?

Sébastien Bahloul : Un certain nombre d'outils sont utilisés. Certains sont génériques : la gestion du code via CVS, le suivi des demandes (issues tracking) via l'outil correspondant, les listes de diffusion, un Wiki et un guide permettant de débuter des travaux de développements.

D'autres sont spécifiques à la plate-forme cible, comme Microsoft Visual Studio, nécessaire pour construire la version exécutable sur l'environnement éponyme. Concernant les services en ligne, l'administration est partagée : l'hébergement est assuré par CollabNet, les outils sont administrés par Sun.

P ! : OpenOffice rémunère-t-il des développeurs, des responsables ?

S. G. : Non. Nous finançons des développements, pas des développeurs.

Il y a trop de bureaucratie !

Le coup de gueule du responsable Open Office, chez Novell - Portrait Michaël Meeks

Michaël Meeks est, chez Novell, à la tête de l'équipe OpenOffice.org. Il contribua à Gnome et fut un des premiers développeurs de Ximian (éditeur notamment de Mono), repris par Novell. Il dirige 17 développeurs, ce qui en fait la première équipe de développeurs, après Sun, éditeur de Star Office.

Michaël travaille depuis 5 ans sur open Office : *" Quand je suis arrivé, le design était laid. On a travaillé sur les widgets, pour que ça ressemble à un OS ".*

Aujourd'hui, Michaël est un homme en colère. Il se plaint de rencontrer des difficultés énormes à coopérer avec les autres structures qui contribuent au projet. Et en particulier avec la SSII mandatée par Sun, Collabnet*, qu'il met en cause. Son "coup de gueule" illustre-t-il la difficulté de faire travailler des développeurs indépendants, distants, sur la base du volontariat, dans le cadre procédural d'un suivi de projet ?

Programmez ! : Combien y a-t-il de développeurs qui travaillent sur Open Office ?

Michaël Meeks : il y en a 17 chez Novell, 80 chez Sun (Star Office), 1 chez RedHat, et pour Collabnet, 1 responsable et 15 développeurs extérieurs
(Ndlr : les informations de Sophie Gautier, dans l'article ci-contre ne sont pas identiques).

P ! : comment marche le projet ?

M. M. : C'est terrible ! Il y a beaucoup de dysfonctionnement dans la relation avec la société en charge du recrutement des développeurs extérieurs !
Sun a mandaté une société extérieure, et ça ne marche pas !
Ma compréhension de leur communauté, c'est qu'il n'y a pas de communauté !
On ne dit plus au développeur si son code est bon. On a accumulé des barrières pour implémenter le code. Le message que le développeur interprète est " on ne vous fait pas confiance ".

P ! : Comment se traduisent les dysfonctionnements ?

M. M. : Les symptômes d'échec d'une équipe open source sont les suivants :
- les gens ne vous répondent pas
- la bureaucratie remplace les relations personnelles
- traiter le besoin de transparence par une mailing list

Pour gérer une telle équipe, la communication est primordiale ?

M. M. : Vous n'avez pas besoin de beaucoup de gens, mais ils doivent être motivés, afin de créer. Miguel (Miguel de Icaza, créateur du projet Mono, Ndlr) a un talent exceptionnel pour créer une communauté.

P ! : Votre équipe, chez Novell, est internationale ?

M. M. : Les développeurs sont de l'Utah, de Boston, de république Tchèque, de Chine. Il y a même un Français...

P ! : Comment communiquez vous ensemble ?

M. M. : Tout à fait normalement : par chat et mail, téléphone... Une fois par semaine, à 14H, seul horaire compatible avec tous les fuseaux horaires, nous faisons une réunion générale, au téléphone. Cela dure une heure, en moyenne, et ça se termine par une liste d'actions.

Jean Kaminsky

** CollabNet se définit sur son site comme le seul fournisseur d'un environnement de développement software basé sur le webweb, conçu spécifiquement pour les équipes de développement dispersées. Collabnet intègre des applications de Software Configuration Management, des outils de gestion du cycle de vie logiciel dans un environnement à la demande. http://www.collab.net/products/index.html*

Mozilla : 240 versions à maintenir...

Même s'il est difficile d'évaluer l'importance de Mozilla aujourd'hui, quelques chiffres permettent de se rendre compte de son ampleur : environ 400 comptes CVS, 700 contributeurs au code, une centaine d'employés à Mozilla, une quarantaine de localisation (sur une centaine de projets).

Ainsi, pour une version de Firefox, il faut mettre à jour et maintenir 120 versions (40 localisations * 3 systèmes). Tout en sachant que Firefox possède deux branches : la version 2 et la version 1.5 soit 240 versions. De quoi donner le tournis ! Sans oublier que la version 3 est en cours de développement.

Mozilla c'est aussi : des dizaines de milliers de bugzilla, plus de 100 000 inscrits sur Spreadfirefox.com, des dizaines de communautés dans le monde et des dizaines de millions d'utilisateurs de Firefox (le principal projet de la fondation Mozilla).

Un soutien des éditeurs

" Plusieurs grandes entreprises fournissent du temps d'ingénieur, comme IBM, Sun, Red Hat, Novell... Mais cela est difficile à mesurer, car nous ne connaissons pas l'origine des contributions : un contributeur peut utiliser son adresse personnelle pour soumettre des patchs tout en étant en mission pour son employeur. De même, nous ne savons pas si un contributeur est à temps complet ou partiel. Par ailleurs, il y a des contrats avec plusieurs grands acteurs qui sont présents dans la boîte de recherche de Firefox. Je pense en particulier à Google, Yahoo et Amazon. Ces liens vers ces sites ont été mis au départ dans Firefox parce que nous avons estimé qu'ils pourraient se révéler pratiques pour les utilisateurs : quand nous avons intégré une boîte de recherche dans Firefox, nous voulions que les résultats soient les plus pertinents possibles, et donc nous avons choisi ces acteurs. Ce n'est qu'ensuite qu'ils nous ont contactés pour s'assurer que nous n'allions pas les retirer de cette boîte de recherche, qui leur ramène beaucoup d'audience. D'autres moteurs, moins performants, nous ont proposé de grosses sommes d'argent pour figurer en page d'accueil et dans la boîte de recherche. Dans la mesure où c'était moins bon pour l'utilisateur, nous avons refusé." commente Tristan Nitot (Fondation Mozilla).

À cette ressource, il faut rajouter les recettes générées par le Mozilla Store (produits dérivés) et les dons. Un projet aussi volumineux que Firefox nécessite une solide organisation mais aussi de l'argent. Il faut bien payer la centaine de personnes de Mozilla...

La méritocratie avant tout.

Sur le côté développement, le développeur peut prendre du galon grâce à la méritocratie, un principe que Mozilla tient à conserver. La relecture du code permet d'évaluer les compétences des développeurs. Et parfois, il peut alors monter dans la hiérarchie Mozilla.

"A la base, c'est le développeur. Ensuite, c'est le "module owner", qui est en charge d'un module. Ensuite, c'est le "super-reviewer", un développeur de très haut niveau chargé de faire la 2e relecture (super-review) du code avant qu'il ne soit inséré dans le produit" explique Tristan Nitot. Mais Mozilla est aussi une véritable entreprise. Le marketing s'organise classiquement même si la communauté a son mot à dire.

Les bras ouverts

Pour contribuer à Mozilla, il faut mettre les doigts dans le code ! En théorie cela est facile : trouver un problème que l'on souhaite résoudre, ouvrir un compte bugzilla (et signaler que l'on travaille au problème), rejoindre le serveur IRC et discuter du problème (cette partie n'est pas obligatoire) avec les développeurs. Après le développement, on soumet le patch dans bugzilla et on demande alors sa relecture.

Ensuite, le développeur doit suivre les demandes du ou des relecteurs. Le patch ne sera pas intégré tant qu'il n'est pas accepté par la phase de relecture.

Bref, avant de vouloir s'attaquer au cœur d'un projet Mozilla, on s'occupe de concevoir un patch... " *Concrètement, les contributeurs potentiels commencent souvent par faire de l'assurance qualité, c'est-à-dire télécharger les dernières versions issues des compilations nocturnes (nightly builds), et vérifier qu'il est possible de reproduire les bogues nouvellement signalés dans Bugzilla. Ça leur permet de se familiariser avec le processus, les us et coutumes et nos normes de développement. Ensuite, ils soumettent leurs patches.*" précise Tristan Nitot.

Et les outils

C'est le grand classique : un éditeur de texte, un compilateur C++, un débogueur. Selon la plate-forme, les outils varient. Ainsi sous MacOS X, ce sera le couple Xcode – GCC, sous Linux, GCC – GDB. "Pour ce qui est du travail collaboratif, le code est partagé via CVS en attendant la migration vers un système plus performant et les discussions se font sur IRC. Bien sûr, le suivi des bogues se fait via Bugzilla, un outil maison maintenant utilisé par un grand nombre de projets libres et propriétaires". Conclut M. Nitot.

Sun et l'Open Source

L'éditeur Sun a entamé un virage radical : tous les logiciels seront à terme, ouverts ! Gros contributeur, Sun a décidé d'ouvrir Java via le projet OpenJDK, sans oublier OpenOffice, OpenSolaris et NetBeans.

L'éditeur souhaite construire autour de ses projets une communauté plus large pour renforcer le nombre d'utilisateurs et surtout pour continuer à supporter et à compléter ses solutions. Sun participe à d'autres projets comme Apache ou Mozilla, sans pour autant être à leur l'origine.

L'investissement de Sun dans les projets varie énormément. Cela dépendra si l'éditeur est ou non à son origine. Si tel est le cas, Sun fournira des développeurs à plein temps, payés par l'éditeur, même si Sun ne décide pas seul de l'évolution des projets. Pour les projets initiés par Sun, le développeur " indépendant " peut participer au développement. Ensuite, la méritocratie joue à plein. Il peut espérer devenir commiter. Le commiter joue un rôle central car c'est lui qui a la haute main sur l'évolution du projet, la maintenance et le développement. Et lorsqu'un code est modifié, celui-ci est vérifié

par deux autres personnes afin de le qualifier. C'est un rôle important des " collaborateurs ". " *On relit aussi le code des autres développeurs* " prévient **Alexis Moussine-Pouchkine.** La maîtrise du développement est très importante pour la réussite d'un projet.

Sur OpenOffice, l'importance de Sun et des développeurs non-Sun est très diverse, selon les fonctions, voire les pays. En France, il y a autant de développeurs Sun que de non-Sun. Par contre, au niveau du projet, ce serait plutôt 80-20 pour les développeurs Sun. " *La situation est satisfaisante. Il ne faut pas que le produit perde en qualité. On voudrait qu'IBM soit plus impliqué, il serait le bienvenu.*" poursuit Alexis Moussine-Pouchkine.

Proposer oui, développer aussi

Si un développeur ou un commiter peut proposer une nouvelle fonction, exemple dans Open Office, il ne doit pas arriver les mains vides. Il doit avoir un plan de développement, définir les ressources nécessaires. Bref, à lui de savoir comment mener à bien sa demande ! Et dès le dépôt de la proposition, idéalement, il faut déjà disposer des développeurs nécessaires... Si cela peut sembler " lourd ", cette méthode doit permettre de garantir le projet et d'impliquer le développeur dans sa proposition. Notons que les discussions sont publiées. " *Soit il y a consensus, soit il y a vote.* " précise Alexis Moussine-Pouchkine. Cependant, le fait qu'un grand nombre de développeurs soit payés par Sun peut orienter la décision.

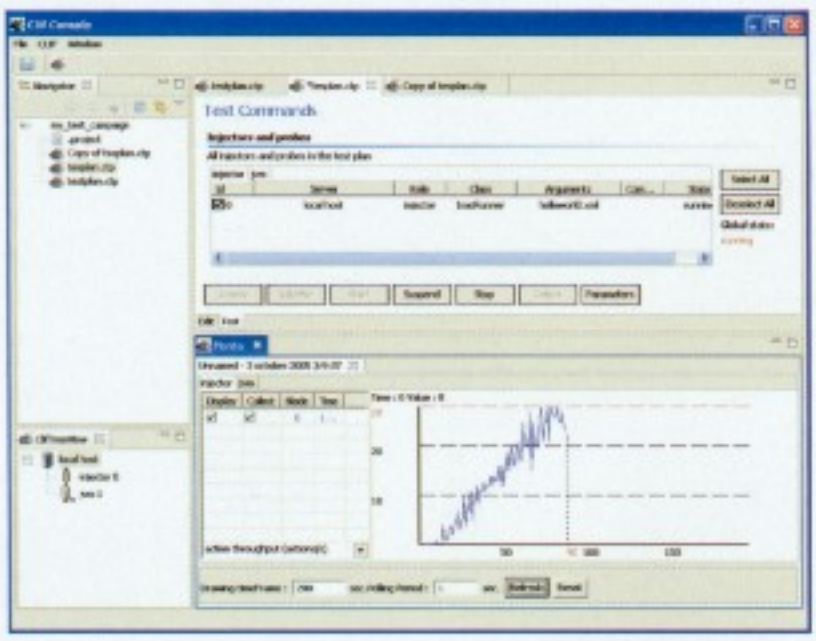
Le " fork ", pas une mauvaise chose...

" *C'est à la communauté de définir les règles de fonctionnement du projet, et notamment de choisir la licence. OpenOffice utilise la LGPL*" commente Alexis Moussine-Pouchkine. Il est donc possible de créer sa propre distribution OpenOffice. " *Le fait de faire des distributions compatibles est une bonne chose.* " analyse Alexis Moussine-Pouchkine. Cela peut renforcer l'audience du projet et donc la communauté.

Objectweb et ses 120 projets

ObjectWeb
Open Source Middleware

À la fin 2006, Objectweb comptait 73 sociétés membres, 2300 individus pré-inscrits dont plusieurs centaines réellement inscrits, 8000 contributeurs, 700 commiters pour environ 120 projets ! Les projets sont à différents stades de maturité.



" *Il n'y a pas de réelle structure. Il y a le chef du projet, les contributeurs. Chaque projet est libre de se structurer. De plus, une société peut être "lead" d'un projet. Nous avons une forge avec des wiki, des sites. Quand un nouveau projet nous est soumis, il est évalué par un conseil technologique qui statue* " précise François Letellier d'Objectweb.

La recherche et développement sont assurés par les membres. " Nous ne faisons pas de développement. " indique François Letellier. De plus, la plupart des développeurs viennent des sociétés membres, même s'il y a tout de même des individuels. " *La société est copyright. La partie juridique est peut être plus claire chez nous. On trouvera plus facilement des services. La licence est un élément stratégique, elle ne doit pas être un frein à l'utilisation. Nous préconisons la licence LGPL et nous l'avons adapté pour notre middleware. Le choix de la licence dépend (aussi) du niveau de maturité du projet. Nous avions un projet en licence BSD, un concurrent l'a repris tel quel !* " analyse François Letellier.

Pour Objectweb, les utilisateurs s'habituent à travailler dans un monde mixte : propriétaire, ouvert. De plus, rien ne sert de tout réinventer à chaque fois. Sur les Fork, Objectweb se veut philosophe, " *c'est la rançon du succès. On évite d'enfermer l'utilisateur. Il faut aussi éviter l'enfermement du développeur. Il faut dire que le fork, pour l'utilisateur, est une plaie* " précise François Letellier.

Retour d'expérience

La collaboration au sein de NetFilter : la couche pare-feu de Linux

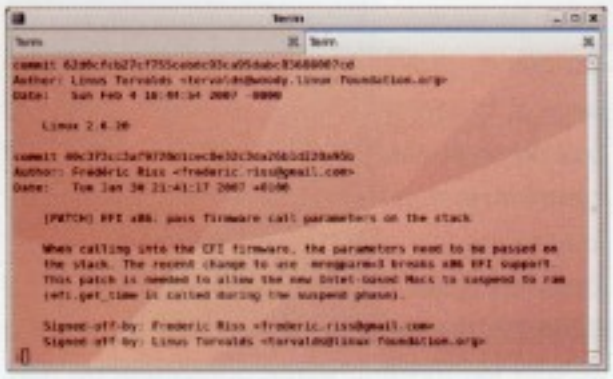
Des milliers de développeurs collaborent pour faire de GNU Linux l'un des meilleurs systèmes d'exploitation. Loin d'être le bazar , la participation au code suit une mécanique très précise. Netfilter, la couche pare-feu du noyau initié par Linus Torvalds ne fait pas exception.

Une architecture pyramidale

Le code de Linux est composé de millions de lignes évoluant rapidement et leur validation ne peut être faite par un seul homme. La tâche a donc été découpée entre plusieurs responsables chargés chacun de maintenir une portion donnée du code. Linus Torvalds reste le responsable des changements stratégiques et de l'inclusion ultime. Chaque mainteneur de branche est libre de reproduire cette architecture pyramidale dans son secteur. C'est le cas du domaine de la couche réseau qui regroupe des entités indépendantes comme ATM ou Netfilter.

Où l'innovation peut venir du bas

Un ajout de fonctionnalités est souvent le fait d'une initiative individuelle. Dans notre exemple, le point de départ a été la découverte que Skype et bon nombre de logiciels de Peer2peer parviennent à établir une connexion directe entre deux machines masquées derrière des routeurs. Une des étapes clés de l'algorithme utilisé est la prédiction d'un numéro de port. En le rendant aléatoire, on empêche l'établissement de ces connexions. J'ai donc décidé de modifier Netfilter dans ce sens. Pour éviter tout conflit avec des développements récents, il faut travailler sur la dernière version de Linux et donc utiliser le système de suivi de version du noyau : git. Cet outil conçu spécialement pour le noyau Linux est pour une grande part dans la réussite du projet et dans son évolution rapide. Il offre les commandes attendues sur tout RCS et des extensions avancées



Les derniers changements de Linux par "git log"

Affichage du bloquage d'un paquet Skype

comme l'envoi par mail de l'ensemble des commits entre deux branches. Ceci permet aux développeurs de se décharger d'une part importante des contraintes d'échanges. Il suffit de rentrer un changelog explicite lors des commits pour générer en une commande les mails diffusant les modifications. Une fois les modifications testées, il restait à soumettre le code à la "Netfilter Core Team", l'équipe de Netfilter dirigée par Harald Welte. La proposition se fait par l'envoi de mails à la liste de diffusion publique netfilter-devel : un mail par patch et, si on a plusieurs patchs, un mail introductif expliquant les motivations du travail effectué. Cette diffusion publique des patchs avant même leur application permet la discussion et les critiques et favorise l'émergence rapide des meilleures solutions. La relecture est tant fonctionnelle que formelle puisqu'elle inclut la vérification du respect du CodingStyle du noyau qui définit de manière très précise le formatage du code. Cette uniformité facilite le maintien du code et la recherche de bogues.

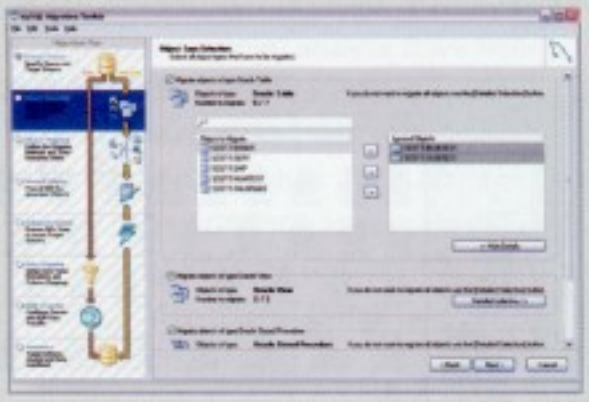
Une remontée rapide

La décision finale est prise par les membres de la Core Team, Patrick Mc Hardy en tête, qui valident ou rejettent la contribution. Ils sont responsables de l'envoi des modifications à David S. Miller, le mainteneur de la couche réseau. git facilite son travail en lui permettant d'appliquer toutes les modifications contenues dans les mails sauvegardés dans une boîte aux lettres. Il ne reste plus alors à Linus Torvalds qu'à fusionner la branche réseau pour voir le code intégré à Linux.

Eric Leblond
Co fondateur d'INL, SS2L et éditeur du pare-feu authentifiant NuFW

MySQL et sa forge

MySQL est l'un des projets les plus connus et populaires. Il représente aujourd'hui une centaine de développeurs internes répartis entre différents pays. " *Nous avons des équipes sur les grandes thématiques, à plein temps* " précise Serge Frezefond de MySQL. Sur les contributions externes, elles se portent beaucoup sur les bugs et les rapports de bugs. L'éditeur dispose de sa propre forge : MySQL Forge. Elle y stocke des projets liés à MySQL réalisés par des contributeurs, du code réutilisable ou encore de la documentation. Il y a plusieurs centaines de contributeurs. Cependant, il n'est pas possible de contribuer directement au cœur de MySQL. "*Nous avons des personnes chargées de la communauté. Il faut naviguer entre les demandes des clients et la communauté* " précise Serge Frezefond. La communauté peut proposer des fonctions, suggérer, commenter la roadmap de MySQL. Il y a des sondages sur éléments à rajouter. " *Une grande partie du développement est interne. Cependant le profiling vient de la communauté. Elle sera peut être intégrée* " commente Serge Frezefond. L'éditeur continue à proposer une version communautaire et entreprise.



JBoss et la communauté

" Aujourd'hui, JBoss compte pas moins de 25 projets, de différentes tailles, le plus gros est le projet du serveur d'application ainsi que Hibernate " précise Sacha Labourey de JBoss. Ces projets sont tous structurés autour de la licence, de la communauté, avec une gouvernance. Parfois JBoss rejoint des communautés déjà constituées et se plie aux conditions existantes.

Le modèle de JBoss est une licence copyleft, avec réciprocité, la LGPL. Cela permet de faire du propriétaire, de vendre son logiciel, mais si le code est modifié, il doit être disponible. *"Avec la licence Apache, il n'y a pas de réciprocité, il n'est pas obligatoire de reverser les modifications"* poursuit Sacha Laborey. Parfois, l'éditeur prend aussi la licence GPL, cette licence a une notion de viralité, qui concerne l'ensemble du code.

" Sur la gouvernance, l'approche est différente. Il s'agit de l'organisation, la vision " idéaliste " est de voter sur tout. Nous avons un modèle ouvert, les votes sont en ligne, mais on ne s'interdit pas de prendre des décisions " poursuit Sacha Labourey. Cette position de l'éditeur a été parfois critiquée. Mais pour pouvoir avancer et fixer des objectifs, il faut savoir trancher et décider. *" Soit on prend la bonne décision, soit on en prend une mauvaise. "* précise Sacha Labourey. Comme le fork est possible, rien n'empêche des développeurs de créer un dérivé... La pression est grande sur les grands projets car il faut concilier le business et la communauté. *" Un projet n'est pas que du code. Nous avons plusieurs outils pour communiquer, décider : les listes de diffusion, les forums ".*

Dans un projet, il faut s'avoir gérer une communauté, la faire vivre, la faire participer, faire évoluer les fonctionnalités, incorporer de nouveaux développeurs, etc. Surtout, comme JBoss ne veut pas cacher ses objectifs à long terme, la roadmap est disponible sur le site de l'éditeur. Ainsi les développeurs souhaitant intégrer la communauté peuvent voir si des fonctions peuvent leur plaire. Au début, cette roadmap n'existait pas.

Qui développe les projets JBoss ? Les développeurs de la communauté avant tout. JBoss embauche régulièrement des développeurs venant des communautés pour renforcer l'équipe interne. Si JBoss souhaite avant tout prendre des développeurs open source / libres, l'éditeur ne ferme pas la porte aux développeurs traditionnels pour réaliser des développements plus spécifiques comme des plug-in, des connecteurs.

Un projet comme JBoss ne s'improvise nullement. *" On formalise (le travail) "* prévient Sacha Labourey. Chaque projet possède un "lead". Au niveau de la société, il y a des " product marketing manager ". Ils gèrent le développement. *" Ils s'assurent que les project leads ont les moyens nécessaires au développement, une roadmap à jour, etc. Le développement est géré au niveau du lead."* conclut Sacha Labourey.

Mais alors comment devient-on un lead jBoss ? Il s'agit avant tout de la méritocratie. Le développeur montre ses capacités, son envie de faire les choses bien, il a aussi la reconnaissance de la communauté.

La distribution Mandriva en chiffres

Mandriva Linux

L'éditeur français possède en interne une équipe de R&D de 35 personnes. Mais la force d'une distribution licence est le nombre de développeurs externes. Mandriva compte un millier d'inscrits sur son site de développement collaboratif Cooker. De l'aveu même de l'éditeur, seul un petit noyau d'une trentaine de contributeurs est particulièrement actif. Pour qu'un projet soit un succès, l'internationalisation constitue un enjeu vital. Pour notre éditeur, il n'y a pas moins de 200 traducteurs pour les différentes langues !

Pour chaque nouvelle version de Mandriva, un cahier des charges est établi suivant les recommandations et les demandes des développeurs internes et des contributeurs. Chaque membre de l'équipe technique de l'éditeur est responsable d'un domaine et gère les relations avec les contributeurs ayant la charge de ce domaine. Si un développeur veut participer au projet Mandriva, c'est possible. La première étape consiste à maîtriser le système et les différentes éditions fonctionnellement mais surtout, techniquement. Le plus souvent, la valeur et les capacités du prétendant se remarquent par le codage de patchs ou de corrections qu'un contributeur déjà actif remarque. Ce contributeur pourra alors proposer d'intégrer le développeur.

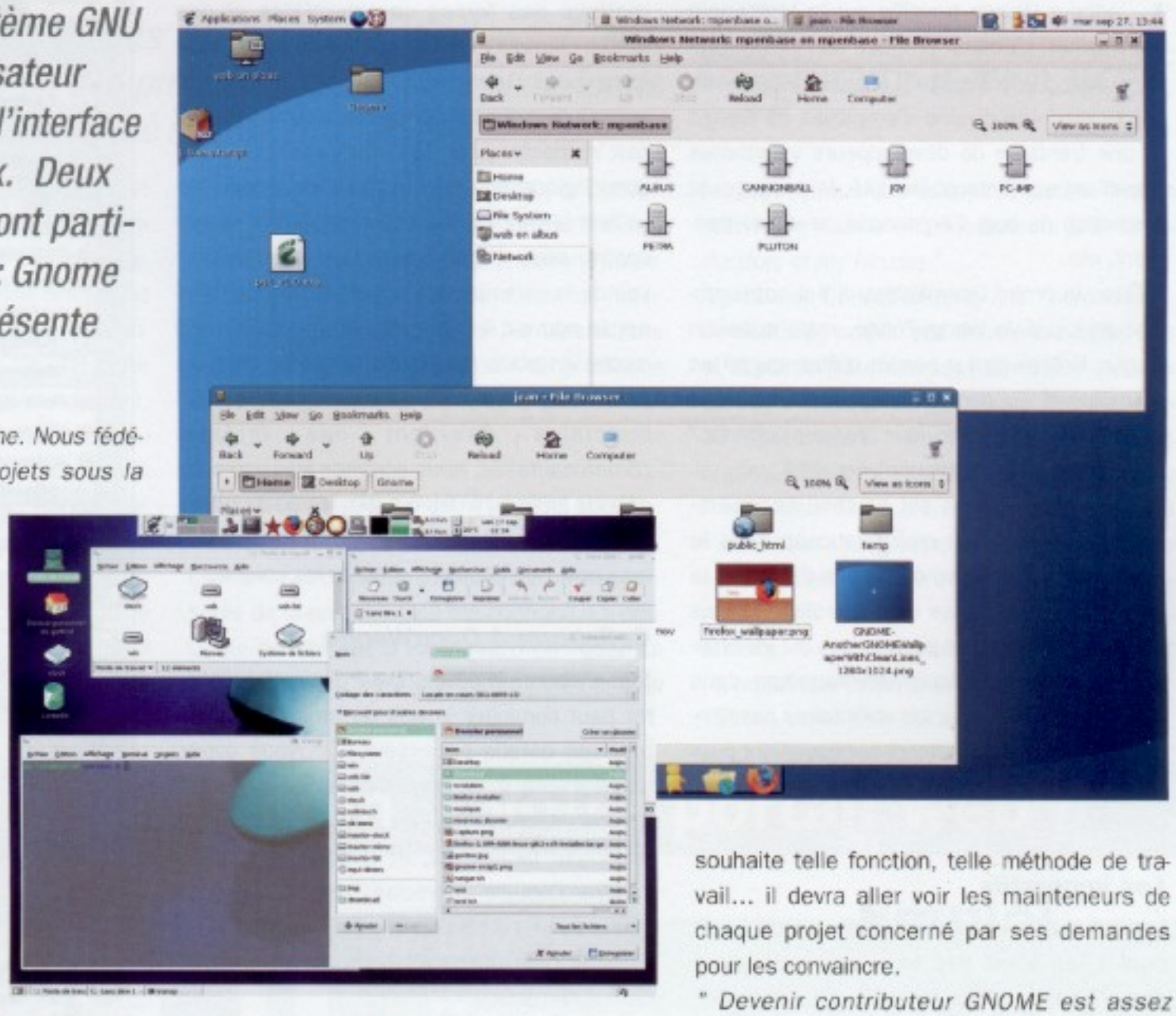
Pour le financement, l'essentiel provient de la vente des boîtes, de la vente en ligne sur le site Mandriva, les accords OEM avec les constructeurs de matériel ou encore les projets grands comptes. L'éditeur rémunère ses développeurs. Certains d'entre eux participent à d'autres projets ouverts comme KDE ou Gnome.

Côté outil, Mandriva utilise Svn pour le système de build. On dispose aussi de Repsys pour accéder rapidement et simplement aux codes. Côté développement pur, ce sont les grands classiques (Gcc, Perl, etc.).

La licence utilisée est la GPL. Ainsi tout outil conçu par Mandriva (et diffusé par l'éditeur) ou par la communauté, doit utiliser cette licence, et doit être reversé à la communauté.

Gnome : un des projets phares de GNU Linux

Dans le système GNU Linux, l'utilisateur peut utiliser l'interface de son choix. Deux interfaces sont particulièrement connues : Gnome et KDE. Mais que représente réellement Gnome ?



"*GNOME est un projet énorme. Nous fédérons des dizaines de projets sous la "grande tente" - les projets qui suivent les conventions de programmation et les cycle de sorties de GNOME, et qui participent pleinement à l'avancement du projet. Pour nos produits (the GNOME Desktop Environment et the GNOME Developer's Platform), nous avons dans chaque cycle de développement des centaines de contributeurs. Le nombre de participants GNOME dépasse largement 1000 en ce moment*" commente Dave Neary de Gnome Fr. Il faut dire que Gnome est le bureau par défaut d'Ubuntu, Red Hat Enterprise, Fedora Core, SuSe Linux, OpenSolaris. Selon des sondages de préférences de son bureau, sur des sites d'utilisateurs GNU Linux, Gnome récolte 35 % contre 55 – 60 % pour KDE, le " grand ennemi ". Soit au moins 5 millions d'utilisateurs. Concevoir et maintenir un bureau et les projets annexes de l'ampleur de Gnome nécessite des moyens financiers. Gnome est financé, essentiellement, par les sociétés – éditeurs utilisant par défaut Gnome (Canonical, Novell, Sun, Red Hat). Ils ont même embauché les développeurs desktop afin de continuer à le faire évoluer. Bien entendu, les développeurs communautaires ne sont pas oubliés et participent activement au projet Gnome. Cela représente plusieurs centaines de volontaires (étudiants, professionnels), dixit Dave Neary. Dans ce cas, il participe souvent selon leur méthode, selon leurs envies.

" *Tous ces intérêts cohabitent facilement grâce à la culture de notre communauté, et à l'alignement des intérêts des acteurs commerciaux avec les intérêts de la communauté. Cela aide également qu'il n'y ait pas une société qui domine, ce qui oblige tout le monde à bien jouer ensemble*" analyse Dave Neary.

Une association pour chapeauter

" *Il y a une association à but non lucratif qui gère le budget GNOME, et qui organise tout les ans des conférences GNOME (GUADEC, ou le Boston Summit), elle fournit un budget à des développeurs GNOME qui voyagent pour promouvoir GNOME dans des conférences, et travaille également avec des sociétés pour fournir un lieu de collaboration stratégique autour de GNOME. Il y a aussi une équipe qui gère les contenus de nos produits – le Release Team - constitué des gens choisis parmi la communauté au mérite* " précise Dave Neary. " *La fondation GNOME a une employée à temps partiel, et cherche à embaucher un chargé d'affaires à temps plein cette année.* " poursuit Dave Neary.

Mais chaque projet se gère à sa façon, il n'y a pas de direction globale. Ainsi si un éditeur souhaite telle fonction, telle méthode de travail... il devra aller voir les mainteneurs de chaque projet concerné par ses demandes pour les convaincre.

" *Devenir contributeur GNOME est assez simple - nous avons une équipe dédiée à l'accueil des nouveaux contributeurs, GNOME Love, une équipe dédiée à aider des femmes à trouver leur place dans le projet (GNOME Women), et une équipe d'études de marché et communication (Marketing Team). Pour les tâches techniques, chaque projet a sa propre hiérarchie, et donc ses propres façons de s'investir. L'équipe GNOME Love est là pour dessiner la carte de la communauté, et aider chacun à trouver la place adapté à ses capacités et ses envies* " commente Dave Neary.

Si un nouveau projet veut faire partie du release set de Gnome, il devra se soumettre à une procédure rigoureuse. " *Il faut proposer au début d'un cycle de développement, ce qui est suivi généralement par un débat très pointu sur l'intérêt du nouveau module, et 6 semaines avant la sortie, le Release Team décide sur les nouveaux modules.* " prévient Dave Neary. Côté outillage, Gnome utilise les classiques des projets ouverts : Subversion, Bugzilla, un Wiki, un réseau IRC, des serveurs (pour les développeurs du projet).

Wengo : de la téléphonie ouverte

L'éditeur Wengo travaille sur la téléphonie internet. L'éditeur propose son propre projet ouvert : OpenWengo. Le projet représente aujourd'hui une dizaine d'employés de Wengo et une trentaine de développeurs volontaires travaillant sur l'interopérabilité, le package, la correction de bug, l'ergonomie, le développement, etc.

" Dans le projet OpenWengo, il y a notre produit principal, le WengoPhone, mais aussi un plug-in FireFox (qui a besoin d'être soigné en ce moment) qui permet aux gens de faire des appels SIP à partir de leur browser préféré. " précise Dave Neary (Wengo).

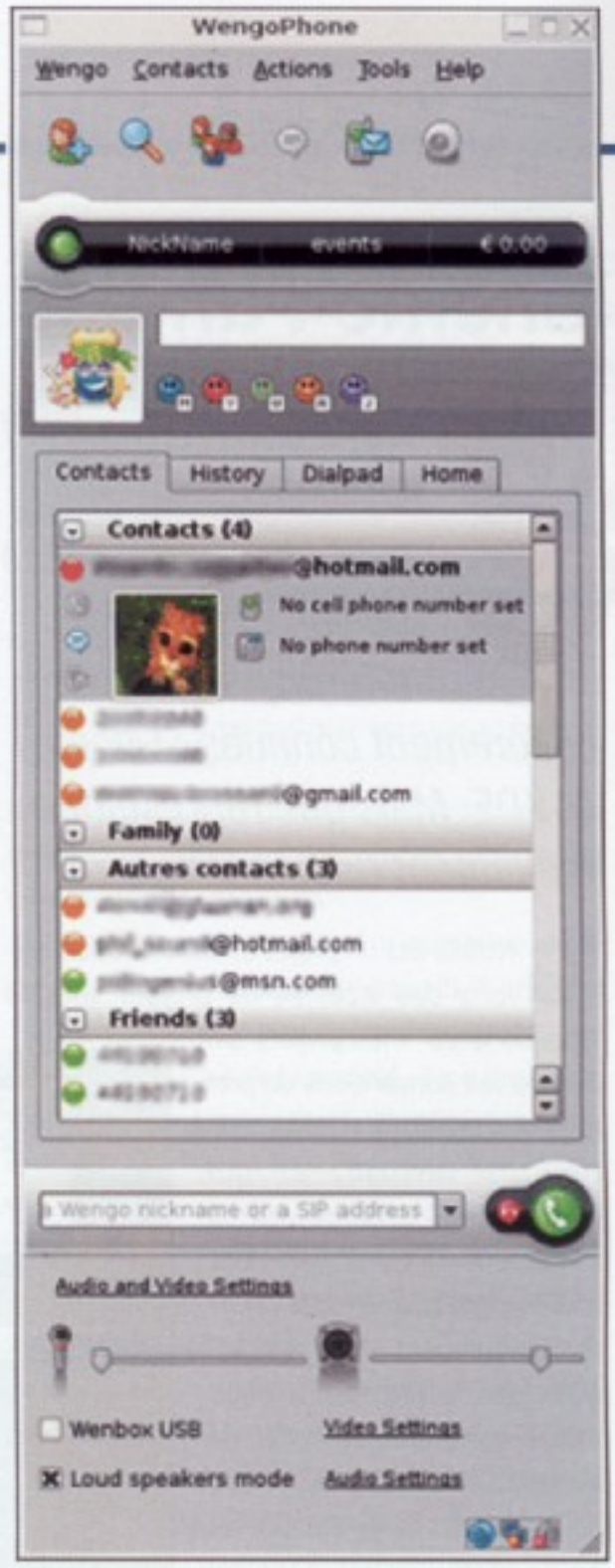
Le projet OpenWengo est soutenu financièrement par Wengo qui croit beaucoup dans le logiciel libre. Et preuve du succès du projet, la communauté ne cesse de s'accroître. *" Nous avons également des partenaires qui investissent du temps de leurs développeurs dans OpenWengo, ainsi que les volontaires passionnées et plusieurs étudiants qui travaillent pour des projets de recherche dessus. "* indique Dave Neary.

Les impératifs

Lorsqu'un éditeur initie ou supporte un projet ouvert, ce n'est pas sans attentes, sans contre partie. *" Pour l'instant, il y a la problématique des forces commerciales et des forces communautaires qui est en train de se jouer pour OpenWengo. Wengo a des exigences de calendrier et de fonctionnalités qu'il faut respecter pour ses investisseurs - il est donc logique que les employés de Wengo travaillent davantage sur ces fonctionnalités. Par contre, nous croyons beaucoup dans le pouvoir de la communauté, et je suis là pour donner le pouvoir à la communauté pour aider à mettre en place les grands lignes de développement du projet. Là où les exigences commerciales divergent des intérêts communautaires, nous sommes transparents, et nous arrivons à des compromis ou à l'entente cordiale "* précise Dave Neary. Bref, il faut trouver le bon équilibre et éviter les conflits.

Collaborer à OpenWengo

Tout d'abord, le développeur désirant collaborer peut consulter le wiki du projet. Il fournit tous les détails nécessaires. *" Nous avons aussi une équipe d'accueil qui essaye d'identifier des tâches que les débutants peuvent accomplir facilement, et d'aider les gens à intégrer l'équipe. "* précise Dave Neary. Côté outils, du grand classique : CVS, des listes de diffusion, une chaîne IRC sans oublier un build-farm automatique.



openwengo

Eclipse, ouvert et fermé

Eclipse est devenu un des projets les plus emblématiques du monde ouvert. Il s'est imposé comme la fondation technique de nombreux outils de développement. IBM fut son créateur. *" Nous participons beaucoup à Eclipse qui représente aujourd'hui 180 projets "* commente Michel Lara d'IBM France. Dans Eclipse rien n'est laissé au hasard. Tout y est formalisé, avec une solide gouvernance.

Sur Eclipse, il existe un " board " dans lequel on vote. *" On est autour de la table. Sur Java, on passe par Sun. Nous ne sommes pas seuls"* commente M. Lara. Eclipse fait partie des plus gros projets, avec un millier de développeurs répartis sur les 180 projets. IBM mobilise plusieurs dizaines de développeurs sur les projets Eclipse dont quelques-uns résident en France. *" Ils y sont à plein temps "* précise M. Lara. Les autres éditeurs membres font la même chose.

Dans Eclipse, il y a un processus industriel pour fixer le calendrier, synchroniser le développement des développeurs, construire le projet, fusionner les sources, etc. Les équipes s'occupant d'Eclipse sont géographiquement dispersées. *" On a besoin d'un formalisme. Eclipse est découpé en projet, sous-projet. Chacun ayant une définition, un planning. Tout est publié ".*

Mais alors, peut-on collaborer librement à Eclipse ? La réponse est non. *" Il faut être élu"* précise M. Lara. Le développeur individuel doit passer par un processus de sélection. Comme pour d'autres projets ouverts, l'argument de la qualité est mis en avant. *" On doit maîtriser la qualité. Il y a une obligation de sélection "* argumente Michel Lara. Il faut dire qu'Eclipse doit livrer des versions les plus propres possibles car elles seront ensuite réutilisées par de nombreux outils tiers, des éditeurs. La plupart des membres de la fondation sont des sociétés.

Il n'existe pas de réel fork d'Eclipse, car cela demanderait un travail considérable. Cependant, il existe des distributions packagées comme MyEclipse mais à chaque évolution d'Eclipse, la distribution doit s'adapter.

DotNetNuke : projet ouvert .net

Il existe de nombreux projets open source Windows / .Net. DotNetNuke constitue un des projets majeurs sur la plate-forme .Net. Sa vie n'a pas toujours été simple…

L'origine du projet vient d'un starter kit .net lancé par Microsoft. Rapidement, un fork avait été lancé sur le projet par des développeurs allemands. L'autre secousse vint quand le créateur de DotNetNuke voulut passer l'environnement dans un modèle payant pour financer le développement. Mais rapidement, la proposition fut annulée. La question du financement est un réel souci. " *Microsoft apporte l'hébergement.* " précise Benoit Sarton de DotNetNuke. En contre partie, DotNetNuke doit incorporer les nouveautés de l'éditeur. Pour la licence, le choix s'est porté sur la très souple BSD. *"Au début, nous n'avions pas de licence"* avoue Benoît Sarton.

Sur les six derniers mois de 2006, le DotNetNuke a représenté plus de 2 000 heures indemnisées (et autant en contributions volontaires), soit environ 100 000 euros. Le financement se fait beaucoup par les bienfaiteurs qui ont donné 64 000 euros. " *La prochaine étape est de générer du revenu par les services* ".

Une dizaine de contributeurs pour la traduction

En France, c'est Benoit Sarton qui est responsable de la communauté francophone. La principale activité est la localisation de l'outil. La " filiale France " est une petite communauté par rapport aux autres communautés. " *Nous avons une dizaine de contributeurs pour la traduction, et les forums* "

À la tête de DotNetNuke, il y a le " core team ", l'exécutif du projet. Les liens entre cette tête et la partie française sont ténus. Ce manque de lien s'explique par l'absence d'un Français au " core team ". DotNetNuke possède une organisation très concentrée. Au-dessous du " core team ", il y a une sorte de conseil. Le " core team " regroupe une vingtaine de personnes. Elles travaillent sur le noyau de l'environnement.

Ensuite, il y a des responsables de modules (le projet étant réparti en modules).

www.programmez.com Votre portail web !

Windows Vista

NOUVEAU

- Toute l'actu **Vista**
- Les articles du magazine en **intégral**
- Les blogs, le forum

Boutique

Achetez les magazines et les articles en PDF

NOUVEAU

Emploi

Un espace emploi dédié

Riche en actualité et en articles dédié à l'emploi, Programmez vous permet enfin de consulter les offres publiées et de postuler directement en ligne

Ecrivez vos Tips

Consultez les Tips des membres et créer les votres. Des livres **Windows Vista** à gagner

Rédigez vos Tips

LES COULISSES DES PROJETS

Témoignage d'un développeur

Retour d'expérience d'un an

J.B. Evain est ingénieur chez db4objects. Il est aussi consultant indépendant. Grâce à db4objects, il participe au monde Open Source. Il nous raconte son expérience

Programmez ! : Tu es impliqué dans plusieurs projets Open Source, quel est ton parcours ?

J.B. Evain : Cela va bientôt faire un an que je travaille pour une société qui sponsorise le développement d'un projet Open Source. Avant d'en arriver là, j'ai comme beaucoup de programmeurs, simplement participé aux développements de logiciels Open Source. Depuis que j'ai commencé à programmer, j'ai été attiré par la philosophie du Logiciel Libre, mais aussi par le mode de fonctionnement des projets Open Source, en particulier des gros projets. J'ai aussi toujours gardé, en dehors de mes études et de mon temps de travail, du temps pour travailler sur des projets personnels qui m'ont permis de monter en compétence sur certaines technologies. Je passais alors beaucoup de temps dans certaines communautés sur Internet qui se regroupent pour partager autour d'une technologie. De par mon parcours professionnel, je me retrouvais la plupart du temps à travailler avec des technologies comme Java ou .Net. C'était l'époque où .net venait de sortir, et où tout restait à faire sur cette plate-forme. On prenait alors la communauté Java et ses idées comme modèle. C'est à ce moment qu'est née l'idée du premier projet Open Source où j'ai réellement participé. Avec un membre d'une de ces communautés, nous avons commencé le développement d'un logiciel implémentant les concepts de la programmation orientée aspect sur la plate-forme .Net.

C'est ce qu'on appelle un tisseur d'aspect ; il a pour nom AspectDNG. Après plusieurs prototypes, nous nous sommes aperçus qu'il nous manquait une librairie pour manipuler les binaires .net. Je me suis donc attelé à cette tâche pendant que mon collègue continuait à travailler sur le tisseur.

Programmez ! : Comment as-tu transformé ce hobby en travail ?

J.B. Evain : Au bout de quelques mois de développement, alors que ma librairie (Cecil) commençait à devenir utilisable, j'ai eu l'idée de la donner à un projet Open Source de plus grande envergure qu'AspectDNG, pour gagner en visibilité, et en utilisateurs. Cela a permis d'en accélérer le développement. J'ai donc contacté les mainteneurs du projet Mono, un projet de grande taille qui vise à développer une implémentation libre du Framework .net. J'ai alors pris pied dans la communauté de Mono dont les développeurs ont commencé à utiliser Cecil. Ce projet a été sponsorisé deux années de suite par Google lors de son Summer of Code, où des étudiants sont payés pour travailler sur des projets Open Source. Aujourd'hui, Cecil est utilisé dans plusieurs sous-projets de Mono, d'autres logiciels Open Source, et même, grâce à sa licence permissive, dans des produits commerciaux fermés.

Non seulement travailler sur ces projets m'a permis de gagner un peu ma vie en tant qu'étudiant, mais elle m'a aussi permis de trouver du travail. J'ai reçu de nombreuses propositions suite à mon travail sur Cecil. J'ai accepté celle de db4objects, Inc ; société qui utilise Cecil dans son produit. Participer à des projets Open Source a été pour moi un formidable outil pour construire ma carrière.

Programmez ! : Comment travaille-t-on dans l'Open Source ?

J.B. Evain : À db4objects, j'ai la chance de travailler pour une société qui développe un logiciel libre, et dont les méthodes sont très différentes des sociétés dans lesquelles j'ai pu évoluer auparavant. Nous sommes tous dispersés dans le monde. Cela nous force à communiquer et à adopter des méthodes "agiles". Nous pratiquons par exemple beaucoup de "Pair-Programming" entre contributeurs. En travaillant à deux sur le même écran distant, cela nous permet de partager la connaissance du code plus facilement, tout en produisant du code de meilleure qualité. Nous nous réunissons tous les six mois pour définir la road-map et les tâches des six prochains mois, en fonction de ce que nos utilisateurs nous soumettent dans notre logiciel de gestion de bugs et de tâches. Ces réunions sont pour nous l'occasion de se retrouver, et de partager entre personnes de cultures souvent très différentes. Cette ouverture est vraiment quelque chose que j'apprécie dans les projets Open Source, et on ne la retrouve pas forcément ailleurs, dans des sociétés où tous les développeurs sont sur le même plateau.

Logiciels libres : des technologies reconnues et des business models qui rapportent

Marginal l'Open Source ? Peu lucratif ? Fi de ces affirmations inexactes ou dépassées. Ce monde est en constant développement avec le soutien des administrations françaises qui ont ouvert la voie au Libre et dans lequel, banques, assurances et industries s'engouffrent. Avant-dernière étape avant de séduire les PME/TPE et les particuliers ?

La France est l'un des pays phares pour le logiciel libre dans le monde. Avec une croissance de 80% en 2006 (+ 79% en 2005) selon l'étude Pac (Pierre Audoin Consultants) Open Source France 2007, ce marché représente 450 millions d'euros en 2006. Pac indique que le marché mondial du logiciel libre est dominé par IBM, Red Hat et Novell mais que " progressivement Oracle et SAP deviennent des acteurs cruciaux sur le marché Linux ". En France, les spécialistes du logiciel libre les plus présents sont Red Hat, Novell, MySQL pour les internationaux et pour les nationaux, IdealX, Linagora, OpenWide, et Mandriva. Pour donner raison à cette étude, une annonce est tombée au cours du salon Solutions Linux 2007 : *"Nous venons de signer un contrat avec PSA Peugeot Citroën pour déployer 20 000 postes de travail Linux et 2500 serveurs Linux sur plusieurs années"*. **Christophe Therrey,** DG de Novell France, est particulièrement satisfait de donner à ses équipes la possibilité *" non pas de pratiquer une migration totale car ce n'est pas encore à l'ordre du jour, mais de permettre une coexistence pacifiée entre Microsoft et Linux. On ne s'affranchit pas complètement de l'aspect propriétaire mais ce qui est sûr, c'est que l'on progresse ! "* Les origines du contrat PSA-Novell remontent à Solutions Linux 2006 et révèlent un modèle économique qui s'ancre. En effet, aux côtés des communautés existent désormais des entreprises bien structurées et reconnues. Elles vendaient traditionnellement leurs services et désormais, pour les éditrices, elles donnent la possibilité du logiciel sous licence. Ce mode de fonctionnement permet de progresser en parts de marché : le modèle de Novell repose en effet sur des solutions Open source et des solutions propriétaires (Netware, Zenworks), et s'il est difficile de séparer les deux dans les comptes, **Christophe Therrey** souligne : *" avec notre solution Suse Linux réellement industrielle, nous prévoyons une forte croissance du CA en France à 2 chiffres pour 2007."*

Deux business models : d'un côté, la coexistence solutions Open Source et licences propriétaires et de l'autre, les services

La cohabitation entre les modèles propriétaires et libres au sein d'une même entreprise est désormais admise. Ainsi, MySQL qui a progressé sur le modèle de la double licence depuis 2000, a également fait le choix des services inédits et déclinés sous toutes leur formes via MySQL Enterprise, lancé en octobre 2006. Cette offre est dédiée aux *" utilisateurs de MySQL souhaitant s'assurer que leurs serveurs déployés sont à jour, sécurisés, et fonctionnent de façon optimale grâce à une suite de services qui aident les DBA et les administrateurs système à être plus productifs. De plus, MySQL Enterprise permet le recours aux experts de MySQL en cas de besoin "*, indique **Bertrand Matthelié,** VP marketing EMEA de MySQL AB. Les entreprises semblent tellement conquises que ce pourrait être ce modèle économique qui l'emporte, en 2007, sur les autres. Mais la plupart des entreprises gardent toujours ce côté "services", à l'instar de Red Hat : *"Red Hat a été un précurseur dans l'adoption d'un modèle économique basé sur l'offre de services autour de solutions Open Source. De nombreux acteurs nous ont emboîté le pas et c'est pour nous la preuve de la réussite du modèle que nous défendons depuis le début. En choisissant les outils Open Source, les entreprises ne sont plus dépendantes de leurs fournisseurs. C'est pourquoi Red Hat s'engage à offrir à ses clients une qualité de service toujours plus grande, en même temps qu'une gamme étendue de solutions ouvertes, sûres et hautement performantes"* déclare **Franz Meyer,** Directeur Europe du Sud, Moyen-Orient et Afrique de Red Hat.

Même son de cloche chez **Pierre Ficheux,** CTO d'Open Wide qu'il définit comme *" une SSLL spécialisée dans l'intégration de composants Open Source, pas un éditeur. Open Wide est reconnue pour son expertise "métiers" et ses différents pôles de compétences qui lui permettent d'accompagner ses clients dans les différentes étapes de leurs projets d'intégration de solutions Open Source. "*

Une force internationale désormais rentable ?

Ceux qui conservaient l'idée d'un monde communautaire hors de toute contingence économique en sont pour leurs frais : *" L'étude Pac a montré que le marché global devrait être multiplié par sept entre 2003 et 2007 "*, note François Brancilhon, PDG de Mandriva qui souligne la bonne santé de son entreprise et son développement à l'international. *" Dans la zone Bric (Brésil, Russie, Chine), Linux et l'Open Source en général connaissent une très forte progression. Par exemple, nous livrons actuellement entre 20 000 et 40 000 machines sous Mandriva au Brésil ! "* Quant à **Olivier Guilbert,** PDG d'IdéalX qui s'est fait surtout connaître avec les systèmes d'authentifications fortes et notamment la signature électronique commercialisée en France et à l'étranger, son analyse est simple : *" Nous ne sommes aujourd'hui qu'au début de ce marché. Pour des raisons économiques mais aussi d'indépendance, les grandes administrations françaises ont choisi l'Open Source. Cette prise de position nous a bien aidés, car le secteur industriel a suivi et c'est aujourd'hui aux banques et assurances de s'y mettre. Maintenant, les entreprises veulent nous suivre mais avec des solutions plus simples. Il faut aller dans ce sens ".*

Grand vainqueur des formations : MySQL avec une croissance de 50%

D'où, l'importance donnée à la formation. Son volume (Linux) a fortement augmenté : *"+30% entre 2005 et 2006, comme Openoffice.org dans le domaine de la bureautique "*, indique Cyril Pierre de Geyer, responsable d'Anaska qui

s'est associé avec Ib-Groupe Cegos, autre spécialiste de la formation, pour concocter le premier observatoire des logiciels libres. Leur but est de quantifier l'impact véritable des logiciels de l'Open Source au sein des sociétés utilisatrices. Premier constat : le volume de formation augmente très rapidement. + 50 % chez Anaska et 115% chez Ib – groupe Cegos depuis la sortie de MySQL 5. Cette version était donc particulièrement attendue par les entreprises. *" Comme Lamp d'une manière générale et PHP plus précisément, MySQL commence à s'imposer comme une institution en entreprise avec de nouveaux cursus destinés aux utilisateurs avancés. En revanche, contrairement à l'image qui colle à MySQL, 70% des stagiaires MySQL ne travaillent pas sur des projets associés au Web "*, souligne Cyril Pierre de Geyer, également consultant, architecte Open Source et auteur de l'ouvrage *" PHP 5 avancé "* (Eyrolles).

Vers un monde Libre simplifié pour entrer dans le quotidien de tous les utilisateurs ?

Le baromètre Anaska-Ib-Groupe Cegos indique également qu'entre 2005 et 2006, PHP a progressé de 20% (25% des formations chez Anaska et 40% des demandes de certification), alors que Java ne progresse que de 5%. Pour Anaska, cela s'explique par le fait que Java est une technologie (langage et plate-forme) établie, largement enseignée en formation initiale et que les équipes qui se forment sont donc de plus en plus rares. Mais la grande leçon de ce baromètre rejoint la conclusion de l'enquête : l'Open Source s'ouvre à un public moins technique. *"Alors que les formations Open Source visaient initialement des développeurs et des responsables d'infrastructure web, les cursus s'enrichissent aujourd'hui de formations dédiées à un public moins technique. C'est notamment le cas des outils de gestion de la relation client comme SugarCRM et de reporting décisionnel tels que JasperSoft et l'ETL de Talend. L'arrivée de ces progiciels s'accompagne d'une demande de formation à la fois technique (déploiement, paramétrage) et fonctionnelle comme par exemple une formation sur le module " clients " de SugarCRM. Les entreprises qui demandent ce type de formations sont plutôt des administrations et de grandes entreprises "*, conclut Cyril Pierre de Geyer. Ce qui signifie que le tournant du monde des logiciels libres est pris et que les seules inconnues à cette équation sont le nombre d'utilisateurs qui y migreront, la progression future du CA de ces modèles économique si particuliers et surtout ... quelles réponses concocteront les adeptes du modèle propriétaire.

■ **Claire Padych**

Des C.A. en forte progression

Fin 2006, IDC indique que les ventes mondiales de serveurs sous Linux ont progressé plus vite que tous les autres OS : le marché Linux a atteint 1,5 milliard de dollars (+5,4% sur un an) et les serveurs Linux représentent désormais 11,8% du marché total des serveurs. Tous les responsables interrogés ont confirmé la bonne santé financière de leur entreprise. Deux exemples révélateurs : " une progression de 70% sur les 5 dernières années " selon Bertrand Mathellié de MySQL, ou " une multiplication par trois entre 2003 et 2006 ", pour Olivier Guilbert d'IdealX. Et cet accroissement de parts de marché induit, tout naturellement, des recrutements dans le secteur avec, pour les développeurs les plus aguerris, d'excellentes perspectives financières !

Le profil-roi : le contributeur en activité

Toutes les entreprises leader sur le marché de l'Open Source recrutent des nouveaux talents, soit par annonce, soit par cooptation. Mais toutes insistent sur la notion " philosophique " autour du Libre que les futurs candidats doivent posséder. *" Le contributeur, la personne qui, au-delà de tout CV est derrière une solution Open Source majeure pourra trouver du travail partout, indique* ***Alexandre Zapolsky,*** *PDG de Linagora. L'Open Source est à la portée de tout le monde, c'est vraiment la meilleure piste actuelle et à venir ! "*

Zend : la philosophie du développement durable

Laurent Bouffiès est directeur marketing et stratégie de Zend France. " Chez Zend, l'aspect communautaire demeure essentiel dans notre mode de fonctionnement. L'une des raisons est sans doute que les deux fondateurs, Andi Gutmans et Zeev Suraski sont deux des architectes à la base du PHP, à l'origine du moteur PHP (le Zend Engine) et ont cherché à conserver cette philosophie. L'entreprise tient à s'inscrire dans une idée éthique de responsabilité sociétale. Le business model de Zend est rentable, avec des outils de développement et d'exploitation Web dédiés et des services associés, pour gérer de manière opérationnelle et professionnelle des applications réalisées en PHP, afin d'industrialiser la filière. Et à côté, un framework sans cesse remis à jour par la communauté des développeurs. J'aime dire que PHP peut aussi signifier People Help People ! En fait, cette technologie est un vecteur idéal pour pénétrer le monde industriel et celui des petites entreprises d'un esprit bottom-up collaboratif parfaitement pragmatique. "

3 questions à ... Philippe Bournhonesque, *Directeur du développement* ***IBM Software Group***

Quelle est l'approche d'IBM en matière d'Open Source ?

Nous voulons des standards ouverts, ne pas travailler en chambre mais avec d'autres afin de faire de l'implémentation en Open Source. L'idée est de multiplier les consortiums et c'est le milieu du Libre qui a fait évoluer nos méthodes. Les équipes sont plus restreintes, sur le modèle des communautés. Nous avons quelques grands leaders de communautés qui sont salariés d'IBM comme Andrew Tridgell, l'un des principaux développeurs de Samba.

Quel est le business model développé par IBM ?

Nous développons trois axes. Depuis trois ans environ, nous sommes entrés sur le marché des " commodités " avec Eclipse. Avant, nous étions attaqués par Microsoft sur ce créneau. Désormais, nous voulons faire moins cher et plus répandu que lui. Autre axe: l'intégration. Le troisième axe est la compétence métier.

Quelle est la plus grande difficulté à laquelle vous êtes confronté dans l'Open Source ?

Elle est de deux ordres. D'une part, Microsoft est toujours là et le monde de l'Open Source le combat. Il y a véritablement deux camps. Ce qui nous sauve c'est que la communauté des logiciels libres est vivante. Mais c'est aussi sa particularité initiale qui nous oblige sans cesse à réfléchir au meilleur modèle économique : un " mix " entre gagner de l'argent d'un côté et avec ces financements, investir dans des développements.

■ **C. P.**

L'Open Source Valley :
bientôt la décision des pouvoirs publics ?

Un projet de pôle de compétitivité sur les logiciels libres dans l'Open Source Valley a récemment été déposé auprès des pouvoirs publics. Il y a unanimité entre la gauche et la droite sur ce projet. Il pourrait consacrer l'Open Source comme système porteur d'emplois et d'avancées technologiques au sein duquel le triptyque recherche publique, industrie et collectivités serait partie prenante.

Suite à la remise du rapport de la Commission sur l'économie de l'immatériel intitulé " réinventer notre modèle de croissance " et défendu par Maurice Levy (Publicis) et Jean-Pierre Jouyet (Inspection des Finances), le ministre de l'économie et des finances Thierry Breton a pris fait et cause à Bercy, le 4 décembre 2006, pour l'Open Source.

Le discours du patron du " Minefi " (qui évolue dans le monde du Libre) est sans équivoque : " L'économie " post industrielle ", c'est l'économie des services et de l'immatériel (...) L'économie de l'immatériel, c'est pour moi l'économie du talent et des technologies."

Et le grand argentier de plaider la cause de l'Open Source: " Exemple d'action, afin de replacer la France en tête de la course aux innovations technologiques : un nouveau pôle de compétitivité du logiciel libre et de l'Open Source. (...) La Commission a raison de s'interroger : où sont les Google, les Yahoo, les e-Bay français ? Actuellement, un nouveau modèle économique et technologique se met en place dans l'industrie de l'informatique autour des logiciels libres de droits (Linux). C'est une nouvelle opportunité majeure qui se présente (...) La France doit saisir cette chance (...) Compte tenu de l'importance majeure de l'industrie du logiciel dans l'économie de l'immatériel, et du dynamisme de notre recherche, des entreprises et des communautés françaises du logiciel libre, je souhaite que ces talents et ces compétences se fédèrent dans le cadre d'un pôle de compétitivité du logiciel libre et de l'"Open Source ". Mes équipes ont déjà entrepris de demander aux acteurs de se mobiliser en ce sens. " *" Le travail avance bien, et je souhaite un aboutissement et une labellisation rapide"*, ont été les paroles de conclusion du ministre Breton. Presque aussitôt, " les acteurs ", **Alexandre Zapolsky,** PDG de Linagora et président de l'Association des sociétés de services en logiciel libre (ASS2L), initiateur du projet, et **Roberto Di Cosmo**, professeur au laboratoire PPS à l'Université de Paris VII- Jussieu, expert académique, lui ont remis leur copie, proposant l'Open Souce Valley dans le cadre de ce nouveau pôle de compétitivité. Outre ces personnalités, le bureau provisoire est également composé de François Bancilhon, P-DG de Mandriva et de Stéphane Fermigier, P-DG de Nuxeo. Enfin, la plupart des entreprises de l'Open Source se sont jointes à l'équipe ainsi que de nombreuses universités, centres de recherche et écoles.

> ***Le travail avance bien, et je souhaite un aboutissement et une labellisation rapide***

Roberto Di Cosmo,
professeur au laboratoire PPS à l'Université de Paris VII- Jussieu

Alexandre Zapolsky,
PDG de Linagora et président de l'ASS2L

" Aujourd'hui, le Libre offre 10 000 emplois en France. A horizon 2010, il devrait en créer 60 000 ", indique Alexandre Zapolsky. *" C'est un mouvement de fond qui se généralise, les possibilités de l'Open Source sont innombrables et le rôle du monde académique est de valoriser les pistes de la recherche "*, confirme Roberto Di Cosmo. *"Un tel pôle n'existe pas. La dynamique créée sera extraordinaire "*, renchérit François Bancilhon. S'il est trop tôt pour connaître la décision de Thierry Breton ou du ministre qui lui succédera à Bercy en raison des échéances présidentielles, et en dépit de grincements de dents des défenseurs des logiciels propriétaires et de certains militants du logiciel libre qui n'ont pas été conviés, l'année 2007 devrait voir la labellisation de ce projet hors normes. Et ouvrir des perspectives intellectuelles et industrielles européennes et mondiales.

■ **Claire Padych**

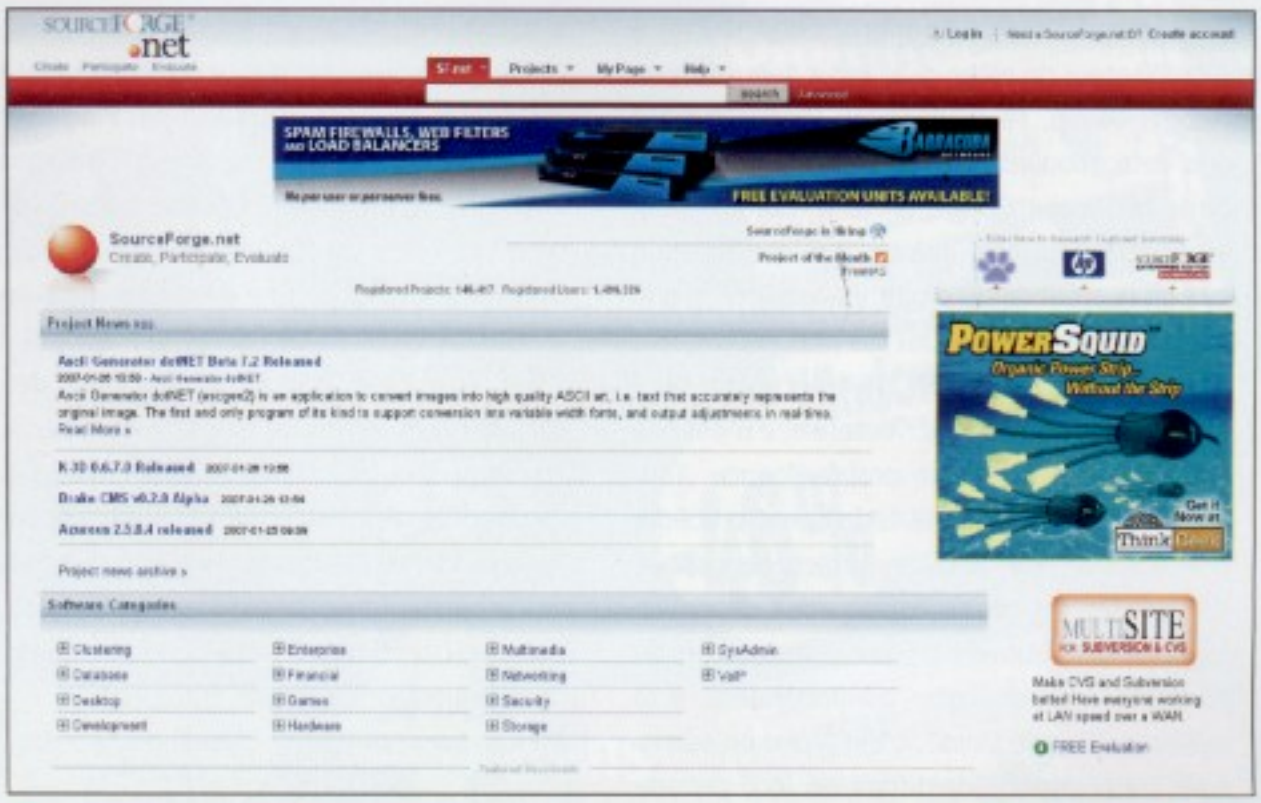
Quels sont les enjeux d'un pôle de compétitivité ?

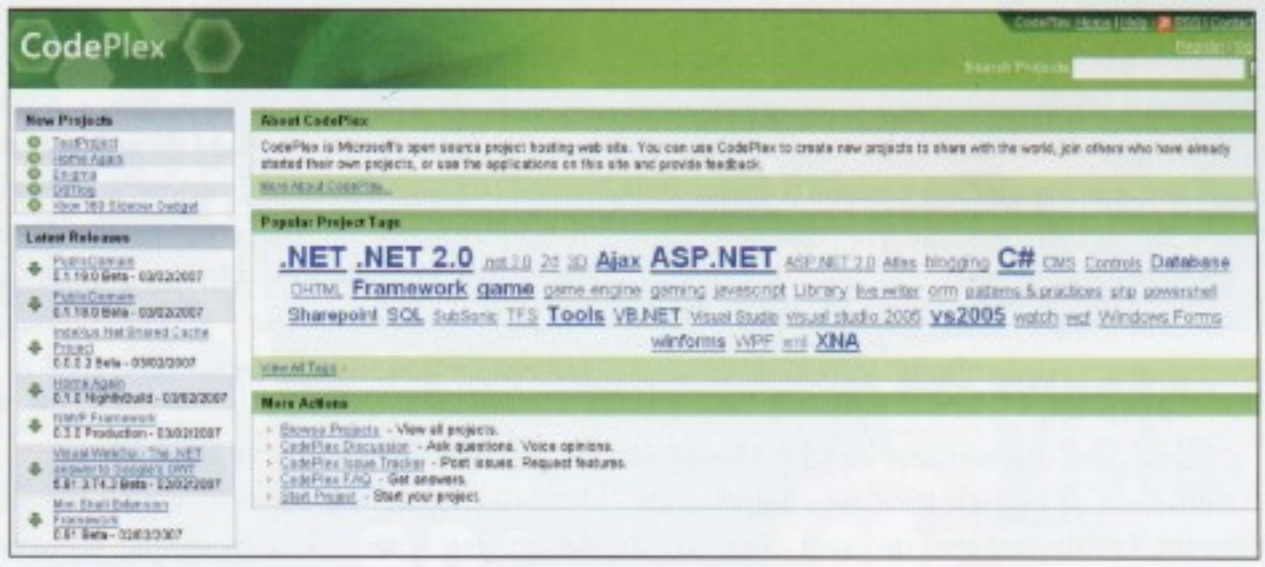
Confrontée aux mutations économiques internationales, la France a initié une politique industrielle de grande envergure : la mise en avant de facteurs clés de compétitivité industrielle, au premier rang desquels se trouve la capacité d'innovation par la R-D. Initiés en 2005 et regroupant les industries locales et régionales, la recherche académique et les collectivités, 66 pôles de compétitivité répondent à un cahier des charges. Celui de l'Open Source Valley a vocation à se développer en Région Parisienne car 70% des offres en 2006 sont localisées dans cette région (source : Lolix).

NOUVEAU : donnez votre avis sur ce numéro
www.programmez.com

Comment héberger un projet libre ?

Le monde du libre repose sur un fondamental incontournable : la communauté. Hors d'elle, point de salut pour un projet libre.





Si avec le temps la communauté globale d'utilisateurs est devenue très importante, celle des développeurs reste nettement plus restreinte. Et parmi eux, il faut parvenir à les regrouper autour d'une idée pour espérer qu'un projet de logiciel libre puisse parvenir à son terme dans les meilleures conditions. Pour cela, le meilleur moyen est de s'afficher, et surtout, de tenter de ne pas être noyé dans la masse. Des services en ligne, de plus en plus nombreux, fédèrent des communautés entières de développeurs en leur offrant une structure et des outils (collaboration, hébergement de site, etc.). Voici un tour d'horizon de l'offre disponible en la matière. Au-delà de la visibilité qu'offrent les services d'hébergement de projets libres au sein de leurs portails, ils mettent également à disposition des équipes de développeurs les outils indispensables à leur collaboration. Pour une équipe déjà en place, il s'agit probablement de l'aspect le plus important lors de la sélection du service qui hébergera le projet. En revanche, si l'on souhaite initier un projet depuis la base, alors la notoriété du site et les caractéristiques des développeurs qu'il fédère seront prioritaires : en effet, si celui-ci doit permettre le "recrutement" de collaborateurs, alors mieux vaut s'adresser à une communauté en adéquation avec le projet. Cette adéquation sera appréciée en fonction de différents critères, comme la langue naturelle principale, la culture technique (plates-formes cibles, langages) ou encore le type de projet. En effet, développer un logiciel de gestion de comptes personnels ou un plug-in pour un framework, par exemple, n'ayant pas du tout la même vocation, les motivations des développeurs sont très différentes.

Les généralistes

• SourceForge

Impossible d'aborder un tel sujet sans s'arrêter sur SourceForge, le premier, et le plus important des services d'hébergement de projets libres au monde (plus de 140 000 projets hébergés début 2007 !). Initialement libre elle-même, la plate-forme a donné naissance il y a quelques années à des clones, comme notamment Savannah, géré par la Free Software Foundation en réaction à l'instauration d'une politique douteuse envers le concept de logiciel libre. Des projets comme JBoss, Firebird, wxWidgets ou même le langage Python utilisent tout ou partie des services proposés par SourceForge à ses utilisateurs. Ces services sont nombreux : outils web avec gestion de droits basés sur les comptes utilisateurs SourceForge, service de téléchargement (avec archivage des versions et mise automatique en miroir), ferme de compilation, système de contrôle des sources CVS et Subversion, système de bug tracking, mailing-lists, etc.

Enfin, comme nous l'évoquions en introduction, SourceForge permet à votre projet d'être accessible, car immédiatement intégré au moteur de recherche du portail ainsi qu'à la "Software Map", listant tous les projets hébergés par catégories.

Adresse : http://www.sourceforge.net

• Google Hosting Services

Est-il un domaine sur internet que Google n'a pas encore investi ? En tout cas, certainement pas celui du logiciel libre, ni celui du développement. Disposant déjà d'une impressionnante base de données de code source libre (Code Search), Google a poursuivi sa contribution au logiciel libre en offrant depuis le début de l'année la possibilité d'héberger totalement ses projets libres.

Comme à son habitude, Google a opté pour la sobriété concernant sa plate-forme. Une interface très simple (austère ?), très légère, et des services limités, centrés sur l'essentiel : une page d'accueil, une section de téléchargement, un wiki (accessible via le serveur Subver-

sion), un bug tracker, et un accès aux sources. Ce dernier est exclusivement basé sur Subversion, CVS ayant été définitivement considéré comme obsolète. L'absence de forum et de mailing-lists s'explique par la possibilité offerte à l'utilisateur de créer, puis lier à son projet un Google Group, réunissant les fonctionnalités des deux modules en question, la puissance de la recherche Google en plus. Comme pour les concurrents, une licence libre est imposée lors de la création du projet. La liste comprend les licences certifiées OSI les plus répandues, comme la GPLv2, la LGPL, la licence Apache 2 ou encore la MIT licence. Toutefois, il n'est pas possible de préciser sa propre licence. Tout possesseur d'un compte GMail peut procéder à la création d'un nouveau projet. Un espace de 100 Mo est alloué aux sources déposées sur le serveur Subversion. Les contributeurs du projet sont ajoutés par l'administrateur, à la méthode "Google Docs", c'est à dire en saisissant simplement l'identifiant de leur compte GMail dans une zone de texte. La gestion des droits est par contre vraiment simpliste, puisque "Project Hosting" ne distingue que deux types de contributeurs : les propriétaires, qui ont tous les droits, y compris d'administration, et les membres, qui n'ont que des droits de modération.

Adresse : http://code.google.com/hosting/

• Codeplex

Codeplex est une véritable invitation au développement de logiciels libre, bien que la plate-forme sous-jacente ne soit pas libre elle-même. Lancé en 2006, le site ne rencontre pourtant qu'un succès limité, avec moins de 1000 projets hébergés.

Les conditions d'hébergement sont très souples : le projet doit être soumis à une licence libre, le projet doit être actif, et enfin, le code source doit être publié (les édicteurs de ces conditions ne semblent pas très familiers avec la notion d'open source...). Autrement dit, rien n'oblige l'utilisateur à proposer un projet utilisant les technologies Microsoft. En principe, en tout cas, car dans la pratique, il est indispensable de disposer de Visual Studio Team Explorer pour exploiter les services principaux du site, comme le contrôle de sources. Sinon, les habituels forums, bug trackers et systèmes de téléchargement ne manquent pas à l'appel, et sont des plus classiques.

Ceci expliquant peut-être pourquoi les projets sont encore peu nombreux aujourd'hui, les développeurs en environnement Microsoft n'étant pas historiquement très impliqués dans la dynamique du logiciel libre. L'avenir seul saura juger de la perénité de cette initiative de la part de Microsoft, même si certains projets de taille hébergés par Codeplex, tel VisualWeb Gui, laissent espérer que le pari puisse à terme être gagné.

Adresse : http://www.codeplex.com

Les spécialisés

En introduction, nous parlions du type de projet comme critère de sélection de sa plate-forme d'hébergement. En effet, dans certains cas, il est préférable de négliger la visibilité pour favoriser un ciblage de ses futurs collaborateurs plus précis. Un exemple des plus pertinents, et s'accordant avec les sujets récemment traités dans le magazine : les forges type CakeForge et RubyForge.

Ces plates-formes offrent des services très similaires à SourceForge, et pour cause : toutes deux utilisent une même version dérivée de la plate-forme SourceForge (SF 2.5, datant de l'époque où tout SourceForge était libre). Mais les projets hébergés, en plus d'être libres sont fortement liés par une composante technique identique. Vous trouverez sur CakeForge essentiellement des projets de contributions liés à la plate-forme elle-même, comme des plug-in, des helper ou encore des scripts d'administration. Au-delà, vous pourrez également y trouver de nombreux projets dont le point commun sera d'être basés sur le même framework, et de vous permettre d'être mis en relation avec des développeurs issus de la même culture technique que vous.

De nombreux frameworks proposent ce genre de services. Certains, comme CakeForge précisément, permettent moyennant une modeste contribution financière, de bénéficier de la plate-forme dans le cadre de projets privés (donc, potentiellement non-libres). Si votre projet repose sur un framework libre, commencez donc par vous renseigner sur son site officiel, pour voir si une forge lui est dédiée.

Conclusion

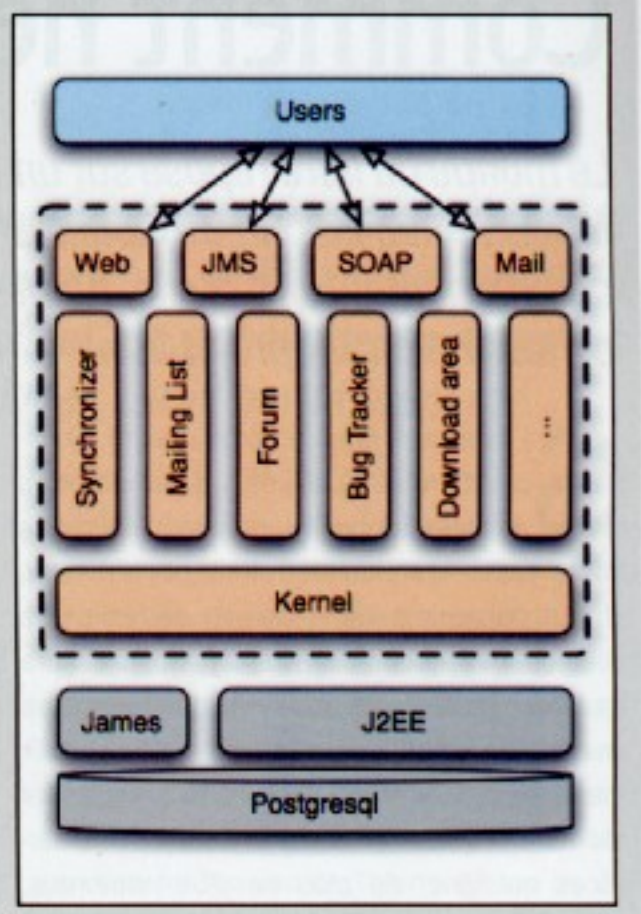
Pendant longtemps, l'hébergement de projets libres est passé systématiquement par SourceForge. Les besoins des utilisateurs se diversifiant, l'offre se diversifie également. Il est très délicat de conseiller telle ou telle forge de manière générale. Ce choix, comme nous l'avons vu, dépend de critères variés, parfois contradictoires. Il y a cependant un point qui est constant, quelle que soit la nature de votre projet : utiliser une plate-forme de ce type, peu importe laquelle, mettra à votre disposition un minimum d'outils nécessaire à une bonne organisation et gestion de votre projet. La méthodologie qu'impose un système de contrôle de sources, un bug tracker ou encore un gestionnaire de tâche ne pourra qu'optimiser le développement, et faciliter le travail en équipe.

Gauthier Delamarre

LibreSource : l'autre forge

Il existe de multiples forges pour les projets ouverts. L'éditeur Artenum édite la forge LibreSource. Il s'agit d'une plate-forme d'hébergement collaborative, accessible partout, avec un accès sécurisé. Il en coûtera 29,90 euros par mois (édition express). Actuellement, l'outil est en version 2.1. Cette version intègre la gestion de source Subversion. Elle améliore aussi la localisation en Français et les espaces wikis. Avec LibreSource, on peut créer des forums, des suivis de bugs. Il est même possible d'interfacer son espace LibreSource avec un annuaire LDAP. Pour développer l'environnement, Artenum a utilisé Java. Un module de gestion de configuration est disponible (LibreSource Synchroniser). Il existe une version community et entreprise.

Pour en savoir plus : http://dev.libresource.org/



Démarrer avec Ruby On Rails

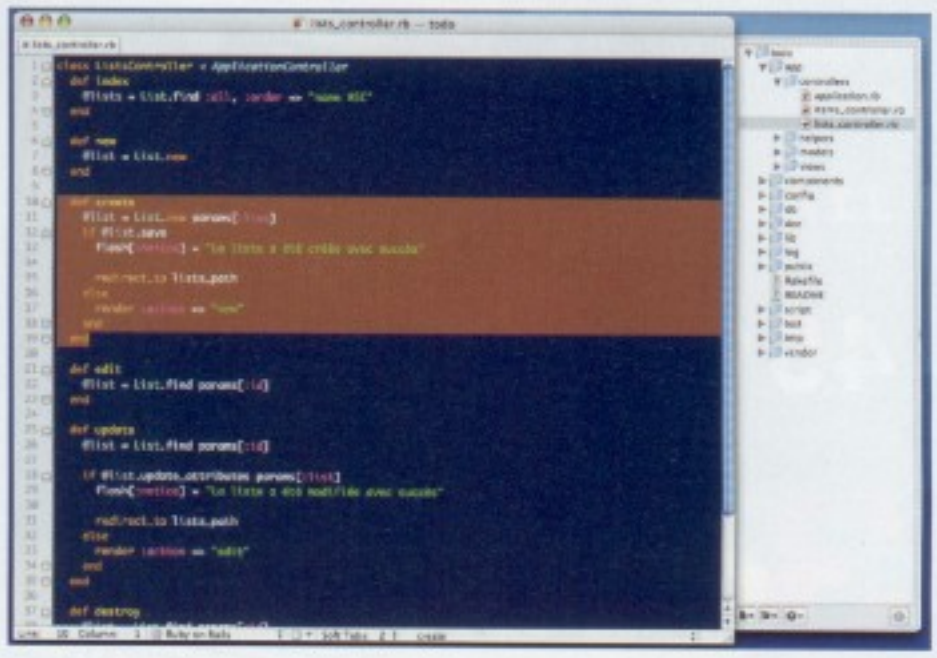
2re partie

Après avoir installé et configuré l'application Todo réalisée avec Ruby On Rails, les modèles et les routes ont été effectués. Il ne reste plus qu'à écrire les contrôleurs et les vues.

Contrôleurs

Les contrôleurs ont été créés dans la première partie, mais il n'y a rien à l'intérieur. Dans un contrôleur, on va avoir une succession de méthodes. Chaque méthode publique représente une action. Voici à quoi va ressembler le contrôleur lists :

```
class ListsController < ApplicationController
  def index # cette méthode correspond à l'action index
    @lists = List.find :all, :order => "name ASC"
  end

  def new
    @list = List.new
  end

  def create
    @list = List.new params[:list]
    if @list.save
      flash[:notice] = "La liste a été créée avec succès"

      redirect_to lists_path
    else
      render :action => "new"
    end
  end

  def edit
    @list = List.find params[:id] # Récupération d'une liste à partir d'un ID
  end

  def update
    @list = List.find params[:id]

    if @list.update_attributes params[:list]
      flash[:notice] = "La liste a été modifiée avec succès"

      redirect_to lists_path
    else
      render :action => "edit"
    end
  end

  def destroy
    @list = List.find params[:id]
    @list.destroy

    redirect_to lists_path
  end
end
```



Les différentes actions sont créées. @list est une variable de classe. On pourra s'en servir dans les vues. Notons que l'action show n'a pas été faite car elle n'est pas nécessaire. Nous utilisons ici le modèle List.

Voici quelques méthodes de ActiveRecord utilisées :

- find pour récupérer un ou plusieurs enregistrements
- new pour créer un nouvel objet
- save pour enregistrer dans la base de données ce nouvel objet
- update_attributes pour mettre à jour les champs
- destroy pour détruire l'enregistrement

Ces méthodes vont envoyer la requête SQL correspondante au serveur. Plus besoin de connaître le SQL, tout est Ruby !

La variable params stocke les informations qui sont envoyées. C'est avec cette variable que l'on pourra récupérer les informations envoyées par un formulaire ou celles qui sont passées à l'URL.

flash[:notice] permet de stocker un message dans une session pour pouvoir l'afficher sur la page d'après (puisque l'on fait une redirection, ce sera sur la page de redirection).

La méthode render permet d'afficher une vue spécifique. S'il n'y a aucun rendu avant la fin de l'action, la vue ayant le nom de l'action sera affichée. Le contrôleur items aura la même structure, à la différence près qu'on utilisera le modèle Item et non List. De plus, il faudra lui ajouter une méthode pour récupérer la bonne liste à chaque appel de ce contrôleur (pour récupérer ensuite les éléments attachés à la liste) et 2 actions, une pour terminer l'action (mais sans la supprimer, ce qui rend l'action destroy inutile) et une pour la réactiver.

```
class ItemsController < ApplicationController
  before_filter :find_list

  def index
    @items = Item.find :all, :order => "text ASC"
  end
```

```
def new
  @item = Item.new
end

def create
  @item = Item.new params[:item]
  @item.list = @list
  if @item.save
    flash[:notice] = "L'élément a été créé avec succès"

    redirect_to items_path(@list)
  else
    render :action => "new"
  end
end

def edit
  @item = Item.find params[:id]
end

def update
  @item = Item.find params[:id]

  if @item.update_attributes params[:item]
    flash[:notice] = "L'élément a été modifié avec succès"

    redirect_to items_path(@list)
  else
    render :action => "edit"
  end
end

def done
  @item = Item.find params[:id]
  @item.done = true
  @item.save

  redirect_to items_path(@list)
end

def redo
  @item = Item.find params[:id]
  @item.done = false
  @item.save

  redirect_to items_path(@list)
end

private

def find_list
  @list_id = params[:list_id]
  redirect_to lists_path unless @list_id
  @list = List.find @list_id
end
end
```

Pour appeler une méthode à chaque chargement de la classe, on utilise la méthode before_filter avec le nom de la méthode que l'on veut appeler. Nous appelons la méthode find_list et comme cette méthode n'est pas une action, il faudra modifier sa visibilité.
Deux nouvelles actions par rapport au contrôleur lists : done et redo pour terminer et réactiver l'élément. Les deux actions ne suivant pas les conventions, il faudra les indiquer dans la configuration de nos routes :

```
map.resources :lists do |list|
  list.resources :items, :member => { :done => :put, :redo => :put }
end
```

Ces deux actions mettent à jour l'enregistrement, on ne doit donc y accéder qu'avec la méthode PUT. Pour accéder aux URL, Rails génère là aussi les méthodes nécessaires : done_item_path(list_id, item_id) et redo_item_path(list_id, item_id). Les méthodes pour construire les URL doivent avoir en premier argument l'ID de la liste car les ressources des éléments sont les enfants des ressources des listes. En mettant la variable @list, Rails va automatiquement reconnaître un objet ActiveRecord et va en extraire l'ID, il n'est donc pas nécessaire de préciser @list.id.

Vues et Helpers

Les vues sont stockées dans app/views. Il y a trois répertoires : lists, items, layouts. lists correspond aux vues qui serviront à l'affichage des actions du contrôleur lists, et c'est la même histoire pour items. Le répertoire layouts permet de stocker le squelette des pages que l'on va faire. Ce squelette permettra de ne pas répéter dans chaque vue, les essentiels de la page : la section <head>, une barre latérale si on veut en mettre... Nous devons appeler notre layout application.rhtml pour ne pas à avoir à le configurer ensuite dans les contrôleurs (toujours ces conventions) :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
  <meta http-equiv="content-type" content="text/html;charset=UTF-8" />
  <title>Todo</title>
</head>
<body>

<p style="color: green"><%= flash[:notice] %></p>

<%= yield %>

</body>
</html>
```

L'écriture est similaire à ce qu'on peut voir dans les autres langages. Nous avons le HTML et les balises <% %> et <%= %> servent à exécuter du Ruby. <%= flash[:notice] %> permet l'affichage du message que l'on a mis dans nos contrôleurs et <%= yield %> représente la vue qui sera insérée dans ce squelette.
Passons aux vues de lists. Il y en a 3 : index.rhtml, new.rhtml, edit.rhtml. L'extension des vues HTML est .rhtml. new.rhtml et

edit.rhtml auront les mêmes champs pour ce qui est du formulaire. Pour éviter d'écrire 2 fois le formulaire, nous allons le mettre dans un partiel _form.rhtml. Il ne peut être appelé qu'avec render, ce ne peut pas être la vue d'une action. Les partiels sont préfixés par _. Commençons donc par construire les 2 formulaires :

```
<% form_for :list, :url => lists_path do |f| %>
   <%= render :partial => "form", :object => f %>
   <%= submit_tag 'Ajouter' %>
<% end %>

<% form_for :list, :url => list_path(@list), :html => { :method => :put } do |f| %>
   <%= render :partial => "form", :object => f %>
   <%= submit_tag 'Editer' %>
<% end %>
```

Le premier code est le formulaire de new.rhtml et le deuxième code le formulaire de edit.rhtml. Nous créons un bloc générant un formulaire en y indiquant la variable de notre contrôleur (:list se réfère à @list), l'URL de destination et la méthode (PUT pour edit.rhtml). Ensuite, nous insérons le bouton de validation avec le texte adéquat. Enfin, nous appelons le partiel qui contiendra les champs qui sont communs aux 2 formulaires. Nous passons la variable f à notre partiel pour pouvoir l'exploiter :

```
<p>Nom de la liste : <%= form.text_field :name %></p>
```

L'objet f prend le nom form dans le partiel. Il permet de générer les champs avec la convention de nommage nécessaire pour manipuler les données du formulaire dans le contrôleur. Prenons la vue new.rhtml, le code HTML résultant est le suivant :

```
<form action="/lists" method="post">
   <p>Nom de la liste : <input id="list_name" name="list[name]" size
="30" type="text" /></p>
   <input name="commit" type="submit" value="Ajouter" />
</form>
```

Tout se génère correctement avec un code valide. L'objet f permet d'insérer un nom spécifique. Le nom list[name] permettra de récupérer la valeur du champ dans le contrôleur avec params[:list][:name].
Pour ce qui est de la vue index.rhtml, ce n'est qu'une simple liste :

```
<h1>Listes Todo</h1>
<ul>
<% @lists.each do |list| %>
   <li><%= link_to list.name, items_path(list) %> - <%= link_to "Editer",
edit_list_path(list) %> - <%= link_to "Supprimer", list_path(list), :method
=> :delete %></li>
<% end %>
</ul>
<%= link_to "Ajouter une liste", new_list_path %>
```

Nous affichons toutes les listes avec un lien pour éditer la liste et un pour la supprimer (il ne faut pas oublier de spécifier la bonne méthode pour ce lien). En cliquant sur la liste, on arrive sur la liste des éléments. Passons maintenant aux vues items. Il y aura les mêmes vues : index.rhtml, new.rhtml, edit.rhtml et le partiel _form.rhtml. Je ne détaillerais ici que la vue index.rhtml car les autres sont similaires à celles de lists.

```
<h1>Eléments de "<%= @list.name %>"</h1>
<ul>
<% @items.each do |item| %>
   <li><%= link_to item.text, edit_item_path(@list, item), :style => item.
done? ? "text-decoration: line-through;" : "" %> - <%= done_or_redo
_item_link @list, item %></li>
<% end %>
</ul>
<%= link_to "Ajouter un élément à la liste", new_item_path %>
```

Nous avons donc ici les éléments d'une liste avec la possibilité d'en ajouter un et d'éditer celui que l'on veut. Petite particularité, la méthode done_or_redo_item_link. Cette méthode permet de créer un lien pour terminer ou réactiver un élément. Elle n'a pas été générée par Rails, mais a été créée en tant que helper. Les helpers se trouvent dans le répertoire app/helpers. Nous sommes dans les vues correspondant au contrôleur items, le fichier de helper est donc items_helper.rb et il ressemble à cela :

```
module ItemsHelper
  def done_or_redo_item_link list, item
    if item.done?
      link_to "Réactiver", redo_item_path(list, item), :method => :put
    else
      link_to "Terminer", done_item_path(list, item), :method => :put
    end
  end
end
```

Une méthode a été créée pour faire un lien permettant de terminer ou réactiver un élément en fonction du champ done de la table items. Ce code est mis dans un helper pour ne pas "polluer" la vue. En effet, mettre toutes ces lignes dans la vue dégraderait la lisibilité du code, il est donc préférable dans ce cas de le mettre dans un helper.

Conclusion

On ne le dirait pas et pourtant, l'application est déjà terminée. Il aura fallu environ 150 lignes de code (en comptant les vues !) ! On se rend compte à cet instant que Ruby On Rails permet de faire les choses très rapidement. L'apprentissage peut paraître compliqué après avoir lu cet article, mais on comprend vite les mécanismes, et la progression est très rapide. Tout est en fait simple à comprendre et à mettre en application. Le code reste très lisible et la maintenance est donc facilitée. A l'heure actuelle, il est indispensable d'avoir des outils alliant productivité et solidité. RoR remplit ce rôle à merveille. Il y a bien d'autres possibilités avec Rails comme l'intégration des web services, du développement par tests, de l'Ajax et d'un système de plug-in permettant d'étendre les possibilités de Rails.

Nicolas Mérouze
Développeur Ruby On Rails
http://www.boldr.fr

Ruby on Rails, AJAX et RJS

Rails intégrait déjà le support d'AJAX à travers les bibliothèques Prototype et Script.aculo.us. La nouveauté la plus marquante de la version 1.1 fut sans nul doute l'arrivée des Templates RJS. En effet si les vues déjà disponibles (fichiers .rhtml et .rxml) nous permettaient de générer facilement du code HTML et XML, cette nouvelle adjonction nous donne la possibilité de générer du JavaScript à partir de code Ruby ! Ce qui va grandement nous faciliter la tâche pour, par exemple, mettre à jour plusieurs éléments d'une page HTML, sans que le navigateur ait besoin de la recharger.

Pour commencer, nous allons créer une nouvelle application, pour cela vous pouvez ouvrir un terminal et taper la commande suivante :

```
rails resolutions_rjs –database=sqlite3
```

Pour ma part, j'utilise généralement SQLite3 pendant le développement et le fait de passer l'option –database=sqlite3 lors de la création de l'application m'évite d'avoir à éditer le fichier config/database.yml. Dans le cas où vous utiliseriez une autre base de données, éditez le fichier et configurez le correctement avant d'aller plus loin. Plaçons nous ensuite dans le répertoire de notre application afin de générer le modèle qui nous permettra de stocker dans la base la foultitude de bonnes résolutions qui ne manqueront pas de nous venir à l'esprit.

```
cd resolutions_rjs
```

Ce modèle sera nommé sans grande surprise Resolution, au singulier, et la table de la base de données correspondante sera donc conventionnellement appelée resolutions, au pluriel et en minuscules.

```
ruby script/generate model Resolution
```

Nous allons maintenant éditer le fichier de migration, qui a été créé par Rails lors de la génération du modèle. Ce fichier db/migrate/001_create_resolutions.rb contient la définition, en langage Ruby, du schéma de la table resolutions, ajoutons lui une colonne titre qui sera de type "chaîne de caractère":

```
class CreateResolutions < ActiveRecord::Migration
  def self.up
    create_table :resolutions do |t|
      t.column :titre, :string
    end
  end

  def self.down
    drop_table :resolutions
  end
end
```

Une fois cette opération effectuée, il ne nous reste plus qu'à lancer la commande de migration rake db:migrate afin que Rails génère le code SQL, compatible avec le SGBDR que nous avons configuré dans le fichier config/database.yml, et charge le schéma dans la base de données. À ce stade, il nous reste encore à créer un contrôleur pour notre future application, nous le nommerons lui aussi resolutions et il contiendra les trois méthodes suivantes : index, ajout et suppression.

```
ruby script/generate controller resolutions index ajout suppression
```

Un nouveau fichier app/controllers/resolutions_controller.rb a été créé et voici ce qu'il contient :

```
class ResolutionsController < ApplicationController

  def index
  end

  def ajout
  end

  def suppression
  end
end
```

Maintenant, il ne nous reste plus qu'à supprimer le fichier public/index.html et à éditer le fichier de routes config/routes.rb, modifiez-le ainsi :

```
ActionController::Routing::Routes.draw do |map|

  map.with_options :controller => 'resolutions' do |m|
    m.accueil ''
    m.ajout '/ajout', :action => 'ajout'
    m.suppression '/suppression/:id', :action => 'suppression'
  end

  # Install the default route as the lowest priority.
  map.connect ':controller/:action/:id'
end
```

Notez au passage l'utilisation de with_options afin de ne pas avoir à répéter l'option :controller => 'resolutions', sympathique non ?

Nous pouvons maintenant démarrer le serveur web avec la commande rubyscript/serveur.
Et si tout se passe bien, en faisant pointer notre navigateur favori à l'adresse http://localhost:3000 nous devrions voir apparaître ceci (ci-contre).

Création du fichier de mise en page

Certes, ceci est loin d'être graphiquement très abouti ni même correct au sens HTML. Mais nous allons y remédier immédiatement en créant un nouveau fichier de gabarit app/views/layout/application.rhtml et en y copiant ce code :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="fr" lang="fr">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <%= stylesheet_link_tag 'style' %>
  <%= javascript_include_tag :defaults %>
  <title>RoR: Résolutions on Rails !</title>
</head>
<body>
  <div id="page">
      <div id="haut">
          <div id="titre_page">
              <a href="/">Mes résolutions pour l'année <%= Time.
now.year %></a>
          </div>
      </div><!- fin div haut ->
      <div id="milieu">
              <%= yield %>
      </div>
      <div id="bas"></div>
  </div> <!- fin div page ->
</body>
</html>
```

Et en rechargeant la page voilà le résultat ;-) :

Il est néanmoins probable, voire même certain, que vous ne verrez pas la même chose tant que vous n'aurez pas récupéré les jolies images qu'il vous suffira de copier dans le répertoire public/images et la feuille de style, nommée style.css, que vous placerez judicieusement dans le répertoire public/stylesheets .

Notez la présence de cette ligne dans le fichier application.rhtml, son importance est capitale car c'est elle qui charge toutes les bibliothèques JavaScript Prototype et Script.aculo.us qui s'avèreront indispensables au bon fonctionnement de notre future application AJAX. Par ailleurs dans le cas où le fichier public/javascripts/application.js est présent il sera chargé lui aussi.

```
<%= javascript_include_tag :defaults %>
```

Création du formulaire d'ajout de résolutions

Créons le formulaire qui va nous permettre d'ajouter nos résolutions, et cela se fera sans qu'il y ait besoin de rafraîchir la page, tout en AJAX s'il vous plaît ! Vous allez vite vous rendre compte qu'avec Rails il s'agit d'une simple formalité.

Nous pourrions simplement créer le formulaire directement dans le fichier de vue app/views/resolutions/index.rhtml, mais nous allons néanmoins utiliser un fichier partiel, tout d'abord parce que c'est plus propre et ainsi nous disposerons en plus d'une première bonne résolution à ajouter (En 2007, j'utiliserais les fichiers partiels !).

Petit rappel au passage, un fichier partiel est un fichier dont le contenu est inséré à l'endroit où il est appelé et dont le nom doit impérativement commencer par un tiret bas " _ " , néanmoins ce tiret ne doit pas apparaître lors de l'appel du partiel. Cela semble étrange à première vue mais on s'y habitue très vite, copiez le code ci-dessous dans un nouveau fichier app/views/resolutions/_form.rhtml :

```
<div id="form-contenu">
<% form_remote_for :resolution,
   Resolution.new,
   :url => {:action => 'ajout'},
   :html => {:id => 'form-resolutions'} do |f| %>

   <p><label for="resolution_titre">Titre:</label>

   <%= f.text_field 'titre', :size => 40 %>

   <%= submit_tag 'Envoyer', :id => 'commit' %></p>

<% end %>
</div>
```

La partie la plus importante ici est l'utilisation de l'assistant form_remote_for qui va générer pour nous tout le code nécessaire pour soumettre les données en arrière-plan à l'aide d'un appel AJAX. Les données transmises par le formulaire, à savoir le contenu du champ texte, seront envoyées à la méthode ajout de notre contrôleur.

Vous trouverez plus de détails concernant cet assistant, et bien d'autres, dans l'API Rails à cette adresse : http://www.rubyonrails.org/api/classes/ActionView/Helpers/PrototypeHelper.html#M000415

Pour vous convaincre de l'importance d'utiliser les assistants de Rails voici le code qui a été généré par remote_form_for :

```
<form action="/ajout" id="form-resolutions" method="post" onsubmit="new
Ajax.Request['/ajout', {asynchronous:true, evalScripts:true, parameters:
Form.serialize(this)}]; return false;">
...
</form>
</div>
```

Certes, nous aurions très bien pu écrire ce code à la main et cela ne changerait rien au niveau du fonctionnement, mais pourquoi s'ennuyer et risquer de commettre des erreurs de frappe quand on dispose d'assistants Rails. Éditons maintenant la vue app/views/resolutions/index.rhtml et remplaçons le code existant par cet appel au fichier partiel, qui au final affichera notre formulaire :

```
<%= render :partial => 'form' %>
```

Rechargez la page. Si vous cliquez sur le bouton "Envoyer" vous constaterez dans le fichier (log/development.log), que la soumission du formulaire AJAX s'est effectuée correctement. Bien entendu, il nous faut maintenant rajouter le code nécessaire dans la méthode ajout de notre contrôleur.

Edition du contrôleur

Ouvrez le fichier app/controller/resolutions_controller.rb et éditez-le :

```
class ResolutionsController < ApplicationController
```

```
  def index
    @resolutions = Resolution.find :all
  end

  def ajout
    @resolution = Resolution.create(params[:resolution])
  end

  def suppression
  end
end
```

La méthode index récupère un tableau d'objets contenant toutes les résolutions stockées dans la base de données. Quant à la méthode ajout, elle crée un objet à partir du modèle Resolution, lui passe les paramètres soumis par notre formulaire et sauve le nouvel enregistrement dans la base. Avec les deux lignes que nous venons de rajouter, notre contrôleur est opérationnel, difficile de faire plus concis et explicite non ?

Affichage AJAX de la liste des résolutions

Créons un nouveau fichier partiel dans lequel nous insèrerons le code permettant d'afficher une et une seule résolution. Ce fichier sera appelé avec en argument la collection d'objets (@resolutions) retournée par la méthode index du contrôleur. Collez ce code dans le fichier partiel app/views/resolutions/_resolution.rhtml

```
<div id="resolution-<%= resolution.id %>" class="resolution">

  <div class="resolution-haut"></div>

  <div class="resolution-texte">
    <h4><%= h resolution.titre %></h4>
  </div>

  <div class="resolution-bas"></div>

</div>
```

Et celui-ci dans app/views/resolutions/index.rhtml juste en dessous de l'appel à render :

```
<div id="liste">
  <div id="resolutions">
      <%= render :partial => 'resolution', :collection => @resolutions %>
  </div>
</div>
```

Maintenant si vous ajoutez une résolution, il ne se passe rien au premier abord. Il nous manque une composante essentielle, en effet la résolution est bien envoyée par le formulaire AJAX, notre contrôleur fait correctement son travail et la sauve dans la base, mais après l'appel à la méthode ajout il nous faut mettre à jour la liste afin d'afficher le nouvel enregistrement.
Nous allons utiliser un fichier RJS. Renommez le fichier app/views/resolutions/ajout.rhtml en app/views/resolutions/ajout.rjs. Attention, si les deux fichiers sont présents c'est le .rhtml qui sera utilisé prioritairement par Rails ! Et comme nous avons judicieusement nommé notre fichier (ajout.rjs), ce dernier sera appelé automatiquement par Rails après exécution de la méthode du même nom (ajout) du contrôleur ...
Voici le code à insérer dans app/views/resolutions/ajout.rjs:

```
page.insert_html :bottom, 'resolutions', :partial => 'resolution'
page.visual_effect :Pulsate, "resolution-#{@resolution.id}", :duration => 1
```

Il se trouve qu'après avoir renommé le fichier ajout.rhtml en ajout.rjs j'ai rencontré un problème de cache avec le serveur WEBrick et le fichier RJS n'était pas exécuté, donc arrêtez et redémarrez le serveur web si besoin. L'objet page que nous utilisons ici est en fait une instance du générateur de JavaScript de Rails (JavaScriptGenerator) qui s'occupe de retourner le code JavaScript adéquat au navigateur. Ici le contenu du fichier partiel _resolution.rhtml sera exécuté et inséré juste après le dernier élément présent dans l'élément div dont l'id html est "resolutions". La méthode visual_effect quant à elle est un assistant qui nous permet d'utiliser facilement les différents effets de la bibliothèque Script.aculo.us. Désormais notre petite application nous permet de stocker et d'afficher nos résolutions, mais elle a tout de même de sévères lacunes. Par exemple, il est possible de soumettre le formulaire sans avoir renseigné le champ " Titre ", ce même champ n'est pas réinitialisé après ajout d'une résolution, et ce serait bien pratique d'avoir une fonction de suppression...

Et pour terminer ...

Dans cette dernière partie, nous allons rajouter la possibilité de supprimer des résolutions et essayer de rendre notre application un tantinet plus robuste. Éditez le fichier de modèle app/models/resolution.rb et rajoutez la ligne suivante juste après la définition de la classe :

```
validates_presence_of :titre, :message => 'doit être renseigné...essayez à nouveau.'
```

Maintenant il est obligatoire de renseigner le champ " Titre " du formulaire pour qu'une nouvelle résolution puisse être créée.
Modifions maintenant le fichier app/views/resolutions/index.rhtml afin de pouvoir gérer l'affichage des messages d'erreur et tant que nous y sommes, pourquoi ne pas implémenter un système de suppression par glisser-déposer sur une poubelle ? cela apportera la touche "Web deux" qui fera pâlir d'envie vos collègues, et avec Rails c'est l'affaire de quelques lignes de code ! Remplacez donc tout le contenu de app/views/resolutions/index.rhtml par ce code :

```
<%= render :partial => 'form' %>

<!- Affichage des messages d'erreur ->
<div id="erreur" style="display: none;"></div>

<div id="liste">
  <!- Suppression des résolutions, oui parfois ça arrive... ->
  <div id="poubelle" class="poubelle"></div>

  <%= drop_receiving_element "poubelle",  :accept => "resolution",
  :hoverclass => "poubelle_pleine",
  :url => suppression_url %>
```

```
  <div id="resolutions">
      <%= render :partial => 'resolution', :collection => @resolutions %>
  </div>
</div>

<!- indicateur d'activité AJAX ->
<div id="spinner" style="display: none;"><%= image_tag 'spinner.gif'
%></div>
```

Et ajoutez ensuite cette ligne à la fin du fichier partiel app/views/resolutions/_resolution.rhtml

```
<%= draggable_element "resolution-#{resolution.id}", :revert => true %>
```

Maintenant, rafraîchissez la page et glissez-déposez une de vos résolutions sur la poubelle, joli hein ? Ceci étant, pour que la suppression soit réellement effective, il nous reste une dernière chose à faire au niveau du contrôleur app/controllers/resolutions_controller., voici le code à insérer après la méthode ajout:

```
def suppression
  # params[:id] correspond ici à *l'id html* de l'élément qui a été déposé
  # sur la poubelle, donc 'resolution-xx', il faut donc extraire 'xx' qui
  # est l'id de la résolution à supprimer
  resolution_id = Resolution.find(params[:id].split('-')[1]).destroy.id rescue nil

  render :update do |page|
   # Si resolution_id == nil c'est que notre résolution a déja été supprimée,
   # ce qui peut-être le cas si on ouvre 2 navigateurs...donc on redirige sur
   # la page d'accueil afin de rafraîchir la liste
    resolution_id ? page[params[:id]].remove : page.redirect_to(:accueil_url)
   end
 end
```

Ce qui est très intéressant ici, c'est la ligne render :update do |page|, en effet Rails nous offre la possibilité de faire du traitement RJS directement à partir du contrôleur, sans avoir à créer spécifiquement un fichier .rjs. Certains puristes rétorqueront peut-être que c'est mal et que ça brise le modèle MVC, en ce qui me concerne je considère que lorsque c'est utilisé à bon escient, c'est une fonctionnalité extrêmement utile...
Ça y est ! Maintenant, vous pouvez effectivement supprimer vos résolutions, mais conservez-en tout de même quelques-unes, ça peut servir.
Pour terminer, nous allons rajouter l'affichage du message d'erreur et un indicateur global d'activité AJAX.
La fonction d'affichage du message d'erreur, appelée le cas échéant par notre fichier ajout.rjs, sera stockée dans un fichier helper juste pour que vous sachiez que les vues RJS ont la possibilité d'utiliser les fichiers helpers, tout comme les vues RHTML ou RXML. Rajoutez donc ce code dans le fichier app/helpers/resolutions_helper.rb :

```
def affichage_erreurs(resolution)
  page.replace_html 'erreur' , "<h2>Une erreur s'est produite !</h2>
<p>#{resolution.errors.full_messages.join('<br />')}</p>"
end
```

Et remplacez le contenu du fichier app/views/resolutions/ajout.rjs par celui-ci :

```
# erreur l'enregistrement n'a pas été sauvegardé
if @resolution.new_record?
  # désactivation du formulaire
  page.form.disable 'form-resolutions'

  page.affichage_erreurs @resolution
  page.visual_effect :Appear, 'erreur'

  # disparition des messages d'erreur
  page.delay(4) do
    page.visual_effect :Fade, 'erreur'
    page.form.enable 'form-resolutions'
   end
else
  page.insert_html :bottom, 'resolutions', :partial => 'resolution'
  page.visual_effect :Pulsate, "resolution-#{@resolution.id}", :duration => 1
  # Réinitialisation du champ de saisie
  page.form.reset 'form-resolutions'
end
```

Et si maintenant vous essayez de soumettre le formulaire sans renseigner le champ " Titre " un message d'erreur sera affiché durant quelques secondes, en plus nous avons même la désactivation du formulaire et la réinitialisation du champ de saisie !

Mise en place de l'indicateur global d'activité AJAX

Un des inconvénients lorsque l'on utilise une interface AJAX c'est que la page n'étant pas chargée, on perd les indications visuelles d'activité auxquelles nous étions habitués, comme par exemple avec Firefox le petit icône tournant affiché en haut à droite de la barre de navigation
Qu'à cela ne tienne, nous allons mettre en place notre propre indicateur d'activité, et qui plus est, grâce aux fonctionnalités offertes par la bibliothèque Prototype, il sera global et se déclenchera à chaque exécution d'une requête AJAX.
Ce sera la seule fois dans cet article où nous écrirons nous-mêmes du JavaScript, mais vous remarquerez que j'ai insidieusement attendu la fin pour vous le proposer !
Ajoutez ce code dans public/javascripts/application.js :

```
Ajax.Responders.register({
        onCreate: function() {
                if(Ajax.activeRequestCount > 0)
                $('spinner').show();
        },
        onComplete: function() {
                if(Ajax.activeRequestCount == 0)
                $('spinner').hide();
        }
});
```

Désormais, une image animée sera affichée en bas de la page à chaque exécution d'une requête AJAX..

Richard Piacentini
Créateur du portail Railsfrance.org et fondateur de la SSLL Nuxos Group, spécialisée dans la formation et le développement avec Ruby on Rails

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

Rédaction interactive
www.programmez.com

Cake : PHP aussi prend le train !

Tout développeur web qui se respecte aura au moins une fois entendu parler de Ruby On Rails (RoR). Pourtant, très peu connaissent vraiment ce framework, dont le problème majeur est de reposer sur Ruby. Ce langage, plutôt performant et assez intéressant techniquement, reste très méconnu. CakePHP est un portage du framework Rails en PHP, ce qui va permettre au plus grand nombre de découvrir les joies du MVC en environnement web.

Un framework MVC force les différents intervenants à se plier à ses contraintes. Comme tout le monde suit les mêmes règles, il est bien plus aisé de s'y retrouver dans l'ensemble du projet. Mais bien entendu, un environnement tel que Cake ne se limite pas à un ensemble de bonnes pratiques. Il met en outre de nombreux composants à la disposition du développeur pour lui faciliter la tâche, que ce soit en terme d'accès aux données (grâce à une implémentation du pattern Active Records) ou bien en terme de génération d'interfaces (avec l'aide des " layout " et autres " helper ").

Cet article vous permettra de vous familiariser avec Cake, d'abord en procédant à son installation, puis en détaillant les mécanismes qui sous-tendent les trois volets du framework cités précédemment : les modèles, les vues et les contrôleurs.

Installation

Installer le framework CakePHP est plutôt simple. L'ensemble des sources et librairies indispensables au bon fonctionnement de l'environnement est fourni dans une unique archive, disponible en téléchargement sur le site officiel www.cakephp.org.

Téléchargez la dernière version stable de Cake, et décompressez l'archive dans votre répertoire de travail (peu importe que ce dossier soit ou non dans l'espace web, car nous modifierons la racine des documents du serveur ultérieurement). L'archive contient un dossier de premier niveau nommé cake_1.1.XX.XXXX (XX.XXXX représentant le numéro de version, régulièrement mise à jour). Au long de cet article, nous considérerons tous les chemins concernant le framework comme relatif à ce dossier, par exemple : /app désignera [votre système]/cake_1.1.XX.XXXX/app.

1. Le serveur Web

La configuration n'est pas compliquée, notamment si vous disposez d'une version standard d'Apache déjà installée sur votre système, pour lequel vous n'aurez alors que très peu de modifications à effectuer : vous devez vous assurer que le module " Rewrite " est présent et chargé (et que les directives des fichiers .htaccess sont autorisées). Pour cela, trouvez votre fichier httpd.conf, et cherchez la ligne suivante :

```
LoadModule rewrite_module modules/mod_rewrite.so
```

Si cette ligne est précédée d'un signe #, supprimez-le. Si elle n'existe pas, ajoutez-la (dans la section " Main server configuration ", en haut du fichier).

Pour autoriser Cake à utiliser le mod_rewrite, cherchez la ligne suivante, toujours dans httpd.conf :

```
<Directory "/chemin/vers/votre/documentroot">
```

Une section incontournable du site de Cake : la boulangerie...

Si en-dessous de cette ligne (et au-dessus <de /Directory>) vous trouvez la ligne :

```
AllowOverride None
```

Modifiez-la pour :

```
AllowOverride All
```

Naturellement, si la première n'était pas présente, ajoutez directement la seconde.

La dernière chose à voir avec Apache est le chemin du Document Root (racine des documents). Le mieux, même si cela n'est pas obligatoire, consiste à le faire pointer directement sur la racine de votre application Cake (pensez à modifier en conséquence le chemin de la section Directory traitée ci-dessus si vous modifier votre DocumentRoot) :

```
DocumentRoot " [chemin vers le dossier de Cake]/app/webroot "
```

Voila, c'en est fini de la configuration d'Apache. Il ne vous reste plus qu'à le redémarrer. Notez simplement que si vous avez d'autres documents sur votre serveur, ils ne seront plus accessibles après ces modifications. Pour pallier à ce désagrément, il vous faut créer un hôte virtuel. Reportez-vous sur http://httpd.apache.org/docs/2.0/vhosts/ pour plus d'informations sur le sujet si vous ne maîtrisez pas cet aspect de la configuration d'Apache.

2. La base de données

Comme pour la plupart des applications PHP, tout ce qu'il y a à faire du côté de votre base (nous considérerons dans cet article que vous utilisez l'inévitable MySQL), c'est créer une base (vide) et disposer d'un utilisa-

teurs ayant tout pouvoir sur cette base. Nous verrons dans la seconde partie comment créer les tables qui nous seront nécessaires avec Cake.

3. Cake

La configuration de Cake est parfaitement triviale. En fait, la seule obligation en la matière consiste à indiquer au framework comment il peut accéder aux données. Cela se fait grâce au fichier /app/config.database.php, qu'il vous faudra créer en renommant le fichier database.php.default, fourni avec l'archive, situé dans le même dossier. Ce fichier contient une classe nommée DATABASE_CONFIG, à laquelle deux propriétés sont affectées. Chacune représente un accès à la base. Nous ne nous intéresserons qu'à l'entrée default. Sachez simplement qu'il est possible de spécifier pour chaque modèle la base de données à laquelle accéder, mais qu'en l'absence de configuration spécifique, tous se connecteront à la base décrite dans l'entrée default.
Parmi les champs déjà présents, dans la plupart des cas, seuls trois sont à modifier pour adapter la configuration à votre système : user, password et database. Inutile de s'étendre sur les valeurs attendues...
Enregistrez ce fichier, votre framework est pleinement fonctionnel !

Les concepts

1. Les modèles

Les modèles, le M de MVC, représentent les objets " métiers ". Cela peut être une facture ou un client dans une application de gestion, ou bien un produit, une catégorie ou encore une page dans un site web. Pour faire simple, les modèles sont les classes nécessaires à la manipulation des données stockées dans la base. Ils représentent, sous Cake, une implémentation du design pattern " Active Record ".
Tous les modèles d'une application Cake doivent dériver d'un modèle maître, une classe nommée AppModel. Cette classe mettra à disposition de vos objets de nombreuses méthodes basiques qui vous feront gagner un temps considérable dans l'écriture de votre application, en vous laissant vous concentrer sur le code spécifique à l'application plutôt que sur les tâches mineures, mais néanmoins indispensables, voire critiques, de gestion brute de données.
De ce fait, beaucoup de modèles Cake, au sein d'une application, sont de simples classes étendues vides. Seuls les objets les plus spécifiques nécessitent l'ajout de fonctions propres à leur traitement.
Par défaut, Cake attend qu'à chaque modèle corresponde une table dans la base, dont les noms doivent correspondre. Bien qu'il soit possible de forcer dans la plupart des cas les conventions de nommage du framework, nous vous conseillons de n'en rien faire, car elles sont suffisamment rigides, mais bien conçues, pour faciliter grandement la collaboration de plusieurs développeurs.

Ces conventions, en quelques lignes, imposent ceci :

- tous les noms de modèles et contrôleurs doivent être en CamelCasing (une majuscule au début de chaque mot composant l'expression, sans espace ni underscore)
- un modèle doit toujours être au singulier
- un contrôleur doit toujours être le pluriel du modèle qu'il contrôle
- les tables de la base doivent être le pluriel du modèle dont elles stockent les données, mais tout en minuscules
- les noms de fichiers et dossiers sont toujours en minuscules, avec un espace pour matérialiser les éventuels espaces entre les mots qui composent l'expression

Cela paraît à la fois compliqué à observer de manière systématique, et en même temps, ne pas forcément être très utile. Pourtant, on s'aperçoit à l'utilisation que l'on se fait très vite à ces usages, et qu'ils facilitent vraiment son " orientation " au sein de l'arborescence du projet (qui est elle-même très contrôlée).
Exemple d'un modèle destiné à gérer une table " utilisateurs " :

/app/models/utilisateur.php :

```
class Utilisateur extends AppModel {

}
```

Comme nous l'avons évoqué plus haut, Cake implémente un mécanisme d'accès aux données conforme au pattern " Active Records " (voir article dans le numéro 94). La conséquence directe pour le développeur, c'est de n'avoir pas à écrire les méthodes nécessaires à la lecture et l'écriture de ces données. C'est au niveau du modèle que ces méthodes sont rattachées : create(), update(), find*(), entre autres, sont disponibles immédiatement. Vous avez cependant toute liberté pour les redéfinir si le besoin s'en faisait sentir.
Mais ce cas de figure est plutôt rare, car les modèles du framework vous permettent également de définir des " callbacks " sur ces opérations. Il s'agit de méthodes qui seront automatiquement déclenchées par certains évènements. Par exemple, vous pouvez définir une méthode beforeSave(), qui sera donc exécutée à chaque appel de la méthode immédiatement après l'appel (mais avant l'exécution) des méthodes create() et update(). Ainsi, vous pouvez manipuler vos données avant de les enregistrer grâce aux méthodes Active Records natives.
Exemple d'encodage d'un mot de passe lors de l'ajout d'un utilisateur :

/app/models/utilisateur.php :

```
class Utilisateur extends AppModel {

   public function beforeSave()
   {
        // les données, relues dans la base ou postées
        // depuis un formulaire sont dans $this->data
        $this->data['Utilisateur']['password'] = md5($this->data['Utilisateur']['password']);

   // valeur de retour obligatoire pour poursuivre l'action create() ou update() du modèle :
   return true;

   // return false aurait pour effet d'annuler l'enregistrement des données

   }

}
```

Vulgairement, ce mécanisme peut s'apparenter aux triggers d'une base de données, mais permet de regrouper tout le code au sein du projet PHP, au sacrifice, certes, des performances globales, mais dans une proportion quasi négligeable.

Enfin, les modèles peuvent être associées les uns aux autres, afin de reproduire les liens existant au sein de la base de données entre les tables. Cet aspect du framework est essentiel, mais en même temps trop complexe pour que l'on puisse rentrer dans le détail. Voyons cependant un exemple. Nous allons d'abord créer un modèle Groupe, que nous lierons à nos utilisateurs :

```
<?php
class Groupe extends AppModel {
   public $hasMany = array('Utilisateurs' => array('className' => 'Utilisateur'));
}
?>
```

La propriété $hasMany est un tableau de toutes les relations de type 1 pour n de notre objet. Une relation est définie par un index (ici, 'Utilisateurs') et un tableau d'options, dont seule className est obligatoire dans toutes les relations simples. Si vous respectez les conventions de Cake, il n'y a rien d'autre à spécifier. Vous devez simplement ajouter un champ groupe_id à votre table utilisateurs pour que les relations soient opérationnelles.
Pour la réciproque, il suffit d'ajouter :

```
public $belongsTo = array('Groupe' => array('className' => 'Groupe'));
```

à votre modèle Utilisateur. Les autres options permettent d'affiner précisément les relations entre objets, en définissant par exemple des filtres, ou encore en rendant des objets dépendants d'autres, de sorte qu'il devient possible de les supprimer en cascade d'une seule ligne de code.

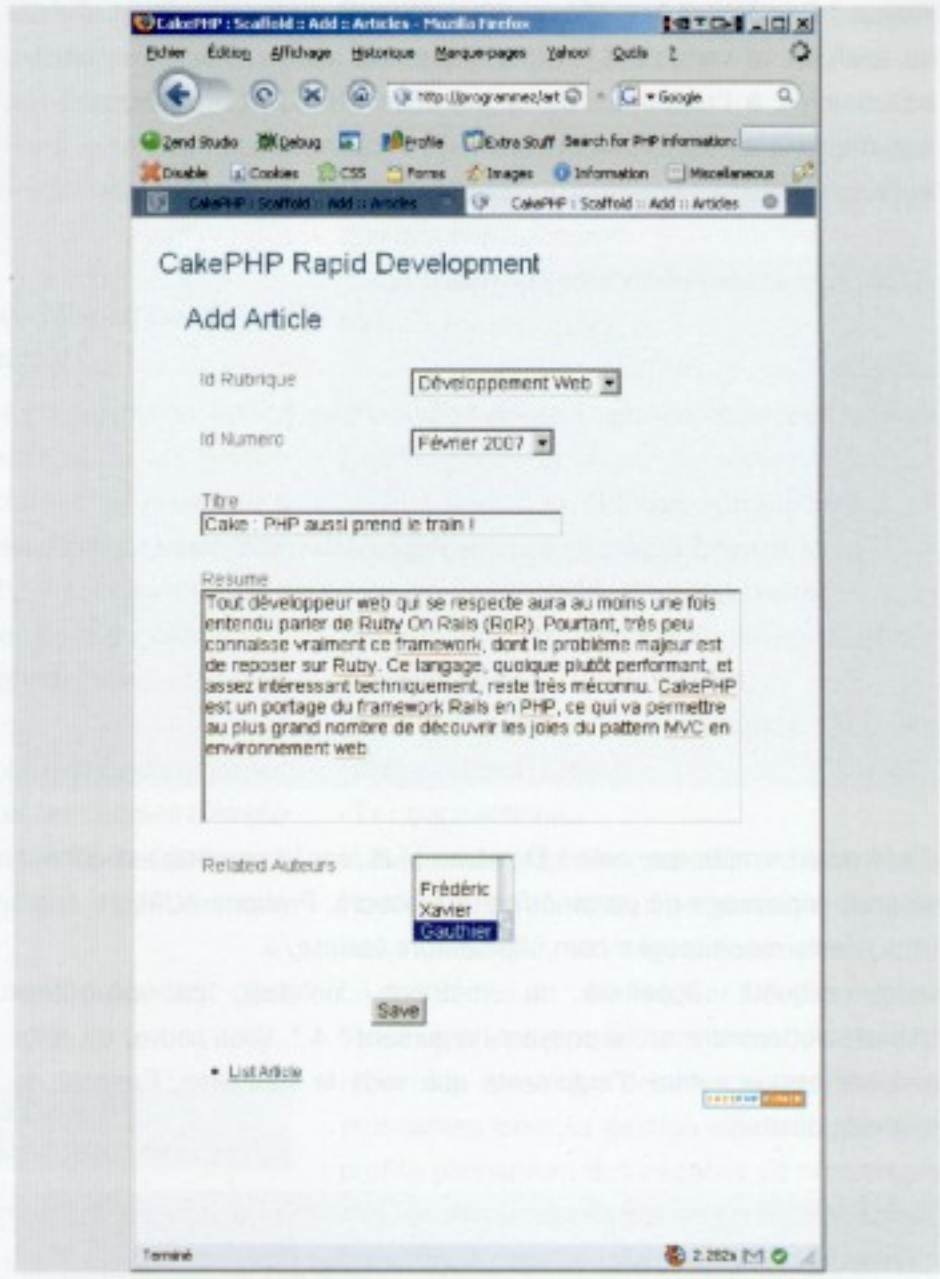
Un site en production sous CakePHP.

2. Les vues

Les vues, comme leur nom le laisse entendre, sont la face visible de l'application (dans le cas d'un site, on pourrait les assimiler aux pages). Dans un environnement CakePHP, elles sont matérialisées par des fichiers " .thtml ", qui ne sont ni plus ni moins que des fichiers PHP. Ce qu'il faut absolument retenir des vues, c'est qu'elles ne doivent en aucun cas intervenir sur les données, bien qu'elles en aient la possibilité technique. Leur rôle consiste exclusivement à afficher des données préalablement déterminées par un contrôleur, et proposer à l'utilisateur l'interface nécessaire à l'appel des contrôleurs en question.
Exemple de vue de debuggage :

```
/app/views/utilisateurs/dump.thtml

<?php
   // pr(array $var) est un raccourci pour print_r() encadré par les balises
<pre> et </pre>
   // $user est une variable affectée par la méthode dump() contrôleur
 UtilisateursController
   pr($user);
?>
```

Pour des facilités de mise en page, les vues peuvent être associées à des " layout ". Ces derniers sont une sorte de " template ", assez restreints cependant, puisqu'ils ne prévoient que deux champs variables : $title_for_layout et $content_for_layout. Toutefois, ils permettent de définir un affichage commun à un ensemble de vues, y compris en embarquant des portions variables : les " elements ".



Les "elements" fonctionnent globalement comme des vues, mais ne sont pas liés à un modèle précis. Les elements sont très pratiques pour afficher les parties d'une interface que l'on retrouve sur de nombreuses vues, comme un menu, typiquement.
Voyons dans le chapitre suivant, traitant des contrôleurs, comment ces différents composants s'imbriquent au sein de la plate-forme.

3. Les contrôleurs

Chefs d'orchestre de l'application, les contrôleurs font la liaison entre les modèles et les vues. Ils sont sollicités depuis l'interface (les vues) pour agir sur les données (via les modèles).
Pour bien comprendre l'utilisation des contrôleurs, il nous faut revenir sur un point déjà abordé dans la première partie de cet article : le mod_rewrite d'Apache. Nous avons expliqué comment l'activer, mais pas pourquoi il était nécessaire.
CakePHP se base sur les URL transmises au serveur pour déterminer les actions qu'il doit effectuer. Le mod_rewrite permet de faire des requêtes sur des URL qui n'existent pas, puis de les réécrire à la volée pour récupérer les arguments passés dans la requête. Un exemple valant parfois mieux qu'un long discours, examinons directement un cas concret. Prenons la requête suivante :
http://www.monsitecake.com/utilisateurs/creer
Cette adresse sera transcrite par le mod_rewrite en :
http://www.monsitecake.com/index.php?url=/utilisateurs/creer
La variable $_GET['url'] ainsi passée à index.php (qui est en réalité le seul script PHP présent dans l'espace web, tous les autres étant ensuite inclus par la méthode " require "). Ce script va ensuite instancier la classe " Dispatcher ", et invoquer sa méthode dispatch(). C'est elle qui va analyser la variable $_GET['url'] pour en extraire les informations nécessaires à l'exécution de la méthode attendue, en séparant les arguments d'après le caractère " / ". Dans notre exemple sera invoquée la méthode " creer() " du contrôleur " UtilisateursController " :

```
/app/controllers/utilisateurs_controller.php

<?php
class UtilisateursController extends AppController {

   public function creer() {
        // layout à appliquer à la vue /app/views/utilisateurs/creer.thtml
        $this->layout = 'public';
   }

}
?>
```

C'est aussi simple que cela ! Et même plus, car le procédé est conservé pour le passage de paramètres utilisateurs. Prenons l'URL :
http://www.monsitecake.com/utilisateurs/delete/4
Cette requête appellera la méthode delete() du contrôleur UtilisateursController en lui passant l'argument " 4 ". Vous pouvez de cette manière passer autant d'arguments que vous le souhaitez. Exemple de méthode delete() :

```
<?php
class UtilisateursController extends AppController {

   public function delete($user_id) {
        $this->autoRender = false;
        // test de l'existence de l'utilisateur
        $user = $this->Utilisateur->findById($user_id);
        if(!$user)
        {
             // l'utilisateur n'existe pas : message d'erreur
             $this->flash('L\'utilisateur n'existe pas', '/',5);
        }
        else
        {
             // il existe on peut bel et bien le supprimer
             $this->Utilisateur->delete($user_id);

             // message de confirmation
             $this->flash('Utilisateur supprimé','/',3);
        }
   }

}
?>
```

Cet exemple nous apprend deux choses très importantes quant à l'utilisation de Cake : d'une part, chaque contrôleur instancie automatiquement le modèle auquel il est associé. Ici, on retrouve cette instance dans $this->Utilisateur. Ce qui nous permet d'accéder directement à toutes les méthodes associées. Ensuite, on peut constater qu'une méthode de contrôleur n'est pas obligatoirement associée à une vue. Dans notre exemple, quelle que soit l'issue de notre requête (erreur ou succès), on affiche un message, puis on retourne à la page d'accueil (paramètre '/' de la méthode flash()). La méthode flash() est disponible dans tous les contrôleurs, et utilise une vue standard qu'il est possible de personnaliser. La propriété autoRender nous permet de spécifier explicitement qu'aucune vue ne doit être rendue à l'issue de l'exécution de la méthode. Ceci est particulièrement pratique dans le cas de méthodes appelées en Ajax qui ne renvoient rien au navigateur. Une petite astuce : au tout début de votre projet, ne vous privez pas d'ajouter un scaffold (littéralement, échaffaudage) à vos contrôleurs. Cela permet de disposer d'interfaces d'ajout, édition, suppression et listage des objets, tenant compte des relations, à l'aide de la simple ligne suivante :

```
public $scaffold = true;

[scaffold.tif : tout ça en une ligne de code !]
```

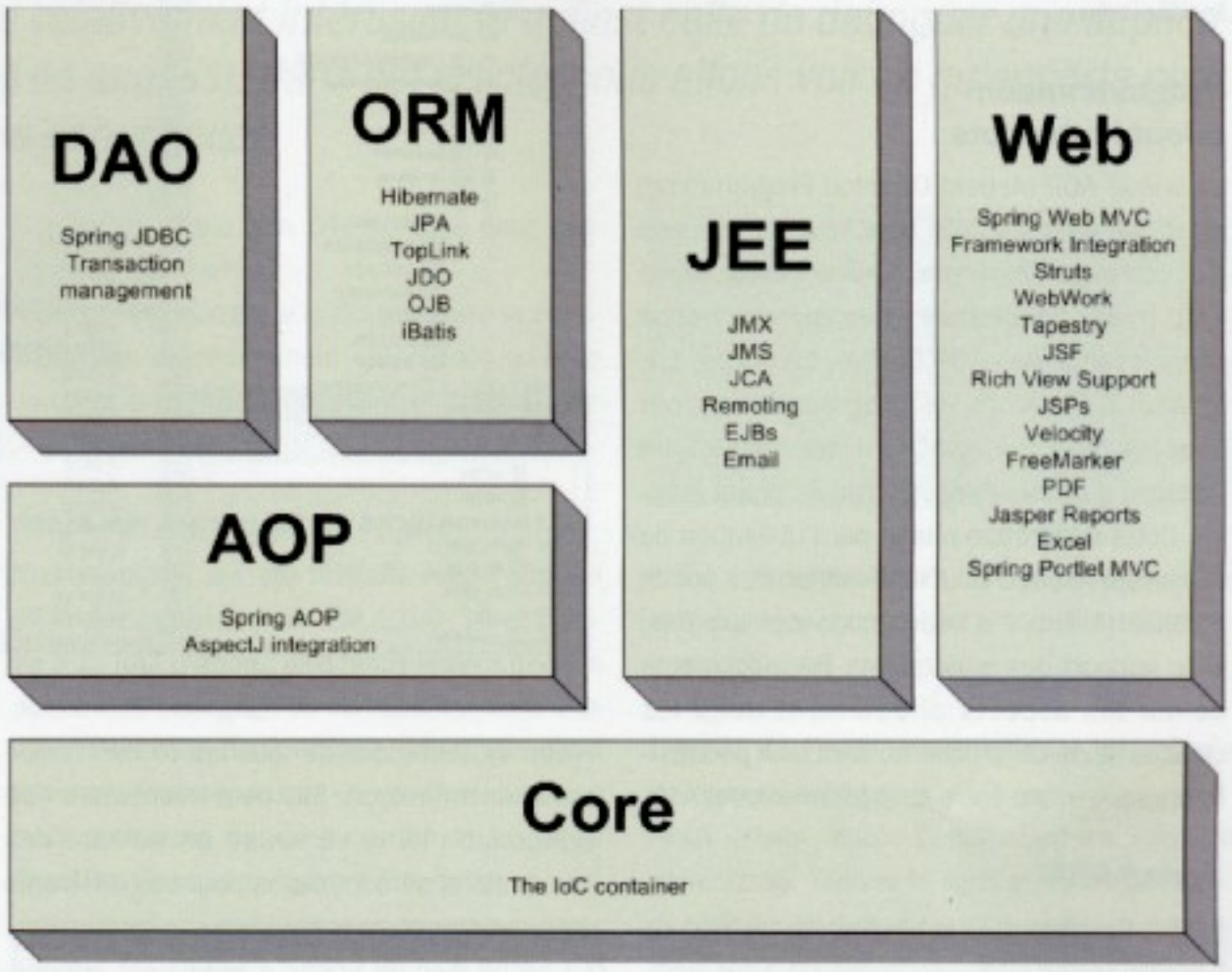
Conclusion

Cette brève découverte de CakePHP et des joies du pattern MVC vous auront malgré tout permis d'en comprendre les grands principes, qui sont vraiment séduisants. Ce qu'un tel pattern ôte de souplesse à votre projet, il vous le rend au centuple en efficacité d'organisation et en fiabilité. La documentation officielle, ainsi que les nombreux sites dédiés, vous permettront, au travers de tutoriels pertinents et concrets, de faire vos premiers pas avec ce framework dont beaucoup vous diront, après y avoir goûté, qu'il n'est plus possible de s'en passer.

■ **Gauthier Delamarre**

Les nouveautés de Spring 2.0

Spring s'est imposé comme LE conteneur léger pour les applications d'entreprise de la sphère Java. Ayant participé au succès de nombreux projets, il est même embarqué officiellement dans certains serveurs d'applications comme Weblogic. Spring 2.0, sorti en octobre 2006, confirme la bonne santé du projet, avec une consolidation du socle et un ensemble de nouveautés très attendues.



Spring reste fidèle à son comportement non-intrusif : les bibliothèques Spring 2.0 restent parfaitement compatibles avec la syntaxe utilisée dans les précédentes versions de Spring. Par ailleurs, Spring 2.0 est compatible avec la plupart des versions de Java (JDK 1.3, 1.4 et 1.5).

Pour définir Spring, on parle souvent de conteneur léger ou de framework Java. L'un et l'autre sont un peu réducteurs, car Spring est à la fois un conteneur léger et un ensemble de classes " utilitaires " pour le développement d'applications d'entreprise. La force de Spring vient du fait que ses 2 composantes forment un tout cohérent, et cela depuis la première version du projet. L'utilisation de Spring encourage certaines pratiques (développement en couches, programmation par interfaces). D'autres pratiques, comme la testabilité des applications, sont facilitées par Spring. On parle souvent de " philosophie " Spring. Cette démarche a été conservée dans la version 2.0, avec de substantielles améliorations, tant sur la partie "conteneur léger" que sur la partie "outils".

Espaces de nommage

Ils constituent une des nouveautés les plus attendues. Ils permettent d'alléger grandement la configuration xml. En effet, dans Spring 1.x, la déclaration des beans se faisait comme suit :

```
<bean id="dataSource" class=" org.springframework.jndi.JndiObjectFactoryBean">
<property name="jndiName" value="jdbc/myDataBase"/>
<property name="resourceRef" value="true"/>
</bean>
```

Déclaration d'une datasource dans la configuration Spring

Toutes les briques de la configuration Spring utilisaient les mêmes balises : bean, property... Pourtant, Spring permet de répondre à des besoins extrêmement variés : configuration Web avec Spring-MVC, configuration des transactions, injection de dépendances...
Cette généricité des balises avait plusieurs conséquences :
- Une configuration xml bien plus volumineuse
- La nécessité de spécifier les classes d'implémentations des concepts utilisés (comme la classe JndiObjectFactoryBean dans l'exemple ci-dessus).

Avec Spring 2.0, l'exemple précédent peut s'écrire comme suit :

```
<jee:jndi-lookup id="dataSource"
jndiName=" jdbc/myDataBase" resourceRef="true"/>
```

La nouvelle syntaxe gagne donc beaucoup en lisibilité. Les espaces de nommage doivent être déclarés en tête de fichier comme dans l'exemple suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation=" http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans-2.0.xsd">
...
</beans>
```

De nombreux espaces de nommage sont disponibles, dont :
- Jee : configuration des éléments J2EE/JEE (datasources, EJB...)
- Tx : transactions.
- Util : fichiers properties, collections d'éléments...

Pour d'autres besoins, il est possible de créer des espaces de nommage personnalisés.
La partie dite de " Middle Tier " de Spring, qui inclut un support pour des frameworks comme Hibernate ainsi que des fonctionnalités très puissantes pour la gestion des transactions, profite pleinement des espaces de nommages XML et voit sa configuration fortement simplifiée. A noter aussi l'ajout du support pour JPA

(Java Persistance API), avec un gestionnaire de transaction dédié et des classes de base pour faciliter l'accès aux données avec cette technologie.

Programmation orientée aspects

La brique AOP (Aspect Oriented Programming) n'est pas en reste : elle bénéficie d'un espace de nommage très fourni pour la configuration XML (mais demandant du coup une bonne connaissance de l'AOP Spring). En Spring 1.x, le seul framework de programmation par aspects était Spring-AOP. Il est maintenant possible d'utiliser AspectJ, issu du projet Eclipse. Cette intégration passe par l'utilisation de la syntaxe AspectJ pour la définition des points d'action (" lieux " d'intervention des aspects) et le support des annotations @AspectJ, pour définir les aspects directement dans les classes Java. On a donc au final une portabilité presque totale entre ces 2 frameworks AOP.

Spring-MVC

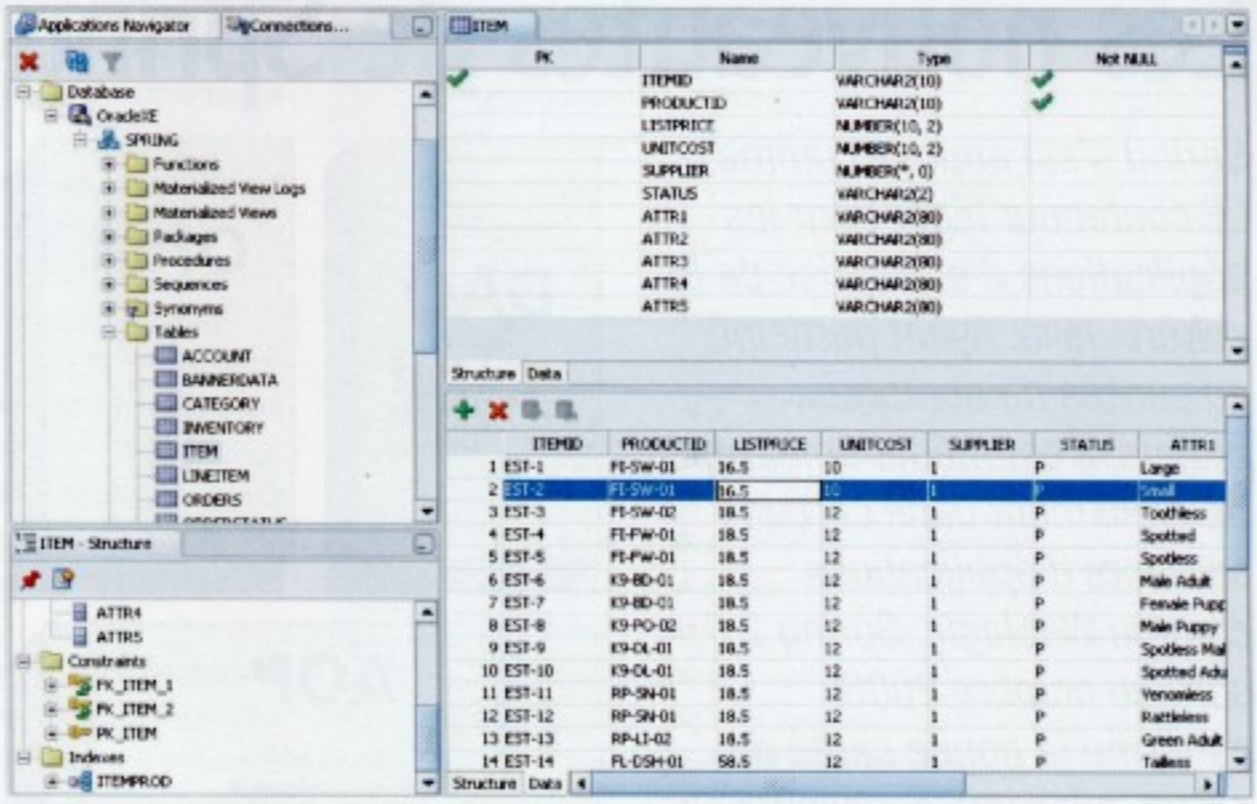
Pour les projets utilisant le framework Web de Spring, Spring-MVC proposait déjà de nombreux avantages : une hiérarchie de contrôleurs bien pensée, une validation des formulaires non-intrusive et une intégration parfaite avec les autres concepts de Spring (ex: injection de dépendance). Spring-MVC accusait cependant un retard en matière de taglibs. En effet, contrairement aux autres frameworks Web de la sphère Java (Struts, implémentations JSF...), Spring-MVC ne proposait presque pas de tags pour les pages JSP. En conséquence, les projets utilisant Spring-MVC devaient utiliser des scripts Java dans les pages JSP, ou développer leurs propres tags. Spring 2.0 introduit la bibliothèque form qui va permettre de combler le retard de Spring-MVC sur le sujet (1).

```
<form:form action="validerClient.sqli" command
Name="clientForm" >
<form:input  path="nom" />
<form:input  path="prenom" />
<input type="submit" value="Virer">
</form:form>
```

exemple de page JSP utilisant les nouveaux tags form

La configuration de Spring MVC a été aussi simplifiée : auparavant, elle demandait la définition systématique de tous les éléments (formulaire, contrôleur). Elle peut maintenant être beaucoup moins verbeuse en suivant des conventions de nommages pour ses différents éléments.

La partie Web de Spring a également été enrichie d'un nouveau framework pour la gestion de portlet, qui emprunte de nombreux concepts à Spring MVC.



Nouvelles portées de bean

Lors de la déclaration d'un bean, Spring permet de paramétrer la portée du bean. En Spring 1.x, seules 2 portées étaient proposées :

- singleton : si une seule instance du bean doit être partagée par l'ensemble des accesseurs.
- Prototype : chaque accesseur récupère sa propre instance du bean.

En Spring 2.0, les contextes les plus courants du monde Web ont été rajoutés pour les applications Web dites "classiques", les contextes de requête et session sont gérés par Spring. Pour les portails d'entreprise, il est également possible d'utiliser le contexte globalSession. Spring 2.0 peut donc "injecter" un bean dans l'un de ces contextes.

De plus, le conteneur gère les relations entre beans de portées différentes, comme l'injection d'un bean de portée "session" dans un singleton.

La documentation

Il serait dommage de ne pas aborder le sujet de la documentation de Spring. Déjà très conséquente pour la version 1, elle s'est encore améliorée dans la version 2. Elle est disponible en différents formats (HTML ou PDF, près de 500 pages !) mais seulement en Anglais. Cette documentation ne se contente pas de traiter des composants de Spring mais propose des informations, des bonnes pratiques sur le développement d'applications JEE et l'utilisation des divers frameworks pour lesquels Spring propose un support (Hibernate, Struts...). On peut dire que la documentation de Spring est un projet à elle seule !

La publication des nouvelles spécifications JEE 1.5 avait engendré quelques inquiétudes dans le monde Spring. En effet, JEE s'étant fortement inspiré des travaux de Spring (injection de dépendances, déclaration "légère" des transactions...), quelques questions se posaient concernant la place de Spring dans les applications Java de demain. Rod Johnson et son équipe ont su montrer leur réactivité en proposant une nouvelle version majeure de Spring, riche en nouvelles fonctionnalités et toujours en avance sur son temps.

Michael ISVY

Groupe SQLI

Arnaud COGOLUEGNES

(1) Cependant, les bibliothèques proposées par Spring-MVC restent moins riches que celles de Struts ou des JSF. Notamment, toute la génération html hors-formulaire n'est pas prise en compte par Spring-MVC (tags de génération des liens html, images...)

Déboguer à distance avec Eclipse

Eclipse propose une fonctionnalité relativement intéressante qui est celle de deboguer une application à distance (remote debugging en anglais). Dans cet article, nous allons voir ce qu'apporte cette technique par rapport au classique debug local.

Le remote debugging consiste à se connecter à une autre machine virtuelle que celle sur laquelle se trouve notre IDE afin d'y déboguer les applications qui s'y exécutent. Les deux machines virtuelles peuvent se trouver sur le même ordinateur mais également sur deux ordinateurs différents du réseau (typiquement une sur l'ordinateur du développeur et une sur le serveur hébergeant les applications).

Cette technique, avec Eclipse, est relativement simple pour le développeur, car elle ne diffère guère d'un debug local.

Il existe deux types de remote debugging :

- homogène : toutes les machines possèdent le même système d'exploitation.
- hétérogène : les différentes machines ne possèdent pas le même système d'exploitation.

Pourquoi déboguer à distance?

Il peut y avoir des différences entre l'environnement de développement et l'environnement de production où sera déployée l'application. Une application qui s'exécutait parfaitement sur le poste du développeur lors de sa conception peut très bien présenter des dysfonctionnements, une fois déployée sur le serveur de production. Il est donc très intéressant de pouvoir la deboguer une fois qu'elle a été déployée.

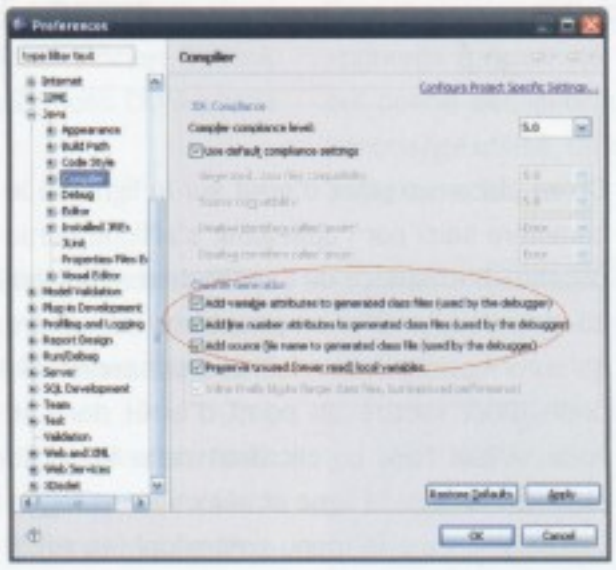
Principe et mise en place

Le principe du remote debugging est une communication par socket. La machine virtuelle distante ouvre un socket et le débogueur va se connecter dessus afin de communiquer avec elle et intercepter les points d'arrêts de l'application à déboguer. Pour mettre en place cette fonction, il faut suivre les étapes suivantes:

- Compiler l'application avec les options de débogage
- Déployer l'application.
- Configurer le fichier de lancement du serveur d'applications afin qu'il se lance avec le débogage à distance activé ou appeler l'application avec les informations de débogage à distance nécessaires si l'application ne s'exécute pas sur un serveur d'applications.
- Créer une configuration d'attache du debogueur d'Eclipse à la JVM distante pour l'application distante.
- Lancer le debogueur d'Eclipse avec la configuration précédemment créée pour qu'il se connecte à l'application sur la machine virtuelle distante.

Mise en œuvre et exemple

Nos exemples ont été réalisés avec Windows XP Professional SP2, le JDK 1.5.0_10 et Eclipse 3.2. Tout d'abord, il va falloir vérifier que les options de debogage du compilateur sont activées. Pour cela, dans Eclipse, il faut aller dans Window->Preferences...->Java->Compiler. Les trois premières cases à cocher de la partie Classfile Generation doivent être cochées.



Mise en œuvre et exemple sans serveur d'applications

Le projet de l'exemple

On crée un projet tout simple, HelloRemote, contenant une classe Hello.java qui va juste afficher Hello World, puis la valeur d'une variable et enfin un caractère saisi au clavier par l'utilisateur :

```
import java.io.*;

public class Hello {
   public static void main(String[] args) throws
 IOException{
        System.out.println("Hello World");

        String titi = "test";
        System.out.println(titi);
        char test = (char)System.in.read();
        System.out.println(test);

    }

}
```

Puis, on crée son archive exécutable HelloRemote.jar que l'on place à la racine du lecteur C.

L'appel de l'application avec les arguments du remote debugging

Nous allons lancer l'application en ligne de commande. Tapons la ligne suivante :

```
java -Xdebug -Xrunjdwp:transport=dt_socket,server=y,suspend=n,address=8000 -jar HelloRemote.jar
```

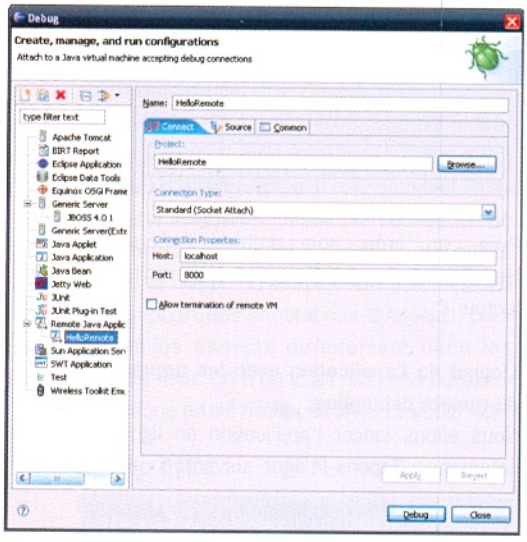
Signification des différents arguments (ces arguments sont propres à la JVM de Sun, ils peuvent être différents pour d'autres machines virtuelles) :

- -Xdebug : démarre la JVM en mode debogage
- -Xrunjdwp : précise comment se connecter pour le debogage à distance
  - transport=dt_socket : précise le type de connexion à la JVM, ici elle se fera par socket
  - server=y : démarrage de la JVM en mode serveur attendant la connexion d'un programme extérieur (ici le débogueur d'Eclipse donc).
  - suspend=y : la JVM se suspend en attendant que le débogueur s'y connecte, elle se relancera une fois celui-ci connecté.
  - address=8000 : spécifie le port où le debogueur pourra se connecter à la JVM

On lance l'application, la JVM passe en attente de la connexion du debogueur. On peut voir qu'elle écoute les connexions sur le port 8000 :

Attacher le debogueur d'Eclipse à la JVM distante

Pour attacher le debogueur d'Eclipse à la JVM et à l'application distante, il faut aller dans le menu Run->Debug puis choisir dans la colonne de gauche Remote Java Application, faire un clic droit dessus et choisir New dans le menu contextuel. Enfin, on choisit un nom pour l'application distante. On arrive alors à l'écran suivant :

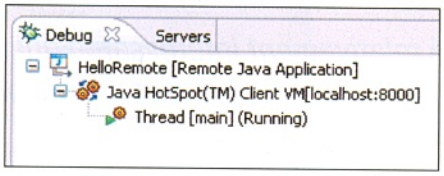


- Name : contient le nom de notre connexion distante (on peut mettre le nom que l'on veut ici)
- Project : spécifie le projet que l'on veut déboguer afin d'avoir accès à son code source.
- Connection Type : précise le type de connexion, ici connexion par socket
- Host : hôte distant auquel on veut se connecter, cela peut être localhost, une adresse IP ou un nom d'hôte.
- Port : spécifie le port sur lequel on se connecte à la JVM distante (il doit être le même que celui que l'on a précisé plus haut).
- Allow termination of remote VM : permet d'arrêter la JVM distante depuis Eclipse.

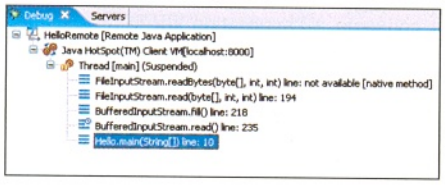
Il faut alors cliquer sur Apply afin de sauvegarder la configuration. Puis, en cliquant sur Debug, on attache le débogueur à la JVM distante. La JVM qui était suspendue (par l'option suspend=y dans la ligne de commande) se relance et l'application se lance. L'application s'exécute donc, elle affiche Hello World et la valeur de la variable. Elle est ici arrivée au niveau où elle attend qu'un utilisateur saisisse un caractère au clavier.

Pour passer de la vue du code à la vue du debogueur dans Eclipse, il faut aller dans Window->Open perspective->Debug.

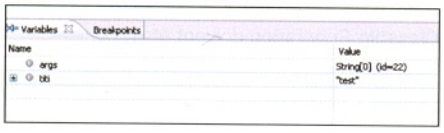
En haut, à gauche, on peut voir que l'application tourne :



En faisant un clic droit sur Thread et en choisissant Suspend dans le menu contextuel, on peut suspendre l'exécution :

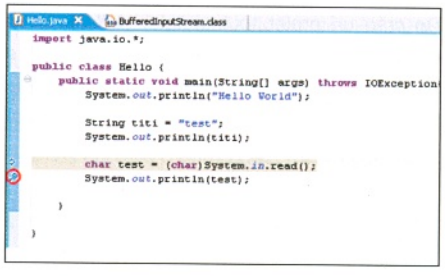


Une fois l'application suspendue, on peut voir dans la partie des variables les valeurs qu'elles ont à ce moment là:



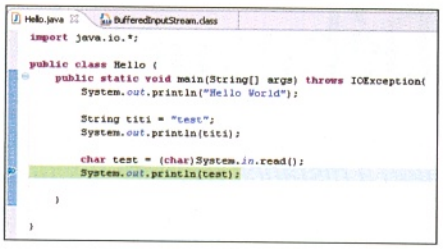
On va placer un point d'arrêt sur la ligne où le caractère saisi par l'utilisateur s'affiche sur la console. L'exécution de l'application s'arrêtera sur ce point d'arrêt et on pourra voir les valeurs qu'auront prises les variables à cet endroit du code. Pour mettre un point d'arrêt dans le code, il faut faire un clic droit dans la partie bleue en face de la ligne et sélectionner Toggle Breakpoint dans le menu contextuel (en re-cliquant sur Toggle Breakpoint, on pourra retirer le point d'arrêt).

Le point d'arrêt est entouré en rouge :

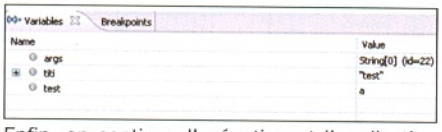


Ensuite, on relance l'application en re-cliquant droit sur Thread dans la partie Debug et en choisissant Resume dans le menu contextuel. On saisit un caractère au clavier.

L'application se suspend de nouveau au niveau du point d'arrêt :



Une fois suspendue par le point d'arrêt, on peut regarder la valeur de ses variables dans la partie Variables :



Enfin, on continue l'exécution et l'application se termine.

Mise en œuvre avec le serveur d'applications JBoss

Le remote debugging d'une application déployée sur le serveur d'applications JBoss diffère peu du débogage d'une application non déployée sur un serveur d'application vu ci-dessus.

Les seules choses à faire seront d'arrêter JBoss, de décommenter la ligne ci-dessous dans le fichier de lancement de JBoss, run.bat situé dans le répertoire <répertoire d'installation de JBoss>\bin puis de relancer JBoss.

L'attachement du debogueur d'Eclipse à la JVM se fera de la même manière que précédemment. Le port de connexion à la JVM à spécifier sera ici le 8787 (celui par défaut dans le fichier run.bat, mais il peut être changé). Une fois le debogueur attaché à la JVM distante par un clic sur Apply dans la fenêtre de configuration d'une connexion à une JVM distante, il faut déployer l'application dans le répertoire de déploiement de JBoss puis, la lancer et pour finir, cliquer sur Debug pour lancer le debogueur et pouvoir la deboguer.

Enfin, l'utilisation du debogueur dans l'application déployée sur JBoss sera rigoureusement identique à précédemment et l'on pourra suspendre l'exécution de l'application et mettre des points d'arrêts où on le souhaite.

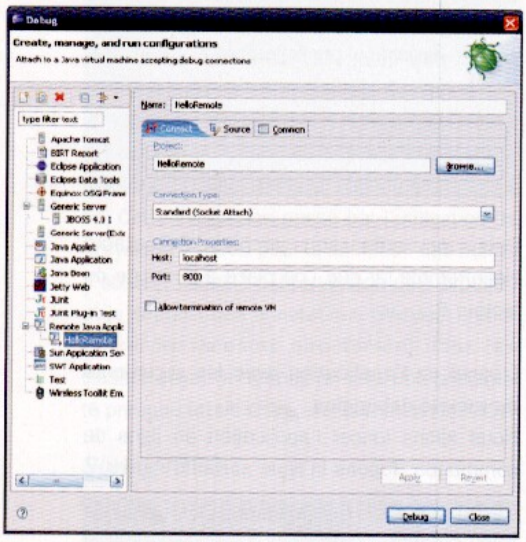
Pour aller plus loin dans la découverte du debogage à distance avec JBoss, voici un excellent tutoriel :

http://www.onjava.com/pub/a/onjava/2005/08/31/eclipse-jboss-remote-debug.html

Alexandre Fillatre
Labo Sun - Supinfo

Attacher le debogueur d'Eclipse à la JVM distante

Pour attacher le debogueur d'Eclipse à la JVM et à l'application distante, il faut aller dans le menu Run->Debug puis choisir dans la colonne de gauche Remote Java Application, faire un clic droit dessus et choisir New dans le menu contextuel. Enfin, on choisit un nom pour l'application distante. On arrive alors à l'écran suivant :

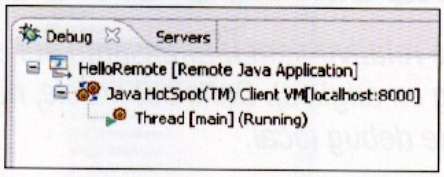


- Name : contient le nom de notre connexion distante (on peut mettre le nom que l'on veut ici)
- Project : spécifie le projet que l'on veut déboguer afin d'avoir accès à son code source.
- Connection Type : précise le type de connexion, ici connexion par socket
- Host : hôte distant auquel on veut se connecter, cela peut être localhost, une adresse IP ou un nom d'hôte.
- Port : spécifie le port sur lequel on se connecte à la JVM distante (il doit être le même que celui que l'on a précisé plus haut).
- Allow termination of remote VM : permet d'arrêter la JVM distante depuis Eclipse.

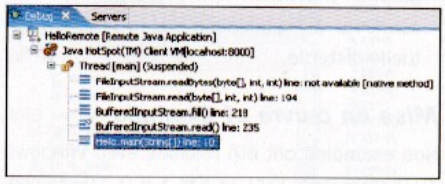
Il faut alors cliquer sur Apply afin de sauvegarder la configuration. Puis, en cliquant sur Debug, on attache le débogueur à la JVM distante. La JVM qui était suspendue (par l'option suspend=y dans la ligne de commande) se relance et l'application se lance. L'application s'exécute donc, elle affiche Hello World et la valeur de la variable. Elle est ici arrivée au niveau où elle attend qu'un utilisateur saisisse un caractère au clavier.

Pour passer de la vue du code à la vue du debogueur dans Eclipse, il faut aller dans Window->Open perspective->Debug.

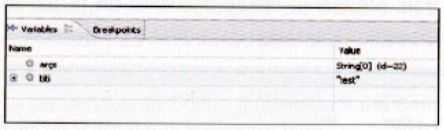
En haut, à gauche, on peut voir que l'application tourne :



En faisant un clic droit sur Thread et en choisissant Suspend dans le menu contextuel, on peut suspendre l'exécution :

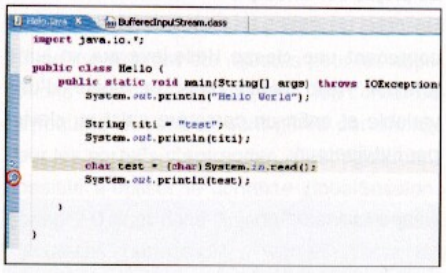


Une fois l'application suspendue, on peut voir dans la partie des variables les valeurs qu'elles ont à ce moment là:



On va placer un point d'arrêt sur la ligne où le caractère saisi par l'utilisateur s'affiche sur la console. L'exécution de l'application s'arrêtera sur ce point d'arrêt et on pourra voir les valeurs qu'auront prises les variables à cet endroit du code. Pour mettre un point d'arrêt dans le code, il faut faire un clic droit dans la partie bleue en face de la ligne et sélectionner Toggle Breakpoint dans le menu contextuel (en re-cliquant sur Toggle Breakpoint, on pourra retirer le point d'arrêt).

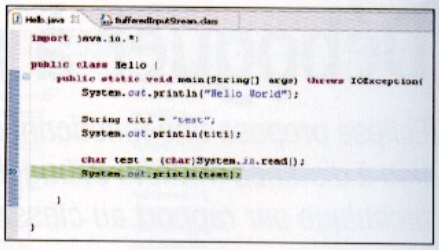
Le point d'arrêt est entouré en rouge :



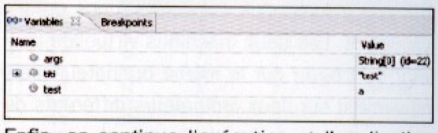
Ensuite, on relance l'application en re-cliquant droit sur Thread dans la partie Debug et en choisissant Resume dans le menu contextuel.

On saisit un caractère au clavier.

L'application se suspend de nouveau au niveau du point d'arrêt :



Une fois suspendue par le point d'arrêt, on peut regarder la valeur de ses variables dans la partie Variables :



Enfin, on continue l'exécution et l'application se termine.

Mise en œuvre avec le serveur d'applications JBoss

Le remote debugging d'une application déployée sur le serveur d'applications JBoss diffère peu du débogage d'une application non déployée sur un serveur d'application vu ci-dessus.

Les seules choses à faire seront d'arrêter JBoss, de décommenter la ligne ci-dessous dans le fichier de lancement de JBoss, run.bat situé dans le répertoire <répertoire d'installation de JBoss>\bin puis de relancer JBoss.

L'attachement du debogueur d'Eclipse à la JVM se fera de la même manière que précédemment. Le port de connexion à la JVM à spécifier sera ici le 8787 (celui par défaut dans le fichier run.bat, mais il peut être changé). Une fois le debogueur attaché à la JVM distante par un clic sur Apply dans la fenêtre de configuration d'une connexion à une JVM distante, il faut déployer l'application dans le répertoire de déploiement de JBoss puis, la lancer et pour finir, cliquer sur Debug pour lancer le debogueur et pouvoir la deboguer.

Enfin, l'utilisation du debogueur dans l'application déployée sur JBoss sera rigoureusement identique à précédemment et l'on pourra suspendre l'exécution de l'application et mettre des points d'arrêts où on le souhaite.

Pour aller plus loin dans la découverte du debogage à distance avec JBoss, voici un excellent tutoriel :
http://www.onjava.com/pub/a/onjava/2005/08/31/eclipse-jboss-remote-debug.html

■ **Alexandre Fillatre**
Labo Sun - Supinfo

LANGAGE : C#
DIFFICULTÉ : ●●○○○
APPLICATION : MOBILE
SOURCE : oui

PocketPC : échanger vos contacts grâce au Compact framework 2.0

L'application réalisée dans cet article utilise le Compact Framework 2.0 (CF2). Ce SDK, tout comme son prédécesseur, s'intègre dans Visual Studio. Seules les versions Visual Studio à partir de la Standard supportent ce SDK, exit donc les versions de Visual Studio Express. Les nouveautés du SDK sont les API managées ouvrant l'accès à PocketOutlook, les API permettant d'accéder à un périphérique GPS, la sérialisation XML.

Les librairies de classes employées sont celles utilisées pour une application WinForm. Seules certaines librairies comme celles liées à PocketOutlook sont spécifiques. Ainsi, le passage d'un développement d'une application Winform à une application CF nécessite peu de modifications. Ces changements peuvent être limités en créant une librairie d'accès aux données et aux API spécifiques. Seules les modifications sur ces librairies seront à faire pour passer d'une plate-forme à l'autre. Une interface graphique développée pour l'une des deux plates-formes devra être refaite pour l'autre.

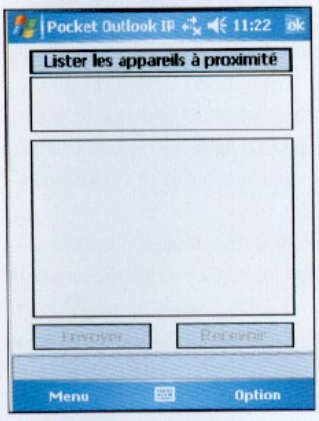
Ce qui va nous intéresser dans cet article, c'est l'interfaçage avec PocketOutlook. Les premières versions du SDK pour PocketPC (Embedded Visual C++ et Visual Basic) permettaient d'accéder à PocketOutlook en utilisant des API COM communément appelées POOM (Pocket Outlook Object Model). Dans le CF2, l'utilisation de ces interfaces COM passe par des API managées.

Application développée

Dans Windows Mobile 5.0, PocketOutlook est un PIM (Professional Instant Messaging) complet. Il stocke ses informations dans des dossiers : les contacts, les rendez-vous, les notes, les tâches mais aussi les SMS et les emails.

A partir de l'interface graphique de PocketOutlook, l'utilisateur peut transmettre un item à la fois à partir de l'un de ces dossiers. L'émission se passe par infrarouge, bluetooth vers un autre terminal mobile (PocketPC, autres PDA, téléphone portable...) ou un PC.

PocketOutlook est limité à l'émission d'un seul item à la fois vers un autre terminal. L'application proposée permet de pallier à cette limitation en laissant la possibilité de transmettre tous vos contacts ou ceux de votre choix vers un autre PocketPC.



Interface graphique

L'interface graphique se présente comme suit (figure ci-contre).

Dans la première partie de l'écran, en cliquant sur le bouton "lister les appareils à proximité", la liste située en-dessous est remplie par le nom des appareils infrarouge détectés. Cette liste est remplie dans la méthode liée au handler du bouton.

En-dessous de cette liste, un ListView affiche la liste des contacts du PocketPC sur lequel est lancée l'application. Le menu Menu/Rafraîchir la liste, met à jour la liste des contacts. Une barre de progression s'affiche pour connaître l'avancement de la mise à jour de la liste. En fin de traitement, cette barre disparaît. En bas de l'écran, une barre de statut affiche les actions effectuées dans l'application, par exemple l'envoi / réception du fichier contenant la liste des contacts.

Le menu Option/Cocher tout, sélectionne tous les contacts, le menu Option/Décocher tout effectue l'opération inverse. La sélection d'un contact consiste à cocher la case à cocher à gauche du contact.

Chercher les périphériques à proximité et initier l'envoi

Pour chercher un périphérique, il faut cliquer sur le bouton : " lister les appareils à proximité ". On remarquera l'utilisation de IrDADeviceInfo qui donne des informations sur le périphérique distant, par exemple l'encodage utilisé, un id unique, le type de matériel quand c'est possible (téléphone, fax, pc).

```
// Recherche et affiche les périphérique infrarouge.
        private void Discover_Click(object sender, System.EventArgs e)
        {
            //Fixe le nombre maximum de périphériques recherchés.
           irdaDeviceInfo = irdaClient.DiscoverDevices(maxDevicesToDiscover);

            // Affiche un message si aucun périphérique n'a été trouvé.
            if (irdaDeviceInfo.Length == 0)
            {
              MessageBox.Show("Aucun appareil infrarouge détecté à proximité."
, "Information", MessageBoxButtons.OK, MessageBoxIcon.Hand, Message
BoxDefaultButton.Button1);
                return;
            }

            lstDevices.Items.Clear();

            //Affiche la liste des périphériques trouvés.
            foreach (IrDADeviceInfo irDevice in irdaDeviceInfo)
            {
                lstDevices.Items.Add(irDevice.DeviceName);
            }

            lstDevices.SelectedIndex = 0;
            if (irdaDeviceInfo.Length > 0)
            {
```

```
            statusBar.Text = irdaDeviceInfo.Length.ToString() + " appareil(s)
distant(s) détectés.";
        }

        // Rend les boutons d'envoi et de réception disponibles
        btnSend.Enabled = true;
        btnReceive.Enabled = true;
    }
```

Envoyer des données via le port infrarouge

Dans notre application, les données envoyées sont un fichier XML contenant les données relatives aux contacts sélectionnés dans l'interface graphique :

```
// Sends a file.
    private void Send_Click(object sender, System.EventArgs e)
    {
        ...

        //Create an IrDA client with a service name that must
        //match the service name of the receiving IrDA client.
        try
        {
            irClient = new IrDAClient(irServiceName);
        }
        catch (SocketException exS)
        {
            MessageBox.Show("Create socket error: " + exS.Message +
" - Did you click Receive on the other device?");
            return;
        }

        // Get the underlying stream of the client.
        Stream baseStream = irClient.GetStream();

        //Get the size of the file to send
        //and write its size to the stream.
        byte[] length = BitConverter.GetBytes((int)fileStream.Length);
        baseStream.Write(length, 0, length.Length);

        //Create buffer for reading the file.
        byte[] buffer = new byte[buffersize];

        // Display the number of bytes being sent.
        int fileLength = (int)fileStream.Length;
        status.Text = "Sending " + fileLength + " bytes";

        // Read the file stream into the base stream.
        while (fileLength > 0)
        {
            int numRead = fileStream.Read(buffer, 0, buffer.Length);
            baseStream.Write(buffer, 0, numRead);
            fileLength -= numRead;
        }

        fileStream.Close();
        baseStream.Close();
        irClient.Close();
        status.Text = "File sent";
    }
```

Structure des données utilisées pour la sérialisation XML

Chaque contact sélectionné pour être envoyé est sérialisé dans un DataSet dédié. L'intérêt de ce DataSet permet de ne pas être dépendant des champs de la structure de la classe Contact de l'API PocketOutlook. Les contacts sont sérialisés puis envoyés sous forme de fichier XML. Dans la solution, c'est le fichier ContactSet.xsd qui définit la structure du DataSet. La méthode ci-dessous sérialise les données dans un fichier XML :

```
/// <summary>
    /// Crée un fichier xml pour l'émission.
    /// </summary>
    /// <param name="filename">Nom du fichier avec le chemin d'accès
complet.</param>
    /// <param name="contacts">Liste de contacts à transmettre.</param>
    static internal void CreateXmlDocument(string filename, List<PimItem>
contacts)
    {
        FileStream fs = new FileStream(filename, FileMode.Create);

        ContactSet contactSet = new ContactSet();

        contacts.ForEach
            (
                delegate (PimItem cd)
                {
                ContactSet.ContactRow contactRow = contactSet.Contact.
NewContactRow();

                    PropertyDescriptor descriptor = TypeDescriptor.Get
Properties (typeof (Contact)).Find ("itemid", true);
                int idx = Session.Contacts.Items.IndexOf (cd);


                PropertyInfo[] properties = Session.Contacts.Items[idx]
.GetType().GetProperties();

                foreach (PropertyInfo pi in properties)
                {
                    if (pi.CanRead && pi.CanWrite)
                    {
                    object ob = pi.GetValue(Session.Contacts.Items[idx], null);
                        if (ob != null)
                        {
                        if (pi.PropertyType.FullName == typeof(DateTime)
.FullName)
                            {
                                DateTime dt = (DateTime)ob;
                                contactRow[pi.Name] = dt.Ticks;
                            }
                    else if (pi.PropertyType.FullName == typeof(Uri).FullName)
                            {
                                Uri uri = (Uri)ob;
```

```
                        contactRow[pi.Name] = uri.AbsoluteUri;
                    }
                    else
                    {
                        contactRow[pi.Name] = ob;
                    }
                }
            }
        }

        contactSet.Contact.AddContactRow(contactRow);
    }
);
contactSet.WriteXml (fs);
fs.Close();
}
```

On notera l'utilisation de la méthode Foreach sur un object List. Cette méthode optimise les itérations sur un objet de type List. Pour le faire fonctionner, il faut définir un delegate qui définit l'action à exécuter pour chaque élément de la liste.

Dans cette partie, la réflexion est utilisée. Ce mécanisme permet de connaître le contenu d'une classe (noms des propriétés, méthodes) afin de lire ou d'écrire des données en y accédant par leur nom. Dans notre exemple, cela permet d'affecter les champs du DataSet (ConstactSet) aux champs de la classe Contact (de la librairie PocketOutook). Cependant, dans ce contexte, il faut que les deux classes aient les mêmes noms de champs. Pour éviter ce lien fort, il faut redéfinir le delegate afin d'affecter de façon unitaire les champs de destination et d'entrée. La "réflection" est à éviter dans le cas où l'application nécessite des temps de réponses rapides.

Le document est ensuite stocké dans le répertoire temp du PocketPC, le nom du fichier émis et reçu est le même ainsi que l'endroit de stockage. Voici ce qui fixe le nom du fichier :

```
Path.Combine(Path.GetTempPath(), "contacts.xml");
```

Sur Windows Mobile 5.0, le répertoire utilisé est : Application Data\Volatile.

Réception de données par le port infrarouge

La réception se déroule en deux phases :

- Création d'une connexion avec le port infrarouge. Celle-ci se mettra en attente de données. La séquence de code suivante lance l'écoute de signaux sur le port infrarouge :

```
int i = lstDevices.SelectedIndex;
irdaEndPoint = new IrDAEndPoint(irdaDeviceInfo[i].DeviceID, irServiceName);
irdaListener = new IrDAListener(irdaEndPoint);
irdaListener.Start();
```

- Réceptionner les données : les données sont réceptionnées en utilisant une technique habituelle avec un buffer que l'on remplit au fur et à mesure qu'elles arrivent.

La réception doit être initiée manuellement en appuyant sur le bouton recevoir. Une fois le fichier reçu, celui-ci est traité par la méthode ComputeContacts :

```
/// <summary>
    /// Insère les contacts contenus dans le fichier xml dans PocketOutlook.
    /// </summary>
    /// <param name="filename">Nom du fichier avec le chemin d'accès
complet.</param>
    internal static void ComputeContacts(string filename)
    {
        ContactSet cs = new ContactSet();
        cs.ReadXml(filename);

        foreach (ContactSet.ContactRow contactRow in cs.Contact)
        {
            Contact contact = Session.Contacts.Items.AddNew();
        PropertyInfo[] propertiesContact = contact.GetType().GetProperties();

            foreach (PropertyInfo pi in propertiesContact)
            {
            if (pi.CanRead && pi.CanWrite && cs.Contact.Columns.Contains
(pi.Name))
                {
                    object ob = contactRow[pi.Name];

                    if (ob != DBNull.Value)
                    {
                if (pi.PropertyType.FullName == typeof(DateTime).FullName)
                        {
                    DateTime dt = new DateTime(long.Parse(ob.ToString()));

                            if (dt != DateTime.MinValue && dt.Year > 1900)
                            {
                                pi.SetValue(contact, dt, null);
                            }
                        }
                else if (pi.PropertyType.FullName == typeof(Uri).FullName)
                        {
                            Uri uri = new Uri(ob.ToString());
                            pi.SetValue(contact, uri, null);
                        }
                        else
                        {
                            pi.SetValue(contact, contactRow[pi.Name], null);
                        }
                    }
                }
            }
            contact.Update();
        }
        cs.Dispose();

    }
```

Utilisation du ListView

L'affichage des contacts utilise le contrôle ListView, ce contrôle permet d'afficher des informations sous différentes formes. Ce contrôle est utilisé dans l'explorateur Windows, l'affichage des programmes installés ou encore pour afficher les icônes des paramètres systèmes. Pour modi-

fier le type d'affichage, il faut modifier la valeur de la propriété View. L'utilisation de ce contrôle dans notre application permet de stocker :
- le label affiché pour représenter un contact (nom et prénom du contact)
- l'objet représentant le contact (une instance de PimItem) que nous décrirons un peu plus loin dans cet article.

Afin d'afficher une liste d'information, le ListView contient une liste d'objets de type ListViewItem. Dans l'application, chaque contact est stocké dans un objet ListViewItem.
L'objet ListViewItem contient des propriétés pour :
- fixer la couleur de fond (BackColor)
- la couleur de la police (ForeColor)
- le texte affiché (Text)
- une instance d'objet (Tag), dans notre application, une instance de PimItem est stockée
- une propriété Checked qui permet de savoir si une case à cocher est cochée ou pas. Dans notre application, une case à cocher est présente à côté de chaque contact. Si la case est cochée, le contact sera envoyé lors de la prochaine transmission.

La méthode ci-dessous crée la liste des contacts dans le ListView de l'application. Cette liste est remplie au début de l'application. Pour la mettre à jour d'utilisation, cliquer sur le menu Menu/Rafraîchir la liste. La méthode ci-dessous est alors exécutée :

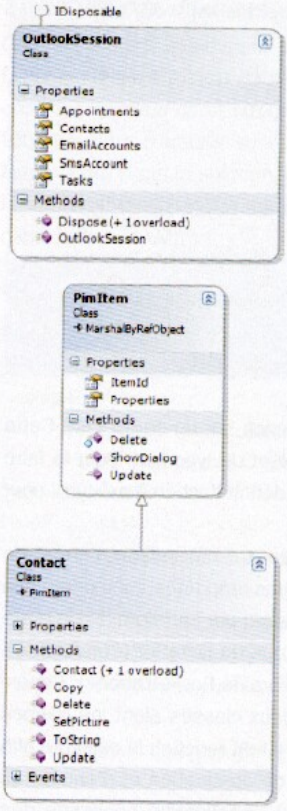
```
/// <summary>
    /// Rempli le ListView par la liste des contacts actuels de PocketOutlook.
    /// </summary>
    private void getContacts()
    {
        progressBarContact.Visible = true;
        progressBarContact.Maximum = Utilities.Session.Contacts.Items.Count;
        progressBarContact.Value = 0;

        lstContacts.Clear();
        lstContacts.Refresh();

        List<PimItem> lstContactsToSend = new List<PimItem>();
        foreach (Contact contact in Utilities.Session.Contacts.Items)
        {
            string detail = contact.FirstName + " " + contact.LastName;

            ListViewItem item = new ListViewItem(detail);
            item.Tag = contact;
            lstContacts.Items.Add(item);
            ++progressBarContact.Value;
        }
        progressBarContact.Visible = false;
        lstContacts.Refresh();
    }
```

Bien veiller à appeler la méthode Refresh du ListView après avoir inséré les contacts, ici lstContacts.Refresh();sinon l'affichage n'est pas modifié.

Lier l'interface aux données de PocketOutlook

Le modèle objet

Nous allons maintenant nous intéresser de plus près au modèle objet utilisé pour exploiter les informations de PocketOutlook. La classe OutlookSession est le point d'entrée pour connaître les types d'informations que l'on peut toucher. Ces types sont définis dans la section Properties du schéma ci-dessous.
Chaque propriété représente un dossier dans PocketOutlook :



- **Rendez-vous :** Appointments
- **Contacts :** Contacts
- **Emails :** EmailAccounts
- **Sms :** SmsAccount
- **Tâches :** Tasks

Le modèle objet d'un contact

Dans la propriété Contacts de OutlookSession, la propriété Items contient la liste des contacts créés par l'utilisateur dans PocketOutlook. Cette classe Contact hérite d'une classe PimItem.
Cette classe est utilisée pour décrire un élément Outlook (contact, email, sms...) de façon unique par le biais de la propriété ItemId.

Les propriétés et les événements ont été volontairement cachés.
Les propriétés contiennent tous les champs affichés dans PocketOutook pour un contact.
Les événements contiennent un événement pour chaque propriété.

Comment faire pour éviter d'écraser un contact existant ?

L'application ne teste pas l'existence d'un contact. Pour rendre cela réalisable, il faut déterminer au préalable les champs discriminants, par exemple les noms et prénoms du contact et effectuer une mise à jour ou une création.

Aller plus loin

A présent, si l'on souhaite modifier le type d'information (rendez-vous, tâches) à émettre, il faut écrire un nouveau dataset typé et écrire de nouvelles méthodes à l'image de ComputeContacts et CreateXmlDocument pour mapper le dataset typé au type d'information qu'on souhaite manipuler. Le choix de passer par un dataset typé réside dans le fait d'avoir un format commun d'envoi / réception d'un type de données (contact, rendez-vous...). La compatibilité de cette application avec les PocketPC 2003 est possible en installant le CF2.0 en allant à l'adresse suivante : http://www.microsoft.com/downloads/details.aspx?FamilyID=9655156b-356b-4a2c-857c-e62f50ae9a55&DisplayLang=fr

Quelques liens :

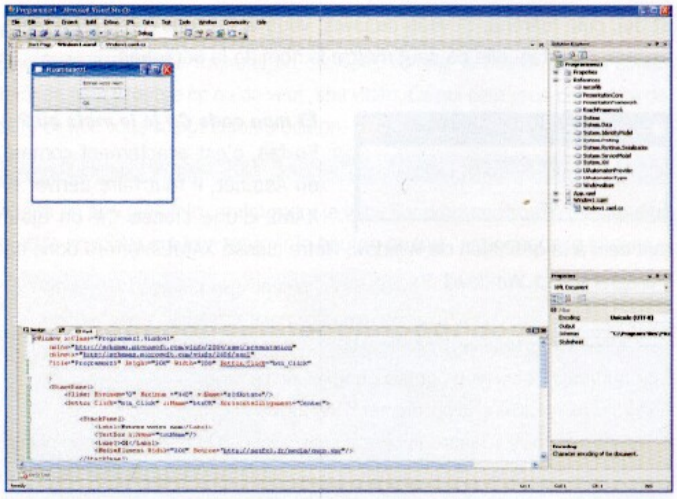
http://msdn2.microsoft.com/fr-fr/library/f44bbwa1.aspx
http://www.pdafr.com/ - http://www.codeppc.com/

■ **Thomas Clarenc**
Chef de projet / Ingénieur concepteur
pour le compte de la société So@t
Certifié MCAD.NET et MCT - Email : tclarenc@hotmail.fr

Mes débuts en WPF

1re partie

Microsoft a lancé cette année la technologie WPF (Windows Presentation Foundation). WPF est une technologie basée sur Direct qui permet de gérer l'affichage des applications tournant en .Net 3.0. WPF est inclus dans Vista pour gérer l'affichage des fenêtres. Ce mille-feuille d'API permet de simplifier le développement en faisant abstraction de la lourdeur de Direct, tout en gardant les performances.



Comment développer en WPF ? Premièrement il vous faudra .Net 3.0 qui est téléchargeable sur le site de Microsoft. Ensuite, le plateform SDK qui inclut des outils très pratiques comme le XAMLPad, il vous faudra aussi les extensions WPF pour Visual studio 2005, et bien entendu Visual studio 2005.

Le langage XAML

Le XAML est un dialecte XML. Il entre en ligne de compte dans la description de l'interface utilisateur, ce qui permet de coder le métier en C# ou en VB.NET et l'interface graphique en XAML.

Tiens ça me rappelle quelque chose !

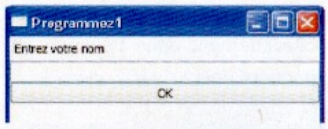
En effet, le principe est déjà utilisé depuis un moment en ASP.Net ou le graphique est décrit par le HTML et le code métier par le C# ou le VB.Net. Tout comme Asp.net, à la compilation, un code MSIL est généré pour l'affichage de l'interface utilisateur.

Comment se présente le XAML ?

Le principe du XAML est simple, regardons une application normale, avec la fenêtre principale, qui contient par exemple un bouton, un label et une textbox. Le bouton contient du texte, ainsi que le label et la textbox, on peut donc dire que la fenêtre est le parent du bouton, de la textbox et du label, et que le texte est l'enfant du bouton, du label et pourquoi pas de la textbox, s'il est nécessaire d'en mettre. Tous ces contrôles ont des propriétés permettant de leur donner une apparence, couleur, taille, etc.

Voila à quoi ressemblerait notre application en code XAML :

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
  <StackPanel>
    <Label>Entrez votre nom</Label>
    <TextBox/>
    <Button>OK</Button>
    <Label/>
  </StackPanel>
</Window>
```



Dans un premier temps on a l'élément Window, qui sera l'élément racine de notre application, puis on a le stackpanel, puis le label, la textbox et enfin le button et un dernier label comme enfant du StackPanel.

Qu'est ce que le StackPanel ?

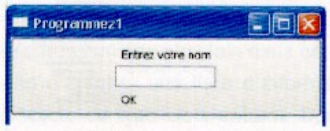
Le StackPanel fait partie de la notion de contenu en WPF, la plupart des contrôles héritent de ContentControl une classe de l'API WPF qui implémente la propriété Content. Cette propriété Content est présente pour mettre du contenu mais l'inconvénient de cette propriété est qu'elle n'accepte qu'un seul enfant, en gros dans la propriété Content de Window, on ne pourrait mettre qu'un bouton, ou une textbox, pas terrible comme nouvelle expérience utilisateur. Heureusement, on a des containers qui peuvent être des contenus de contrôle et avoir plusieurs contrôles enfants. Le stackPanel en fait partie, mais on a aussi le Grid, ou le dockpanel. Chacun de ces contrôles permet un layout particulier. Ici le stackpanel affiche les contrôles en pile (comme son nom l'indique car stack veut dire pile).

Mais le Bouton, il hérite de quoi ?

Button aussi hérite de contentcontrol, on pourrait donc imaginer demander le nom de l'utilisateur dans le bouton et avoir des contrôles enfants comme contenu du bouton :

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:sys="clr-namespace:System;assembly=mscorlib"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
  <StackPanel>
    <Button>
      <StackPanel>
        <Label>Entrez votre nom</Label>
        <TextBox/>
        <Label>OK</Label>
      </StackPanel>
    </Button>
    <Label/>
  </StackPanel>
</Window>
```

Et oui, cela marche, Si vous essayez, vous verrez que le bouton a une textbox dans laquelle on peut mettre le nom de la personne.



Et mon code C# je le mets où ?

En fait, c'est exactement comme en Asp.net, il faut faire dériver le XAML d'une classe C# en ajoutant ceci à la définition de Window, Notre classe XAML héritera donc de " programmer1.Window1 " :

```
x:Class="programmer1.Window1"
ce qui nous donnera un entête complet de ce style :
<Window x:Class="programmer1.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   Title="programmer1" Height="300" Width="300"
   >
```

Ensuite, on ajoute le branchement à l'événement dans l'élément du bouton, notre code complet en XAML donnera ceci :

```
<Window x:Class="programmer1.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   Title="programmer1" Height="300" Width="300"
   >
   <StackPanel>
   <Button Click="btn_Click">
      <StackPanel>
         <Label>Entrez votre nom</Label>
         <TextBox x:Name="txtName"/>
         <Label>OK</Label>
      </StackPanel>
   </Button>
    <Label x:Name="lblWelcome"/>
  </StackPanel>
</Window>
```

J'ai ajouté les noms des contrôles afin de pouvoir y accéder à partir du C#. Je compile d'ailleurs une première fois afin que les noms apparaissent dans l'IntelliSense.
Ensuite, je tape dans le code C# l'affichage de bonjour, plus le nom saisi dans la TextBox :

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
```

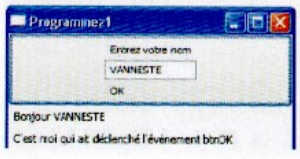
```
namespace programmer1
{
   /// <summary>
   /// Interaction logic for Window1.xaml
   /// </summary>

   public partial class Window1 : System.Windows.Window
   {

      public Window1()
      {
         InitializeComponent();
      }
      protected void btn_Click(object sender, RoutedEventArgs rea)
      {
         lblWelcome.Content = "Bonjour " + txtName.Text;
      }
   }
}
```

A la compilation, deux classes sont créées, une en C# pour le code métier et une autre en C# aussi qui sera la résultante du XAML pseudo compilé, les deux sont ensuite compilées en code MSIL. Ainsi le XAML passe par le langage C# puis MSIL et enfin natif.

Mais qu'est ce que le RoutedEventArgs ?

Comme vous pouvez le voir dans la gestion de l'événement on a un RoutedEventArgs. La notion d'événement routée est assez nouvelle présentée comme cela. Reprenons notre code, mais ajoutons un label qui affichera le contrôle qui a déclenché le clic sur le bouton. Le XAML donne ceci :

```
<Window x:Class="programmer1.Window1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   Title="programmer1" Height="300" Width="300"
   x:Name="wndMain"
   >
   <StackPanel>
   <Button Click="btn_Click" x:Name="btnOK">
      <StackPanel>
         <Label>Entrez votre nom</Label>
         <TextBox x:Name="txtName"/>
         <Label>OK</Label>
      </StackPanel>
   </Button>
    <Label x:Name="lblWelcome"/>
    <Label x:Name="lblEventParent"/>
   </StackPanel>
</Window>
```



Maintenant, ajoutons du code dans le C# pour afficher le nom du contrôle qui a déclenché l'événement :

```
        protected void btn_Click(object sender, RoutedEventArgs rea)
        {
            lblWelcome.Content = "Bonjour " + txtName.Text;
            lblEventParent.Content ="C'est moi qui ait déclenché l'événement
" +  ((ContentControl)(sender)).Name;
        }
```

Quand on l'exécute, il nous informe que c'est le bouton btnOK qui a déclenché l'événement, plutôt logique. Mais si maintenant notre code ressemble à ceci :

```
<Window x:Class="programmer1.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="programmer1" Height="300" Width="300"
        x:Name="wndMain"
        Button.Click="btn_Click"
        >
    <StackPanel>
    <Button Click="btn_Click" x:Name="btnOK">

        <StackPanel>
            <Label>Entrez votre nom</Label>
            <TextBox x:Name="txtName"/>
            <Label>OK</Label>
        </StackPanel>
    </Button>
      <Label x:Name="lblWelcome"/>
      <Label x:Name="lblEventParent"/>
    </StackPanel>
</Window>
```

Comme vous pouvez le voir, c'est la fenêtre qui gère tous les clics sur les boutons, dorénavant. Si on exécute l'application, elle nous dit que c'est le contrôle wndmain qui déclenche l'événement, en fait maintenant, le clic sur un bouton dans cette application est centralisé au même endroit. Ce qui est plutôt bien mais il nous faut à présent récupérer le vrai nom du contrôle qui a déclenché l'événement.
Corrigeons le code de l'application :

```
        protected void btn_Click(object sender, RoutedEventArgs rea)
        {
            lblWelcome.Content = "Bonjour " + txtName.Text;
            lblEventParent.Content ="C'est moi qui ait déclenché l'événement
" +  ((ContentControl)(rea.OriginalSource)).Name;
        }
```

Maintenant on n'utilise plus sender mais rea.OriginalSource qui retourne le nom du contrôle qui a déclenché l'événement et non celui qui gère l'événement.

Mais a quoi cela peut servir ?

Dans beaucoup de cas, on a la même réaction sur le clic de plusieurs contrôles. Par exemple, si vous faites : nouveau document, vous pouvez cliquer sur fichier/nouveau document, ou sur l'icône de la barre d'outils, ou utiliser le raccourci clavier CTRL+N. Si c'est un même contrôle, à savoir la fenêtre qui gère un nouveau document, vous n'avez qu'un seul endroit ou mettre le code.

Et quoi d'autre ?

Une foule de choses. Prenons un exemple, comme le bouton est un conteneur on peut y mettre ce qu'on veut, une vidéo. Ce qui peut vous permettre de faire ce que vous voulez comme bouton. Voici ce qu'il faut ajouter au bouton :

```
<MediaElement Width="200" Source="http://netfx3.fr/medio/gwen.wmv"/>
```

Allons un peu plus loin, maintenant je veux lier deux contrôles, c'est-à-dire mettre un slider qui fera tourner le bouton quand j'actionnerai le slider :

```
<Window x:Class="programmer1.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="programmer1" Height="300" Width="300"
        x:Name="wndMain"
        Button.Click="btn_Click"
        >
    <StackPanel>
      <Slider Minimum="0" Maximum ="360" x:Name="sldRotate"/>
    <Button Click="btn_Click" x:Name="btnOK" HorizontalAlignment="Center">

        <StackPanel>
            <Label>Entrez votre nom</Label>
            <TextBox x:Name="txtName"/>
            <Label>OK</Label>
     <MediaElement Width="200" Source="http://netfx3.fr/media/gwen.wmv"/>
        </StackPanel>
    <Button.LayoutTransform>
     <RotateTransform Angle="{Binding ElementName=sldRotate,Path=Value}" />
    </Button.LayoutTransform>
    </Button>
      <Label x:Name="lblWelcome"/>
      <Label x:Name="lblEventParent"/>
    </StackPanel>
</Window>
```

Dans ce code, je rajoute un slider que je nomme sldRotate, je lui mets une valeur minimum de 0 et maximum de 360 (comme les degrés d'une rotation). Ensuite, je rajoute un binding sur l'angle de rotation du bouton avec la syntaxe suivante :

```
"{Binding ElementName=sldRotate,Path=Value}"
```

Voila, rien de plus, si j'actionne le slider le bouton tourne, et comme le bouton contient une vidéo la vidéo tourne aussi. La textbox peut aussi être remplie si vous tournez le bouton. Bref, tout est dynamique.

Conclusion

WPF est une assez grande révolution dans le développement d'interfaces graphiques sous Windows. Basé sur le vectoriel et DirectX, il permettra de n'avoir quasiment plus de limite à ce qu'on souhaite faire. Prochainement, nous aborderons le WPF/E (la version client léger du WPF), qui permet de développer des applications multi-plates-formes avec un déploiement aussi simple qu'une page HTML. Et pour finir nous examinerons la 3D et comment elle se présente sous WPF.

■ **Xavier VANNESTE**
http://netfx3.fr - http://blogs.developpeur.org/xvanneste/

BIRT : quand Eclipse génère des rapports

Nombreuses sont les entreprises souhaitant créer (rapidement et simplement) des reports, pour le bilan de fin d'année, bilan périodique, ... De nombreux outils sont déjà disponibles sur le marché des outils décisionnels comme ceux de Business Object, Cognos, ...

BIRT (Business Intelligence and Reporting Tools) permet d'utiliser Eclipse pour créer des reports. La force de ce projet nommé BIRT réside dans le fait tout d'abord qu'il est Open Source, ensuite qu'il se matérialise sous la forme d'un plug-in Eclipse. Le projet est sorti en version 1.0 le 8 novembre 2004 et est actuellement en version 2.1.1.

Il est également intégré à Calisto (distribution Eclipse intégrant plusieurs plug-in : BIRT, CDT, Data Tools Platform, EMF, GEF, GMF, WTP, Visual Editor, ...) : http://www.eclipse.org/callisto/, ce qui contribue ainsi à son succès auprès des développeurs. Le terme Open Source peut parfois signifier que le projet n'a pas de réel support, hormis celui assuré par ses " commiters ". En plus de la communauté des développeurs, BIRT est supporté par la société Actuate (http://www.actuate.com) qui est spécialisée dans les applications de reporting d'entreprise depuis plusieurs années.

BIRT se décompose en modules : les deux principaux restent le BIRT Designer et le BIRT Engine (pour la visualisation). La visualisation des reports peut bien évidemment être faite dans Eclipse, mais le plus souvent on préfèrera les héberger sur un serveur d'application Java EE grâce à une WebApp : le BIRT Viewer.

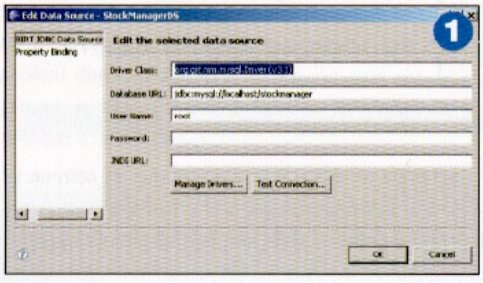
La puissance de ces reports réside dans leur dynamisme, très utile pour connaître la position de ses comptes / factures / charges / stocks / ... de manière instantanée. Ces reports peuvent se baser bien évidemment sur une base de données, ce qui sera certainement la majorité des cas, mais peut également s'appuyer sur des fichiers XML, fichiers plats (fichier CSV, fichier texte, ...) ou un script retournant des données. Si l'on souhaite obtenir des reports statiques, cela est bien évidemment possible. On n'est d'ailleurs pas obligé d'utiliser le BIRT Viewer, on peut se contenter d'exporter le report au format HTML ou PDF afin de le diffuser.

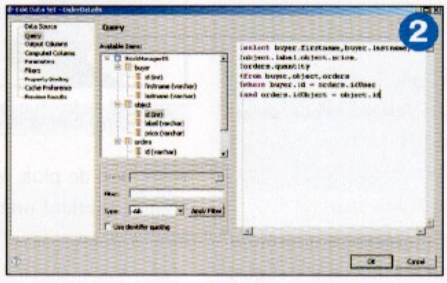
Installation

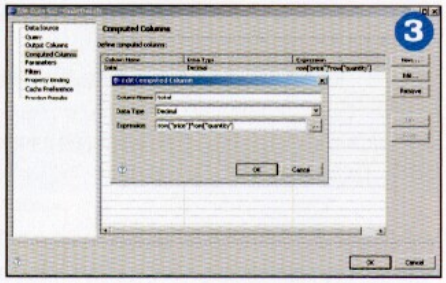
Pour pouvoir l'utiliser, vous aurez le choix soit de l'installer sur un Eclipse si vous en possédez déjà un, soit de récupérer la version "All-in-One" (dans cette version le plug-in est déjà installé sur une version d'Eclipse), soit de préférer le BIRT RCP Designer qui est basé sur Eclipse RCP mais se veut uniquement un designer de report. La version 2.1.1 de BIRT requiert Eclipse 3.2, GEF 3.2 et EMF 2.2. Vous pouvez également récupérer l'API iText qui vous sera nécessaire si vous souhaitez exporter vos reports en PDF. Vous trouverez toutes les informations sur cette API sur le site http://www.lowagie.com/iText/. Il vous faudra également les JDBC Drivers relatifs aux bases de données que vous souhaiterez utiliser. Dans cet article nous utiliserons MySQL.

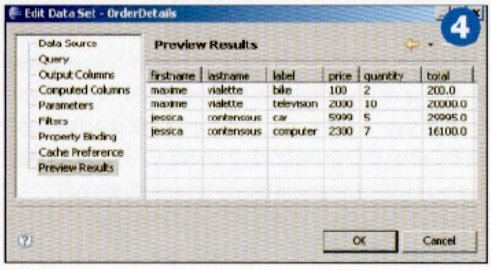
Prise en main

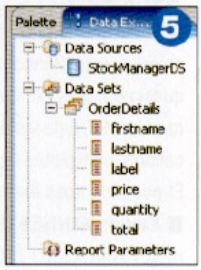
Le designer intègre un éditeur visuel se basant sur Eclipse RCP, nous y retrouverons donc le principe des perspectives et des vues déjà bien connu des utilisateurs d'Eclipse. Pour ceux qui ne connaissent pas Eclipse ou même qui ne sont pas développeurs, pas d'inquiétude, l'utilisation ne nécessite aucun code. L'éditeur vous propose simplement de

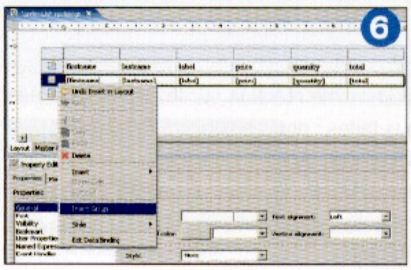












construire le modèle utilisé pour créer votre report par glisser/déposer de manière très conviviale.

Il est possible de créer des tableaux dynamiques, des listes, ... et des graphiques (en barre, en ligne, en zone, en camembert, des nuages de points, ...). BIRT est réellement un outil de reporting... basé sur Eclipse, à ce titre, il intègre une documentation via la vue " Cheat Sheets " qui se décompose en plusieurs sections à la façon des tutoriaux de nombreux plug-in Eclipse.

Cas pratique

Pour débuter la construction de votre report, il vous faudra créer un projet avec le menu " File " \ " New " \ " Project... " \ " Business Intelligence and Reporting Tools " \ " Report Project ". Pour ceux qui connaissent déjà Eclipse, sachez que vous pouvez créer des reports dans n'importe quel type de projet, vous n'êtes donc pas obligé de créer un " Report Project ". Par souci de regroupement et d'organisation, la création d'un tel projet reste cependant conseillé. Eclipse vous demandera alors si vous souhaitez utiliser la perspective " Report Design " qui vous permet d'adapter les menus d'Eclipse à la confection de report. Acceptez ce choix puis dans la vue " Navigator ", faites un clic droit sur votre projet et choisissez " New " \ " Report ". Vous venez de créer un fichier portant l'extension .rptdesign, qui est celle utilisée pour les reports BIRT. L'assistant de création du report vous demandera également de choisir le template que vous souhaitez appliquer à votre report. Vous aurez le choix entre les modèles suivants : Blank Report, My First Report, Simple Listing, Grouped Listing, Dual Column Listing, Chart & Listing, Dual Column Chart & Listing, Side by Side Chart & Listing. Pour commencer nous choisirons " Blank Report ", mais dès que vous aurez pris l'outil en main, vous utiliserez très rapidement les autres modèles disponibles.

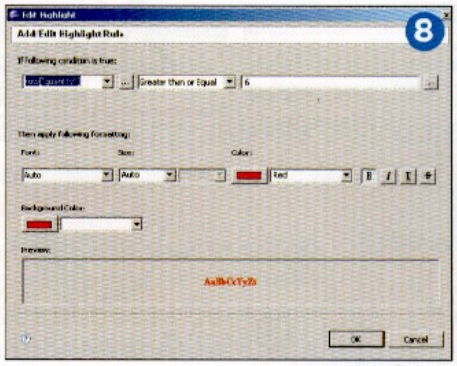
La première des étapes à réaliser va être la création de la liaison à une source de données. Nous allons utiliser MySQL, pour cela, il va falloir définir le type de base de données, la chaîne de connexion, l'utilisateur, le mot de passe ainsi que le JDBC Driver et la classe à utiliser. Voici un exemple de l'assistant de création de DataSource **(Figure 1)**. Nous utiliserons ici une base de données sous MySQL, il faut donc récupérer le pilote correspondant. Une fois cette source de données créée (vous pouvez bien évidemment avoir plusieurs DataSources dans un même report), il faudra créer des DataSets (jeu d'enregistrements) avec l'aide de l'assistant **(Figure 2)**.

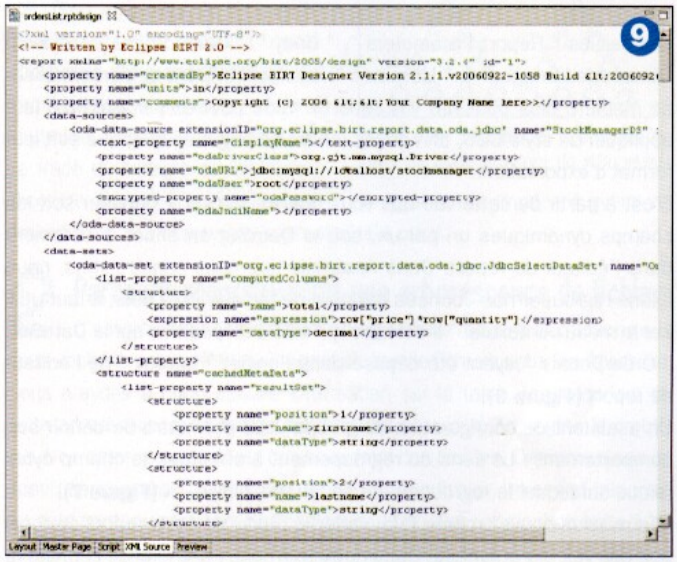
Nous ne détaillerons pas toutes les fonctionnalités proposées par BIRT, cependant, l'une d'elles très appréciée pourra être le fait d'avoir les colonnes du DataSet générées (à utiliser par exemple pour les données calculées). Des notions en SQL seront tout de même nécessaires pour créer votre DataSet : notamment pour la syntaxe, les jointures, ...

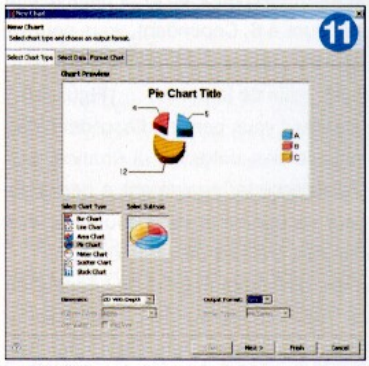
Dans la partie gauche de la fenêtre de cet assistant, vous trouverez différentes options afin de personnaliser ce dernier : DataSource, Query, Output Columns, Computed Columns, Parameters, Filters, Property Binding, Cache Preference, Preview Results.

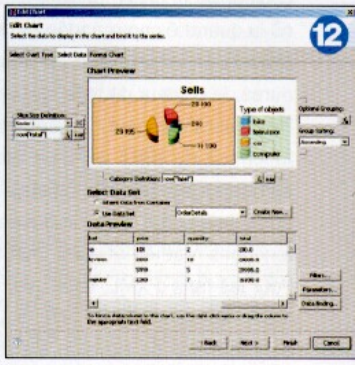
Les parties " Data Source " et " Query " vous permettront de revenir aux étapes précédentes de l'assistant de création du DataSet. La partie "Output Columns" permet de visualiser le type de retour des colonnes de votre jeu d'enregistrements et éventuellement de leur attribuer des alias et nom d'affichage.

La partie " Computed Columns " de l'assistant sera particulièrement pratique pour effectuer des opérations de calcul **(Figure 3)**.







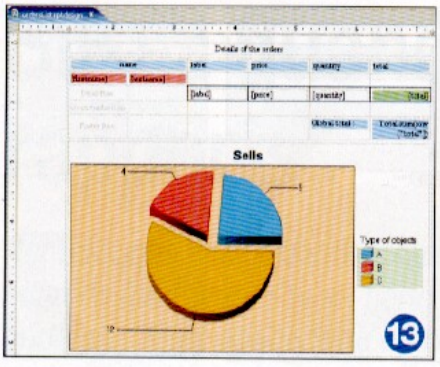


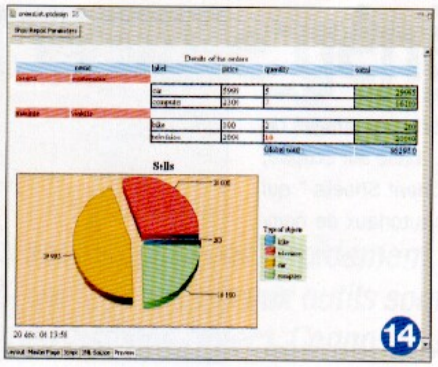
Vous pouvez également choisir de rendre votre formulaire dynamique en fonction des entrées faites par un utilisateur. C'est ce qui vous sera accessible via la partie " Parameters ", "Filters" et "Property Binding". Vous n'avez donc plus à mettre vos opérations de restriction dans votre SQL, mais directement et facilement grâce aux assistants proposés.

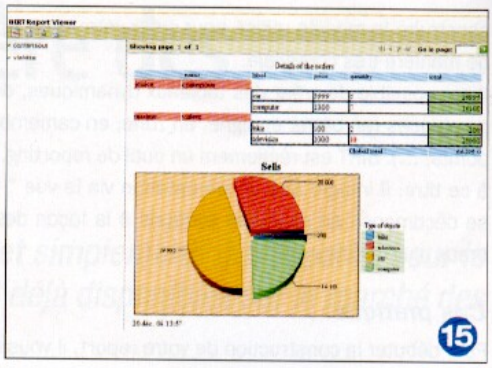
Enfin, une étape de prévisualisation de votre jeu d'enregistrement vous sera proposée avant de pouvoir l'intégrer dans votre report **(Figure 4)**.

Vous pouvez aller plus loin au niveau des DataSets en créant des " Join Data Set " qui vous permettront de faire du regroupement de données provenant de différents types de DataSources (base de données et XML par exemple).

La vue " Data Explorer " d'Eclipse vous permettra d'obtenir le résumé des DataSources & DataSets présents dans votre projet **(Figure 5)**.







Les parties " Report Parameters ", " Body ", "MasterPages", "Styles", "Embedded Images" & " Librairies " vous permettront de personnaliser de manière plus poussée vos reports. Vous pourrez par exemple leur appliquer un style CSS, afin de les rendre homogènes quel que soit leur format d'exportation.

C'est à partir de cette vue que vous pourrez glisser / déposer soit les champs dynamiques un par un, soit le DataSet en entier directement dans l'éditeur de report. Dans l'exemple que nous construisons, nous allons regrouper nos données par nom de famille. Pour cela, il faut utiliser le menu contextuel " Insert Group " mis à disposition sur le DataSet, " OrderDetails " ayant été déposé dans l'onglet " Layout " de l'éditeur de report **(Figure 6).**

Un assistant de configuration du groupe vous permettra de définir son comportement : Le sens du regroupement à effectuer, le champ dynamique sur lequel le regroupement se fait, les filtres, ... **(Figure 7).**

Mieux, vous pouvez même faire varier le rendu de votre report en fonction des valeurs affichées. Dans notre exemple, j'ai choisi de changer la couleur de la police en rouge et de mettre la police en gras dans le cas où la quantité commandée est supérieure à 6. Cependant, vous pourrez dynamiser votre report de manière plus marquée : changer les bordures, la couleur de fond de la cellule, taille de la police, ... **(Figure 8).**

Un assistant appelé " Expression Builder " vous permet d'accéder à des fonctions de BIRT dédiées aux calculs des dates, à la finance, aux sommes, aux tests booléens, ... et d'accéder également à des fonctions JavaScript très pratiques pour récupérer la date courante, des fonctions mathématiques, manipuler des chaînes de caractères, ...

Pour les fans d'XML, vous avez un des différents onglets de l'éditeur du report qui se nomme " XML Source " qui vous permettra de visualiser la définition de ce dernier. En réalité, les fichiers .rptdesign sont des fichiers XML analysés par le BIRT Engine et le BIRT Viewer afin de créer le rendu en temps réel de celui-ci **(Figure 9).**

Pour le moment, nous avons vu comment afficher le contenu d'un DataSet dans le report de manière très rapide, cependant, beaucoup d'autres formes de mise en page sont possibles : Texte dynamique (lié à un élément du DataSet), liste, grille, graphique (2D, 3D) **(Figure 10).**

Il y a une grande diversité de choix dans les graphiques qui sont exportables selon plusieurs formats : PNG, JPG, BMP et SVG. Un assistant de création des graphiques permet de le paramétrer très simplement **(Figure 11).**

Une fois le choix du type de graphique fait, et la liaison avec un DataSet réalisée, l'assistant vous proposera une prévisualisation du résultat dans l'onglet " Select Data ". Il suffit alors de spécifier le champ dynamique qui sera positionné et sur quel axe du graphique, afin de voir la prévisualisation du rendu **(Figure 12).**

Une fois la définition de votre graphique effectuée à l'aide de l'assistant, vous pouvez revenir à l'onglet " Layout " de l'éditeur du report afin d'effectuer les modifications des différents éléments que vous souhaitez afin de le finaliser **(Figure 13).**

Une astuce pour organiser votre report plus facilement : utilisez dans la palette l'objet " Grid " pour positionner dans ses cellules les éléments (tables, graphiques, ...) composant votre report. Toujours dans l'éditeur de report, vous pouvez obtenir le rendu de celui-ci grâce à l'onglet " Preview " **(Figure 14).**

Si vous souhaiter modifier à nouveau votre graphique car vous n'êtes pas satisfait de votre prévisualisation, vous pouvez revenir à tout moment à l'onglet " Layout " afin d'éditer votre report.

Lorsque votre report est finalisé, vous pouvez l'exporter en HTML, PDF, dans le BIRT Viewer **(Figure 15).**

L'avantage de la visualisation dans le BIRT Viewer est que vous pourrez avoir accès à une page d'index, tout comme dans la visualisation d'un report en PDF. Cette option est accessible seulement si vous avez défini des groupes dans votre report, et si celui-ci comporte un listing de plusieurs pages.

Conclusion

Le reporting est plus que vital pour les entreprises, mais les outils restent chers ou, dans le cas d'outils gratuits, assez limités en fonctionnalités. BIRT se présente avec un nombre impressionnant de fonctionnalités faciles à mettre en œuvre. Vous pouvez réaliser des reports croisés qui permettront ainsi de naviguer entre vos différents reports de votre projet via un simple clic sur une cellule d'un de vos tableaux ou sur un élément de votre graphique afin d'accéder à un autre report ... Ce projet est très réactif : des packs de langages sont disponibles, une implémentation sous PHP a été réalisée, des outils commerciaux intègrent BIRT, ... Pour obtenir des informations complémentaires à cet article rendez-vous sur le site du projet : http://www.eclipse.org/birt.

N'hésitez pas à vous rendre aux adresses suivantes pour visionner des tutoriaux flash : http://download.eclipse.org/birt/downloads/examples/reports/2.1/tutorial/tutorial.html et http://download.eclipse.org/birt/downloads/examples/misc/BIRT2.1Demo/EclipseDemo.html

Grâce aux différentes API de BIRT, vous pouvez tout à fait intégrer des fonctions de report dans vos applications web, vos clients RCP, en ligne de commande, ... les possibilités sont multiples.

Maxime Vialette
Ingénieur d'étude et développement chez Completel
Certifié SJCP & MCAD.Net
Membre du laboratoire SUPINFO des technologies Sun

LANGAGE : JAVA
DIFFICULTÉ : ●●●●○
APPLICATION : MATHEMATIQUE
SOURCE : OUI

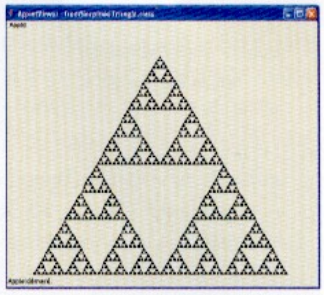
Les algorithmes récursifs

Omniprésents dans le domaine de l'intelligence artificielle ou des fractales, les algorithmes récursifs constituent une part importante de l'algorithmique. Nous faisons connaissance avec quelques unes de leurs particularités.

Dans notre introduction à l'algorithmique du mois dernier nous ne nous sommes intéressés qu'à une catégorie bien définie d'algorithmes: les algorithmes itératifs. Ces algorithmes, comme leur nom l'indique, sont codés avec des boucles. Cette approche est la plus évidente pour le débutant en programmation, notamment parce que, traditionnellement, l'apprentissage d'un langage présente le codage des boucles en premier lieu. Ouvrez par exemple un ouvrage d'initiation à C ou Java. Tout l'arsenal des boucles for, while y sera passé en revue et après cela il sera mentionné que le langage supporte la récursivité. Vous y verrez l'inévitable fonction de calcul d'une factorielle, probablement pauvrement codée. Ainsi en va-t-il dans le monde des langages impératifs et/ou objets tels que C, C++, C#, Java, etc. Dans le monde des langages fonctionnels, moins largement connus, les choses vont autrement. En effet des langages comme Haskell ou Erlang ne disposent tout simplement pas d'instructions de boucle telles que for ou while. Et même l'affectation de variable est interdite. Ainsi, voudrait on avoir un compteur de boucle i que la ligne de code i=i+1 ne pourrait tout simplement pas exister. Peut-on seulement programmer quelque chose avec un langage qui ne connaît pas les boucles et les affectations ? Cela paraît impossible.

1 La récursivité, une technique puissante

Non seulement programmer avec ces langages est possible, mais, lorsqu'on y est habitué, c'est même plus facile qu'avec les langages impératifs. La technique employée est la récursivité, ce qui consiste en des fonctions qui s'invoquent elles mêmes jusqu'à la terminaison d'un calcul. Les langages fonctionnels règnent essentiellement dans les milieux universitaires. C'est compréhensible, car la récursivité s'applique tout naturellement aux calculs mathématiques. Cependant son intérêt ne s'arrête pas là. Nous n'allons pas utiliser aujourd'hui de langage fonctionnel. En effet, la récursivité a toute sa place dans le monde de la programmation impérative. Ainsi un algorithme de tri performant et bien connu, le QuickSort, est récursif. Ce tri est implémenté sous tous les langages impératifs. L'intelligence artificielle utilise surabondamment la récursivité. Ainsi les algorithmes MixMax, avec ou sans élagage AlphaBêta, sont récursifs. (Nous invitons le lecteur à se reporter aux articles "Un jeu de Puissance 4 en Lisp" de programmez! 56, 57 et 58. Bien que Lisp, langage réputé fonctionnel, soit employé, le codage du jeu est réalisé entièrement dans un style impératif). En fait, c'est quand la simple boucle for n'est d'aucun secours que la récursivité nous tire d'affaire. Par exemple lorsqu'il s'agit de parcourir une arborescence inconnue au moment du codage. Ainsi un algorithme MixMax parcourt une arborescence de coups possibles à la recherche du meilleur. L'arbre des coups dépend de la situation du jeu et n'est pas connu au moment du codage. Plus simplement, et sans parler d'intelligence artificielle, nous pouvons souhaiter parcourir l'arborescence d'un système de fichiers. Dans le monde de la programmation impérative, la récursivité a souvent mauvaise réputation, en raison de mauvaises performances, dit on, tandis que les langages fonctionnels sont optimisés pour. S'il est vrai, et normal, que les langages fonctionnels sont optimisés pour la récursivité, il ne faut pas sous-estimer ce qu'il est possible de faire avec une bonne technique de codage et un compilateur capable ainsi que nous allons le voir. Bien employée, la récursivité est élégante et le code est concis et efficace. Sa grande force est de fractionner une opération en une multitude d'opérations identiques. Le tracé du triangle de Sierpinski ci-contre en est une illustration. Un programmeur impératif averti ne devrait donc pas écarter cette possibilité a priori.



Tracé d'un triangle de Sierpinksi en Java. Le code provient de Wikipedia.

2 Parcourir récursivement une arborescence de fichiers

C'est très naturel d'élaborer un algorithme récursif. Supposons que nous voulions parcourir toute l'arborescence de fichiers sous un répertoire. Nous n'avons à priori aucune information sur la taille, la profondeur de cette arborescence. La boucle for est donc proscrite. Analysons le problème. Une arborescence, c'est des noeuds et des feuilles. Dans notre cas un noeud sera un répertoire, et une feuille sera un fichier. Un noeud (répertoire) peut contenir plusieurs autres noeuds et feuilles, et ainsi de suite. L'algorithme récursif de notre fonction "walk" est tout simple et découle directement de la description du problème. Le voici en pseudo code :

```
fonction walk (noeud)
    pour tout item contenu dans noeud
        si item est un fichier, afficher son nom
        sinon si item est un autre_noeud
            afficher son nom
            appeler walk(autre_noeud)
```

Il est bien évident que lorsqu'on écrit un algorithme récursif, il convient de veiller très attentivement à la terminaison des opérations, sinon le programme va boucler indéfiniment. Implémentons notre algorithme en Java. Nous nous basons sur la classe java.io.File qui permet de savoir si une noeud de l'arborescence est un fichier ou un répertoire. La classe permet aussi d'énumérer tous les descendants d'un noeud. Le code est donc immédiat :

```
package fred;

import java.io.File;

public class Walk {
    private int nb_items = 0;

    private void walk(File file) {
        if(file.isDirectory()) {
            System.out.println(file.getName());
            File[] files = file.listFiles();
            for(File sub : files) {
```

```
            walk(sub);
            }
        }
        else System.out.println(
            "- file - " + file.getName());
        nb_items++;
    }

    public int getNbItems() {
        return nb_items;
    }

    public static void main(String[] args) {
        String root = "C:"  + File.separator + "temp";
        Walk w = new Walk();
        w.walk(new File(root));
        System.out.println(
            root + " compte " + w.getNbItems() + " items");
    }
}
```

```
Output - Recursivite (run-single)
graphecomp
-- file -- arbreCouvrantMinimal.htm
-- file -- bellmannFord.htm
-- file -- Chap0.htm
-- file -- Chap1.htm
-- file -- Chap2.htm
-- file -- Chap3.htm
CoBin
-- file -- CoefBin.html
-- file -- CoefBiner.html
-- file -- dijkstra.htm
Finished building Recursivite (run-single).
```

Parcours récursif d'une arborescence de fichiers par notre programme Java.

L'illustration ci-contre montre une partie de la sortie de programme que vous trouverez sur le Cd-Rom accompagnant le magazine. Fondamentalement notre code ne fait qu'une opération, lire le contenu d'un noeud, et cette unique opération est répétée jusqu'à ce que l'arbre soit totalement parcouru. Nous avons bien fractionné une opération en une multitude d'opérations identiques et exprimé très simplement un problème a priori complexe.

3 Performance des algorithmes récursifs.

Pourquoi les programmes récursifs écrits avec un langage impératif sont-ils réputés moins performants que leur équivalent itératifs ? Parce que les appels de fonction imbriqués obligent le compilateur (ou l'interpréteur le cas échéant) à construire des cadres de piles à chaque appel. Si fondamentalement, la complexité (cf. Programmez! 94) de l'algorithme est la même, ces constructions de cadre de pile sont autant d'instructions invisibles dont l'exécution se répercute finalement sur le temps d'exécution. Cependant de bonnes techniques de codage permettent d'obtenir des effets surprenants. Etudions cela à partir de l'inévitable exemple de la factorielle, et avec un langage et un compilateur dignes de ce nom: C++ et gcc 4.1.1. Rien n'empêche d'écrire le calcul d'une factorielle avec un algorithme itératif, une simple boucle :

```
#include <iostream>
#include <cstdlib>
#include <string>

using namespace std;

int fact_iter(int valeur)
{
   int fact = 1;
   for(int i=2; i<=valeur; i++)
      fact *= i;

   return fact;
}

int main(int argc, char* argv[])
{
   string in;
   cout << "Donnez un nombre entier positif" << endl;
   cin >> in;
   int valeur = atoi(in.c_str());
   if(valeur < 0)
   {
      cout << "saisie incorrecte" << endl;
      exit(1);
   }
   if(valeur == 0 || valeur == 1)
   {
      cout << "Factorielle " << valeur
<< " = " << 1 << endl;
      exit(0);
   }
   cout << "Factorielle " << valeur
<< " = " << fact_iter(valeur) << endl;
   return 0;
}
```

Le coeur de ce programme est la fonction fact_iter. Nous avons compilé avec l'option d'optimisation -O2. Nous demandons en outre à ce même gcc de générer un listing assembleur pour notre code (gcc -O2 -S fact_iter.cpp). Nous en extrayons notre fonction fact_iter :

```
_Z9fact_iteri:
.LFB1421:
   pushl   %ebp
.LCFI0:
   movl    $1, %eax
   movl    %esp, %ebp
.LCFI1:
   movl    8(%ebp), %ecx
   cmpl    $1, %ecx
   jle     .L4
   movl    $2, %edx
   incl    %ecx
   .p2align 4,,15
.L5:
   imull   %edx, %eax
   incl    %edx
   cmpl    %ecx, %edx
   jne     .L5
.L4:
   popl    %ebp
   ret
```

Ce code est, comme on s'y attendait légitimement, tout ce qu'il y a de concis et performant. Après construction d'un cadre de pile (pushl %ebp, etc...) une seule fois à l'entrée de la fonction, l'exécution se réduit à des sauts entre

.L5 et jne .L5. Parfait, rien à redire. Récrivons notre programme avec l'idiome classique de la fonction factorielle récursive: (nous ne reproduisons que cette fonction ici. Le lecteur voudra bien se reporter au Cd-Rom pour tous les exemples complets, y compris les listings en assembleur).

```
int fact_rec(int valeur)
{
  if(valeur == 1)
      return valeur;
  else
      return valeur * fact_rec(valeur - 1);
}
```

et examinons maintenant le code assembleur généré :

```
_Z8fact_reci:
.LFB1389:
  pushl   %ebp
.LCFI0:
  movl    %esp, %ebp
.LCFI1:
  subl    $8, %esp
.LCFI2:
  cmpl    $1, 8(%ebp)
  jne     .L2
  movl    8(%ebp), %eax
  movl    %eax, -4(%ebp)
  jmp     .L4
.L2:
  movl    8(%ebp), %eax
  decl    %eax
  movl    %eax, (%esp)
  call    _Z8fact_reci
  movl    %eax, %edx
  imull   8(%ebp), %edx
  movl    %edx, -4(%ebp)
.L4:
  movl    -4(%ebp), %eax
  leave
  ret
```

Point n'est besoin de comprendre beaucoup d'assembleur pour se rendre compte que ce code est bien plus mauvais. C'est à l'instruction call _Z8fact_reci que le bât blesse le plus. Cette instruction correspond à l'appel de fact_rec dans le corps de notre fonction C++. Cet appel provoque la ré-éxécution du code dès le dèbut. Et dans ce cas, le cadre de pile (pushl %ebp, etc.) est construit autant de fois que la fonction s'invoque elle même. Outre le ralentissement, nous encourons aussi un débordement de la pile, car la taille de celle-ci n'est pas illimitée.

4 La récursivité terminale, ou la performance retrouvée

Si nous pouvons facilement abandonner, ainsi que nous l'avons vu, l'approche récursive pour le calcul d'une factorielle, nous ne pouvons pas abandonner l'idée d'écrire des algorithmes récursifs en général. Ce qu'il nous faut, c'est une technique de programmation qui remédie au problème. Cette technique s'appelle la récursivité terminale. Le terme ne doit pas prêter à confusion avec la terminaison des algorithmes récursifs mentionnés plus haut. Nous dirons qu'une fonction est dite récursive terminale lorsque tous ses appels récursifs sont terminaux. Un appel récursif est terminal lorsqu'il est la dernière instruction exécutée dans le corps d'une fonction et que sa valeur de retour ne fait pas partie d'une expression. Dans notre exemple précédent nous avons

```
return valeur * fact_rec(valeur - 1);
```

Cet appel est bien le dernier du corps de la fonction mais sa valeur de retour est multipliée avec la variable nommée valeur. Nous n'avons donc pas un appel récursif terminal. Pour y remédier, une technique très efficace est l'emploi d'une variable dite accumulateur. Une telle variable n'est qu'un argument supplémentaire à la fonction et n'est jamais déclarée dans le corps de la fonction. Voici donc notre fonction ré-écrite:

```
int fact_rec_term(int valeur, int accu)
{
  if(valeur == 1)
      return accu;
  return fact_rec_term(valeur-1, valeur*accu);
}
```

Cette technique est bien connue des programmeurs fonctionnels et est omniprésente dans du code Erlang par exemple. Nous avons maintenant une fonction récursive terminale. Mais direz vous, un paramètre supplémentaire de fonction C++ ça passe par la pile, alors où est le gain ? Ça doit même être pire en fait ? Non, car un bon compilateur est capable de détecter un appel récursif et dans ce cas il utilise toujours le même cadre de pile au lieu d'en créer un à chaque appel. Quant à la variable supplémentaire, elle est rangée dans un registre tout simplement. Gcc est capable d'une telle optimisation. Donnons la commande :

```
g++ -O2 -foptimize-sibling-calls -S fact_rec_term.cpp
```

et examinons le code assembleur de notre fonction :

```
_Z13fact_rec_termii:
.LFB1421:
  pushl   %ebp
.LCFI0:
  movl    %esp, %ebp
.LCFI1:
  movl    8(%ebp), %edx
  movl    12(%ebp), %eax
  cmpl    $1, %edx
  je  .L3
  .p2align 4,,15
.L6:
  imull   %edx, %eax
  decl    %edx
  cmpl    $1, %edx
  jne     .L6
.L3:
  popl    %ebp
  ret
```

Surprise! Ce code est même plus concis que celui obtenu à partir de l'itération. La partie principale, entre .L6 et jne .L6 présente les mêmes performances que celle de l'algorithme itératif. Nous ne perdons donc pas en performance ce que la récursivité apporte en expressivité.

■ **Frédéric Mazué** - *fmazue@programmez.com*

NOUVEAU : donnez votre avis sur ce numéro www.programmez.com

Comprendre le Design Pattern Décorateur

Le pattern Stratégie nous a enseigné les bienfaits de la composition en programmation objet. Lorsqu'il s'agit d'étendre les fonctionnalités d'une classe dont on ne possède pas le code, ou à l'exécution, une autre forme de composition peut entrer en jeu : le pattern Décorateur.

Reprenons la conception de notre Wargame fictif en Java et résumons la situation. Nous programmons avec le paradigme objet. Tout le monde s'accorde pour dire que la programmation objet est souple, et permet la réutilisation de code. Cependant le mois dernier, nous avons constaté qu'une importante hiérarchie de classes non seulement ne permet qu'une réutilisation de code médiocre voire nulle, mais conduit à de véritables cauchemars lorsqu'il s'agit de faire évoluer le code. Bien que l'héritage soit au centre de la programmation objet, son emploi excessif apparaît pénalisant, paradoxalement. Nous avons constaté grâce au Design Pattern Stratégie que la composition est une alternative beaucoup plus souple. Aujourd'hui nous étudions une autre façon de composer les objets: le Design Pattern Décorateur.

1 Toujours plus de fonctionnalités pour une classe

Le Design Pattern Stratégie nous a dispensé de créer une hiérarchie trop lourde en offrant la possibilité de faire varier dynamiquement les algorithmes utilisés par les clients que sont nos unités de combat. Pour un type donné, par exemple un soldat, la faculté de tirer est déléguée à l'arme qu'il détient. Nous avons ainsi un soldat adaptable dynamiquement, ce qui est ce que nous voulions et ce qui reste pleinement valable aujourd'hui. Seulement voilà, notre soldat, tout adaptable qu'il est, reste un soldat. Notre chef de projet, ami fidèle ;), intervient de nouveau et nous fait remarquer qu'un soldat qui acquiert de l'expérience au combat peut devenir une unité de combat d'élite, c'est-à-dire plus performante au tir, plus rapide et plus furtive dans ses déplacement etc. En plus ou à côté d'être une unité d'élite, notre soldat, continue notre chef de projet, peut être doté d'une aptitude spéciale. Par exemple être doué pour l'espionnage, ce qui correspondrait à la faculté d'accumuler des renseignements au cours de ses déplacements. Le chef de projet omet soigneusement de préciser que l'idée est valable pour d'autres types d'unité de combat, par exemple un avion ou un navire. Il ne le précise pas, mais nous devons anticiper le problème, sinon il mettra à nouveau notre code par terre à son prochain passage.

2 Premières approches

En première approche nous pouvons choisir de modifier le code de nos classes. Soit une hypothétique classe Soldat :

```
class Soldat {

   // toutes les autres méthodes ici, puis:

   boolean espion = false;

   public void acquiertEspionnage() {
       espion = true;
   }

   public String deplacement () {
       String dep = "Deplacement";
       if(espion)
           dep += " acquisition de renseignements";
       return dep;
   }
}
```

Ce qui ne va pas avec cette approche est qu'il faut modifier du code existant. D'abord cela introduit des risques de nouveaux bugs. Ensuite, on retrouve le même problème que le mois dernier: dans notre hypothétique jeu de Wargame nous pouvons nous trouver avec des centaines de classes à modifier. En plus, dans un développement réel sur lequel interviendraient plusieurs entreprises différentes, il est très probable que le code source écrit par l'entreprise A soit inaccessible à l'entreprise B qui devrait faire les modifications. Une telle approche est donc probablement la dernière chose à envisager. En deuxième approche, on peut se dire qu'un soldat d'élite est avant tout un soldat. De même un soldat espion est avant tout un soldat. Donc il "n'y a qu'à" dériver de nouvelles classes de la classe Soldat :

```
class SoldatEspion extends Soldat { /* etc */}
class SoldatElite extends Soldat { /* etc */ }
```

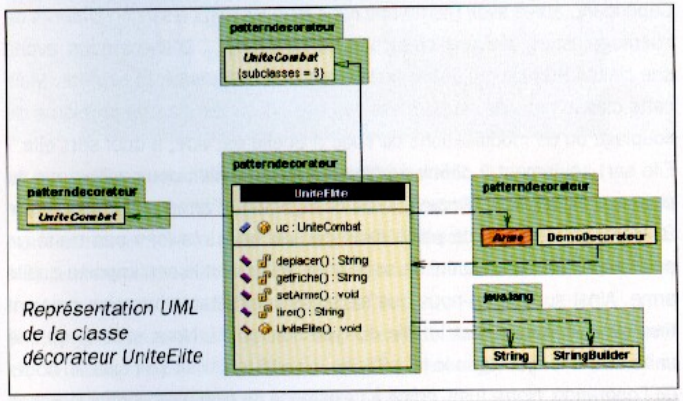
Bref, faire jouer à plein le polymorphisme. Mais cette approche ne vaut rien non plus. Si nous pouvons nous dispenser de disposer du code source, nous retombons à nouveau dans le problème mentionné le mois dernier et ci-dessus. Cela peut conduire à créer des centaines de classes nouvelles. Chaque fois qu'une unité est susceptible d'avoir des capacités d'élite ou d'espionnage, nous devons écrire deux classes. Le nombre de classes va littéralement exploser, et gare aux maux de tête si nous devons modifier quelque chose plus tard. Autre souci, que fait-on si une unité peut devenir à la fois élite ou espionne ? Java, notre langage choisi pour notre exemple ne supporte pas l'héritage multiple.

```
class SoldatEliteEspion
   extends SoldatESpion, SoldatElite
       { // IMPOSSIBLE!! }
```

Nous pourrions nous en tirer en redéfinissant des interfaces supplémentaires. Ce qui au passage montre le bien fondé de notre principe de conception "Préférer implémenter une interface plutôt qu'une classe". Mais là aussi les répercussions sur le code sont trop lourdes. Et quand bien même le ferait-on, un problème persiste. Nous voulons qu'un Soldat puisse acquérir des capacités au cours du jeu, ce qui revient à dire qu'un objet doit être modifié dynamiquement. Avec une architecture de classes dérivées, ce n'est pas si simple. Il faudrait par exemple écrire une classe SoldatEspion dont le constructeur prendra une classe Soldat en argument, recopier l'état du Soldat (points de vie, moral, arme, etc.) et ainsi de suite pour chaque type d'unité. Avec tout le travail et encore une fois les risques de bugs que cela comporte.

3 Le Design Pattern Décorateur

Pour avoir rencontré ces problèmes moult fois, les spécialistes de la programmation objet on défini le pattern Décorateur, qui est une application des principes de conceptions à la base des Design Patterns et récapitulés ci-contre . Les quatre premiers principes nous sont déjà connus. Le dernier s'impose quand on n'a pas accès au code source. Voici la définition du pattern décorateur: "Il attache dynamiquement des responsabilités supplémentaires à un objet. C'est une alternative souple à la dérivation pour étendre les fonctionnalités d'une classe". Concrètement, on le réalise en écrivant une classe de même type que celle dont on veut étendre les fonctionnalités. La classe décorateur expose les mêmes méthodes. Le constructeur du décorateur reçoit en argument une référence sur la classe à étendre et le décorateur conserve cette référence. Enfin le décorateur ajoute son propre comportement soit avant soit après avoir délégué le reste du travail à l'objet décoré. Cette démarche permet d'étendre les fonctionnalités d'une classe à l'exécution. Et cette extension n'est pas limitée. Une classe peut être décorée plusieurs fois de suite par plusieurs décorateurs différents.



Représentation UML de la classe décorateur UniteElite

Principes de conception à retenir

- Encapsuler séparément ce qui varie.
- Préférer implémenter une interface plutôt qu'une classe.
- Préférer l'agrégation à l'héritage.
- Limiter le couplage entre les objets.
- Les classes doivent être ouvertes à l'extension, mais fermées à la modification.

Voici notre jeu remanié (Code sur le Cd-Rom). Les classes UniteElite et UniteEspion sont des décorateurs :

```
package patterndecorateur;

interface Arme {
   public String tirer(UniteCombat iuc);
}


class Fusil implements Arme {
   public String tirer(UniteCombat detenteur) {
      if(detenteur instanceof Humain)
         return "tir fusil";
      else
         return "";
   }
}

interface IUniteCombat {
   public String deplacer();
   public String tirer();
   public void setArme(Arme arme);
   public String getFiche();
}

abstract class UniteCombat implements IUniteCombat {
   protected int pv; // points de vie
   protected Arme arme;
}

abstract class Humain extends UniteCombat {
}

class Soldat extends Humain {
   public String deplacer() {
      return "Deplacement";
   }

   public String tirer() {
      return (arme == null) ? "" : arme.tirer(this);
   }

   public void setArme(Arme arme) {
      this.arme = arme;
   }

   public String getFiche() {
      return "Soldat";
   }
}

// Cette classe dérive de UniteCombat seulement
// pour la compatibilité de type
class UniteElite extends UniteCombat {
   UniteCombat uc;

   UniteElite(UniteCombat uc) {
      this.uc = uc;
   }

   public String deplacer() {
      return uc.deplacer() +
         " rapide et furtif";
   }

   public String tirer() {
      String tir = "Action efficace: ";
      tir += uc.tirer();
      return tir;
   }

   public void setArme(Arme arme) {
```

```
        uc.setArme(arme);
    }

    public String getFiche() {
        return uc.getFiche() + " elite";
    }
}

// Idem, dérivation pour compatibilité de type
class UniteEspion extends UniteCombat {
    UniteCombat uc;

    UniteEspion(UniteCombat uc) {
        this.uc = uc;
    }

    public String deplacer() {
        return uc.deplacer() +
            " acquisition de renseignements";
    }

    public String tirer() {
        return uc.tirer();
    }
    public void setArme(Arme arme) {
        uc.setArme(arme);
    }

    public String getFiche() {
        return uc.getFiche() + " espion";
    }

public class DemoDecorateur {
    public static void main(String[] args) {
        System.out.println("Debut partie");
        UniteCombat[] armee = new UniteCombat[2];
        armee[0] = new Soldat();
        armee[0].setArme(new Fusil());
        armee[1] = new Soldat();
        armee[1].setArme(new Fusil());
        System.out.println(armee[0].tirer());
        System.out.println(armee[1].tirer());
        System.out.println(armee[0].deplacer());
        System.out.println(armee[1].deplacer());
        System.out.println(armee[0].getFiche());
        System.out.println(armee[1].getFiche());

        System.out.println("Promotion d'un soldat");
        armee[1] = new UniteElite(armee[1]);
        System.out.println(armee[0].tirer());
        System.out.println(armee[1].tirer());
        System.out.println(armee[0].deplacer());
        System.out.println(armee[1].deplacer());
        System.out.println(armee[0].getFiche());
        System.out.println(armee[1].getFiche());
        System.out.println(
            "Les unités acquierent un don special");
        armee[0] = new UniteEspion(armee[0]);
        armee[1] = new UniteEspion(armee[1]);
        System.out.println(armee[0].tirer());
        System.out.println(armee[1].tirer());
        System.out.println(armee[0].deplacer());
        System.out.println(armee[1].deplacer());
        System.out.println(armee[0].getFiche());
        System.out.println(armee[1].getFiche());

        System.out.println("Fin partie");
    }
}
```

Examinons ce code. Les méthodes exposées par les décorateurs sont bien les mêmes que celles des objets décorés. Selon le cas, le décorateur fait son travail en amont ou en aval du travail principal délégué à l'objet décoré. Cependant, après avoir mentionné dans tous les sens les inconvénients de l'héritage, nous dérivons beaucoup nous-mêmes.... D'abord nous avons une classe Humain qui dérive de la classe abstraite unité de combat. Mais cette classe est vide, et ce n'est pas elle qui va induire des problème de couplage ou de modifications de code ;) Si elle est vide, à quoi sert elle ? Elle sert seulement à définir un type. Le mois dernier, nous avons mis en place un mécanisme dynamique de changement d'arme grâce au pattern Stratégie. Mais faute de place dans l'article nous n'avions pas traité un point important: toute unité de combat ne doit pas utiliser n'importe quelle arme. Ainsi supposons-nous que seuls les combattants humains peuvent tirer avec un fusil. C'est le rôle du type humain ici. Nous savons qu'une unité de combat délègue le tir à l'arme. L'unité transmet son type au cours de l'opération. Notre fusil, grâce à l'existence du type peut vérifier que son possesseur est qualifié pour s'en servir :) Cette solution demande cependant à être améliorée, ce que nous remettons à plus tard.

Nos décorateurs dérivent de la classe abstraite UniteCombat. Mais cette dérivation ne fait pas non plus intervenir le polymorphisme. Tout ce que nous voulons obtenir est une concordance de type. Ainsi dans notre tableau armee (méthode main), une classe décorée reste du type abstrait UniteCombat, ce qui est ce que nous voulions. Ceci montre la puissance du pattern. Il y a plus: en supposant que des classes concrètes "Navire" ou "Avion" puissent, elles aussi, devenir des unités d'élites, nous pourrions les décorer avec UniteElite alors même que lorsque nous avons écrit cette dernière, nous n'avons pas un seul instant songé à l'existence de l'avion ou du navire. Nous avons comme le mois dernier réalisé un agrégat entre des objets faiblement couplés. Et comme souhaité, le pattern décorateur nous permet bien d'étendre des classes sans disposer des sources. Enfin, la décoration peut être appliquée plusieurs fois successivement. Notre exemple illustre cela en décorant un même objet d'abord avec UniteElite puis avec UniteEspion. Décorateur est un pattern très puissant. Mais comme toute chose il a aussi ses inconvénients. Par exemple la JDK fait grand usage du pattern pour les flots d'entrées/sorties. Le résultat est qu'on peut vite se perdre dans le grand nombre de classes que cela induit et que la construction d'un flot précis implique un code fastidieux, fait de constructeurs imbriqués, ainsi que tout programmeur Java (ou .Net) le sait. Pour remédier à ce problème potentiel, existent les patterns Fabrique et Monteurs, mais ceci est une autre histoire ;)

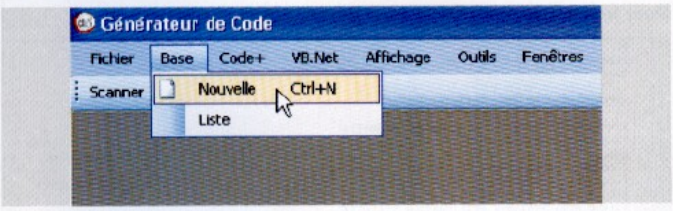
■ **Frédéric Mazué** - *fmazue@programmez.com*

DataBase2Code s'occupe du code et des données

Le logiciel DataBase2Code permet de générer une couche d'accès aux données qui facilitera votre développement. Les Tables et Champs seront transformés en classes et fonctions, ce qui aura l'avantage d'avoir l'IntelliSense des objets de la base directement dans votre environnement de développement (par exemple Zend ou PhpEdit).

DB2C vous permet aussi de vous affranchir des tâches répétitives en utilisant un moteur de scripts qui permet d'adapter le code en fonction de la structure.

Nous allons commencer notre prise en main en créant une nouvelle connexion :

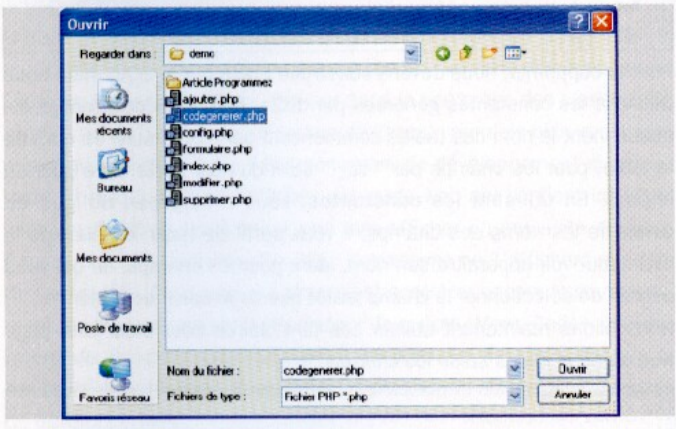


et nous allons nous laisser guider par l'assistant.

Vous renseignez le nom que vous voulez donner à ce projet, l'espace de nom ne servant à rien pour les projets PHP (il est utilisé pour les projets VB), vous pouvez le laisser tel quel. Ensuite, vous créez votre chaîne de connexion, un assistant peut vous y aider.

Renseignez ensuite le type de base, le nom du serveur, ainsi que le nom de la base de données, le nom de l'utilisateur ayant le droit de se connecter et son password.

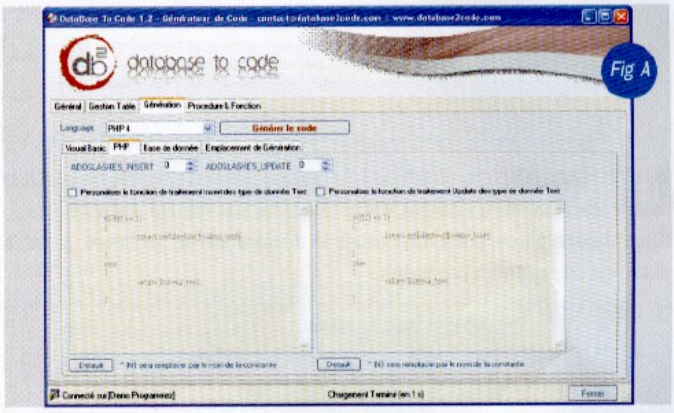
Sélectionnez où vous souhaitez effectuer la génération du code, soit dans le presse papier (après chaque génération vous devrez faire Coller dans le fichier texte contenant le code) ou directement dans un fichier.



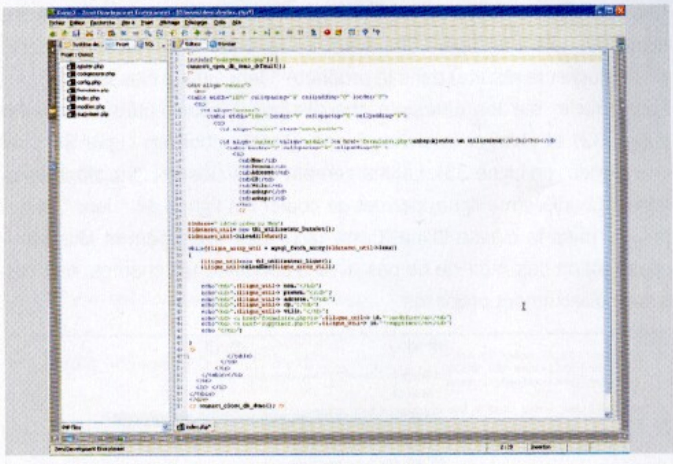
Dans ce cas, dans type de fichier, sélectionnez PHP et recherchez le fichier qui va contenir le code généré.

L'assistant possède maintenant toutes les informations et peut effectuer le scan et la génération du code. **(Fig. A)**

Sélectionnez l'onglet Génération et choisissez comme type de langage PHP 4, ensuite vous pouvez modifier la manière dont on gère les Add-SLASHES, pour le moment, gardez cette valeur à 0 et cliquez sur générer et ensuite sur Fermer. Nous allons pouvoir commencer à coder et



voir ce qui a été généré a partir de ZEND. Nous allons commencer par faire une page qui permette d'afficher les données :



Pour commencer, il faut inclure le fichier généré, ensuite nous établirons la connexion vers la base de données.

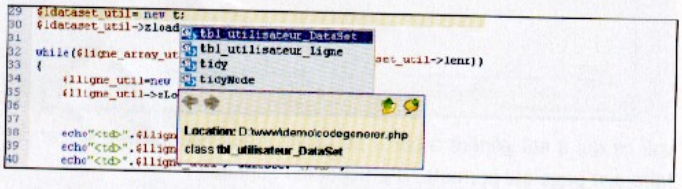
Il nous suffit d'appeler la fonction connect_open_db_demo_default().

Nous allons mixer du php et du Html, je ne vais pas ici détailler la programmation, je me base sur le principe que vous connaissez déjà PHP ainsi que le HTML. Le code généré contient un ensemble de classes, toutes commencent par le nom de la table et sont ensuite suivies de leur type. Deux types de classes sont générés, la classe Ligne, elle gère une ligne d'une table de la base et permet d'effectuer les insert, select et update sans avoir à taper une requête sql. La classe DataSet permet de charger un ensemble de lignes mais ne permet pas de mise à jour de la base de données, elle est néanmoins utilisée pour créer une page qui affiche un ensemble de données. Il vous est également possible de combiner la classe Ligne à la classe DataSet pour mettre à jour les infor-

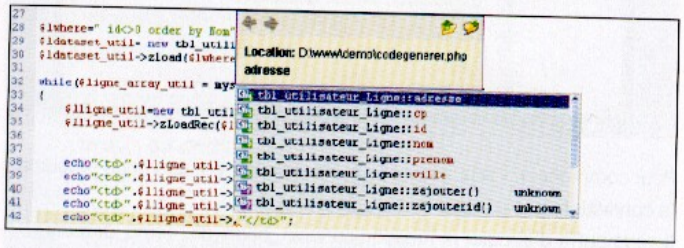
mations contenues dans la classe DataSet. Nous allons commencer par créer notre première page PHP qui affiche les informations depuis la table tbl_utilisateur. Elle contient aussi la possibilité de rajouter une personne dans la base de données et de modifier ou supprimer une personne existante, nous verrons ce code dans les parties correspondantes. Le code PHP devrait ressembler à ceci :

```
<?
$lwhere=" id<>0 order by Nom";
$ldataset_util= new tbl_utilisateur_DataSet();
$ldataset_util->zload($lwhere);

while($ligne_array_util = mysql_fetch_array($ldataset_util->lenr))
{
    $ligne_util=new tbl_utilisateur_Ligne();
    $ligne_util->zLoadRec($ligne_array_util);


    echo"<td>".$ligne_util-> nom."</td>";
    echo"<td>".$ligne_util-> prenom."</td>";
    echo"<td>".$ligne_util-> adresse."</td>";
    echo"<td>".$ligne_util-> cp."</td>";
    echo"<td>".$ligne_util-> ville."</td>";
    echo"<td> <a href='formulaire.php?id=".$ligne_util-> id."'>modifier</a></td>";
    echo"<td> <a href='supprimer.php?id=".$ligne_util-> id."'>supprimer</a></td>";
    echo "</tr>";

}
?>
```

Pour effectuer la boucle du tableau qui affichera les personnes, nous allons utiliser la classe tbl_utilisateur_dataset, qui est déclarée en ligne 29.
Nous voyons que si nous effectuons la déclaration de la classe, la table nous est directement proposée.



Nous utilisons la fonction ->zLoad() en lui passant en paramètre where (ligne 30), elle s'occupera de construire la requête SQL avec le where, et de stocker le résultat dans la propriété " lenr " de la classe.
Pour boucler sur les éléments chargés, nous allons utiliser un while (Ligne 32) et charger la classe Ligne (Construction en Ligne 34 puis chargement en Ligne 35). Le chargement de la classe Ligne via la fonction ->zLoadRec(ma ligne) permet de copier les lignes de " lenr " directement dans la classe Ligne. Cette technique nous permet lors de la construction des <td> de ne pas avoir à connaître les champs, ils nous seront directement proposés.



Il vous est aussi possible à ce moment de modifier la classe ligne et d'effectuer une mise à jour de la ligne dans la base de donnée (zUpdate)
Le résultat de cette page Web doit ressembler à cela :

Ajouter un utilisateur

Nom	Prenom	Adresse	CP	Ville		
DIEUDONNE	Sebastienma	rue	68000	COLMAR	modifier	supprimer

Nous allons maintenant passer au code qui permet d'ajouter, modifier et supprimer une ligne.
Pour cela, nous créerons trois fichiers qui contiendront respectivement le code pour chacune de ces fonctions.

Pour l'ajouter,

```
<?
include("codegenerer.php");
include("config.php");
connect_open_db_demo_default();

$ligne_util = new tbl_utilisateur_Ligne();
$ligne_util->nom=$_GET['textbox_nom'];
$ligne_util->prenom=$_GET['textbox_prenom'];
$ligne_util->adresse=$_GET['textbox_adresse'];
$ligne_util->cp=$_GET['textbox_cp'];
$ligne_util->ville=$_GET['textbox_ville'];
$ligne_util-> zajouterid();

connect_close_db_demo();


Redirect("index.php");


?>
```

Nous effectuons d'abord les include du fichier généré (pour avoir les classes) et du fichier config (pour gérer les sessions). Ensuite, nous nous connectons, déclarons une nouvelle classe ligne, et chargeons cette classe avec les valeurs des variables de session. Nous rajoutons la ligne avec zajouterid et fermons ensuite la connexion. Puis nous retournons à la page index.

Pour le Modifier,

```
<?
include("codegenerer.php");
include("config.php");

connect_open_db_demo_default();

$ligne_util = new tbl_utilisateur_ligne();
$ligne_util->zload($_GET['valider']);
$ligne_util->nom=$_GET['textbox_nom'];
$ligne_util->prenom=$_GET['textbox_prenom'];
$ligne_util->adresse=$_GET['textbox_adresse'];
$ligne_util->cp=$_GET['textbox_cp'];
$ligne_util->ville=$_GET['textbox_ville'];
$ligne_util->zmodifier();

connect_close_db_demo();

Redirect("index.php");

?>
```

Nous faisons la même chose que pour l'ajout à une différence près, nous effectuons un chargement de la ligne avec un zLoad() de l'id.

Pour le supprimer,

```
<?
include("codegenerer.php");
include("config.php");

connect_open_db_demo_default();

$lsql="delete from ".zt_TBL_UTILISATEUR." where ".zc_TBL_UTILISATEUR_ID."='".$_GET['id']."'";

db2c_db_demo_execute($lsql);

connect_close_db_demo();

Redirect("index.php");

?>
```

Pour le supprimer, nous devons construire une requête SQL, mais nous utilisons les constantes générées par db2c. Toutes les constantes qui contiennent le nom des tables commencent par " zt_ " suivi du nom de la table, pour les champs par " zc_ " suivi du nom de la table puis du champ. En utilisant les constantes, vous n'êtes pas obligés de connaître les noms des champs, il vous suffit de taper le début de la table pour voir apparaître son nom, idem pour les champs, ce qui vous permet de sélectionner le champ plutôt que de le saisir en entier.
Nous allons maintenant utiliser ces formulaires pour créer une page web où nous allons saisir les champs.

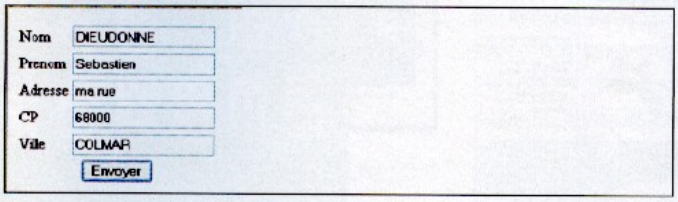
```
<? session_start();
include("codegenerer.php");

connect_open_db_demo_default();

if (isset($_GET['id']))
{
echo "<form name='form1 method='get' action='modifier.php'>";
}
else
echo "<form name='form1 method='get' action='ajouter.php'>";

if (isset($_GET['id']))
    {
        $ligne_util = new tbl_utilisateur_Ligne();
        $ligne_util->zLoad($_GET['id']);
    }

?>
<table border="0">
```

Nous allons inclure le code généré, nous connecter, et vérifier si l'id est initialisé (s'il est initialisé il contient un id de ligne, on est donc dans un cas de modification, dans le cas contraire on est dans un ajout). S'il est initialisé, on charge une classe Ligne.
Nous pourrons commencer à construire les champs. Je n'afficherai que la construction du champ nom, mais ils sont tous identiques, nous testerons si l'id contient une valeur, si oui, la classe a été chargée et on peut affecter une valeur dans le champ.

```
<td><td>Nom </td><td>
<?
    if (isset($_GET['id']))
        echo '<input  name="textbox_nom" type="text" id="textbox_nom" value="'. $ligne_util-
    else
        echo '<input name="textbox_nom" type="text" id="textbox_nom">';
?>
</td></tr>
```

```
</td></tr>

    <tr>
      <td colspan="2"><div align="center">
<?
    if (isset($_GET['id']))
    {
        echo " <input name='valider' type='hidden' id='valider' value='".$_GET['id']."'>";
    }
    else
    {
        echo " <input name='valider' type='hidden' id='valider' value='ajout_utilisateur'>";
    }


?>
<input type="submit" name="Submit" value="Envoyer">

</div></td>
      </tr>
    </table>
  </form>
<? connect_close_db_demo(); ?>
```

Enfin, nous ajoutons le bouton et la déconnexion, et au final, vous devez obtenir ceci :



Conclusion

Durant la phase de conception, nous avons une aide totale pour la modélisation, les tables et les champs, ce qui facilite le développement. Nous ne pouvons pas nous tromper dans le nommage des classes, les erreurs ne sont plus remontées lors de l'exécution mais bien avant, à la compilation (même dans l'environnement de développement si celui-ci vous l'indique). Le fait d'afficher une page web qui contient un appel vers une fonction qui n'existe pas fera afficher un message d'erreur. Pour ce qui est d'une requête SQL, le message ne s'affichera que lors de l'exécution. Pour ceux qui aimeraient accélérer encore le développement, ils peuvent utiliser le générateur de scripts (Menu Code+ -> Générateur de Script) il vous permettra de produire du code, suivant la structure de votre base de données, et s'adaptera ainsi a vos méthodes de développement.
Vous pourrez ainsi passer de ceci :

```
<DB2C.TABLE>

<?
include("codegenerer.php");
include("config.php");

[PHP_CONNEXION_OPEN_DEFAULT]();

$ligne_util = new [TABLE.NOM_CODE_LIGNE]();

  <DB2C.CHAMPS>
  $ligne_util->[CHAMPS.NOM_CODE]=$_GET['textbox_[CHAMPS.
NOM_CODE]'];
  </DB2C.CHAMPS>

</DB2C.TABLE>

$ligne_util-> zajouterid();

[PHP_CONNEXION_OPEN_DEFAULT]

connect_close_db_demo();

Redirect("index.php");

?>
```

à ce code :

```
<?
include("codegenerer.php");
include("config.php");

connect_open_db_demo_default();

$ligne_util = new tbl_utilisateur_Ligne();


  $ligne_util->id=$_GET['textbox_id'];

  $ligne_util->nom=$_GET['textbox_nom'];

  $ligne_util->prenom=$_GET['textbox_prenom'];

  $ligne_util->adresse=$_GET['textbox_adresse'];

  $ligne_util->ville=$_GET['textbox_ville'];

  $ligne_util->cp=$_GET['textbox_cp'];


$ligne_util-> zajouterid();

connect_open_db_demo_default

connect_close_db_demo();

Redirect("index.php");

?>
```

Pour télécharger l'outil : www.database2code.com

■ **Sébastien Dieudonné** - *Responsable Informatique - 3Ma Group - www.3magroup.com - s.dieudonne@3magroup.com*

Vista : un drôle de jeu...

Ouf ! Après cinq années de développement et d'annonces retardées, Vista est enfin là ! Belle interface, réorganisation des tâches, sécurité améliorée, assistance dans tous les coins, ok, c'est tout neuf. Mais que fait vraiment Vista pour les joueurs ?

Le noyau technique du nouvel environnement ludique, c'est évidemment le fameux DirectX 10 : une version rénovée de l'API, dont la dernière mouture remontait quand même à 2004. DX 10 vaut surtout par sa gestion étendue de la 3D, en interaction totale avec de nouvelles cartes graphiques, nVidia comme ATI, qui lui sont et seront dédiées. Concrètement, il en résulte une gestion des textures, polygones et autres shaders qui exploite sans retenue la mémoire interne de la carte, avec une meilleure gestion des ressources du ou des processeurs graphiques. En même temps, la moindre sollicitation du processeur central du PC laisse la place à des processus non-graphiques plus nombreux et complexes. Bref, de quoi rêver à des sommets ludiques jamais atteints, tel le fameux Crysis, pour l'instant seul fer de lance de la " révolution " DX10 – Vista : un jeu de tir dans un univers de science-fiction dont les premières images, hallucinantes de finesse et de réalisme, ne sont pas celles d'une séquence animée, mais du jeu animé en temps réel ! On devrait donc voir sortir dans les prochains mois d'autres titres aussi spectaculaires, bâtis autour du moteur de l'allemand CryTek, et techniquement indissociables de Vista. L'argument est-il suffisant pour faire changer de carte graphique et d'environnement Windows joueurs et développeurs ? Pas sûr, d'autant qu'évidemment, pour ne pas fâcher tout le monde, une version de DirectX 9 est intégrée à Vista, pleinement compatible avec les " vieux jeux "...

Le Games Explorer

Vista propose aussi un gestionnaire de jeux intégré à toutes ses versions, appelé Games Explorer. C'est l'équivalent, somme toute, de Mes Documents, reconverti à la sauce ludique : toute installation de jeu est automatiquement dirigée vers ce dossier. Une fenêtre spécifique propose une interface adaptée : jaquette de chaque jeu en guise d'icône, informations sur le genre, l'éditeur, la version, etc. En outre, le Games Explorer peut surveiller en permanence les patchs, mises à jour et add-on proposés en ligne et vous les signaler. Ou encore aider votre choix de jeu par des critères de recherche en connexion directe avec des sites spécialisés. Ou enfin établir un indice de performance pour chaque titre pour déterminer s'il tournera plus ou moins bien sur votre machine. D'accord, voilà qui peut être séduisant pour le joueur occasionnel... Mais pour le hardcore gamer, nul doute que cette assistance permanente, fortement tachée de marketing à la Microsoft, sera insupportable et systématiquement contournée ! Sans compter le probable espionnage en arrière-plan sur la validité des jeux d'origine...

Enfin, un dispositif de contrôle parental est disponible dans le Games Explorer. De quoi bloquer tel jeu, ou limiter les parties à certaines plages horaires... Les parents inquiets des interminables sessions de World of Warcraft vont se régaler ! Conflits en perspective...

Un label finalement ambigu

Dès le lancement de Vista, il apparaît donc une lourde contradiction dans sa gestion de l'univers des jeux : d'un côté, des performances réservées à DX 10 qui sont censées attirer les hardcore gamers, de l'autre des procédures d'assistance et de surveillance inadaptées, qui ne peuvent que les faire fuir... Le problème se résoudra sans doute simplement par l'apparition rapide d'un craqueur du style Anti-Spy, version Vista, apte à activer ou désactiver les services cachés de Windows !

Le dispositif ludico-marketing de Vista se complète par l'arrivée d'une nouvelle dénomination : " Games for Windows ". Un label pour l'instant purement Microsoft, déjà apposé sur Flight Simulator X ou le dernier Ages of Empires, mais qui entend remplacer le clas-

sique " jeu PC ". Des exigences de développement précises le régissent pour les éditeurs : une compatibilité parfaite avec les processeurs 64 bits comme avec le Games Explorer et la manette XBox, l'exploitation (versions Premium et Ultimate de Vista) des fonctions du Media Center, dont l'affichage en 16/10 sur le PC, ou sur un téléviseur à l'aide de la télécommande Windows Media Center. Ajoutez à cela une norme d'installation dite " simplifiée ", qui semble pour l'instant assez floue, et en tout cas mal adaptée aux MMORPG, pourtant grandes stars du moment...

Au bilan, il semble que Vista devrait mettre longtemps avant de conquérir le cœur des vrais joueurs. Quant aux développeurs, c'est surtout en matière de titres grand-public que ce nouvel environnement et sa tentative de normalisation peuvent leur offrir, à court terme, des perspectives intéressantes. Mais au prix de sévères efforts d'adaptation.

ACTUS

Pas glorieux peut-être, mais rentable

Le développement d'utilitaires grand-public comme " Mon Intérieur en 3D " ne vous intéresse pas et vous semble même indigne de votre génie ? Économiquement, vous avez totalement tort ! Car, à votre avis, qui est, selon le panel " Total Software Retail " de GFK France, le 1er vendeur de logiciels en France en 2006 ? Micro-Application ! Oui, Micro-Application qui, avec entre autres, des titres comme Impression, Habitation, ou Code de la Route, raffle 15% du marché ! Et a vendu, en 2006, près de 2,5 millions d'exemplaires, pour un chiffre d'affaires de 26 millions d'Euros... De plus, c'est un marché en explosion, avec une progression hallucinante de 42,75 % par rapport à 2005 !

N-Gage : distribution numérique

Nokia vient de lancer son nouveau canal de distribution numérique pour les jeux N-Gage. Le portail en ligne permet aussi bien l'essai de démos que l'achat de versions complètes. En vedette, évidemment, les hits comme Pathway to Glory, High Seize ou Payload. L'opération se déroule en deux temps : c'est sur son PC que l'utilisateur télécharge d'abord le jeu, pour en copier les fichiers sur une carte-mémoire. Il l'insère alors dans le lecteur externe de sa N-Gage et procède à l'installation. Evident, mais astucieux. www.n-gage.com/gameshop

Paysages et terrains 3D pour jeux réalistes

Bionatics vient de présenter à Imagina 2007 un nouveau logiciel de modélisation 3D de paysages, très impressionnant. Une des spécialités les plus étonnantes de LandSim 3D est sa capacité à interpréter des photos, issues de stallites ou non, pour en tirer des reconstitutions de terrain très précises. Il intègre et traite aussi aisément les données vectorielles importées de logiciels 2D et 3D. Enfin, il ajoute un éventail exhaustif d'outils de modélisation avec formes pré-installées, aussi bien en environnement " naturel " qu'urbain. L'ensemble, quoi qu'initialement destiné aux études et architectures d'environnement, se conforme aussi remarquablement au développement de terrains pour jeux réalistes, jeux de combat à la Counter Strike ou jeux de stratégie en temps réel. www.bionatics.com

Second Life en Open Source ?

Vous connaissez sans doute Second Life, cet univers virtuel mis en place par les californiens de Linden Lab, où certains candidats de nos élections présidentielles ont même maintenant leur QG promotionnel... La grande nouvelle, annoncée par Philip Rosedale, fondateur de Linden Lab, est qu'une partie de l'environnement de Second Life devient Open Source : " Nous plaçons désormais les développements du client entre les mains des résidents (du monde virtuel) et des développeurs. " En fait, une partie seulement ! C'est-à-dire, pour l'instant, l'application Viewer. Mais c'est elle qui gère la plupart des interactions entre les habitants du monde...

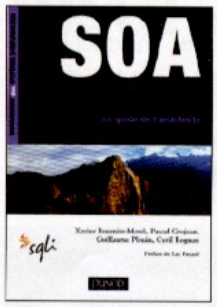
Linden Lab garde bien sûr le contrôle de l'infrastructure globale. D'autres " ouvertures " en open source se feront peu à peu : l'ambition affichée est de donner aux habitants de Second Life une meilleure maîtrise de leur environnement virtuel et, bien sûr, de développer ainsi l'esprit communautaire du lieu. Que de bonnes intentions ! Vous pouvez télécharger dès maintenant le code source de Viewer sur http://secondlife.com.

Plus d'outils sur AMD !

AMD, pour rester dans la course, vient d'annoncer un élargissement de la gamme de ses outils de développement sur ses processeurs graphiques. GPU PerfStudio, encore en bêta version, permet une analyse en temps réel du flux graphique, avec une amélioration de performance par la détection des " goulôts". GPU Shader Analyzer se spécialise, lui, dans le contrôle et la gestion optimale des effets shaders en pixel comme en vertex. Diverses versions des divers outils, plus ou moins bêta, sont téléchargeables dès maintenant. www.amd.com/us-en/Processors/DevelopWithAMD

PAR FRANÇOIS TONIC

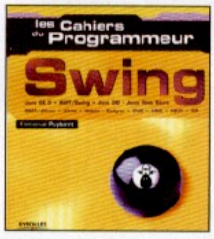
SOA



- **Difficulté : *****
- **Editeur : Dunod**
- **Auteur : collectif**
- **Prix : 29 €**

L'architecture logicielle a été particulièrement secouée par l'apparition des web services et surtout par les architectures dites orientées services ou SOA. Architecture flexible, la SOA demeure parfois obscure par ses concepts. Elle modifie l'approche applicative et la manière même de développer. Les auteurs tentent de la démystifier et de poser les fondations pour la comprendre en tentant de répondre aux : où, quand, comment, pourquoi. Ils démontent la couche de cette architecture, comment définir une plate-forme SOA. Faut-il privilégier l'approche composite ou les services web ? Comment s'organise un projet SOA ? Faut-il mettre en œuvre des framework, la notion de MDA ? Sans oublier quelques standards de base à connaître (UDDI, WSDL, etc.). Un ouvrage instructif même si la partie offre du marché est déjà partiellement obsolète à cause de la constante évolution de l'offre. À lire et à méditer ! Écrit par les experts SOA de SQLi.

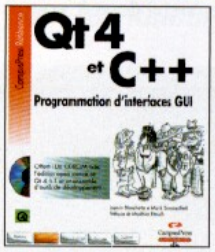
Les cahiers du programmeur : Swing



- **Difficulté : *****
- **Editeur : Eyrolles**
- **Auteur : Emmanuel Puybaret**
- **Prix : 39 €**

Swing est la librairie d'interface Java la plus complète et la plus connue, même si SWT d'Eclipse la concurrence de plus en plus. L'ouvrage n'a pas l'ambition d'être un guide exhaustif de Swing, il faudrait plusieurs milliers de pages. L'auteur se concentre sur l'utilité de Swing, l'environnement de développement, l'affichage 2D et 3D, les classes pour gérer le glisser-déposer, les composants graphiques... De très nombreux exemples de code ponctuent les explications. L'auteur a eu la bonne idée de mettre des références à Windows, GNU Linux et MacOS X. Par contre, dommage de se limiter à Eclipse. L'auteur, pour montrer la puissance et le développement Swing, s'appuie sur un projet d'une application 3D d'aménagement intérieur. On va ainsi de la conception UML au déploiement Java Web Start.

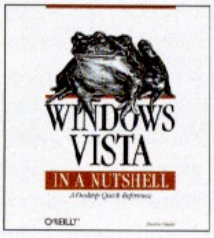
Qt 4 et C++ : programmation d'interface GUI



- **Difficulté : *******
- **Editeur : CampusPress**
- **Auteur : collectif**
- **Prix : 45 €**

La librairie C++ multi-plate-forme Qt de Trolltech est une référence en la matière. La version 4.1 renforce sa richesse fonctionnelle. Mais comment l'exploiter au mieux ? Cet ouvrage procure une plongée au cœur même de Qt. Les auteurs abordent trois niveaux : notions de base, intermédiaires, avancées. Cela va très loin : comment créer un plug-in, les notions d'embarqué, la gestion du réseau, Unicode, XML, le traitement des threads. Indispensable aux développeurs C++ ! Livré avec un CD-Rom contenant les sources et Qt 4.1.

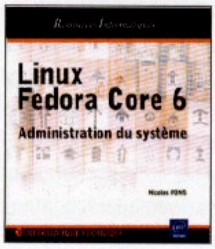
Windows Vista in a nutshell



- **Difficulté : *****
- **Editeur : O'Reilly**
- **Auteur : Preston Gralla**
- **Prix : 31 €**

Vous cherchez un ouvrage quasi exhaustif sur Windows Vista, allant à l'essentiel ? Cet ouvrage est fait pour vous. Cette collection O'Reilly est une référence en la matière. On y trouve la description des fonctions, le chemin pour y accéder, éventuellement la commande Prompt, des conseils. Un must ! En Anglais.

Linux Fedora Core 6



- **Difficulté : ****
- **Editeur : Eni**
- **Auteur : Nicolas Pons**
- **Prix : 27,14 €**

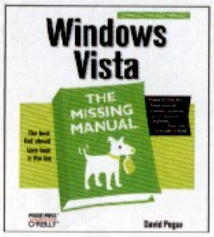
Sortie il y a peu, la distribution Fedora Core 6 est l'une des plus réputées et des plus complètes du marché. Ce livre intéressera les administrateurs et surtout l'utilisateur avancé. On trouve les classiques du genre : gestion de disques, des paquetages, des utilisateurs, des ressources système. L'auteur apporte de nombreux exemples de mise en place et de configuration. Plutôt clair et précis, idéal si vous voulez démarrer rapidement ou peaufiner Fedora Core. Seul regret : l'absence de la distribution sur CD ou DVD.

Ubuntu efficace

- **Difficulté : ***
- **Editeur : Eyrolles**
- **Auteur : Lionel Dricot**
- **Prix : 29,90 €**

Ubuntu est la distribution star du moment ! On débute par la phase de migration si vous partez de Windows. L'auteur brasse les thèmes traditionnels : installation, configuration, Internet, bureautique, le shell, etc. Dommage qu'il n'y ait pas de section virtualisation cela aurait été intéressant. Pour le reste, l'ouvrage est de bonne tenue et apporte un bon démarrage à cette distribution. Bonus: la distribution livrée sur CD !

Windows Vista : the missing manual



- **Difficulté : *****
- **Editeur : O'Reilly**
- **Auteur : David Pogue**
- **Prix : 31 €**

À peine Windows Vista disponible pour tout le monde, que l'utilisateur avancé ou débutant voudra en savoir plus sur son nouveau système... L'auteur décortique toutes les nouveautés de Vista et surtout selon les différentes éditions. On y trouvera les bases du bureau, du menu démarrer, l'explorateur, les gadgets, la sécurité, les applications, le matériel, etc. À chaque élément, on trouvera la présentation, son utilité, son utilisation et surtout des tips, des petits bonus pour mieux l'utiliser ! C'est complet et clairement écrit ! En Anglais.

Nouvelle rubrique livres

www.programmez.com

L'information à la carte

1 1 an ECO

Recevez le magazine chaque mois
et ***économisez 20 €***

11 ***Numéros***
Prix au numéro : 65,45 €

45 €
(Prix France métropolitaine)

-30%

2 1 an TOUT NUMERIQUE

Lisez chaque mois le magazine seul
au format PDF
(téléchargement)

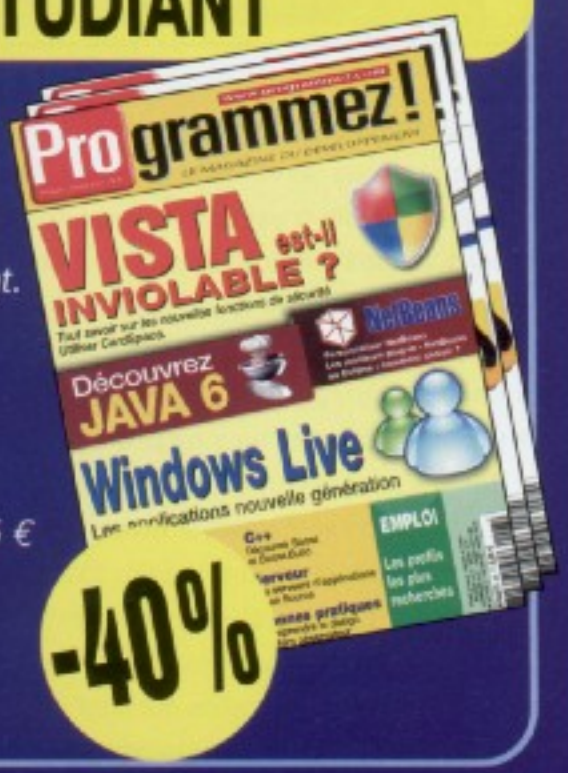
Inscription :
www.programmez.com

11 ***Numéros***

35 €
Tarif Monde entier

sur la base tarif France
-46%

3 1 an ETUDIANT



Vous devez justifier de votre statut d'étudiant.
Economisez 26 €

11 ***Numéros***
Prix au numéro : 65,45 €

39 €
(offre réservée France métropolitaine)

-40%

4 2 ans

Abonnez-vous pour 2 ans et recevez le livre « Windows Vista », installation, configuration et administration de Benoît Lanlard (ENI, 464 pages).

22 ***Numéros***
Prix au numéro : 130,90 €
+ livre 27 €

90 €
(offre réservée France métropolitaine dans la limite des stocks disponibles)

ECONOMISEZ 68€

OUI, je m'abonne ! ou abonnement en ligne : www.programmez.com

☐ **ABONNEMENT 1 an ECO** au prix de 45 € TTC. *Tarif France métropolitaine.*
Tarifs hors France métropolitaine : CEE et Suisse : 51,83 € - Algérie, Maroc, Tunisie : 55,95 € - Canada : 64,33 € - Tom : 79,61 € - Dom : 62,84 € - Autres : nous consulter

☐ **ABONNEMENT 1 an ETUDIANT (11 numéros)** au prix de 39 € TTC. ***Offre limitée à la France métropolitaine.***
Photocopie de la carte d'étudiant obligatoire

☐ **ABONNEMENT 2 ans + livre «Windows Vista» (22 numéros)** au prix de 90 € TTC. ***Offre limitée à la France métropolitaine.***

☐ M. ☐ Mme ☐ Mlle Entreprise : ..... Fonction : .....

Nom : ..... Prénom : .....

Adresse : .....

Code postal : ..... Ville : .....

Tél : ..... E-mail : .....

☐ Je joins mon règlement par chèque à l'ordre de Programmez !
☐ Je souhaite régler à réception de facture

A remplir et retourner sous enveloppe affranchie à :
Programmez ! - Service Abonnements - 22 rue René Boulanger - 75010 Paris.
abonnements.programmez@groupe-gli.com

Programmez!
LE MAGAZINE DU DÉVELOPPEMENT

Offre limitée, valable jusqu'au 31 mars 2007

PROG 95

Le renvoi du présent bulletin implique pour le souscripteur l'acceptation pleine et entière de toutes les conditions de vente de cette offre.

Conformément à la loi Informatique et Libertés du 06/01/78, vous disposez d'un droit d'accès et de rectification aux données vous concernant.

Par notre intermédiaire, vous pouvez être amené à recevoir des propositions d'autres sociétés ou associations.

Si vous ne le souhaitez pas, il vous suffit de nous écrire en nous précisant toutes vos coordonnées.