Programmez !
www.programmez.com
Mensuel - Décembre 2007 - N°103
LE MAGAZINE DU DÉVELOPPEMENT

Microsoft Visual Studio 2008

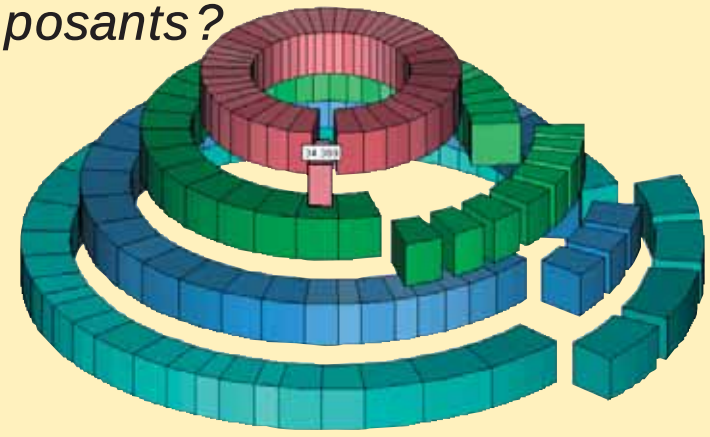
Découvrez les grandes nouveautés

Google s'attaque aux mobiles avec Android

Programmation par composants

Simplifiez-vous la vie !

Pourquoi utiliser des composants. Exemples concrets avec Spring, PHP et Java. Peut-on concilier UML et composants ?



Développement Multicore

Comment optimiser un code pour plusieurs processeurs

RIA
OpenLazlo : l'autre Flash

Java
Les exceptions en Java

Web
Intégrer Google Gears à Xulrunner
WebKit : créez votre navigateur !

MacOS X
Découvrir Core Animation

Fichier
Manipuler les microformats

Printed in France - Imprimé en France -
BELGIQUE 6,45 € - SUISSE 12 FS -
LUXEMBOURG 6,45 € - Canada 8,95 $ CAN
DOM Surf 6,90 € - TOM 940 XPF - MAROC 50 DH
M 04319 - 103 - F: 5,95 €

WINDEV WEBDEV WINDEV Mobile

PLATEFORME PROFESSIONNELLE
DE DÉVELOPPEMENT (AGL)
Windows, .Net, WebServices, Java, RAD

NOUVELLE VERSION 12

JUSQU'AU
21 DÉCEMBRE 2007

ACHETEZ WINDEV ET RECEVEZ 1 PC POUR «1 EURO DE PLUS»

Les détails sont sur www.pcsoft.fr
ou appelez-nous.

Parmi les 500 nouveautés:

- Accès natif à **SAP**
- Fond de page PDF
- Gestion des exigences
- Débogage à distance
- Compilation «JITc»
- Fonctions d'administration réseau
- Réplication automatique
- Sauvegarde à chaud
- Héritage de modèle
- Nouveau RAD
- 100 Nouvelles fonctions **Java**
- 39 Nouvelles fonctions **PHP**
- 50 Nouvelles fonctions **Linux**
- Débogueur PHP
- Web 2.0/Ajax

500 NOUVEAUTÉS

www.pcsoft.fr

Demandez le dossier technique gratuit (en couleurs, en français), accompagné de 112 pages de témoignages et d'un DVD. Version Express Gratuite.

Tél Province **04.67.032.032** Tél Paris **01.48.01.48.88** **info@pcsoft.fr**

Fournisseur Officiel de la Préparation Olympique

XIII

SOMMAIRE N°103

Donnez votre avis sur ce numéro
www.programmez.com/magazine_satisfaction.php

CD-Rom 103 PROGRAMMEZ !

MySQL 5.1.22
Système de gestion de base de données relationnelles, Windows.

MySQL 6.0.2 alpha
Système de gestion de base de données relationnelles, Windows, Linux.

MySQL connector .net 5.1.3
Permet de développer des applications .Net nécessitant des connexions à des bases de données MySQL, Windows.

Infragistics NetAdvantage for .Net 2007 vol.3
NetAdvantage for .NET est une suite de contrôles, composants et outils pour la plate-forme .Net permettant aux développeurs de réaliser des applications et interfaces utilisateur, Windows. Version complète 20 jours.

Pervasive PSQL v10
Solution complète de gestion des données et de développement d'applications stratégiques. version complète, 30 jours- Windows.

SweetDev Ria
Allez au-delà d'Ajax sans vous soucier de Javascript ! La bibliothèque de tags Ajax permet de créer rapidement vos applications ou de faire évoluer vos sites JSP / Struts.

.Net framework 3.5 bêta 2
NET Framework comprend tout ce dont vous avez besoin pour exécuter les applications développées à l'aide du .NET Framework, Windows.

Phoenix SDK
Nom de code du framework d'optimisation et d'analyse de Microsoft, Windows.

_LE JOURNAL DE NOTRE INFRASTRUCTURE

_69e JOUR : Tout ce que nous voulons, c'est une information bien précise. Gilles la tenait presque, mais il a fait un faux mouvement. Comment trouver des informations commerciales fiables dans des montagnes de données contradictoires qui s'accumulent sans fin ?

_Gilles vient d'attraper un panda en peluche !

_71e JOUR : J'ai trouvé ! Grâce aux solutions IBM de gestion de l'information, nous rendons l'information plus claire et partageons une définition commune des données sources pour gagner en cohérence et en fiabilité. Je peux créer une seule et unique banque d'informations à partir des différents systèmes sources. Et ainsi tout le monde peut prendre de meilleures décisions.

_Ouf... On était à court de jetons.

IBM

Information Management

Téléchargez notre livre blanc sur la gestion des données de référence
IBM.COM/**TAKEBACKCONTROL**/ACCURATE/FR

L'info permanente

Programmez!
LE MAGAZINE DU DÉVELOPPEMENT

Rédaction : redaction@programmez.com
Directeur de la Rédaction : Jean Kaminsky
Rédacteur en Chef : François Tonic
Ont collaboré : F. Mazué, S. Leroux, G. Renard, F. Dewasmes, D. Leroux, E. Chenu, T. Snyder, T. Templier, J. Pauli, P. Desfray, J. Joubert, L. Guillois, C.Seiwald, F. Bordage, M. Alexandre, A. Ricotta, E. Vautherin, J. Chable, J.M. Maman.
Dessin : Jissey
Maquette : AJE Conseils
Publicité : Régie publicitaire, K-Now sarl
Pour la publicité uniquement : Tél. : 01 41 77 16 03
coordination@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
Ce numéro comporte un CD Rom.

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 : 30 € (Monde Entier) souscription en ligne.

Édito

Designer + Développeur = ?

On semble découvrir les bienfaits de l'interface, de l'ergonomie grâce au web 2 et surtout aux technologies Flex, AIR, Silverlight, JavaFX, Xul, etc. L'heure est à une collaboration étroite entre designer et développeur ! Ah bon, une bonne interface peut aider l'utilisateur…? À croire que l'on était passé à côté de ce principe depuis 15 ans !

J'ai créé mes premières interfaces utilisateurs à cette époque, sous DOS, Windows 3.x, MacOS et même Atari ST. J'ai toujours été sensible à l'ergonomie. Dans la SSII où j'étais employé, on me disait toujours : « penses utilisateur final, penses ergonomie, penses utilisation au quotidien ». La contrainte était à la fois simple et compliquée : un écran 800 * 600 - mais le plus souvent c'était le bon vieux 630 * 480, une liste de champs de saisies, de menus, de listes, de tableaux avec affichage ou masquage à la volée d'options selon les choix de l'utilisateur.

Je passais des heures, voire parfois des jours et des nuits entières à concevoir une interface avec les développeurs, pour trouver le bon agencement, la bonne présentation et la tester. Bien sûr nos premiers outils étaient limités, il était déjà compliqué de gérer le multifenêtrage sous MS-DOS… High Screen fut un réel progrès dans notre quête ergonomique. Mais le saut en avant s'effectua avec Visual Basic. Bien qu'habitué à la programmation visuelle avec HyperCard, j'ai été immédiatement conquis par VB 1.0, et même VB sur DOS !

Le développeur, s'il n'est pas sensibilisé, ne peut comprendre l'ergonomie d'interface. Longtemps, l'interface a supplanté l'ergonomie. Or l'un ne va pas sans l'autre. Une interface n'est pas une fin en soi. C'est pourquoi le designer prend une place de plus en plus importante encouragé par les applications RIA / RDA qui nécessitent un travail collaboratif entre designer et développeur.

Dans un logiciel, qu'on le veuille ou non, l'interface, sa présentation, son ergonomie priment de plus en plus sur son contenu. Un des effets « pervers » des systèmes modernes (Vista ou MacOS X et les dernières distributions Linux intégrant la 3D ou des effets à la Vista / Apple) est que l'utilisateur attend désormais le même résultat dans les applications. Il veut des gadgets sur son bureau. Il surfe sur des applications riches avec des animations, une interface dynamique avec des effets partout : il le veut pour son site web, son application web !

Aujourd'hui, nous ne sommes qu'au début de cette fusion intime entre le développeur et le designer. La disponibilité d'outils collaboratifs, l'apparition de l'intégration faisant le lien entre les deux, vont dans ce sens. Le développeur a-t-il le choix ?

François Tonic - ftonic@programmez.com

Souriez avec Jissey
www.programmez.com
chaque semaine, un gif animé

PROCHAIN NUMERO

N°104 - Janvier - Parution : 28 décembre 2007

Construire et déployer les applications

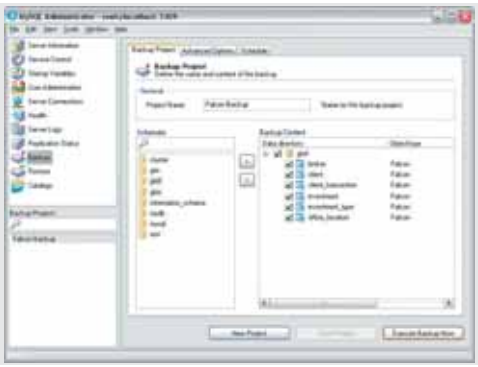
Comment gérer et automatiser les build de ses projets ? Les outils et les techniques

Gestion du cycle de vie

Pas de projets sans ALM ! Les techniques, les bonnes pratiques, les outils disponibles

SGBD

MySQL sort l'artillerie lourde



Le leader des bases open source, MySQL, sort en cette fin d'année la version 5.1 de son SGBD (inclus sur le CD du numéro). Cette version contient plusieurs nouveautés très attendues par les utilisateurs et surtout les entreprises : le partitionnement (indispensable pour segmenter une grosse base en plusieurs bases), la réplication par ligne (ou row based) et la réplication mixte, ainsi que la réplication de cluster MySQL. Autre nouveauté, l'introduction d'un support basique de Xpath. On bénéficie aussi d'une programmation de tâche et d'événement. Cela va permettre de réaliser des actions sur la base de données. À noter la disponibilité depuis peu d'un plug-in MySQL pour Visual Studio. Le support de XML est présent, à un niveau basique mais l'éditeur ne sait pas exactement ce qu'il adviendra de XML dans MySQL. Les responsables attendent la réaction des utilisateurs.

Mais cette 5.1 n'est qu'une étape vers la 6.0 qui doit apparaître fin 2008. Actuellement, seul le moteur de stockage Falcon est disponible. La première pré-version globale de la v6 sera disponible peut-être courant décembre, au plus tard en janvier prochain. Cette version doit consolider les fonctions présentes dans la 5 et 5.1 et notamment dans la qualité et l'intégration des données, la sécurité et les moteurs de stockage. La 6.1 rentrera en phase bêta vers la mi-2008 pour une disponibilité dans le courant de l'année. La v7, dont les premières spécifications circulent déjà, aura pour axe la sécurité avec données, la BI. La disponibilité n'est pas attendue avant fin 2009. Quant à l'ajout de code Google dans MySQL, l'éditeur se montre plus prudent sur l'intégration, Si Google a effectivement proposé des améliorations, c'est aux équipes de MySQL de les valider. Désormais aucune date ou version n'est précisée.

SGBD

IBM lance DB2 9.5

IBM propose depuis fin octobre la version 9.5 de DB2.Il s'agit d'une consolidation de la DB2 9.0. On remarquera tout de même que XML est toujours mieux intégré (notamment sur la mise à jour des données XML, le requêtage ou la transformation XSLT). La gestion des ressources constitue un autre point fort de cette version et tout particulièrement la gestion de la mémoire. On notera que l'édition Linux est maintenant multithread. Sur le partitionnement, DB2 9.5 redistribue automatiquement les données quand on modifie la taille des bases partitionnées. Côté développement, on a Data Studio Developer Workbench (toujours basé sur Eclipse). Son originalité est qu'il permet au développeur aussi bien qu'au DBA de développer des applications classiques. Les premières réactions face à l'outil sont plutôt bonnes. Notons que la version communauté, édition gratuite de DB2, évolue elle aussi en version 9.5. Le futur lointain s'appellera DB2 Cobra (DB2 10.0), prévue pour 2009.

INTÉGRATION

Magic Software enrichit son offre

L'éditeur continue à faire évoluer ses solutions de développement et d'intégration : iBolt et eDeveloper. Ce dernier est actuellement en version 10.1 SP2. Cette version apporte le support Windows Vista, la disponibilité d'un debugging distant, le support des web services et des headers SOAP. On peut avec eDeveloper 10.x faire des applications fonctionnant sur un client riche de type RDA, ou RIA, le tout dans un

modèle Java Web Start. On déploie alors une infrastructure logicielle (serveur + client). Magic a choisi de s'appuyer sur l'interface SWT et non Swing. Sur l'outil d'intégration, iBolt, l'éditeur travaille sur des services packs de la version 2.5 (qui s'appuie sur eDeveloper 9.4), la v3 est attendue l'an prochain et s'appuie sur eDeveloper 10. L'architecture de l'outil changera peu, hormis sur le support du XML et des métadonnées.

OUTIL

Rogue Wave publie la V10 de SourcePro C++

L'éditeur Rogue Wave Software publie une nouvelle release de ses bibliothèques multi-plates-formes : SourcePro C++. Assurant une portabilité entre Windows, Linux et Unix, SourcePro C++ comporte des modules pour : collections, string processing, multithread, internationalisation, protocoles réseau, accès base de données indépendant de la BDD, calculs mathématiques. SourcePro V10 améliore notamment la gestion des BDD, et gère de nouvelles plates-formes, dont Vista 32 et 64 bits.

OUTIL

IBM tente de contrer les hackers

L'éditeur a dévoilé un nouveau logiciel ayant pour objectif de protéger les applications web (d'entreprise) contre les attaques. Cette annonce fait suite au rachat par IBM de WatchFire avec le lancement de Rational AppScan. Les sites dits Web 2, et notamment Ajax, causent de réels soucis de sécurité et d'intrusions car la sécurité est souvent oubliée. AppScan a pour but d'identifier les vulnérabilités d'une application et de générer le rapport lié. L'outil propose les meilleures pratiques à mettre en œuvre selon la faille, la vulnérabilité. AppScan est capable de scanner les applications Ajax et Flash.

Altova® DatabaseSpy™ 2008 – The unique multi-database query and design tool.

ALTOVA® databasespy™ 2008

Finis les outils d'interrogation contraignants!

Passez à DatabaseSpy™ 2008 et gérez toutes vos bases de données à partir d'une interface agréable d'utilisation.

Nouveautés dans la version 2008:

- Création et édition de vues de bases de données
- Création et édition de procédures stockées
- Edition du contenu de bases de données interactives
- Autocomplètement SQL avancé

Altova® DatabaseSpy 2008, conçu par les créateurs de XMLSpy®, est un outil multi-bases de données de gestion et de modélisation de données. Véritablement exceptionnel, DatabaseSpy simplifie la gestion des données en offrant des fonctionnalités logiques et avancées pour une fraction du prix des solutions à base de données unique. Rédigez des requêtes précises en toute confiance et obtenez des résultats clairs et facilement négociables. Créez et modifiez des structures de bases de données visuellement en déposant des tableaux dans le volet de conception et en établissant des liens de relations entre eux. Vous pourrez même copier des structures existantes dans plusieurs bases de données de types différents. Gérez vos données autrement!

Téléchargez DatabaseSpy 2008 dès aujourd'hui:
www.altova.com

DatabaseSpy est également disponible au sein de MissionKit™, l'offre groupée Altova déjà primée.

ALTOVA®

REPORTING

Crystal Reports cuvée 2008

Business Object a sorti début novembre la version 2008 de son outil Crystal Reports. Une des nouveautés majeures est la compatibilité avec Adobe Flex. Il est maintenant possible d'embarquer des rapports faits sous Crystal Reports dans son application SWF, Flex Builder servant toujours à créer l'application et l'interface ainsi que l'accès au rapport et son intégration par un web service. On dispose aussi d'un panneau de configuration pour accéder aux valeurs sans avoir besoin de rafraîchir. D'autre part, l'aperçu du report se veut toujours plus flexible avec de nouvelles options pour les filtres et le formatage des données. Parmi les autres nouveautés, on peut retenir : génération de codes à barres, un design amélioré, une gamme de produits simplifiée, le support du multilangue plus souple, un pilote pour SalesForce, un nouveau SDK Server pour .Net. Sur le support de Java, on dispose d'un composant spécifique « Crystal Reports for Eclipse ». L'éditeur espère améliorer la prise en compte de Java et Eclipse dans quelques mois.

IDE

OpenKomodo : vers de nouveaux IDE libres ?

Voici un nouvel IDE initié par ActiveState : OpenKomodo. Open Source, l'outil est multi-langage et multi-plate-forme. On peut y développer en JavaScript, Ruby, Python, etc. Il vise essentiellement le développement avec les langages dynamiques. OpenKomodo est extensible et sous licence MPL, GPL et LGPL. Actuellement en version alpha, il fonctionne sous Linux, Windows et MacOS X. Il utilise le framework Mozilla et XUL pour son interface. Son approche peut déconcerter lors des premières utilisations. L'organisation de l'espace de travail ne ressemble pas à ce que l'on trouve dans d'autres IDE. Il fournit de nombreuses fonctions comme l'intégration avec firebug. OpenKomodo a pour but de stimuler la communauté en offrant une plate-forme d'IDE libre sur laquelle d'autres développeurs pourront développer leurs propres environnements.

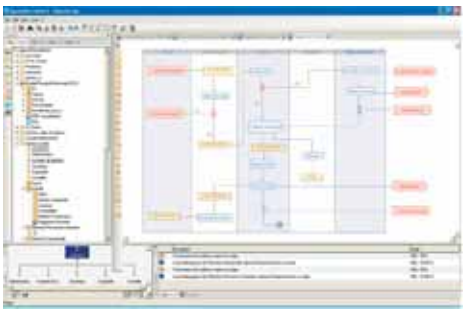
IDE

NetBeans 6.0 : dernière ligne droite !

La 6e version majeure doit être disponible tout début décembre. NetBeans 6 se pose plus que jamais en concurrent d'Eclipse avec l'apparition des langages dynamiques, notamment le support de Ruby, Jruby et Rails (éditeur, gestion de projet, debugger). On bénéficie d'un tout nouveau module d'installation et d'un centre de mise à jour amélioré. Le constructeur d'interface (Matisse) a été amélioré avec le support des applications Swing Database (JSR 295 et des API de persistence) ainsi que le framework Swing application. Le profiling a été complété et amélioré avec l'apparition de Jmeter. Le développement mobile bénéficie d'une remise à niveau avec, principalement, la présence d'un environnement de développement de jeux et un designer de GUI tout neuf. SOA et UML n'ont pas été oubliés, avec un renforcement des outils XML pour les web services. Pour les développeurs MacOS X, la partie C et C++ est désormais disponible. On notera aussi des nouveautés sur les API plates-formes. Sur la roadmap prévisionnelle, les développeurs NetBeans ont débuté le travail sur la v7 depuis quelques semaines. La version finale est attendue pour fin novembre 2008 ! Il est à prévoir des mises à jours 6.x courant 2008.

MODÉLISATION

Objecteering 6.1 disponible



L'éditeur Objecteering a sorti la version 6.1 de son environnement de modélisation UML 2. Il supporte désormais BPMN, SysML et une nouvelle API Java de paramétrage MDA. BPMN permet de modéliser des processus métiers, SysML se destine aux systèmes techniques. Objecteering 6.1 apporte plus de 280 contrôles de cohérence en temps réel afin d'assurer une vérification en continu de la qualité des modèles, ce qui garantit aussi la qualité des livrables issus des transformations et générations depuis les modèles (PIM, PSM, code, base de données, frameworks, chaîne de production, documentation, etc.). Côté langage, l'outil supporte toujours C#, C++, Java, SQL, CORBA, Fortran. Objecteering 6.1 gère la traçabilité, fournit des générations documentaires paramétrables pour documenter les exigences, les modèles et le code produit. L'intégration de Objecteering 6.1 avec la plate-forme Eclipse en fait un outil de développement puissant et complet pour Java, pareillement sous Visual Studio.

FRAMEWORK

Spring 2.5 arrive

Les dernières pré-versions de Spring 2.5 sont disponibles depuis la mi-novembre. Cette version doit apporter une meilleure flexibilité des modèles de configuration Java 1.4 et 5, avec comme focus particulier, le support de Java 5 tel que les annotations. Le développeur bénéficiera de nouveaux contrôleurs web ou encore le support de Junit 3.8, 4.4 et de TestNG. On notera aussi un support étendu d'AspectJ pour la programmation par aspect. Pour la migration entre Spring 2 et 2.5, la tâche devrait être simple (selon les développeurs).

Agenda

NOVEMBRE

Le 10 décembre, auditorium de la cité des sciences, Paris La Villette
Paris on Rails, édition 2007
Pourquoi Rails révolutionne le développement web ?
http://paris.onrails.info/

Les 11 et 12 décembre au CNIT, Paris La Défense
La 4ème Convention Dématérialiser
"Les nouvelles frontières de la dématérialisation pour les entreprises et les collectivités"
www.dematerialiser.com

Le 12 décembre 2007 de 8h30 à 16h00, Club Med World, Bercy Village 75012 Paris. Sybase organise un forum sur le thème **des nouveaux enjeux du Data Management**
http://dm.sybase.fr/?elqPURLPage=115#

Le 14 décembre, Paris VII[e], Rue de l'Université, DotNet User Groupe organise une **après-midi sur VS 2008 et .Net 3.5** dans les locaux de Microsoft France. www.dugfr.org

JANVIER

Le dimanche 20 janvier 2008
Salon Studyrama des Métiers de l'Informatique. http://www.studyrama.com/salons_details.php3?id_article=14157

29, 30 et 31 janvier 2008 CNIT – Paris-La Défense. **Solutions Linux / Open Source 2008** - www.linuxsolutions.fr

ETRANGER

Du lundi 10 décembre 2007 au vendredi 14 décembre 2007, Anvers Belgique. **JavaPolis 2007**. Le rendez-vous Java incontournable en Europe
http://www.javapolis.com/

Mozilla entre RDA et la prochaine génération de la plate-forme

Il y a quelques semaines l'annonce de Prism, un projet sorti des laboratoires de développement de l'éditeur, avait jeté un petit trouble. Tristan Nitot confiait quelques heures plus tard à Programmez ! : " c'est un projet de laboratoire avant tout ! ".

Prism permet d'utiliser des applications web sur son bureau via un runtime, un peu comme Adobe AIR. Il repose sur XULrunner. On sort du navigateur pour travailler sur son bureau, sans créer de nouvelles applications de type RDA ou RIA. Dans Prism, on indique l'adresse du site de l'application web pour pouvoir l'utiliser. On peut ainsi utiliser des applications comme Gmail ou Facebook directement sur son bureau, au lieu d'avoir son navigateur... De plus, on dispose d'un système de raccourcis. Ainsi, avec un simple double-clic, Prism lance depuis le bureau une fenêtre avec le site correspondant. L'installation d'un runtime est nécessaire. Celui-ci fonctionne sous Windows, Linux et MacOS X. Pour Mozilla, l'objectif est de pousser en avant un web ouvert basé sur les standards, loin des plates-formes propriétaires comme Adobe ou Microsoft, même si, là aussi, un mouvement d'ouverture se réalise. Le projet Prism, anciennement connu sous le nom de code WebRunner doit donc permettre aux utilisateurs de profiter des sites web, applications web sur le desktop même si actuellement, aucune personnalisation n'est possible. La question de rajouter des fonctions de gestion locale (ex. : accès aux fichiers locaux) est en cours. Il faudrait alors modifier les applications pour qu'ils puissent fonctionner pleinement sous Prism.

Vers Mozilla 2 !

L'objectif du projet Mozilla, est une réécriture afin de réorganiser le code et l'architecture, une optimisation complète du moteur et préparer aussi la déclinaison mobile de Firefox. À terme, il s'agit d'en faciliter la maintenance.

Sur Mozilla 2, parmi les modifications profondes, on notera la prise en compte d'ECMAScript 4, une orientation JIT pour la machine virtuelle JavaScript. Sur les API, il aura des ajouts d'API et certaines seront externalisées. XPCom ne sera pas jeté entièrement. Le projet Tamarin (moteur ActionScript) débutera son intégration. Le graphisme sera un focus important avec les prototypes video, SVG, l'accélération graphique.

On devrait voir les premières versions de Mozilla 2 après la disponibilité de Firefox 3. La bêta qui a déjà plusieurs mois de retard devrait arriver très rapidement (si ce n'est pas déjà le cas quand vous lirez ces lignes). La version 3.5 devrait arriver vers mi-2008, la version 4.0 du navigateur, début 2009.

Android : Google lance l'offensive de l'open mobile !

Mené par Google et soutenu par l'alliance OpenHandset, Android propose une plate-forme alternative ouverte et libre. Le tout premier SDK est disponible depuis le 12 novembre dernier.

Android est le nom d'une société rachetée par Google il y a deux ans, travaillant activement sur la mobilité open source et Linux. Google n'est pas parti seul à l'aventure mais avec une trentaine de partenaires (éditeurs et constructeurs) tels que Wind River, HTC, T-Mobile, Intel, eBay. On remarquera cependant l'absence des principaux acteurs de la mobilité tels que Nokia, Symbian et des gros opérateurs mondiaux comme Orange, Vodaphone. Android se veut : ouvert, proposer une égalité entre les applications, un modèle de développement simplifié et repousser les limites de l'intégration et de la convergence.

Architecture d'Android

Grossièrement, comme le montre le schéma, Android repose sur 5 couches :

- couche système avec le noyau Linux (v2.6) : Android repose avant tout sur un Linux intégrant les fonctions de base (les pilotes matériels pour USB, réseaux, caméra, wifi, etc.)
- librairies : disponibilité des différentes librairies techniques et fonctionnelles (OpenGL, SSL, WebKit, données avec SQLLite)
- runtime Android : comme pour .Net et Java, les applications Android reposent sur un runtime comprenant des librairies et une machine virtuelle (Dalvik VM)
- frameworks applicatifs : fournit les accès aux API et librairies d'Android pour les développeurs.
- Applications : Android dispose d'applications de base puis les applications tierces que l'on installe.

Le SDK Android

Donc depuis la mi-novembre, on dispose d'une version préliminaire d'Android. Il est disponible sous Linux, Windows et MacOS X (uniquement Intel). Il faut disposer d'un JDK 5 ou 6 et d'Eclipse (3.2 ou 3.3.x) pour développer et Ant. On dispose d'un plug-in spécifique Android pour Eclipse, Android Development Tools (ADT) que l'on télécharge directement à partir du panneau de mise à jour / installation d'Eclipse. On trouve après téléchargement un nouveau projet : Android. Le SDK est livré avec la documentation adéquate, un émulateur et une archive Android.jar. On peut créer, tester, debugger, compiler des applications Android. Il est possible de développer en code natif C/C++.

Structure des applications

Une application Android (quand on crée un nouveau projet) se compose de différents " blocks " : activity, Intent Receiver, Service et Content Provider. Toutes les applications n'ont pas besoin de tout cela. Un des éléments majeurs à posséder : AndroidManifest.xml. Il s'agit d'un fichier XML déclarant les composants de l'application, les fonctions et les ressources nécessaires.

Activity

Il s'agit basiquement de l'écran de son application. Une activity est implémentée dans une classe étendant la classe Activity. Une application comporte plusieurs écrans, donc à chaque écran il faut disposer d'une activity…

Intent et Intent Filters

Permet d'aller d'écran en écran. Intent fournit alors ce dont l'application à besoin pour changer d'écran comme donnée, comme action.

Intent Receiver

Permet d'exécuter une action dans une application provenant d'un événement extérieur comme par exemple une notification, une sonnerie téléphonique. On utilisera alors la gestion de notification pour avertir l'utilisateur.

Service

Un service est basiquement un code ayant une longue vie et fonctionnant sans interface, par exemple play list du lecteur multimedia. On communique avec le service (actif) via l'interface qu'il expose.

Content Provider

Sert à stocker les données des applications dans des fichiers, une base SQLLite ou tout autre mécanisme prévu à cet effet. La plupart du temps, les applications Android fonctionneront dans des process Linux (nous sommes dans un environnement Linux). Le process est créé par l'application s'exécutant quand l'application en a besoin. La gestion des process n'est pas directement gérée par l'application mais par le système lui-même. Et donc le développeur devra comprendre comment les blocks Android interagissent d'eux-mêmes avec les process et la gestion de ceux-ci.

Les outils disponibles

Le SDK livre en standard plusieurs outils :

- un émulateur pour tester et debugger ses applications
- un Dalvik Debug Monitor Server : moniteur de debug pour la machine virtuelle
- Android Debug Bridge
- Android Asset Packaging tool : pour créer des fichiers apk contenant les binaires et ressources de l'application à déployer
- Android Interface Description Language : pour générer le code de l'interface interprocess
- SQLlite : base de données utilisée par défaut par Android
- Traceview : outil de trace et d'analyse
- Mksdcard : pour créer des images disques utilisables dans l'émulateur pour simuler la présence d'une carte de stockage
- ActivityCreator : pour utiliser la génération Ant.

Les premiers téléphones Android sont planifiés pour la mi-2008. D'ici là, de nouvelles versions du SDK seront disponibles en téléchargement.

Site : www.android.com

François Tonic

NETFECTIVE
Creative Technology

Editeur du logiciel BLU AGE™ et propriétaire de la marque Opteams™

Le groupe NETFECTIVE TECHNOLOGY, spécialiste des technologies Objets, UML® et MDA®, propose un catalogue complet de formations et de séminaires.
La totalité des sessions proposées sont disponibles en mode inter et intra entreprise.
Retrouvez-nous dans nos locaux à Suresnes (92) ou Bordeaux (33).

Plus d'informations sur www.netfective.com
ou tout simplement contactez-nous au 01 56 05 88 00 / 01 56 05 60 91.

Formations Concepts Objet et UML

AOO-1j **Approche Orientée Objet** (1 jour)
3 décembre 2007; 7 janvier 2008; 4 février 2008; 7 mai 2008; 9 juin 2008; 29 septembre 2008...

AC-UML-4j **Analyse et conception avec UML 2.x, OCL 2.x & MDD** (4 jours)
4 décembre 2007; 8 janvier 2008; 5 février 2008; 13 mai 2008; 10 juin 2008; 30 septembre 2008...

MD-3j **Application d'UML 2.x avec MagicDraw™** (3 jours)
14 janvier 2008; 11 février 2008; 16 juin 2008; 13 octobre 2008...

Formations Java, Java EE

JAVA-4j **Programmation avec le langage JAVA** (4 jours)
28 janvier 2008; 25 mars 2008; 1er décembre 2008...

J2EE-4j **Développement d'applications web pour la plateforme Java EE** (4 jours)
26 février 2008; 21 avril 2008; 15 décembre 2008...

STRUTS-3j **Développement d'applications web avec Struts** (4 jours)
3 mars 2008; 28 mai 2008; 15 septembre 2008...

JSF-3j **Développement d'applications web avec JSF** (3 jours)
6 mars 2008; 2 juin 2008; 18 septembre 2008...

HIBER-3j **Gestion de la persistance avec Hibernate** (3 jours)
11 mars 2008; 30 juin 2008; 23 septembre 2008...

SPRING-3j **Développement d'applications avec Spring** (3 jours)
17 décembre 2007; 17 mars 2008; 15 juillet 2008; 20 octobre 2008...

Formation XML

XML-CM-3j **Technologies XML : Conception et mise en oeuvre** (3 jours)
28 avril 2008, 1er septembre 2008...

Formations BLU AGE™

BLU-INT-1j **Présentation synthétique des outils & méthodes MDD™ de BLU AGE™** (1 jour)
19 décembre 2007; 18 janvier 2008; 15 février 2008; 21 mars 2008; 19 mai 2008; 20 juin 2008; 4 juillet 2008; 5 et 26 septembre 2008; 23 octobre 2008...

BLU-MDD-5j **Une approche pragmatique du MDA : MDD™ avec BLU AGE™** (5 jours)
21 janvier 2008; 18 février 2008; 31 mars 2008; 21 mai 2008; 23 juin 2008; 8 septembre 2008; 6 octobre 2008; 24 novembre 2008...

BLU-SFC-5j **BLU AGE™ Software factory configuration** (5 jours)
7 avril 2008; 21 juillet 2008...

BLU-EMD-5j **EMDD avec BLU AGE™ - Comment construire des applications complexes avec BLU AGE™** (5 jours)
10 décembre 2007; 14 avril 2008; 7 juillet 2008; 25 août 2008; 17 novembre 2008; 8 décembre 2008...

Formations 2007/2008

BLU AGE™
J2EE .NET APPLICATION GENERATOR

BLU AGE™, le logiciel innovant issu du département de R&D de NETFECTIVE TECHNOLOGY, permet la génération automatique d'applications Java EE et/ou .NET sans recours aux techniques et ressources de développement.

'You design, We generate'
'Vous dessinez, Nous générons'

Découvrez BLU AGE™
Démonstration en ligne de génération d'application.
Agenda : 4 & 13 décembre (français), 11 décembre (anglais).

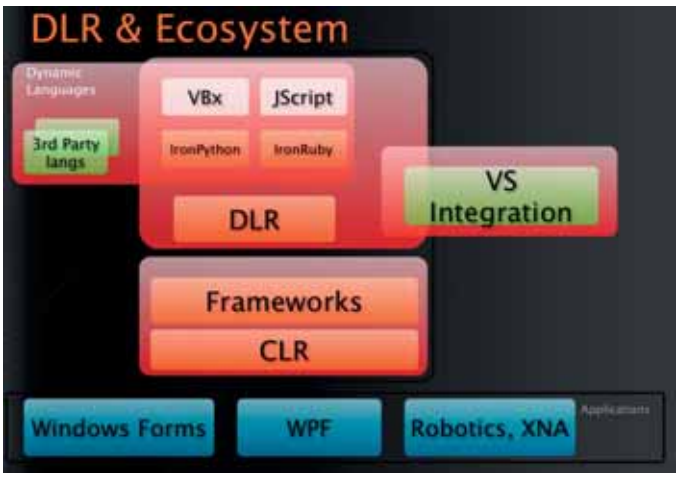
Inscrivez-vous gratuitement aux prochains webinars sur www.bluage.com.

TechEd 2007 : *Visual Studio 2008 en cadeau !*

Début novembre s'est déroulée la traditionnelle conférence développeur Microsoft européenne, à Barcelone, la TechEd. Plus de 4000 développeurs se sont pressés dans les salles durant une semaine pour apprendre, écouter, toucher les nouveaux produits. Plus intéressante que l'édition précédente, TechEd 2007 a créé une véritable surprise avec l'annonce de la sortie de Visual Studio 2008 et du framework .Net 3.5, ainsi que la possibilité de découvrir de nombreux projets disponibles courant 2008…

Durant la session inaugurale, l'éditeur a rappelé l'importance des outils de développeurs : Visual Studio, Visual Studio Team System, la gamme express et Popfly. Quelques chiffres : un million de développeurs professionnels sous Visual Studio 2005, 17 millions de téléchargement des versions Express, 25% des développeurs utilisent Team System, et pas moins de 100 000 utilisateurs de la version bêta de Popfly. Le but de Microsoft est d'adresser l'ensemble des développeurs, de l'amateur averti aux professionnels. La sortie de .Net 3.5 et du futur Sync Framework va dans le sens d'une intégration toujours poussée entre les applications, les technologies et les outils. Cela passe aussi par la documentation dans MSDN : avec les galeries de code, les wiki. Le Software + Services est un autre axe de la stratégie de l'éditeur avec les disponibilités de documents sous forme de Blue Print, des bonnes pratiques ainsi que d'un plug-in pour Visual Studio.

L'écosystème constitue plus que jamais un élément vital pour Microsoft et la vitalité du marché du développement. Ainsi, il y a



plus de 200 partenaires produisant plus de 2 000 produits ! La nouveauté passe par une modification de licencing, ce changement concernera Visual Studio 2008 et son SDK. Il n'y aura plus de limites pour les partenaires pour créer sur Windows ou Visual Studio des solutions… À cela, se rajoute un accès au code source de Visual Studio pour les partenaires " premier " (dans le programme Visual Studio Industrie Partner). Cette volonté passera notamment par l'utilisation de Visual Studio Shell. Il s'agit en quelque sorte d'une coquille vide fournissant l'infrastructure Visual Studio dans laquelle on ajoute ses modules, ses fonctions, ses applications. Très souple, il est personnalisable. Surtout, on peut déployer sa solution avec Visual Studio Shell. Un SDK est disponible. Popfly n'a pas été oublié dans cette ouverture. La nouveauté est la disponibilité de Popfly Explorer (uniquement sous Windows). Il s'agit d'un plug-in destiné à Visual Studio. On peut y créer des projets Popfly ou encore accéder à des projets de n'importe où via Popfly. Surtout, Il intègre désormais les gadgets silverlight.

Côté agenda, 2008 sera chargé avec en particulier : SQL Server, Windows Server, Biztalk v6, IIS 7, Expression Studio 2, Team System (alias Rosario), Popfly Explorer, le framework Sync, sans oublier Live 2.0. Mais surtout, pour les deux ans à venir, il faut s'attendre aux nouveautés suivantes : Visual Studio 10, Silverlight vNext (version 2.0 ?), ainsi que le framework .Net 4.0.

D'autre part, avec l'arrivée du framework 3.5, on aura droit à d'autres sorties plus ou moins rapides. L'une d'elles concernera ASP.Net Futures. Ce futur ASP.Net inclura : le projet Astoria (voir plus bas), un meilleur support d'Ajax et celui de Silverlight, l'apparition d'un framework MVC et de contrôle de données dynamiques. La CTP est prévue courant décembre.

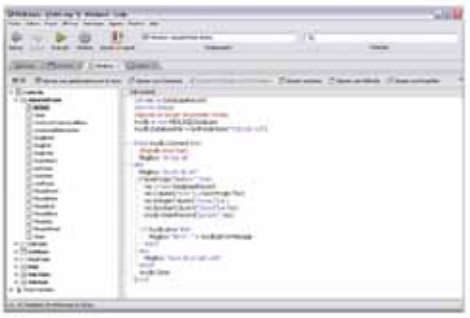
SQL Server 2008 : l'autre vedette

La donnée a été une des vedettes de cette TechEd, avec de nombreuses sessions techniques sur le sujet (ainsi que sur Linq ou le futur framework Sync), qui furent aussi très populaires. François Ajenstat, directeur de SQL Server, a durant plus d'une heure dévoilé l'ensemble des nouveautés du SGBD maison qui tiennent en 4 piliers : plate-forme de données d'entreprise, au-delà du relationnel (notamment avec le XML et les nouveaux types de données, le développement dynamique et enfin l'intégration des données dans son environnement. Le déve-

REALbasic 2007

Cross-Platform that Really Works.*

Développez facilement des applications graphiques ou consoles pour Windows, Mac et Linux à partir d'un source unique



Débuggez à distance : Testez votre application sur Mac à partir de votre PC sous Windows ou Linux et vice versa

Accédez facilement à tous les SGBD du marché : MySQL, MS SQL, PostgreSQL, REAL SQL Server, sources ODBC, ...

Développez des application multilingues : REALbasic est totalement Unicode pour gérer tous types d'alphabet et des applications multilingues.

Compilez des applications natives pour Windows, Mac et Linux en quelques clicks

Déployez facilement : REALbasic ne nécessite ni machine virtuelle, ni DLL et génère des exécutables autonomes pour les trois plates-formes.

Développeurs Visual Basic, portez vos applications sous REALbasic très rapidement sur trois plate-formes dans un environnement familier

Environnement avancé : multi thread, XML, http, TCP/IP, SOAP, orienté objet, Server sockets, SSL et beaucoup plus encore

Support technique gratuit : Support technique par email pour plus d'efficacité

Offre spéciale Programmez

-25%

Offre Spéciale réservée aux lecteurs de programmez !

Pour bénéficier de cette offre, rendez-vous directement sur notre site : www.realsoftware.com/programmez.

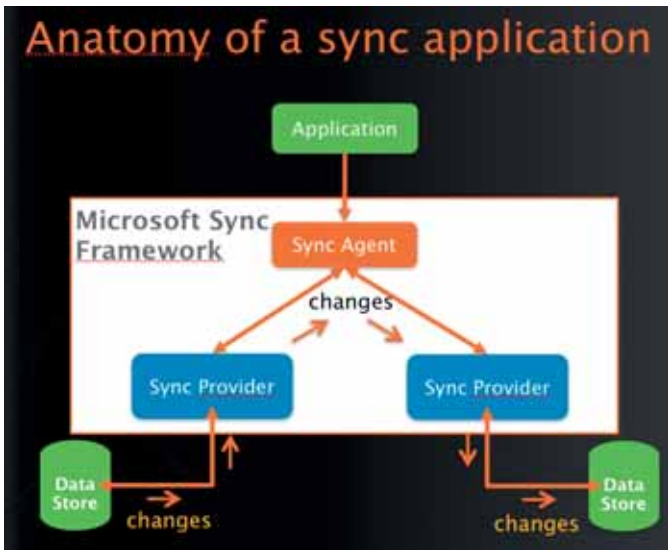
Pour tout renseignement complémentaire, appelez le 08 70 40 94 92 ou contactez nous par email : commercial@realsoftware.fr

25% de remise sur REALbasic PRO Edition soit l'environnement de développement complet pour seulement 300 Euros H.T., déploiement illimité et support par email compris

Téléchargez la version d'évaluation gratuite sur :

www.realsoftware.com/download

*l'outil cross plate-forme qui fonctionne vraiment



loppeur bénéficiera de nombreuses nouveautés comme Linq, un designer graphique pour Entity, de nouveaux modèles de données ou encore comment Sync va interagir avec SQL Server pour synchroniser les données déconnectées (ou non), l'environnement collaboratif et le serveur. Une nouvelle CTP arrivera très prochainement pour une disponibilité finale en février prochain. La version suivante devrait apparaître vers 2010-2011 sous le nom de SQL Server vNext++.

Une exposition réduite mais de qualité

Contrairement à l'an dernier, la partie exposition était plus réduite mais les exposants avaient des produits intéressants à montrer. Nous avons été particulièrement impressionnés par la qualité des nouveaux composants (Infragistics, Gaïa, Software FX), avec une mention spéciale à IdentityMine et des composants WPF haut de gamme, blendables, que l'on peut utiliser directement sous Blend ! Une série d'une dizaine de composants exploitant au mieux les possibilités graphiques de WPF. Un autre stand proposait une solution de développement dédié aux technologies vocales (à base de web services) : Web21C SDK. Le Micro Framework tenait aussi un petit stand. Si aujourd'hui, ce mini framework se destine avant tout aux spécialistes de l'embarqué, de l'enfoui, Microsoft travaille à la prochaine version et surtout à des outils de développement qui pourraient apparaître dans les 12 à 18 mois.

Du Web et encore du Web

Le développement web n'est pas oublié, quelques dizaines de sessions étaient spécifiquement dédiées au Web, aux outils, aux technologies. La gamme Expression et la collaboration développeur – designer occupèrent quelques sessions mais sans plus. Par contre, Silverlight, Live, ASP.Net et même Popfly eurent droit à une audience élargie. Ainsi, si IronPython est disponible depuis un an en v1, la v2 apparaîtra début 2008 et la v3 début 2009. Occasion pour Microsoft de faire connaître les nouvelles API et fonctions de Windows Live ou de Virtual Earth. Les langages dynamiques furent aussi mis à l'honneur et notamment IronPython et tout ce qui tourne autour de la DLR (CLR mais dédié au monde dynamique comme Ruby, Python, etc.).

Et après ?

Microsoft a profité de Barcelone pour commencer à parler du futur. Ainsi, nous avons pu voir quelques idées autour de VB.NexT. Un des avenirs de VB passera par le langage dynamique et son intégration dans la DLR. Pour VB Next, les axes de réflexions concernent la simplification de la syntaxe, la concurrence, le modèle de données, le workflow.

Toujours dans les langages, Microsoft travaille activement sur les langages dynamiques et surtout fonctionnels (avec l'intégration à court terme de F#). La programmation concurrente constitue un axe important, notamment avec le langage C Omega. Occasion aussi de voir de plus près la version 2.0 de PowerShell, disponible en CTP. On notera parmi les nombreuses nouveautés : le remoting, la possibilité de créer des espaces limités d'exécution, des améliorations dans le langage ou encore le support de WMI. Toujours parmi les projets dévoilés ou montrés, on citera encore le projet Astoria. Il s'agit d'une interface de données REST qui doit simplifier l'accès aux données et les affichages, leurs manipulations. Actuellement en CTP, la version finale est attendue mi-2008.

La prochaine édition se tiendra à Barcelone en novembre 2008.

■ **François Tonic**

Projet Oslo (2009) : la prochaine plate-forme applicative

Oslo est un projet ambitieux car il s'agit de construire une nouvelle plate-forme entièrement intégrée incluant les outils, les frameworks, les parties serveurs et données. Oslo n'est donc pas un outil mais avant tout une plate-forme. L'objectif est de pouvoir y construire des applications composites, faire de la SOA, du SaaS. Oslo s'articulera autour de 5 zones :

- le framework .Net 4.0 avec d'importantes évolutions des couches workflow et communication
- le middleware Biztalk : importante refonte pour le bâtir sur WCF et WF, avec une forte intégration SOA et surtout du BPM.
- Les services : avec notamment la disponibilité de l'Internet Service Bus et des Biztalk Services.
- Les outils : disponibilité de nouveaux modeleurs. Team System " Rosario " constitue la première étape. Ces nouveaux outils seront disponibles avec Visual Studio 10.
- Disponibilité d'un référentiel. Pour la première fois, Microsoft proposera un référentiel commun à l'ensemble des outils. Pour le moment, il n'a pas de nom de code ou de nom. Il sera embarqué dans les outils.

Les premières pré-versions (ou CTP) seront disponibles à la fin de l'année et durant l'année 2008. La disponibilité générale n'est pas attendue avant fin 2008, ou plus sûrement courant 2009. Côté migration, sur le .Net framework, le travail sera sans doute limité et sur Biztalk, tout sera fait pour réduire les problèmes. Pour le moment, nous ne savons pas comment sera intégrée la mobilité qui demeure un axe important pour l'éditeur. Sur Live, les services en ligne permettront de consommer les applications composites et services créés et hébergés par Oslo. Pareillement pour Office, qui devient un conteneur naturel des applications Oslo. Enfin, sur la possibilité d'utiliser ces applications en dehors de l'environnement Microsoft, là encore, cela reste à préciser.

Visual Studio 2008 et .Net 3.5 sont disponibles !

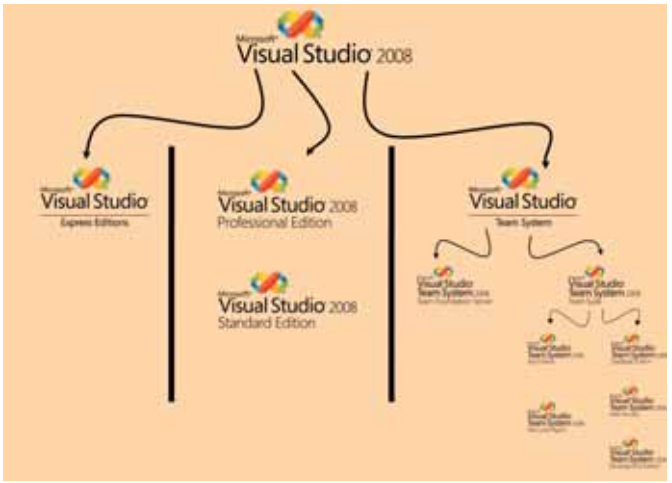
Microsoft l'a annoncé durant le TechEd 2007 se tenant à Barcelone début novembre. Le .NET Framework et Visual Studio 2008 sont disponibles pour les abonnées MSDN et les développeurs quand vous lirez ces lignes.

Cette annonce, bien que très attendue de la communauté des développeurs, n'était en soi pas une surprise majeure, mais plus une suite logique des évènements. L'annonce a été faite par S. Somasegar (Corporate Vice President of the Developer Division). Les versions françaises arriveront début 2008, sans doute pour les prochains TechDays, en février 2008.

Cette annonce n'est pas dénuée de sens car Microsoft au travers de ce premier lancement, mettra du label 2008 sur une gamme de produits bien plus large, à savoir : Windows Server 2008, SQL Server 2008 et aussi Visual Studio 2008 qui prend tout son sens dans ce nouvel écosystème.

Mais cet effet d'annonce et l'énergie déployée par le marketing de Microsoft sont-ils réellement justifiés, cette course effrénée à l'innovation n'est-elle pas seulement une volonté d'alimenter le marché à grand coups de publicité et ainsi d'activer les ventes de licences, même si nous sommes " in fine " dans ce type de spirale... ou bien le label " 2008 " de ces gammes de produits apporte-t-il réellement une valeur ajoutée non négligeable au développeur et de ce fait à l'entreprise qui l'emploie ?

Abordons donc les grands axes stratégiques de Visual Studio 2008, son positionnement, ses valeurs ajoutées ainsi que le .NET Framework 3.5, la plate-forme de développement à laquelle il est associé.



Visual Studio 2008 : 10 ans d'historique !

Cette version 2008 de Visual Studio célèbre les 10 ans d'existence de cette gamme de produits, regardons quelques instants en arrière et observons les challenges en terme de développement que Visual Studio a permis à nos entreprises de réaliser.

La majorité des développeurs se souviendront des débuts de Visual Basic 6.0 avec la mise à disposition, dans une certaine mesure, d'une solution de développement d'application de type Windows client/serveur, sous forme d'expérience RAD (Rapide application development). Ces débuts, qui nous semblent déjà très loin, ont permis de poser les bases d'une évolution constante en cohérence avec les attentes du marché. De ce fait les développeurs ont pu bénéficier des mises à disposition d'un environnement, ou pourrions-nous dire, d'ores et déjà d'une interface de développement de plus en plus riche et de plus en plus en adéquation avec les attentes et contraintes des développeurs contemporains.

Le passage de la version 97 de Visual Studio à la première version labellisée .NET était un changement radical de stratégie pour Microsoft. Cette approche du développement ouvrait de nouvelles portes aux développeurs vers l'étape qui devait suivre la notion d'applications distribuées : les applications Web et Web Services. Celles-ci ont été historiquement à la base des évolutions attendues par le marché au tra-

vers de la tendance " Software as a Services " ou " Software + Services " mise en avant par les grands acteurs de ce monde comme Google, IBM, Microsoft, … Le nouveau challenge de cette gamme 2008 de produits était de satisfaire les attentes des utilisateurs ou des clients demandant de plus en plus des solutions mises à leur disposition : plus de puissance, plus d'expérience utilisateur, plus d'usages, plus de facilité, plus de valeur ajoutée, plus d'interactions au travers de solutions basées sur le Web. Le tout en cohérence totale avec les nouveaux outils Microsoft de l'utilisateur final que sont Windows Vista ou encore Office 2007.

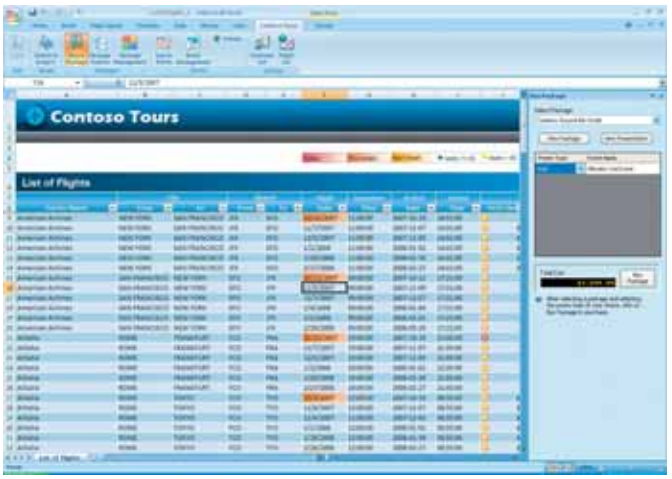
Visual Studio 2008, après une brève utilisation, laisse clairement transparaître les réponses à toutes ces questions. Au travers de nouveaux outils et fonctionnalités il vous permettra de construire une meilleure expérience utilisateur pour vos clients, d'accroître significativement votre productivité ainsi que d'étendre votre expérience collaborative avec vos collègues développeurs. Vos collègues designer graphiques prennent une importance non négligeable grâce à la gamme Expression adossée il y a peu à cette gamme de produits développeur !

Visual Studio 2008 : 3 axes majeurs !

Fort de son historique et ainsi de ses réussites ou échecs dans ce type de solutions, la nouvelle version de Visual Studio 2008 compte 3 nouveaux axes d'évolutions majeures que sont :

1. L'expérience utilisateur
2. La collaboration
3. La productivité

Comme le montre le schéma suivant, Visual Studio 2008, dans la philosophie de ses versions précédentes vous guidera et vous aidera à réaliser mieux, plus vite et en mode collaboratif vos solutions applicatives de demain !



Abordons pas à pas chacun de ces grands axes.

L'expérience utilisateur accrue par Visual Studio 2008. Le premier axe de construction de cette nouvelle version de Visual Studio couvre lui-même les trois modes de développements que sont les applications Winform, Webform et Officeform (entendez solutions basées sur Microsoft Office). Les challenges sont divers et variés :

- **Winform :** Création d'application exploitant les nouveautés techniques et le rendu proposés par Windows Vista.
- **Office :** imbriquer les données métier ainsi que les logiques fonctionnelles au sein des solutions bureautiques maîtrisées par la majorité des utilisateurs
- **Webform :** Attente accrue des applications Web, comportements plus intuitifs, intégration du multi-média (WebTV, Live, …)

Les réponses de Visual Studio 2008 sont à la hauteur des challenges cités ci-dessus :

- **Winform :** Nouveau designer et nouveaux contrôles WPF, support du rendu graphique vectoriel, formes et design.
- **Office :** Intégration complète et simplifiée des possibilités d'extension de Microsoft Office au sein de Visual Studio 2008. Déploiement simplifié avec Click-Once, connexion de Microsoft Office à des applications métier au travers de son interface utilisateur ou du Custom Task Panel.
- **Webform :** Nouveaux outils HTML/CSS/AJAX, intégration plus forte et intuitive des solutions ASP.NET Ajax, meilleure compatibilité cross-browser.

Le second axe qu'est **le mode collaboratif de Visual Studio 2008** apporte également son lot de nouveautés autour de sujets sensibles comme la gestion du cycle de vie des applications, le cycle de vie de leur développement ou encore le travail de plus en plus accru avec les designers d'applications, que celles-ci soient de type lourdes, légères ou riches.

Les challenges collaboratifs sont également à ce titre nombreux et variés, pour cause :

- **Gestion du cycle de vie :** évolution des entreprises qui gèrent de plus en plus leurs bases de données séparément du reste des applications, ces bases de données sont généralement conçues et maintenues par des professionnels
- **Cycle de vie des développements :** souhait des entreprises d'évoluer vers des méthodes de développement agiles permettant une intégration continue des évolutions.
- **Travail avec les designers :** Besoin de structurer et d'optimiser encore et toujours le travail entre les designers d'applications et les concepteurs d'applications.

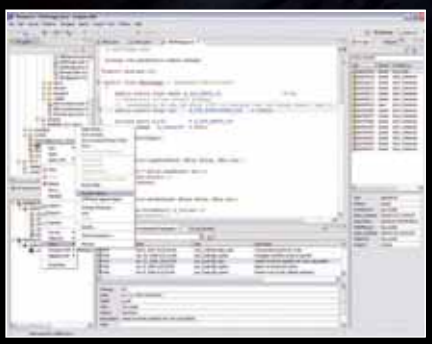
Visual Studio 2008 apporte à nouveau son lot de réponses adaptées :

- **Gestion du cycle de vie :** Team System 2008 inclut de nouveaux rôles orientés vers les professionnels de la base de données : design, test et roll-back sur les changements de topologies de bases de données intégrées aux processus de développement existants.
- **Cycle de vie des développements :** Team System 2008 comprend de nouvelles fonctionnalités permettant l'intégration continue d'évolutions applicatives tout en optimisant la stabilité et les performances de vos solutions.
- **Travail avec les designers :** Visual Studio 2008 permet une intégration et un partage des projets avec les outils de la gamme Expression. Gamme de produits Microsoft dédiée aux designers et intégrateurs. Au-

Perforce | Le système rapide de gestion de configuration logicielle

Présentation du module d'extension de Perforce pour Eclipse

Pour travailler avec Perforce dans une interface IDE Eclipse.



Module d'extension de Perforce pour Eclipse

Le module d'extension de Perforce pour Eclipse permet aux développeurs d'accéder facilement au système de GCL Perforce depuis leur interface IDE Eclipse. Il propose les fonctionnalités suivantes :

- Accès rapide à l'historique complet des fichiers
- Prise en charge complète du développement collaboratif, avec possibilité de fusionner les fichiers
- Possibilité de travailler hors ligne lorsque la connexion avec le serveur Perforce est indisponible
- Outil de comparaison des fichiers et prise en charge du suivi des défauts intégrés
- Prise en charge de la fonction de refactoring de l'environnement Eclipse

Le module d'extension de Perforce pour Eclipse prend en charge les systèmes d'exploitation Windows et Linux. Et ce n'est que l'un des nombreux outils intégrés dans le système de GCL Perforce.

eclipse FOUNDATION MEMBER

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.

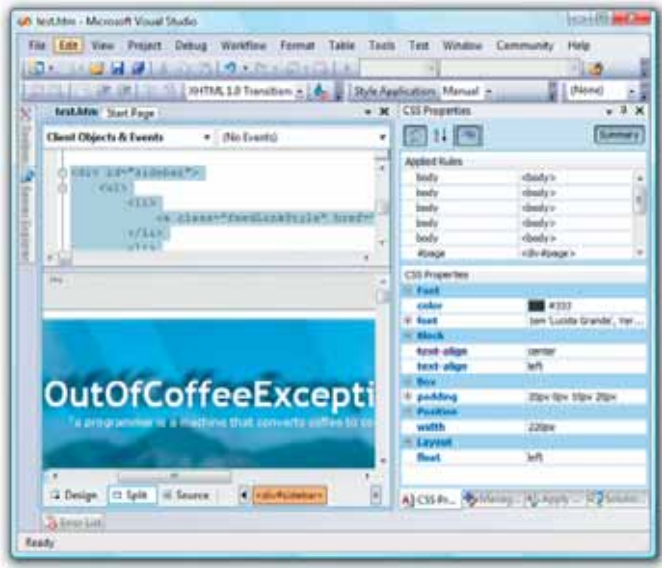
delà, de ces premiers éléments, Visual Studio 2008 intègre de nouveaux éditeurs visuels pour CSS, l'HTML split, …

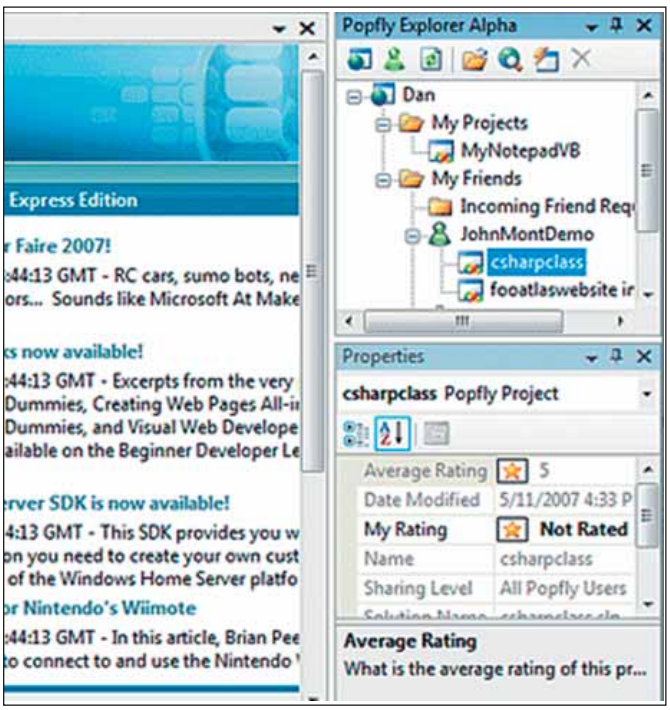
Abordons enfin le dernier axe majeur de Visual Studio 2008 : **l'augmentation de la productivité** des développements de solutions .NET. Ce dernier axe est certes un des plus importants pour l'ensemble des sociétés de développement ou de toute entreprise exploitant les outils de la gamme Visual Studio.

Pour ne pas rompre l'approche prise depuis le début, analysons les challenges rencontrés par les équipes de Redmond sur ce dernier axe de développement de Visual Studio 2008. Tout comme dans les deux premiers axes, ce dernier est approché selon des points de typologie de développement comme le développement Web Client-Side, le souhait d'un IDE unique permettant de développer sur les différentes versions du .NET Framework (et notamment en version 3.5), l'optimisation des développements de Workflow et de solutions de communication ainsi que la gestion optimisée des données. Chacune de ces typologies fut visiblement l'objet d'une grande attention de la part de Microsoft qui apporta face à chacune d'entre elles, au travers de Visual Studio 2008, des solutions pragmatiques !

Passons en revue ces challenges :

- **Développement Web Client-Side :** Optimisation des modes de développement de solutions orientées Web 2.0 et du debugging associé.
- **Un IDE unique pour le .NET 2.0 – 3.0 – 3.5 :** Souhait de n'avoir qu'un seul IDE afin de développer sur les différentes versions du .NET Framework.
- **Workflow et Communication :** Attente d'une optimisation des coûts de maintenance ainsi que l'optimisation du support de solution à architecture de services hétérogènes.



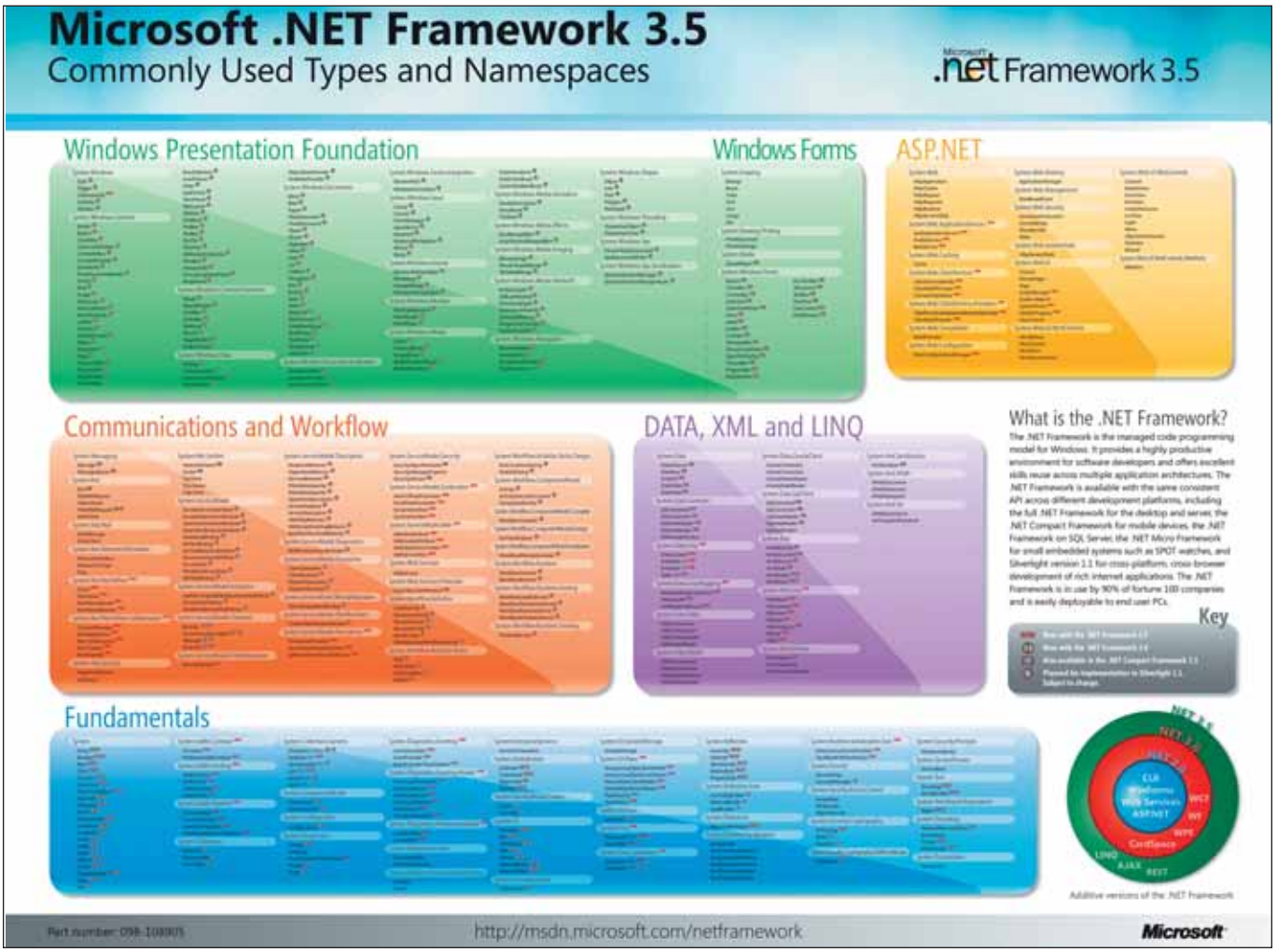


- **Data :** Souhait des développeurs d'homogénéiser l'accès aux types hétérogènes de données (XML, relationnels, …) avec pour challenges l'accès, le requêtage, la manipulation ainsi que la mise à jours des données.

Voici les solutions mises en face de chacun de ces challenges :

- **Développement Web Client-Side :** Visual Studio 2008 intègre le support de l'Intellisense pour Javascript ainsi qu'un mode de debugging associé. Vous y retrouverez également une optimisation de la vue permise sur votre code au travers du Split ou encore une meilleure gestion des CSS.
- **Un IDE pour le .NET 2.0 – 3.0 – 3.5 :** Visual Studio 2008 intègre le multi-targeting pour les versions 2.0, 3.0 et 3.5 du .NET Framework. Celui-ci se chargera d'adapter les spécificités de votre projet ainsi que ses propres spécificités en fonction de la version du .NET Framework visée. Cette démarche permettra d'éviter l'installation de la version de Visual Studio adaptée au Framework visé. Cette fonctionnalité, déjà bien connue sur d'autres environnements de développement était plus que nécessaire au sein de cette nouvelle version de l'IDE phare Microsoft.
- **Workflow et Communication :** Visual Studio 2008 intègre le mode design de workflows pour des processus métier et permet également une unification des modèles de programmation pour une grande variété des modes de communication.
- **Data :** Visual Studio 2008 intègre le support du nouveau langage de requête de données " LINQ " en son sein, au travers d'une possibilité de manipulation de données en tant qu'objets ou tout au moins sous forme de part naturelle des modèles de programmation.

Cette nouvelle version de Visual



Studio risque ainsi d'en étonner plus d'un de par sa diversité et sa capacité à évoluer et correspondre à vos besoins, que vous soyez développeur, Chef de projet, Architecte au travers de la version de Team System par exemple ou encore Designer lors de la collaboration ou intégration de votre travail !

Le .NET Framework 3.5

Impossible de poursuivre un article relatif à Visual Studio 2008 sans mentionner la version du .NET Framework. Dans ce cas-ci, le .NET Framework 3.5. Bien que Visual Studio 2008 permette de travailler sur différentes versions du Framework, attardons-nous quelques instants sur cette nouvelle version !

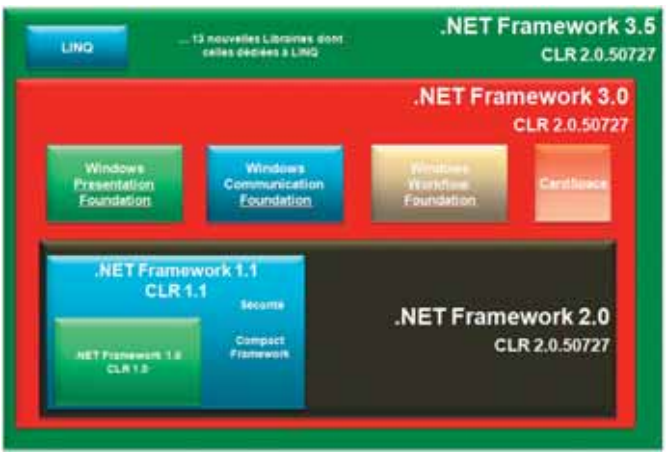
Observez ainsi la structure suivante présentant les différentes évolutions et organisation du .NET Framework depuis sa version 1.0 jusqu'à sa toute dernière version, soit la 3.5.

L'insistance quant à la notion de CLR est volontaire. Effectivement, les versions 2.0 – 3.0 et 3.5 reposent sur la CLR 2.0.50727 et permettent ainsi de mieux comprendre l'une des nouvelles fonctionnalités phare de Visual Studio 2008 : le multi-targeting!

N'hésitez pas à consulter l'article suivant, afin de comprendre en détail la notion de multi-targeting de Visual Studio 2008 : http://www.asp-php.net/tutorial/asp.net/linq-1.php?page=3

Cependant, voici pour rappel, quelques dates clés :

- 2002 : .NET Framework 1.0
- 2003 : .NET Framework 1.1
- Fin 2005 : .NET Framework 2.0
- Fin 2006 : .NET Framework 3.0



- Fin 2007 : .NET Framework 3.5

Ces dates ont pour objectif, pour ceux qui ne le sauraient pas encore, de vous montrer l'historique et la pérennité de cette plate-forme de développement.

Les couleurs attachées au .NET Framework 3.0 (Rouge) et au .NET Framework 3.5 (Vert) ne sont pas dénuées de sens puisque celles-ci permettent d'introduire la notion de :

- **Red Bits :** incluant les notions relatives à WCF, WPF, WF. L'objectif étant, dans ce cas, de délivrer des services pack de compatibilité afin de préserver un haut niveau de compatibilité

ascendante des solutions développées sur les versions précédentes du .NET Framework.

- **Green Bits :** incluant les librairies introduisant les nouvelles fonctionnalités dans la plate-forme de développement. Celles-ci reposent en partie sur les Red Bits et impliquent des changements légers de celles-ci. Dans ce cadre, l'objectif est de garantir au maximum que l'installation de toute nouvelle librairie de ce type n'impactera pas les applications existantes développées sur toute version précédente du .NET Framework.

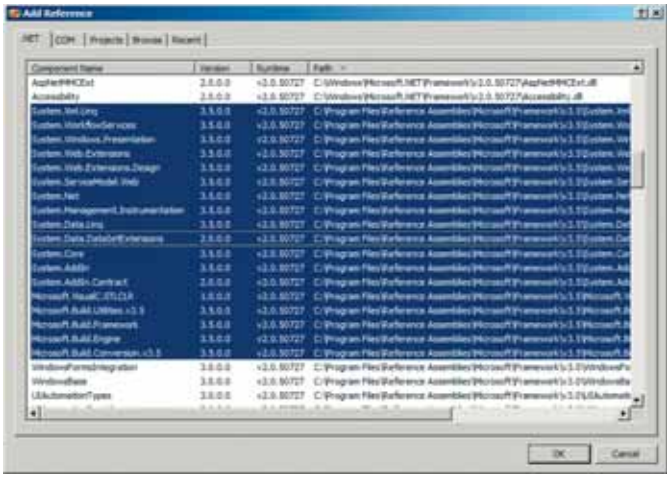
Cette double notion de Red et Green Bits avait été annoncée de longue date par S. Somasegar au travers de son blog, le 18 Mai 2006.

En parcourant celui-ci, vous retrouverez les notions qui ont guidées les équipes de Redmond dans l'évolution vers la version 3.5, dont une majeure, qui était de minimiser l'impact des nouveautés et fonctionnalités sur les composants en place et ainsi de travailler plutôt sur une évolution de la plate-forme au travers de nouvelles librairies et non pas sur une refonte de celle-ci !

Visiblement le terme **" pérennité "** prend tout son sens encore une fois ici ! Abordons ainsi les nouvelles librairies du .NET Framework 3.5. Différentes pistes s'offrent à nous :

- L'explorateur d'objet de Visual Studio 2008
- L'application " .NET Reflector "
- L'explorateur de Windows
- L'explorateur de librairies exploitées dans l'ajout de référence à tout projet …

L'exploitation de cette dernière option au travers d'un projet .NET quelconque créé sur la version 3.5 du Framework permettra de vite se rendre compte des nouveautés en librairies proposées par celle-ci. Effectivement, après un rapide tri des librairies disponibles dans l'onglet " .NET " en



cliquant sur l'intitulé de la colonne " Path ", vous pourrez observer les nouvelles librairies du .NET Framework 3.5.

Vous retrouverez ainsi les librairies propres au .NET Framework 3.5 ou tout au moins dans ce cas celles faisant partie du package de livraison de celui-ci. Vous remarquerez à toutes fins utiles la version du Runtime associé qui repose bien sur la CLR 2.0.

Focalisez-vous sur les librairies relatives à la structure du .NET Framework, celles inhérentes à l'espace de nom " System ".

Vous y retrouvez pas moins de 13 nouvelles librairies !

Voici une brève description de celles-ci :

- System.AddIn.dll et System.AddIn.Contract.dll = implémente un nouveau modèle d'AddIn au niveau de la CLR. A ce titre, retrouvez le blog de la Team Corp ayant en charge cette notion : http://blogs.msdn.com/clraddins/

- System.Core.dll – Cette librairie peut être considérée comme étant la librairie principale des " Green Bits ", entendez les nouvelles librairies du .NET Framework 3.5.

Un seul coup d'œil au travers du .NET Reflector vous permettra de comprendre que cette librairie couvre plusieurs Espaces de Noms comme présenté dans la capture ci-dessous :

- **System.Linq.*** = implémentant entre autre LINQ to Objects
- **System.Diagnostics.***
- **System.Security.***
- **…**

- System.DirectoryServices.AccountManagement.dll = implémentation d'un wrapper pour les APIs de type Active Directory.

- Implémentation de **LINQ** :
  - **System.Data.DataSetExtensions.dll** = implémentation de LINQ to DataSet.
  - **System.Data.Linq.dll** = implémentation de LINQ to SQL.
  - **System.Xml.Linq.dll** = implémentation de LINQ to XML.

- System.Net.dll = implémente des APIs de "Peer to Peer" et " Peer to Peer Collaboration ".

- System.Management.Instrumentation.dll = implémente un provider manage pour WMI 2.0 " Windows Management Instrumentation " (Cette librairie est combinée à l'espace de nom System.Management dans System.Core.dll).

- System.Web.Extensions.dll = implémentation de la technologie ASP.NET AJAX ainsi que des Services relatifs à l'implémentation cliente des applications. Vous y retrouverez également, après un petit coup de Reflector, l'implémentation de l'espace de nom : **" System.Web.UI. WebControls "** incorporant des nouveaux contrôles ASP.NET 3.5 comme le **ListView,** le DataPager ou encore le **LinqDataSource.**

- System.Windows.Presentation.dll = implémente le support de WPF pour la librairie System.AddIn.

- System.WorkflowServices.dll et System.ServiceModel.Web.dll = amélioration de WF et WCF.

Vous voici donc maintenant bien armés face à ces nouvelles librairies du .NET Framework 3.5. Vous l'aurez compris, LINQ prend une place relativement importante dans cette nouvelle version de .NET.

Ainsi pouvons-nous imaginer que la version 4.0 ou peut-être encore la version 3.5 du .NET Framework intègrera de nouvelles librairies annoncées lors du Keynote, comme le projet Sync Framework (http://msdn2. microsoft.com/en-us/sync/default.aspx).

Sources et liens

- http://msdn2.microsoft.com/en-us/vstudio/aa700830.aspx
- http://www.microsoft.com/downloads/details.aspx?familyid=d2f74873-c796-4e60-91c8-f0ef809b09ee&displaylang=en
- http://weblogs.asp.net/scottgu
- http://blogs.msdn.com/somasegar
- http://www.microsoft.com/downloads/details.aspx?familyid=7B645 F3A-6D22-4548-A0D8-C2A27E1917F8&displaylang=en
- http://www.microsoft.com/downloads/details.aspx?displaylang=fr&FamilyID=17319eb4-299c-43b8-a360-a1c2bd6a421b

Grégory Renard
CTO Wygwam
Microsoft Regional Director
Microsoft MVP (Most Valuable Professionnal) INETA Lead Belgium
http://www.wygwam.com/
http://blogs.developpeur.org/redo/ - WygwamTV

LEONARDI

l'ihm

EN TOUTE SIMPLICITÉ !

ATELIER "QUICK START" LEONARDI 17 ET 18 DEC. 500€ HT REMBOURSABLES.

Votre application de gestion multilingue avec plus de 100 vues de 20 types différents, en DHTML/Ajax, en Swing ou en plugin Eclipse, connectée à un SGBD et un bus JMS.

« il vous faut combien de temps pour la réaliser ? »

Si votre réponse est moins d'une semaine, inutile de vous rendre sur notre site, ni de télécharger la version gratuite de LEONARDI, sinon il est temps de passer à la vitesse "*Model-Driven*"...

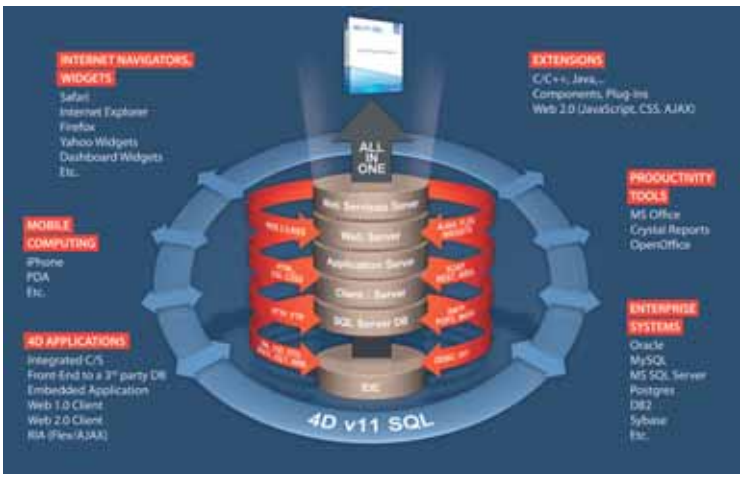
LYRIA

Pour en savoir plus sur la solution Leonardi, rendez-vous sur notre site www.lyria.com ou envoyez-nous un courriel à info@lyria.com

Dossier en ligne
www.programmez.com

4D v11 SQL : Innovation et développement durable

*4D v11 SQL est un environnement de développement intégré ouvert et évolutif grâce à l'**adoption de nombreux standards** comme le support du **SQL,** la **prise en charge du SVG,** l'intégration d'**Unicode,** ainsi que le support étendu du **XML, ODBC** et des **Web Services SOAP.***
Ce qui explique d'ailleurs le fait que la technologie 4D soit au cœur de milliers de solutions dans le monde depuis plus de 20 ans.



4D v11 SQL présente un système d'administration complet en rassemblant tous les outils nécessaires au contrôle, à la maintenance, à la sauvegarde et au compactage des fichiers de données et de structure.
Ce centre de sécurité et de maintenance permet aussi de configurer la sauvegarde périodique des données en cours d'exploitation (possibilité de sauvegarde par **miroir logique**) et la restitution automatique des archives pour récupérer intégralement les données après incident. Les fichiers d'historique (système de **journalisation**) enregistrent les opérations effectuées sur les données de la base (possibilité d'annuler les opérations). Tous les paramètres sont accessibles par programmation afin de personnaliser le système de sauvegarde.

4D v11 SQL comporte une base de données intégrée hautement performante (accès au fichier de données sur 64 bits, taille du **fichier de données illimitée,** plus de 32 000 tables et champs par table possible, 1 milliard d'enregistrements par table, etc.), un environnement de développement professionnel riche et rapide (éditeur de structure, de formulaires, de méthodes, d'utilisateurs et mots de passe, explorateur d'index, débogueur, correcteur orthographique, vérificateur de syntaxe, compilateur, générateur d'applications, etc.) et un code source unique (plus de 900 commandes) permettant de déployer des solutions multi-plates-formes, à la fois **Mac OS et Windows.** 4D v11 SQL intègre également un **serveur de données** qui ouvre l'accès aux données depuis un poste client, un **serveur d'application** qui permet le développement collaboratif, un **serveur Web Services** qui publie l'application instantanément, et un **serveur SQL** ultra-rapide qui interroge la base de données en langage SQL de manière native (en interne) ou depuis et vers une autre base de données du marché.

Un environnement de développement standard

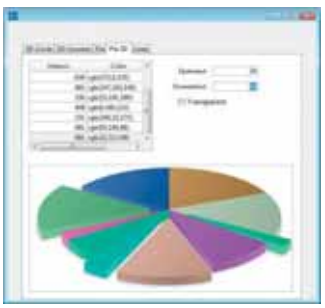
XML

Le langage de 4D est doté d'un jeu de commandes permettant la création de documents **XML** selon les deux méthodologies :

- **DOM** (Document Object Model) : construction, analyse et manipulation de l'arbre XML en mémoire avant sa mise à disposition ;
- **SAX** (Simple API XML) : lecture séquentielle permettant notamment la gestion de fichiers volumineux.

4D v11 SQL dispose également en standard d'un processeur **XSLT** permettant la transformation à la volée de documents **XML.**
Le serveur Web interne de 4D permet de diffuser des documents **XML** dynamiquement alimentés par les données de la base de données.

Prise en charge native des images et moteur SVG



4D v11 SQL offre la possibilité de manipuler des images vectorielles ou bitmap dans de nombreux formats natifs (Jpeg, Png, SVG, ...) et intègre un moteur de rendu **SVG** (format de fichier graphique vectoriel) pour le plus grand bénéfice de vos interfaces qui pourront afficher les fichiers **SVG** dans les champs ou les variables de type image. L'utilisation la plus courante du **SVG** est la publication de données statistiques ou cartographiques. Générez par exemple vos graphiques en **SVG** à la volée et modifiez-les directement par programmation tout en les visualisant dans vos formulaires !

Unicode, XLIFF

4D v11 SQL utilise maintenant de l'**UTF-16** (et s'aligne ainsi sur les OS Windows et Mac) et parfois l'**UTF-8,** notamment pour le Web et les

imports/exports. Le forum 4D est d'ailleurs déjà affichable en **japonais** et bientôt en **chinois** !
4D v11 SQL prend en charge le support natif des **regex** et le standard **XLIFF** pour l'adaptation linguistique des textes et libellés de l'interface. Cette technologie est utilisée en interne pour les applications 4D. Les développeurs 4D ou de plug-in peuvent tirer parti de cette nouveauté dans leurs propres applications et plug-in personnalisés.
XLIFF (XML Localization Interchange File Format) est un standard dédié aux processus de traduction et d'adaptation linguistique. Il permet d'établir une correspondance entre une langue source et une langue cible à l'intérieur d'un fichier **XML.**
Le standard **XLIFF** constitue de fait une alternative aux systèmes basés sur les ressources. Divers outils, dont de nombreux logiciels gratuits, permettent la gestion des fichiers **XLIFF.**

Compatibilité accrue

Certifiée sur les derniers OS (OS X 10.5 **Léopard** et Windows **Vista**), 4D v11 SQL apportera notamment aux utilisateurs Macintosh un surcroît de vitesse avec le support des machines Mac Intel, grâce à une nouvelle architecture 100% **Universal Binary.**

Accessibilité et ouverture de vos applications métiers avec les technologies modernes

Code 4D

Une méthode 4D peut être appelée localement et par défaut dans la totalité de votre application. Elle peut s'exécuter sur le poste client ou sur le serveur. Mais une méthode 4D peut également être appelée via un **process http,** une **requête SQL,** une **autre méthode** ou encore une **requête SOAP** selon que vous la rendiez :

a. Disponible pour le Web (via 4DACTION, 4DMETHOD ou 4DSCRIPT).
b. Accessible via les **Web Services** avec la possibilité de publier et de mettre à jour le **WSDL** de façon **automatique** ! Les requêtes et réponses des méthodes de Service Web utilisent des messages **SOAP** (Simple Object Access Protocol) envoyés par l'intermédiaire du serveur Web natif de 4D. Les messages **SOAP** et les données qu'ils contiennent suivent les règles syntaxiques et les restrictions d'**XML.**
c. Accessible à partir d'une autre méthode 4D dans une autre application par l'intermédiaire des composants. Une base 4D standard peut devenir un composant d'une autre base 4D.
d. Disponible via **SQL** (on peut utiliser les paramètres de retour d'une méthode directement dans une requête **SQL**).

Tout ceci se paramètre simplement en cochant une option dans la fenêtre des propriétés d'une méthode (Figure ci-contre).

Rich Internet Application

2006 a vu l'émergence de nouveaux acteurs du monde Internet : les applications Internet riches (***RIA*** en anglais). Le Web jusqu'alors dédié à la présentation d'informations permet désormais d'exécuter localement une application, au sein d'un navigateur, déchargeant ainsi les appels à un serveur applicatif pour optimiser la vitesse d'exécution et minimiser le transfert de données. Ce type d'application a également la particularité d'être très léger et de soigner l'interface graphique pour améliorer l'expérience utilisateur. Plusieurs technologies permettent dès aujourd'hui de créer des ***RIA,*** parmi lesquelles **AJAX,** Silverlight™ de Microsoft® et **Flex**™ de chez Adobe® (cf. n°102 de Programmez).

Ajax

4D Ajax Framework, composant intégré à 4D Web 2.0 Pack, combine quant à lui la puissance d'**AJAX** et celle des bases de données 4D :

- **XHTML** et **CSS** pour la création de l'interface utilisateur
- **DOM** pour la manipulation dynamique de cette interface
- **XML** et **XSLT** pour l'échange d'information avec le serveur
- **JavaScript** pour la logique applicative

Ce framework **AJAX** permet à tout développeur de bénéficier de la puissance d'**AJAX** pour concevoir des applications Web 2.0 sans avoir au préalable à maîtriser les **CSS** ou le **JavaScript.** Grâce aux bibliothèques mises à leur disposition, les développeurs produiront rapidement de puissantes applications professionnelles.

Flex

L'accès aux données

Bien que les **RIA** présentent de nombreux avantages, l'accès aux données depuis un client Flex dans un navigateur reste un enjeu majeur. Depuis la version 2003, 4D propose un serveur **HTTP** intégré à la base de données et au serveur applicatif facilitant le développement et le déploiement d'un serveur Web qui hébergerait des pages pouvant contenir une application Flex.
Bien plus encore, 4D permet de publier, en un seul clic et sans réécriture, des méthodes en **Web Services** grâce à un serveur **SOAP** également intégré. Ces **Web Services** sont compatibles et consommables par un client Flex avec le composant mx:WebService, voire même en quelques clics avec le Data Wizard bientôt disponible dans la prochaine version 3 de Flex™ Builder™.

Au-delà du serveur HTTP : le serveur SQL et 4DFlexLib

Le mode d'accès aux données d'une base 4D, par un client **Flex,** peut parfois s'avérer contraignant. Chaque requête s'effectue en mode déconnecté, ne garantissant pas l'intégrité des données, n'identifiant pas son origine et nécessitant surtout d'écrire une API sur le serveur pour pouvoir effectuer des modifications (insertion, suppression ou modification de données).

La version 4D v11 SQL intègre un nouveau et puissant serveur **SQL.** Ce serveur est accessible via un protocole propriétaire, identique à celui utilisé dans le driver **ODBC** de 4D. Lors de la dernière conférence des développeurs 4D à Memphis en octobre dernier, 4D a dévoilé la sortie prochaine d'une librairie de classes Flex ayant pour nom de code **4DFlexLib. 4DFlexLib** permet à n'importe quel développeur Flex d'exécuter en quelques lignes des requêtes SQL sur une base 4D v11 SQL. Cette librairie est bâtie en 3 couches dont seules 2 seront disponibles pour les développeurs Flex :

couche 3 : une couche haut niveau de service comprenant un composant SQLService similaire aux composants HTTPService ou WebService de Flex. Ce composant SQLService permet d'effectuer des requêtes **SQL** sur le serveur SQL 4D en quelques lignes de code Flex.

Déclaration simple d'un service SQLService en MXML

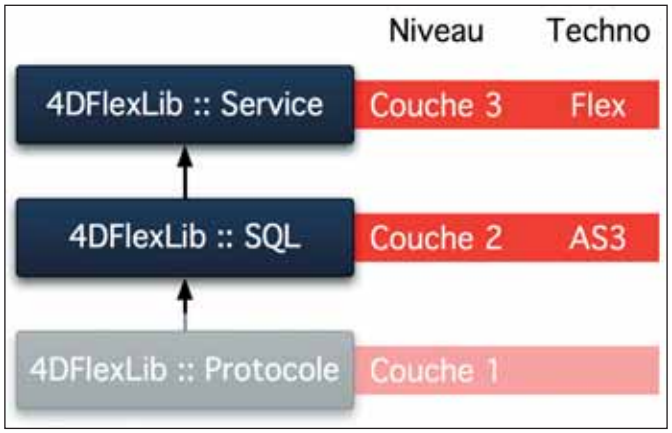
```
<fourD:SQLService
    host="my4DSQLServer.myDomain.com"
    userName="sqlUser"
    result="resultHandler(event)"
    fault="faultHandler(event)"
>
<mx:Script>
    <![CDATA[
        private function resultHandler(event:ResultEvent):void
        { trace(event); }

        private function faultHandler(event:ResultEvent):void
        { Alert.show(event.toString()); }
    ]]>
</mx:Script>
```

couche 2 : une couche de plus bas niveau qui permet d'interagir de manière plus précise avec le protocole **SQL,** et disponible pour les développeurs **Flash** (pas besoin de composants Flex, i.e. de l'espace de nom mx).

couche 1 : la couche de plus bas niveau qui gère l'implémentation du protocole **SQL** lui-même et qui ne sera pas accessible directement par les développeurs.

Grâce au composant mx:SQLService, il est possible d'ajouter, de sélectionner, de modifier, ou de supprimer des données d'un serveur 4D, très simplement et sans écrire une ligne de code côté serveur !



4DFlexLib et la sécurité

La sécurité des transactions effectuées sur un réseau reste un enjeu majeur en Flex et est souvent critiquée par nombre de développeurs. **4DFlexLib** permet de créer une session sécurisée en cryptant les informations de connexion, évitant ainsi de transmettre les noms d'utilisateur et mots de passe en clair. Elle garantit aussi la provenance des données prévenant ainsi le client Flex ou le serveur de toute attaque extérieure éventuelle.

RDA ?

L'année 2008 verra arriver massivement de nouvelles technologies de type ***RDA*** (Rich Desktop Applications) qui pourront exécuter des applications de type **RIA** hors d'un navigateur. Ces applications permettent à la

fois de s'affranchir des contraintes du navigateur et de pouvoir accéder au système d'exploitation. Chez Adobe, c'est la technologie ***AIR*** (Adobe Integrated Runtime) qui est sur ce terrain. Un nouveau terrain auquel **4DFlexLib** s'adapte et sur lequel 4D porte déjà attention.

Sources de données externes

4D v11 SQL contient un ensemble de commandes de haut niveau permettant d'accéder aux sources de données compatibles **ODBC** via le **SQL.** Elles sont d'ailleurs utilisables avec le tout nouveau moteur **SQL** natif qui rend vos données accessibles au plus grand nombre d'applications, mais surtout qui permet aux applications 4D d'accéder directement à la plupart des systèmes d'information actuels. Notre article paru dans le magazine précédent était d'ailleurs consacré à ce tout nouveau moteur **SQL** multithread préemptif. Vous pouvez également importer et exporter des données vers une source **ODBC.** De plus, 4D v11 SQL permet d'interroger directement vos autres applications 4D sans avoir besoin du driver **ODBC.**

Conclusion

Grâce à l'augmentation spectaculaire des capacités de stockage de la base de données 4D v11 SQL, vous pourrez désormais aisément anticiper les besoins constants de montée en puissance des solutions d'entreprise. 4D v11 SQL fait entrer vos développements dans une nouvelle ère de communication. La productivité de développement légendaire de 4D a encore été améliorée, avec plus de **200 nouvelles fonctions,** incluant des **composants** de nouvelle génération, la **prise en charge native des images** s'émancipant dorénavant de QuickTime, la publication

de graphiques **SVG,** le support natif des **regex,** et bien plus encore. 4D v11 SQL vous permet plus que jamais de vous appuyer sur les standards du marché avec un nouveau moteur SQL natif intégré au cœur de son architecture ainsi que la généralisation de l'emploi des technologies XML et Unicode. 4D v11 SQL est la solution la plus ouverte parmi les versions proposées par 4D à ce jour, offrant au développeur une palette d'options inégalée.

■ **Denis Leroux**

Un bureau eXtreme Programming

Nous continuons l'exploration des bureaux de développeurs. Ce mois-ci, nous nous arrêtons chez Thales, dans un bureau où on développe des logiciels temps-réel critiques et embarqués pour l'avionique. Attention, c'est de la haute voltige !

Le bureau d'Emmanuel CHENU

La complexité des besoins des clients, le volume des développements à réaliser et la rigueur imposée par les exigences de sécurité ne peuvent être appréhendés que par un travail d'équipe. C'est en confrontant les points de vues, en partageant les expériences et en parallélisant certains travaux que nous pouvons développer des logiciels de qualité dans les délais attendus.

Praticiens du développement logiciel agile, nous avons trouvé dans l'eXtreme-Programming des recommandations pragmatiques pour organiser bureaux et postes de travail afin de favoriser le travail en équipe.

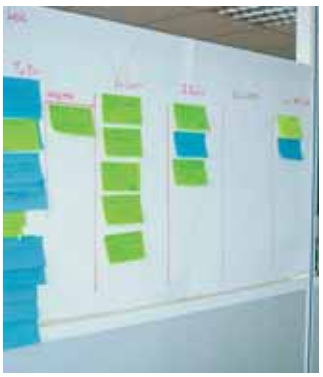
L'espace projet

Notre équipe de quatre développeurs logiciels partage le même bureau. Il est directement voisin de notre salle de réunion et de celui partagé par le chef de projet et les experts métier. Cette proximité nous permet de privilégier la communication orale en face à face et renforce la cohésion de l'équipe.

Le bureau du logiciel

Les développeurs logiciels sont tous installés autour d'une grande table afin que personne ne se tourne le dos. Nous communiquons toujours en face à face, sans avoir à interrompre les activités en cours. Ainsi, tout membre de l'équipe peut aisément solliciter ses collègues pour le moindre conseil. De plus, partager le même bureau favorise la communication passive: chacun est à tout moment au courant des activités des autres, même s'il n'est pas directement impliqué.

Dans cet espace, chaque poste est adapté au travail en binôme et aux relectures de code. En effet, chaque table peut accueillir deux développeurs qui s'échangent le clavier et la souris devant le même écran.



Tableaux blancs et post-it géants

Nous modélisons beaucoup en groupe et au marqueur en utilisant la notation UML. Les conceptions sur lesquelles nous sommes en train de travailler sont ainsi affichées en grand format. Ceci nous permet de revenir fréquemment sur nos décisions de design pour les conforter ou les remanier au fur et à mesure des informations que nous collectons en codant et en testant.

Un des tableaux est un grand " kanban " représentant notre flux de travail. Lors de réunions quotidiennes de moins de dix minutes, nous faisons avancer les tâches, représentées par de petits post-it, répartis le long des cases du " kanban ".

Cela nous permet de ressentir quotidiennement l'avancement de l'équipe et de l'afficher en toute transparence.

Un autre tableau est dédié aux améliorations potentielles. Chaque idée de remaniement de code, de test ou d'automatisation de procédure manuelle est résumée sur un post-it dédié, collé sur le tableau. Les idées applicables peuvent être élues pour traverser le " kanban " du flux de travail.

Des post-it représentant les jalons du projet sont collés sur un grand calendrier affiché au mur. Ainsi, l'équipe ressent d'un coup d'oeil le temps qui la sépare de ces moments clés.

Au centre du bureau commun, un panier à courrier contient les numéros de [Pro]grammez ;o) En effet, nous discutons volontiers des articles qui nous touchent (intégration continue avec CruiseControl, tests de recette avec Fitnesse, développement model-driven avec génération de code contre des pratiques centrées sur le code et les tests, etc.).

Ah, oui, nous avons un cinquième développeur dans le bureau : le poste d'intégration continue sur lequel tourne CruiseControl ;o)

Un environnement virtuel adapté au travail en équipe

Les développeurs partagent un outil de gestion de configuration, un outil de gestion des faits techniques, un intranet éditable pour échanger de l'information et un outil d'intégration continue qui publie la santé du logiciel et en notifie l'équipe par mail.

Bien sûr, les développeurs utilisent un IDE, connecté à un compilateur natif et un compilateur croisé pour le processeur cible. Nous testons avec la suite xUnit, un outil de couverture structurelle et éventuellement Fit ou Fitnesse pour les tests de recette. Enfin, la panoplie est complétée par des outils de qualimétrie et de traçabilité des exigences.

Sur Thales – Aerospace Division

Présence à bord de tous les aéronefs civils et militaires

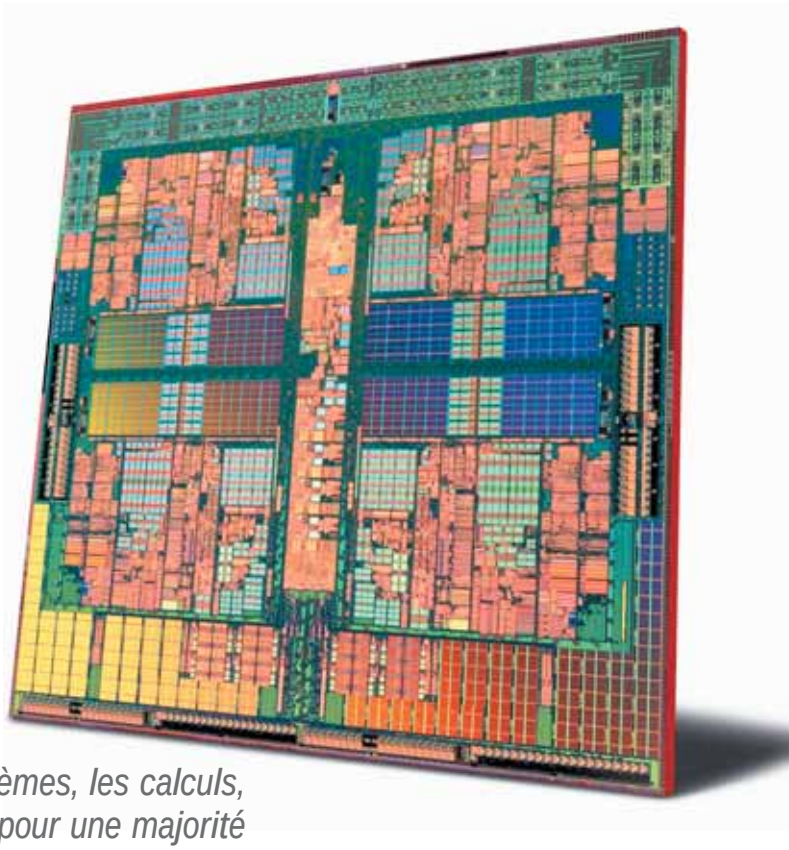
Mission :

- concevoir, développer, produire et intégrer des équipements et des systèmes stratégiques pour la sécurité des vols et la réussite des missions, pour les performances et l'efficacité économique des aéronefs civils et militaires
- offrir des services clients personnalisés

Emmanuel CHENU
Thales - Aerospace Division

Votre bureau nous intéresse, témoignez : redaction@programmez.com

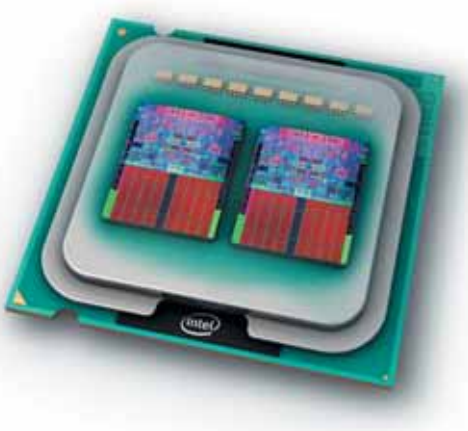
Du multicore pour vos applications



Il y a à peine 3 ans, les constructeurs misaient sur la multiplication des processeurs pour gagner en performances, ainsi que sur la course aux Ghz. Aujourd'hui, chaque processeur peut embarquer 2, 3, 4, ou plus, cœurs sur un seul processeur. Cette débauche de puissance brute est la bienvenue pour les applications gourmandes en ressources, les systèmes, les calculs, les rendus 3D, les jeux. Mais reconnaissons que pour une majorité d'utilisateurs et d'applications, le multicore reste un gadget.

Cependant, pour les éditeurs et les développeurs, ne pas considérer la programmmation multicore dans les applications serait une erreur. Car l'avenir va clairement dans cette voie. Encore faut-il savoir ce que l'on entend par là et comment on parvient à exploiter plusieurs cœurs dans son code sans pour autant perdre du temps à ré-architecturer un code existant. C'est un des défis des outils et langages. Comment faciliter la prise en compte des processeurs à plusieurs cœurs ?

Soyons clairs : aujourd'hui, ce Saint Graal n'existe pas ! Même si certaines solutions actuelles et futures faciliteront le travail.



Qu'entend-on par développement multicore ?

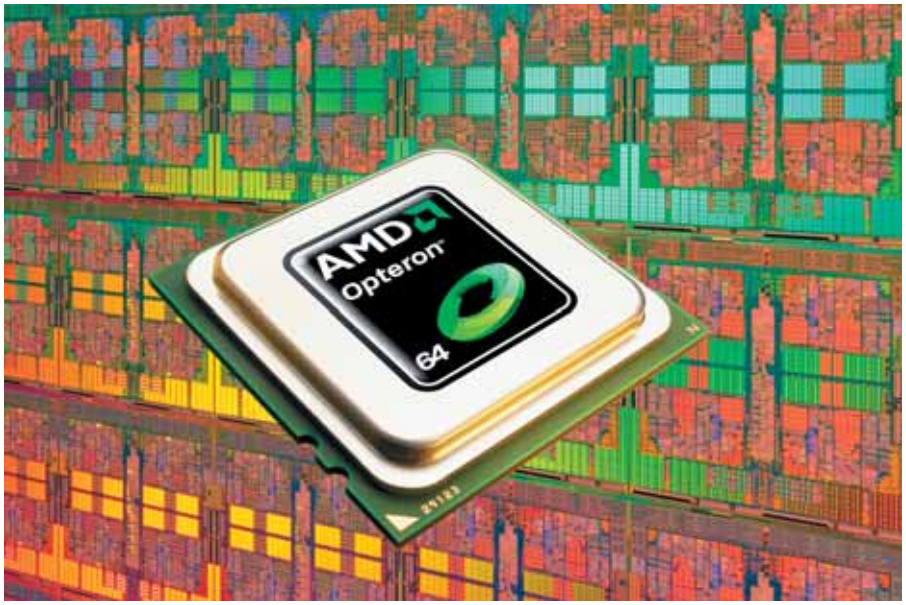
Il s'agit d'adapter ou d'écrire un code source capable d'exploiter plusieurs cœurs ou plusieurs processus, de répartir la charge, de monter en charge, etc. De bien exécuter les tâches, de la synchroniser entre les cœurs. Il y a quelques mots- clés à se rappeler : parallélisation du code, multithreading. Aujourd'hui, une application " moderne " devrait être multithreadée, c'est-à-dire posséder plusieurs tâches durant son exécution. Le multithread permet de découper son application en tâches et de les faire évoluer indépendamment les unes des autres. Cela permet de mieux exploiter les ressources matérielles et systèmes. Cependant, cela nécessite une architecture de thread bien définie, une coordination et synchronisation des threads, une gestion fine de ceux-ci. Et c'est là qu'il faut introduire la notion de programmation concurrente. Le parallélisme est un problème dépassant le code car il concerne les données, les tâches, le calcul, etc. C'est pour cela que l'on parlera facilement de parallélisation des données et des tâches.

Retenez déjà trois notions de bases : répartition, traitement des problèmes (pour arriver à une exécution exacte), et maintenabilité. La répartition (ou scalabity) est la technique permettant de répartir des données, des tâches entre plusieurs cœurs, 2, 4, 8, 16 et de les exploiter au mieux. En étant scalable, un code doit être capable d'exploiter un nombre de cœurs non prévus à l'origine par le développement et ce, sans devoir modifier le code. Mais encore faut-il déterminer le bon parallélisme pour les données et les tâches, car il en existe plusieurs, comme le pipeline ou exécution en cascade.

Race condition et deadlock

Mais il faut avouer que la programmation parallèle est aussi difficile à faire qu'à tester et traquer les problèmes. Déjà débugger une application à threads multiples n'est pas une partie de plaisir… On peut alors rencontrer deux problèmes d'exécution :

- la Race Condition : lorsque des tâches s'exécutent sur des processeurs ou cœurs différents et que l'on a une mauvaise coordination, cela peut fournir un accès concurrent à une variable avec modification de sa valeur pouvant aboutir à une erreur du résultat final. Très difficile à traquer, à déboguer
- Le Deadlock est un problème assez vicieux. Lorsque plusieurs tâches attendent la libération de ressources (de manière interdépendan-



te), il peut en résulter une attente infinie si les programmes (ou les tâches) qui sont interdépendants ne libèrerent pas les ressources.
Si on code " parallèle ", encore faut-il obtenir un code maintenable, capable d'évolution et d'être débogué. C'est là un problème que l'on aurait tendance à oublier trop rapidement. La notion de maintenabilité du parallélisme est conditionnée par les techniques de programmation pour créer ce parallélisme : faut-il coder en dur en C, C++, Fortran ou autre langage, ou faut-il introduire des notions d'abstraction.
L'abstraction en programmation parallèle permet de soulager le développeur d'une partie du code en évitant de devoir coder les interactions, l'architecture, la concurrence, etc. Il faut donc envisager l'utilisation de librairies, API, frameworks parallèles comme OpenMP ou Parallel FX.

Utiliser l'abstraction

S'il y a quelques années, la tendance était de coder en bas niveau le parallélisme, la tendance aujourd'hui est plutôt de procéder à un parallélisme abstrait, abstrait dans le sens : utiliser des frameworks, librairies prenant en charge tout ou partie du parallélisme dans mon application.
La première étape est d'identifier le code que l'on peut et souhaite paralléliser. Les exemples les plus fréquents sont le traitement des données, les calculs, l'imagerie, la vidéo, l'audio. Puis de trouver une bibliothèque répondant à notre besoin. N'oublions pas que tout ne peut pas être parallélisé et que cela n'aurait pas de sens. Cette abstraction permet une économie de temps, de code et offre un niveau de maintenance correct. Pour une parallélisation classique, comme sur des boucles for, une librairie de type OpenMP suffira largement. Et son usage se limite souvent à une ligne de code et/ou une modification de la boucle. Les librairies sont clairement l'avenir du parallélisme.

Et le compilateur alors ?

De nombreux compilateurs permettent d'optimiser un code pour le multithread ou encore le multiprocesseur / multicoeur. On pourra passer par des directives de compilation ou encore l'autovectorisation du compilateur qui est capable de détecter les parties du code utilisant les unités vectorielles des processeurs. Cela signifie que le compilateur détecte et vectorise le code en question afin de l'optimiser ainsi que les appels. On remplace le code normal par des instructions vecteurs. Sous GCC 4.x, on utilise le framework Tree-SSA, utilisant le SSA (Static Single Assignment).
On doit s'attendre à ce que les compilateurs supportent dans les prochaines années le parallélisme, comment ? Cela reste à définir. Mais il ne fait pas de doute non plus que les langages auront en standard, ou via des extensions, le support de notre parrallélisme et le compilateur aura pour tâche de résoudre le mapping entre les différents cœurs… La question est de savoir jusqu'où le compilateur pourra et saura aller. Faut-il sauter une étape et carrément initier des langages parallèles au lieu d'ajouter des extensions ?
Il existe déjà de tels langages plus ou moins avancés. On citera pour exemple : NESL, mpC. Il existe aussi de multiples travaux, concrets ou non sur l'encadrement et l'outillage du codage parallèle. Chez IBM, on peut évoquer le Model-Dirven Development Tool for Parallel Applications. Il s'agit d'une extension UML pour générer du code parallèle. On utilise alors des patterns sur lesquels s'appuient les modèles UML. Le code généré est du C++. Côté Eclipse, on peut citer Parallel Tools Platform et un des sous-ensembles : Parallel Language Development Tools. Ils permettent d'analyser le code MPI / OpenMP en C, C++ et Fortran, d'avoir un debugger parallèle, des outils de profiling, la possibilité de bâtir des applications MPI, etc.

Le mois prochain, nous continuerons notre découverte du monde merveilleux du multithread et du multicore.

■ **F.T.**

Quelles règles pour le parallélisme ?

Il n'y a pas véritablement de Bible sur le sujet. Cependant, voici quelques pistes, notamment préconisées par Intel :
- penser dès le départ " parallèle ". Ne pas coder une application parallèle comme une application normale.
- Définir les fonctions, parties du code à paralléliser
- Utiliser les bons outils, les bonnes bibliothèques
- Éviter d'user et d'abuser des verrous qui peuvent perturber le fonctionnement parallèle
- Ne pas oublier de gérer la mémoire et son allocation. Il s'agit d'une partie importante car une mauvaise gestion mémoire cause bien des soucis.

Une autre question peut aussi survenir : faut-il penser tâche (task) ou thread ? À cette question, nous n'avons pas de réponse, car cela dépend du contexte de l'application, si elle est juste multithread et non parallèle.

Parallel FX : la parallélisation facile ?

Pour l'éditeur, le multicore est un enjeu important pour améliorer les performances des applications et pour mieux utiliser les ressources matérielles désormais disponibles auprès des fondeurs tels que Intel et AMD. Les solutions proposées promettent beaucoup, reste à les concrétiser.

Aujourd'hui, la solution multicore pour paralléliser le code s'appelle Parallel FX (ou PFX) aussi connu sous son nom complet : Parallel Extensions for the .Net Framework. Il s'agit, comme son nom l'indique, d'un nouveau framework managé pour la parallélisation des données, des tâches, du scheduling ainsi que de la coordination sur du matériel adapté. PFX se veut un modèle le plus simple possible pour le développeur sans à avoir besoin (dixit la présentation) de jouer avec la complexité des threads ou encore la programmation concurrente. Cependant une bonne maîtrise de cela s'avèrera indispensable pour architecturer au mieux son projet. PFX inclut plusieurs librairies, les principales étant : Plinq et Task Parallel Lib.

Plinq

Plinq est un élément clé de PFX. Il permet d'optimiser les performances des requêtes Linq (à des endroits précis du code) sur plusieurs cœurs. Théoriquement, la montée en charge que procurera Plinq permettra des performances de traitement à la hausse avec l'accroissement du nombre de cœurs. Bien entendu, Plinq est intéressant uniquement sur des volumes conséquents de données et/ou des requêtes consommatrices de ressources. Plinq accepte les requêtes Linq-to-objects et Linq-to-XML (mais pas Linq-to-SQL et Linq-to-Entities). Il utilise tous les opérateurs Linq. Il suffit de rajouter deux éléments : assembly System.Concurrency.dll et l'encapsulation du source de données dans IparallelEnumerable<T> avec un appel à la méthode System.Linq.ParallelEnumerable.AsParallel. Cet appel doit s'assurer que le compilateur est bien relié à System.Linq.ParallelEnumerable. AsParallel et l'Enumerable de Linq.

Les buts de Plinq sont :

- appliquer une parallèlisation aux données dans l'exécution des requêtes Linq
- préserver le modèle de développement Linq, sans modifier fondamentalement l'interface
- on garde donc les opérateurs.

Plinq sera disponible aux développeurs C#, VB et C++ (à partir du .Net 3.5). Fonctionne sous Windows sur processeurs Intel et AMD, multicore de 2 à 64 processeurs.

Plinq propose 3 modes de traitement :

- Pipeline : le thread qui fait l'énumération est séparé des threads dédiés à l'exécution de la requête. On possède des threads de production et un thread consommateur.
- Arrêt / Reprise : là, tous les threads travaillent pour produire le plus rapidement possible la sortie. Lorsque la requête se termine, le thread d'énumération s'occupe de la sortie.On exploite mieux les processeurs (mieux que dans le pipeline).
- Enumération inversée : on fournit une fonction lambda à Plinq qui est exécutée une fois pour chaque élément de la sortie. On parallélise mieux que dans les autres solutions mais la gestion est plus lourde (avec notamment l'emploi d'une API ForAll).

Quel mode choisir ? Microsoft ne préconise aucun mode en particulier, tout va dépendre du contexte applicatif et des besoins Linq spécifiques. Car, selon son contexte, tel mode peut grever les performances au lieu de les améliorer.

Task Parallel Lib

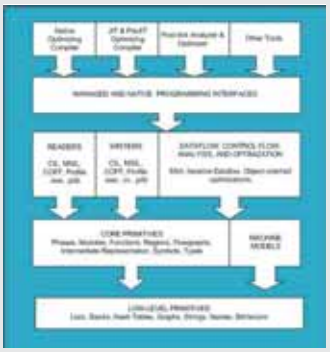
Autre composant important de PFX, la Task Parallel Lib ou TPL. À l'image de Plinq, TPL vise à simplifier la programmation parallèle en facilitant le codage. Son but : transformer rapidement un code séquentiel en code parallèle. Et les tâches parallèles s'exécuteront sur tous les processeurs disponibles. Cette librairie fonctionne avec le framework .Net 3.5 et ne nécessite pas d'extension de langage.

Boucle normale	Boucle parallèle

For (int i=0 ; i<100 ; i++) {
A [i] = a[i] * a[i] ;
}
	Parallel.For (0, 100, delegate(int i) {
	a[i] = a[i] * a[i] ;
	}

Parrallel.For est une méthode statique normale. Ce n'est pas au développeur de s'occuper de la répartition des tâches mais à TPL qui détecte la présence de plusieurs cœurs et se charge de faire la répartition pour exploiter la puissance disponible. Si l'application tourne sur une machine mono processeur, la boucle fonctionnera de manière séquentielle. Mais TPL n'est pas l'outil miracle et actuellement, la librairie ne garantit nullement une synchronisation correcte du code parallèle utilisant la mémoire partagée. À charge au développeur de vérifier si le code parallélisé peut l'être. Il ne fait pas de doute que le développeur verra dans cette librairie une aide précieuse.

Phoenix : le futur du compilateur selon Microsoft



Au-delà de PFX, l'éditeur travaille sur les prochains compilateurs qui auront un rôle bien plus important qu'aujourd'hui dans le parallélisme du code. Phoenix est un ensemble d'outils servant à cette prochaine évolution. Ce framework est une extensibilité du système pour créer des binaires et des assemblies de langages intermédiaires. Phoenix doit contribuer à construire des outils prédictibles, plus sécurisés. Cela passe une nouvelle génération de langages et de compilateurs.

Site : http://research.microsoft.com/phoenix/

Nouvelles Editions des Compilateurs Intel® C++ et Fortran 10.1

La nouvelle version offre le meilleur support pour créer des applications multi-thread sur Windows*, Linux* et MAC OS* X. Seules les éditions professionnelles des compilateurs Intel proposent l'étendue d'optimisations avancées et de capacités multi-threading qui incluent la vectorisation, l'auto-parallélisation, OpenMP*, récupération anticipées des données, déroulages de boucles, et les bibliothèques hautement optimisées de construction de blocs de threading, de traitement mathématiques et de multimédia.

Maintenant disponible sous forme de pack C++/Fortran

Nouveautés des compilateurs C++ et Fortran 10.1:

En plus des autres, support de Debian et Ubuntu
Support de OpenMP* Microsoft et de OpenMP* GNU
Tuning des processeurs: Penryn avec le switch -xS, Bonnel (enfouit) avec le switch -xL

Les compilateurs et bibliothèques sont faits les uns pour les autres:

Les Compilateurs Intel® C++ et Fortran parallélisent automatiquement votre code et l'optimisent en performances pour profiter au mieu des architectures multi-core.

L'Intel® Math Kernel Library met à votre disposition des fonctions mathématiques multi-threadées qui surpassent les performances de codes compilés individuellement ainsi que celles d'autres bibliothèques .

Les Intel® Integrated Performance Primitives (C++ seulement) incorporent des fonctions hautement optimisées qui accélèrent le développement du traitement de média, de cryptographie et du signal.

Les Intel® Threading Building Blocks (C++ seulement) regroupent des routines vérifiées et affinées pour simplifier le développement d'aplications multi-thread robustes et capables de monter en charge.

"D'ici 10 ans, un programmeur qui ne pense pas 'parallèle' ne sera plus un programmeur."

James Reinders
Evangéliste en Chef des Outils Logiciels Intel

intel Software Elite Reseller

microsigma

Votre VAR, partenaire à valeur ajoutée officiel

N°Azur 0 810 120 240

Tel 01 30 82 04 54 Fax 01 39 69 93 31
www.microsigma.fr/intel
intel@microsigma.fr

Votre spécialiste Fortran, C++, XML et outils de développement

Au service de ses revendeurs et de ses clients depuis 1984

Utilisation optimale des processeurs multi-core

Comment tirer pleinement parti de plusieurs processeurs ou du multi-core en limitant la réécriture du code ? Il faut que l'application soit capable de montée en charge tout en garantissant un ordre de traitement souvent indispensable dans des processus métier.

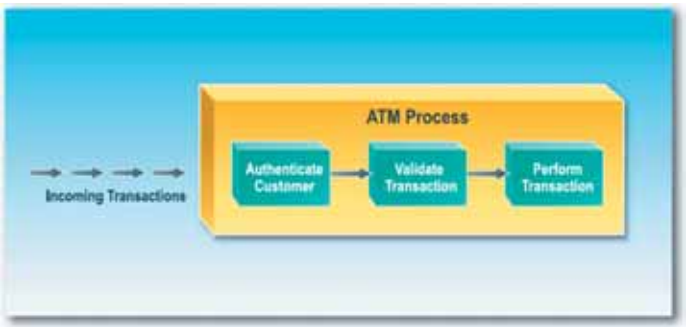
Il n'a pas fallu longtemps pour que les serveurs et ordinateurs de bureau soient équipés de processeurs à cœur multiples (de 2 à 4 actuellement). La puissance de calcul globale théoriquement disponible est ainsi augmentée. Dans les faits, un logiciel dont les performances dépendent uniquement de la rapidité du processeur sur lequel il s'exécute, ne tirera pas parti de ces nouvelles architectures. En effet, les fréquences d'horloge sont plus basses sur des architectures multi-core. Il faut donc avoir une autre approche.

Les solutions les plus couramment envisagées pour pallier ce problème sont de deux types :

- confier au système d'exploitation le soin de dispatcher les traitements sur les CPU disponibles.
- reprogrammer le code pour avoir un multi-thread propre, et du code parallèle.

Cependant, dans les processus métier hébergés par ces serveurs, on rencontre le plus souvent un flux de traitements devant être exécuté avec le plus grand débit possible, avec de surcroît le besoin d'une garantie partielle ou totale d'ordre dans les traitements. Dans ce contexte, les solutions habituelles sont peu satisfaisantes.

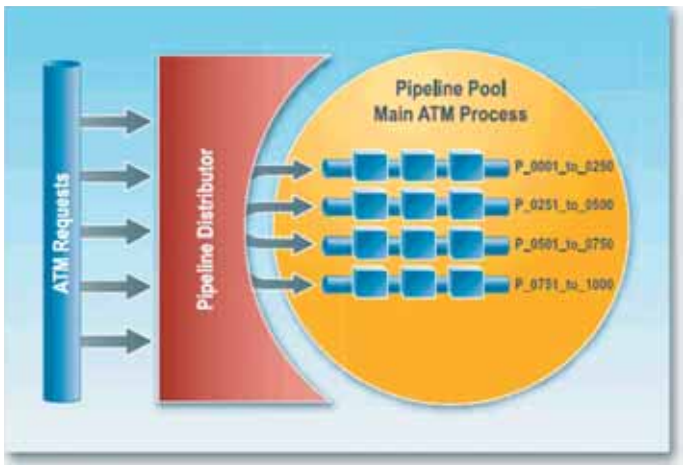
La délégation au système d'exploitation est bien adaptée pour exécuter de manière concurrente plusieurs applications n'ayant rien à voir entre elles, ce qui n'est pas le cas de figure des processus métier. La programmation multi-thread permet d'accélérer un traitement unitaire, mais n'apporte pas de solution pour augmenter le débit. Pire, un code déjà multi-thread sera plus lent une fois porté sur une machine multi-core, car les fréquences d'horloge sont plus basses.



Exemple de processus métier : distributeur bancaire de billets.

Pipeline Evaluation Expression	Pipeline Name
/atmtrans/[branch_id >= 0001 and branch_id <= 0250]	P_0001_to_0250
/atmtrans/[branch_id >= 0251 and branch_id <= 0500]	P_0251_to_0500
/atmtrans/[branch_id >= 0501 and branch_id <= 0750]	P_0501_to_0750
/atmtrans/[branch_id >= 0751 and branch_id <= 1000]	P_0751_to_1000

Exemple de règle de distribution fixe, basé sur le contenu du tag " branch_id ", lui-même sous le tag " atmtrans ".



Architecture pipeline associée à l'exemple de distribution.

La solution HydraSCA de Rogue Wave Software

L'éditeur propose une solution logicielle pour répondre à ces besoins sur les processus métiers : HydraSCA. L'atelier graphique livré avec HydraSCA permet de développer des services (au sens SOA) en intégrant éventuellement du code existant C++ ou Java ainsi que des Web Services, puis de déployer ces services sur les machines cibles.

L'outil implémente en natif un mécanisme dénommé " software pipelines ". Basé sur un concept de parallélisation sophistiquée, ce mécanisme assure une montée en charge sans réécriture de code, et garantit l'ordre de traitement souvent indispensable dans des processus métier. C'est ce mécanisme qui garantit la pleine exploitation des ressources multi-core et plus généralement des multiprocesseurs.

Les processus métier sont répartis à tous les niveaux disponibles : différents cores d'un processeur, différents processeurs d'une machine, et même différentes machines d'un parc (architecture cluster).

Le débit global est multiplié d'un facteur quasiment égal au nombre total de cores mis en œuvre.

Implémentation

Le processus métier à accélérer est supposé se présenter sous forme d'une suite de services (au sens SOA) appelés les uns après les autres.

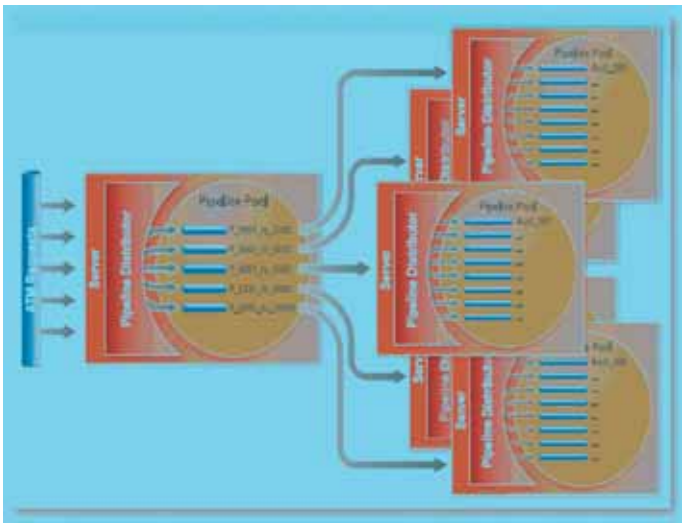
Le mécanisme " software pipelines " s'appuie sur les composants logiciels suivants : Un " Distributeur ", plusieurs " Pipelines " formant un " Pipeline pool ", un ou plusieurs " Agents ". Le distributeur est le point d'entrée du flux de traitement. Sa fonction est

de distribuer sur les pipelines le flux en fonction de critères de distribution. Un pipeline est un canal d'exécution de services, capable de traiter le flux qui provient du distributeur de manière ordonnée. Un pipeline pool désigne l'ensemble des pipelines associés à un distributeur. Un Agent est le run-time d'exécution de services, également réceptacle d'exécution du distributeur et de tout ou partie des pipelines.

Pour une petite configuration, on utilise un Agent unique, tournant sur un serveur, et hébergeant le distributeur et l'ensemble des pipelines. Pour une parallélisation massive, la configuration habituellement préconisée est la suivante : d'une part un serveur dédié à l'Agent hébergeant le distributeur, d'autre part autant de serveurs que nécessaire, chacun avec un Agent hébergeant une partie des pipelines. Le rôle de l'architecte système consiste à déterminer les éléments suivants : règle de distribution, nombre global de pipelines, répartition des pipelines sur les différents serveurs.

Règle de distribution

L'architecte définit les critères de distribution en choisissant parmi trois possibilités :



Architecture pipeline avec distribution multi niveau

- Implémentations standard telle que round-robin, load balancing : se définissent dans le paramétrage du distributeur.
- Critères de distribution fixes : permettent de distribuer en fonction de valeurs ou de tranches de valeurs présentes dans les champs du flux entrant. Se définissent simplement dans un fichier de configuration XML associé au distributeur.
- Critères de distribution dynamiques : plus souples, impliquent l'écriture de code implémentant l'algorithme de calcul de la règle de distribution.

Performances

On constate que le distributeur est un point de passage obligé pour toutes les requêtes entrantes. Sa performance conditionne donc la performance de l'ensemble. Rogue Wave Software a apporté un soin tout particulier à la mise au point de ce composant. En particulier, le temps de distribution est conditionné par la durée d'analyse du flux entrant (en général un flux SOAP) afin de déterminer vers quel pipeline router la requête. Aussi, le distributeur sait gérer deux modes d'analyse du flux entrant.

- Mode SAX : lecture " au fil de l'eau " du flux SOAP et décision d'aiguillage dès que les éléments discriminants ont été lus
- Mode DOM : chargement complet du flux SOAP en mémoire, permettant des calculs sophistiqués du critère de distribution.

Pour de très grosses configurations, il est possible de gérer une distribution multi niveau, en cascadant des distributeurs entre eux :

- Le premier étage distribue selon un critère simple rapide à calculer.
- Chaque pipeline alimenté par ce distributeur se contente de transférer à un distributeur secondaire.
- Les distributeurs secondaires gèrent chacun leur pool de pipelines selon un critère plus sophistiqué.

Conclusion

La solution présentée apporte une réponse pouvant aider à exploiter le multi-core avec les avantages suivants :

- Pleine utilisation des multi-core sans code spécifique
- Montée en charge par rajout de processeurs et simple reconfiguration
- Mode " cluster " (grappe de serveurs) en natif.

Jean Vidames

Les outils pour la parallélisation et le multithreading

Il existe de nombreux outils et librairies pour coder en parallèle. Difficile de dresser un tableau exhaustif. Nous avons retenu l'essentiel de l'offre actuellement disponible.

Les outils des fondeurs

Intel

Outils	Commentaires
CFinder	Outil gratuit pour checker le fonctionnement concurrent des threads d'une application. Permet de détecter le nombre de processeurs, les threads actifs.
Vtune Performance Analyser	Facilite le profiling et l'optimisation des applications. Autorise le tuning sur processeur 32 et 64, s'intègre à Visual Studio et Eclipse. Disponible sous Windows et Linux.
Compilers	Intel propose toute une gamme de compilations Linux, Windows et MacOS X supportant les derniers processeurs et le multithreading, supporte : OpenMP, auto vectorisation, etc.
Code Coverage Tool	Outil inclus dans les compilateurs. Il permet de voir le code non testé, non traité. Génération d'un rapport après analyse.
Thread Checker	Outil destiné à aider le développement à mettre au point des applications multithread. Supporte Windows et Linux, les processeurs multiprocesseurs.
Threading Building Blocks	Permet une montée en charge des threads grâce à cette librairie runtime en C++. Disponible sur Windows, Linux et MacOS X
Math Kernel Library	Librairie de fonctions mathématiques hautement optimisées
Integrated Performance Primitives	Librairie multimedia hautement optimisée.

Amd

Outils	Commentaires
CodeAnalyst	Suite d'outils pour analyser finement les applications et identifier les pistes d'optimisation.
Performance Library	Collection de routines applicatives pour accélérer les applications et le développement, le debuging. Une méthode " simple " pour passer en multithread.
Core Math Library	Collection de librairies mathématiques optimisées pour les processeurs AMD.
SimNow	Plate-forme d'émulation pour les processeurs AMD, facilite le débuging et le profiling des applications
GPU PerfStudio	Outil d'analyse temps réel pour amélioration les performances 3D et graphiques (sur cartes ATI).
PIX Plugin	Plug-in pour mieux supporter DirectX et détecter les problèmes de performances.

Divers

Outils	Commentaires
GCC 4.x	La 4 de GCC, le compilateur GNU, propose une autovectorisation du code. Cette technique aide à optimiser un code pour les processeurs actuels.
OpenMP	Librairie pour créer des applications parallèles, en mémoire partagée. Disponible sur de nombreux systèmes et langages.
MPI	Message Passing Interface. Protocole de communication utilisé par les applications parallélisées. Apporte une montée en charge et une disponibilité dans un cadre distribué. MPI est devenu un standard de facto. Il s'agit d'une spécification.
SMP Seesaw	Utilitaire Windows permettant de régler la charge entre deux processeurs.
Sun Studio 12	Cette version de la suite de développement de Sun inclut de nouveaux outils et compilateurs prenant en charge le multithreading dont un nouvel analyseur de Thread et des compilateurs C, C++ et Fortran pour la parallélisation, inclut le support OpenMP.
HydraSCA	Édité par Rogue Wave, cette solution inclut un mécanisme de software pipelines pour paralléliser le code sans réécriture.

NI LabVIEW 8.5 exploite la puissance du multicœur de façon automatique

Disponible depuis septembre dernier en version 8.5, l'environnement de développement LabVIEW de National Instruments se distingue notamment par un support du multicœur optimisé. Rappelons d'abord que LabVIEW est un langage de programmation graphique par flux de données. Il offre des structures de programmation classiques (boucles, séquences…) que le développeur agence graphiquement sur son écran. Cette approche, en plus d'être intuitive, favorise naturellement la parallélisation des tâches. Alors qu'en monocoeur, l'environnement se débrouille avec l'exécution linéaire pour rendre apparent le parallélisme, le recours au multicœur ouvre grandes les portes d'une exploitation optimale de ce parallélisme natif. En pratique, l'utilisateur à deux possibilités : soit il laisse à LabVIEW le soin d'analyser les flots de données parallèles et de répartir en conséquence, dynamiquement, les tâches entre les différents cœurs; soit il décide lui-même de cette répartition. Dans tous les cas, le mariage du multicœur et du caractère intrinsèquement parallèle du langage se traduit par des gains considérables en termes de performances (jusqu'à x1,98 pour 2 cœurs par exemple en mode automatique). Cette optimisation automatique ou non du multicœur est encore plus intéressante avec le module NI LabVIEW Real-Time, destiné au développement des applications déterministes fiables.

Pour davantage d'informations : www.ni.com/labview/fr

Et aussi une rubrique multicoeur : http://www.ni.com/multicore/

Pourquoi peindre avec les doigts ?

Visualisation Java pour clients riches et Ajax

ILOG JViews 8.1, la dernière version de la suite d'outils graphiques Java d'ILOG, couvre l'ensemble des fonctionnalités de visualisation avancée.

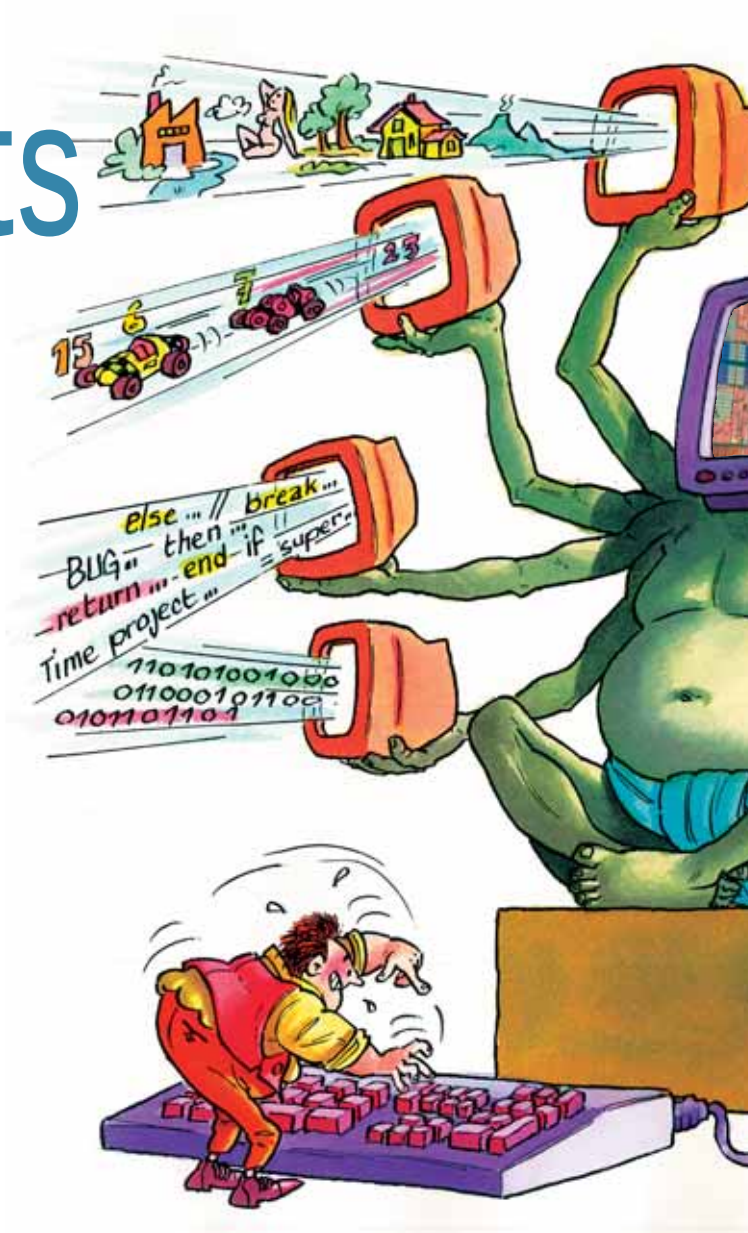
ILOG JViews 8.1 offre :

- Des composants graphiques puissants : diagrammes, courbes, tableaux de bord, cartographie, diagrammes de Gantt
- Des services évolués : agencement automatique de graphes, affichage performant pour des jeux de données volumineux
- Plusieurs techniques de déploiement : clients riches, applications Web interactives Ajax, Eclipse/RCP et portails
- Une expertise prouvée dans les industries les plus exigeantes : Informatique, télécoms, transport, énergie et défense.

Testez un de nos produits Java dès aujourd'hui http://jviews.ilog.com

ILOG
Changing the rules of business™

La programmation par composants



À quoi correspond réellement le développement par composants, ou avec composants, ou encore, orienté composants ? Dans ce grand dossier, nous allons vous apporter des exemples concrets de développement par composants (sous Java, PHP et Spring) mais aussi tenter d'en définir le cadre et de savoir à quelles évolutions on peut s'attendre.

Le salut par les composants ?

La notion de composants couvre une réalité diverse. On peut fournir les pistes suivantes :

- utilisation d'objets, de classes, prêts à l'emploi, de type Infragistics, Ilog, Software FX, etc. On les rajoute dans l'IDE de travail,
- usage de frameworks, de librairies,
- composants métiers internes ou externes : réutilisation du code

La notion de composants recouvre aussi une réalité que l'on oublie encore trop souvent : la réutilisation du code. Ainsi, un " code snippet " peut être assimilé à un composant, certes un peu spécial. La notion de réutilisation est importante car on ne réinvente pas la roue à chaque fois. On se constitue des bibliothèques de code, de modules, dans lesquelles on puise pour les nouveaux projets. On parlera même de développement piloté ou orienté composants ou POC. Le POC est souvent lié à la notion de réutilisation du code.

Toute la question est de savoir ce que l'on veut réutiliser et dans ce cas, définir la granularité du code pour être générique mais pas trop, spécifique, mais pas trop non plus. Il faut donc réfléchir au processus, à sa gestion. Il faut impérativement bien penser la chose.

Dans la programmation par composants, les frameworks sont particulièrement répandus. Ils sont la plupart du temps gratuits et /ou open source, et souvent matures et pérennes. Pour une même fonction, on peut trouver plusieurs frameworks. Leur valeur ajoutée tend à disparaître et ils deviennent des composants de commodité, bref, juste un élément technique facilitant la programmation en prenant en charge une partie du code, le développeur n'ayant plus qu'à créer la glue.

POC, DSL, Asset

Un des problèmes du développement par composants est le niveau de granularité. C'est-à-dire où intervient le composant. Nous pensons que dans la plupart des cas, le composant intervient à un niveau assez bon, au niveau code, objet et classe. Si on parle de librairie, de frameworks, on monte légèrement en abstraction pour arriver au middleware technique (bref, au-dessus du code). Cependant, nous restons assez bas. Pour être plus abstrait, donc plus haut, il faut changer de paradigme de développement. On parlera alors plus volontiers de composants métiers, de programmation orientée composants, voire de développement orienté Asset ou encore DSL (Domain Specific Language). L'aspirine n'est pas loin…

Même si vous vous dites " ce n'est pas mon domaine ", toutes ces approches impactent directement votre manière de coder, de structurer un projet. Bref, l'impact sur l'architecture est réel. L'architecture d'une application développée par composants ne sera pas identique à une application codée de bout en bout. L'architecte, le chef de projet ou le développeur définissent au préalable : les différentes couches de l'application, les composants rentrant dans le projet (quelle que soit la couche car le composant peut intervenir sur toutes).

Développement par Asset

L'Asset est une notion de plus haut niveau que le composant. Par Asset, il faut comprendre une collection comprenant des composants, les règles d'utilisation, des codes, de la documentation. Il constitue un point d'entrée

unique à partir duquel, on accède à l'ensemble des éléments de l'Asset. On peut aussi y trouver des cas de tests, les exigences, etc. Pour IBM, c'est une des tendances et ce vers quoi, le développement, en entreprise (?), ira dans le futur. L'éditeur propose depuis quelques années une solution : Rational Asset Manager. L'outil a pour objet de :

- retrouver facilement les Assets existants (recherche par index, intégration avec les IDE, bibliothèque d'Assets),
- créer et packager les Assets : supporte la réutilisation des Assets du cycle de vie (modèle, patterns, code, cas de tests, etc.,
- mise en avant des Assets

Cependant, s'il constitue une solution intéressante, il faut avouer que pour le moment, il n'a guère enthousiasmé les éditeurs.

Le DSL : l'ultime solution ?

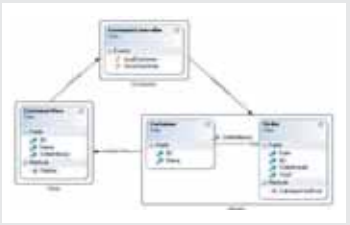
Les DSL, ou langage spécifique de domaine, constituent une tendance lourde depuis 1 – 2 ans chez quelques éditeurs comme IBM Rational et Microsoft. Pour faire simple, tout architecte, chef de projet, développeur peut créer son propre " langage " pour un besoin spécifique. Pour Microsoft, le DSL est une réponse aux développements de plus en plus complexes et surtout aux besoins spécifiques d'un développeur, d'une entreprise. Le DSL sera intégré dans le futur Team System Rosario. Cependant, à l'heure actuelle, l'outillage est trop rudimentaire, trop brut pour généraliser simplement l'usage du DSL.

■ **François Tonic**

MVC ou MVP ?

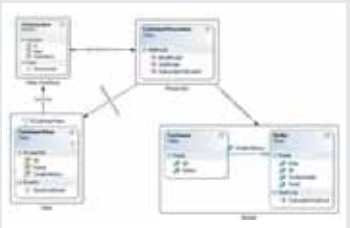
Pour de nombreux développeurs utilisant les design patterns et les bonnes pratiques, une question revient souvent : quelle différence entre le Model View Controller, ou MVC et le Model View Presenter, ou MVP ? La réponse n'est pas aussi évidente et elle s'avère complexe. Il y a plusieurs raisons à cela. Avant de vouloir comprendre les différences, examinons comment le pattern travaille et quels sont les bénéfices à l'utiliser dans un développement. Que ce soit MVC ou MVP patterns, ils sont abondamment utilisés dans le développement orienté objet pour séparer l'interface des couches métiers. Il y a de nombreux frameworks qui utilisent ce type de patterns : Struts, ROR, Microsoft CAB, etc.

Le pattern MVC



Le pattern MVC est un pattern de présentation d'interface (UI) ayant comme focus la séparation entre l'affichage (View) et la couche technique / business (Model). Le pattern sépare cela à travers trois composants : l'affichage s'occupe du rendu via les objets d'interface, le contrôleur a la responsabilité de répondre aux actions déclenchées par l'interface et le modèle s'occupe des comportements et des gestions des états / événements (bref le back office). Dans de nombreuses solutions, les trois éléments interagissent les uns avec les autres. On peut aussi avoir des variantes (ex. : Front Controller Pattern).

Le pattern MVP



Ce pattern est un pattern de présentation d'interface (UI) basé sur les concepts du pattern MVC. Là, nous avons 4 composants : l'affichage (view), l'affichage d'interface, le présentateur (presenter) qui fait le lien entre l'affichage d'interface et le modèle et enfin, le modèle. Dans certaines implémentations, on dispose d'un contrôleur servant de couche de persistance.

Le bénéfice

Avant d'utiliser un pattern, le développeur doit considérer les avantages et inconvénients. Il y a bien entendu des avantages à utiliser MVC ou MVP. Une des contraintes est la phase d'apprentissage et de maitrîse. Il faut utiliser le bon pattern pour le bon problème, pour la bonne architecture :

- l'indépendance de l'interface du modèle est une bonne chose.
- On sépare clairement le travail de chaque couche, de chaque composant. Facilitant la maintenance, l'évolution de l'application
- Mieux tester. Une telle séparation facilite l'écriture et l'exécution de tests unitaires.
- On favorise la réutilisation de code, notamment grâce à la séparation code / interface.
- On force le développeur à créer une véritable couche d'accès aux données.
- Être plus flexible, plus adaptatif aux évolutions.

■ **Todd Snyder** (développeur web chez Infragistics)

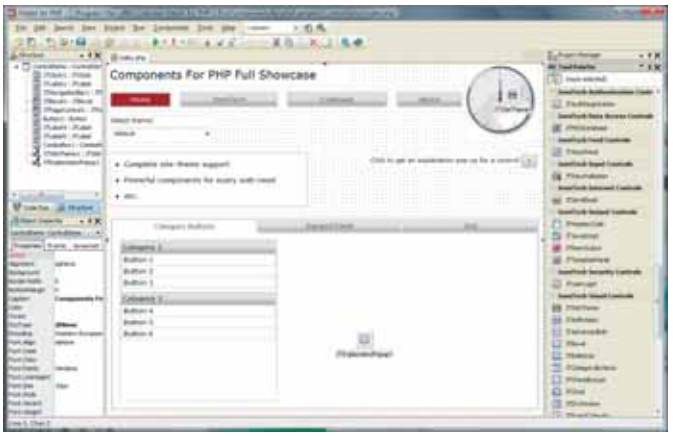
Panorama des composants

Il existe de nombreuses solutions open source ou commerciales proposant des composants clé en main. Un très grand nombre se concentre sur les objets d'interfaces pour les applications et le web. Il n'y a pas de règle pour choisir le composant idéal. À vous de tester et d'adopter…

N'oubliez pas que les composants concernent aussi les plug-in, add-in, librairies, composants d'objets et les frameworks.

PHP

Comme de nombreux langages et IDE, PHP possède une grande variété de composants open source et commerciaux. Voici quelques exemples :
JTC : suite de composants PHP utilisables avec CodeGear Delphi for PHP. Elle comprend plusieurs dizaines de composants pour l'interface, les menus, boutons, barre de progression, etc.



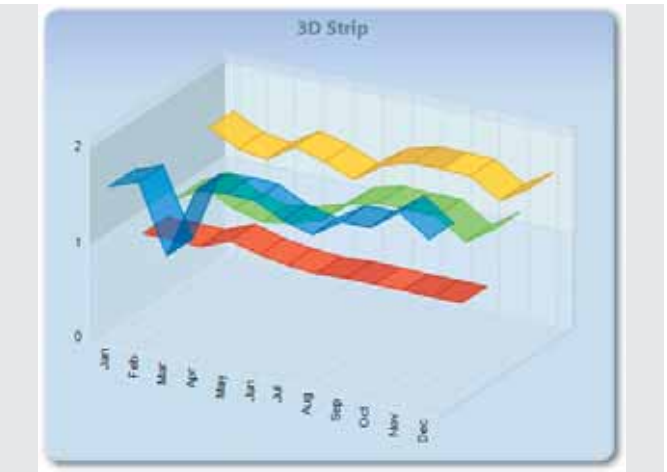
eZ Components : suite de composants d'entreprise pour PHP. Permet de réduire le temps de développement. On dispose de nombreux composants.
Il existe aussi de nombreux frameworks MVC pour PHP : Zend Framework, Limb, EZ Publish, LogiCreate, Phrame, Akelos, phpDrone, Cake, Castor, Inek, Ismo, Nexista, Copix, Prado, SolarPHP, Qcodo, struts4php, Zoop.

Infragistics

NetAdvantage : Le leader des composants Windows / .Net améliore très régulièrement ses composants. Il fonctionne sous COM, ASP.Net et Windows Forms. Supporte les applications mobiles (et le TabletPC), web et desktop. L'ensemble des composants reprend le look & feel Windows et Office. Pour unifier la base technique des différents composants, Infragistics a développé son propre framework : Presentation Layer Framework, son objectif est de minimiser le code et de pouvoir réutiliser facilement les objets. .Net 2 et Visual Studio 2005 sont supportés ainsi que depuis peu Ajax. Les sources des composants sont disponibles en C#.
JSuite : composants JFC et JavaBeans, se basant sur Swing afin de développer des applications Java avec des interfaces riches. Très complet, ce pack permet de simplifier le développement Java. Existe aussi NetAdvantage for JSF, composant Ajax pour JSF.
NetAdvantage for WPF : suite de composants spéciale WPF ! On dispose d'une dizaine d'objets d'interface. À noter qu'Infragistics prépare aussi des composants Silverlight 1.1.

Software FX

Autre éditeur bien connu du monde des composants, Software FX propose deux gammes de produits : ChartFX et GridFX. Aujourd'hui, l'éditeur supporte .Net, Java, Com et tout dernièrement Silverlight permettant d'utiliser ensemble les deux entités grâce à des add-on ChartFX.

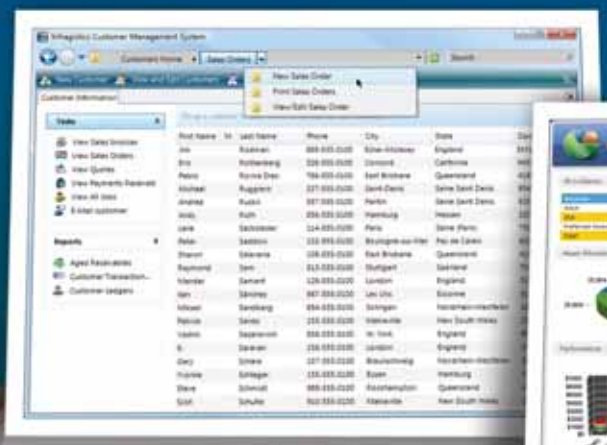


ChartFX : Écrit dès le départ nativement en C#, ChartFX s'utilise aussi bien avec VS 2005 qu'avec les versions précédentes ou Borland C# Builder. Il est orienté, comme son nom l'indique, affichage de données, données. Il est expert en diagrammes et en manipulation de données XML.
GridFX : suite de composants exclusivement composée de grilles personnalisables offrant de nombreuses fonctions : intégration avec ChartFX, sidebar, intégration des barres d'outils et palettes flottantes, fonctions Ajax intégrées, etc.

Ilog

Diagrammer pour .Net : outil de création de diagrammes simples et complexes que l'on intègre dans les Win et Web Forms. On peut créer toutes sortes de représentations. Il est compatible avec ASP.Net Ajax et dispose de trois outils d'édition pour construire des éditeurs de modélisation, de processus métiers et de diagrammes UML. Il dispose aussi d'un SDK incluant des exemples et est accessible à tous les langages .Net.

"Des **résultats** efficaces pour une productivité accrue –
autant pour nos **développeurs**, que pour nos **utilisateurs**."
- Responsable de projet



Navigation intuitive Vista™ Breadcrumb pour
Windows Forms NetAdvantage pour ASP.NET et Gauges

NetAdvantage pour ASP.NET Graphiques et Gauges

NetAdvantage for .NET 2007 Vol. 3

La boite à outils exceptionnelle pour la conception et le développement de vos interfaces utilisateur

Windows Forms

Une expérience consistante pour les utilisateurs de Microsoft Office - D'Office 2007 et du ruban à Windows® Vista et les innovations du Navigation Breadcrumb, nos contrôles pour interfaces utilisateur métamorphosent votre application avec des expériences utilisateur sans précédent

ASP.NET

Afficher plus de données, afficher du savoir - Produisez des documents PDF/XPS de haute fidélité à partir de vos applications AJAX autorisant vos utilisateurs à collaborer, partager et les laisser transférer les données grâce aux fonctionnalités importer/exporter d'Excel

Des Graphiques et Gauges époustouflants

Concevez des tableaux de bord professionnels - Des graphes et Gauges de hautes fidélités pour ASP.NET et Windows Forms fournissant instantanément une compréhension limpide des Key Performance Indicators

Application Styling™

Design Once, Style Everywhere - Créez rapidement et appliquez vos normes corporatives à travers les contrôles et applications pour Windows Forms et ASP.NET conçus avec NetAdvantage

Document Exporting Engine

Des rapports pour exécutifs - Notre composant PDF/XPS Document Exporting pour Windows Forms et ASP.NET permet de créer des documents imprimables et des impressions magnifiquement formatées à partir de nos contrôles interfaces utilisateur

Pour de plus amples informations:
infragistics.com/dotnet

sales-europe@infragistics.com

N° Vert 0800 667 307

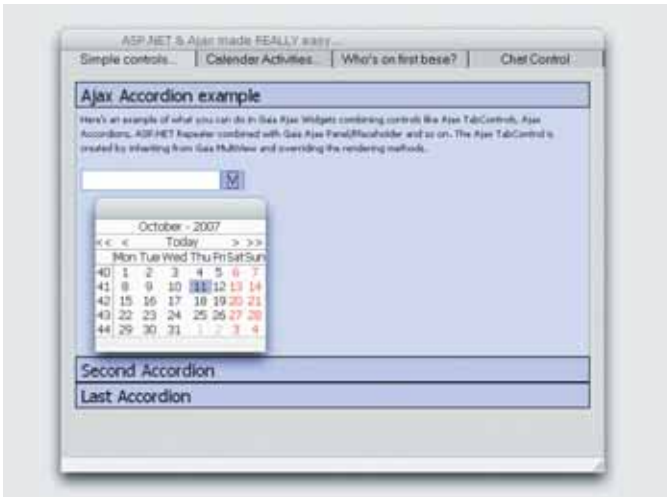
Infragistics®
Powering The Presentation Layer

WINDOWS FORMS ASP.NET WPF JSF

grids scheduling charting toolbars navigation menus listbars trees tabs explorer bars editors & more

Jviews : ensemble de composants et d'outils Java pour créer des applications riches. Cette version améliore encore le support d'Ajax dans l'outil (support apparu début 2006). Cette amélioration porte sur la cartographie. Surtout, la version propose une édition interactive des diagrammes et graphiques à partir d'un navigateur. D'autre part, Jviews 8 inclut deux nouveaux outils : Symbol Editor (éditeur de symboles) et Dashboard Editor (éditeur de synoptiques et de tableaux de bord pour simplifier la création d'interface de supervision). Les tableaux de bord peuvent être déployés en client riche sur les postes clients ou sur un navigateur.

Gaïa



Cet éditeur propose des widgets Java, pour les développeurs ASP.Net. Compatible avec les principaux systèmes et navigateurs, Gaïa propose de nombreux composants prêts à l'emploi : bouton, grilles, menu, listes, arbres, etc. Version d'essai disponible.

IdentityMine

Blendables : 10 composants dédiés au monde WPF s'intégrant à Expression Blend. Permet d'exploiter les fonctions avancées de WPF de l'interface.
À tester d'urgence ! Un peu cher mais efficace !

ComponentOne

Autre éditeur bien connu du monde des composants, ComponentOne propose les mêmes tendances que ses concurrents : Silverlight, Ajax. L'offre principale est toujours Studio Enterprise. Il s'agit d'un ensemble de composants d'interfaces pour les tableaux, grilles, boutons, arbres, etc. Cette gamme propose des éditions WPF, ASP.Net, WinForms, ActiveX et Mobile (pour Compact Frameworks).

Divers		
PowerTCP PowerTCP	Commercial	Composants orientés communication et sécurité Web (SSL, FTP…)
Indy Sockets Nevrona	Open Source	Suite pour TCP/IP. Compatible avec VB.NET, Delphi, Kylix et C++ Builder
Gamme ultimate Dundas Software	Commercial	Gamme de suites de composants MFC / C++. Le Ultimate Toolbox comprend une centaine d'objets. Existe aussi un Ultimate Grid et TCP/IP
PDFKIT.NET Tall Components	Commercial	Composants pour créer et gérer des documents PDF sous .NET. Vient en complément de PDF.NET. Remplace une précédente offre.
AcxChart / acxImage JavaSide.com	Gratuit	Composants pour créer des graphiques et des images en ASP.NET.
Gamme Xtreme Toolkit Codejock software	Commercial	Ensemble de composants .NET et VC++. Large choix d'objets d'interface.
Tx Text Control The Imaging Source Europe	Commercial	Composant pour implémenter des fonctions bureautique dans une application. Supporte d'autres langages que .NET.
Dundas pour .NET Dundas Software	Commercial / Open Source	Divers composants .Net ou ASP.NET dont un grapheur. L'éditeur propose aussi plusieurs composants Open Source
Free SMTP.NET Quicksoft	Gratuit	Composant SMTP pour créer et envoyer des mails. Ecrit en C#.
Jclass Quest Software	Commercial	Composants pour créer des tableaux, graphs, reporting et impression. Il existe aussi les déclinaisons serveurs et desktop.
Formula One Tidestone	Commercial	Outil pour publier les données sur le Web via des tableaux.
Mabry Mega Pack Mabry	Commercial	Contient 22 composants de toute sorte (avec version OCX et VBX). Existe aussi en version .NET
Dataconnect for ADO Datadirect Technologie	Commercial	Composant de connexion ADO. Existe aussi en .NET
SmartUI Xceed	Commercial	Ensemble de composants de style Office 2000 / XP. Création par code et visuellement. Objet d'interface essentiellement. L'éditeur propose aussi bien d'autres composants (compression, Internet…)
EasyMail Objects Quicksoft	Commercial	Ensemble d'objets dédiés aux fonctions de Mail. Version .NET disponible.
ExpressPack for VCL Developer Direct	Commercial	Pack de composants pour Delphi orienté interface et contrôle d'interface. Toute une gamme de composants est proposée (existe pour d'autres IDE et langages).
Rave Reports Nevrona	Commercial	Composants de reporting. Support VCL et CLX.
ICE Nevrona	Open Source	Kit pour créer des installeurs. Fonctionne aussi avec C++ Builder.

François Tonic

Composant d'accès aux données avec Spring

Dans cet article, nous allons détailler la manière de mettre en oeuvre un composant d'accès aux données avec le framework Spring, aussi bien au niveau de la définition du contrat, des implémentations que de l'assemblage de ses constituants afin de l'utiliser dans une application. Nous verrons alors comment tirer parti des fondations du framework telles que l'injection de dépendances et la programmation orientée aspect afin de programmer par contrat avec Spring.

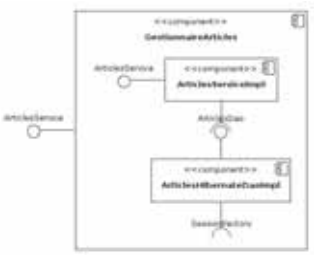
La programmation orientée composant vise à définir clairement les entités fournissant un certain nombre de fonctionnalités. On parle à ce niveau de service mis à disposition par le composant. De plus, le contrat de ce dernier est défini par l'intermédiaire de son interface regroupant la liste des méthodes utilisables. Cette approche offre également la possibilité de complètement décorréler le contrat du composant de son implémentation. Cette dernière correspond aux différents mécanismes internes du composant mis en oeuvre pour fournir le service, les utilisateurs du composant n'ayant pas connaissance de ces mécanismes.

L'objectif principal de cette approche consiste dans la modularisation et la centralisation des traitements afin de les réutiliser à différents niveaux et dans différentes applications. L'approche fondée sur les composants se positionne donc complètement dans la philosophie DRY (Don't Repeat Yourself) visant à réduire la duplication de code afin de faciliter l'évolutivité et la maintenabilité des applicatifs.

Au niveau de leur utilisation, les composants peuvent aussi bien s'intégrer dans les applications qu'être accédés à distance par l'intermédiaire de serveurs de composants distribués. Le point capital à ce niveau consiste en la distribution et la maintenabilité des versions des composants, notamment dans le cas d'une utilisation par plusieurs applications.

Le composant mis en oeuvre ici permet d'accéder à des informations relatives à des articles et stockées dans une base de données. Ce composant possède une granularité importante et sera donc composé de plusieurs POJO (classes Java simples) afin de le structurer. Ces derniers correspondent à un service métier et une entité d'accès aux données (Data Access Object ou DAO).

La figure suivante décrit la structure du composant de gestion des articles, composant dénommé *GestionnaireArticles*.



Comme le montre la figure, le composant GestionnaireArticles expose l'interface ArticlesService du service métier. En interne, le composant se fonde sur l'interface SessionFactory d'Hibernate, framework utilisé ici afin d'accéder à la base de données. Tous les constituants du composant global se fondent sur l'injection de dépendances, brique de base de Spring, afin d'intéragir les uns avec les autres.

Contrat du composant

Nous allons tout d'abord expliciter le contrat global de notre composant par l'intermédiaire d'une simple interface Java.

Comme le composant de notre article permet de manipuler les données d'articles dont les valeurs sont contenues dans une table article, ses traitements internes s'appuiront sur une classe Java Article. De son côté, l'interface ArticlesService met à disposition les différentes méthodes standard de manipulation d'articles (CRUD: Create Retrieve Update Delete), comme l'illustre le code suivant :

```
public interface ArticlesService {
    List<Article> getListeArticles();
    Article chargerArticle(long articleId);
    void ajouterArticle(Article article);
    void modifierArticle(Article article);
    void supprimerArticle(Article article);
}
```

Notons que, pour simplifier, les traitements du service métier délèguent uniquement ses traitements vers ceux du DAO (Data Access Object), l'interface du DAO étant ainsi similaire à celle du service.

Implémentation du DAO

Concentrons nous maintenant sur la mise en oeuvre des traitements relatifs à l'accès aux données dans le DAO du composant, traitements se fondant sur les frameworks Hibernate et Spring.

La classe du DAO doit implémenter l'interface ArticlesDAO et peut également étendre la classe HibernateDaoSupport de Spring afin de bénéficier directement de la méthode d'injection de dépendances pour la SessionFactory tout en ayant accès à la classe centrale du support d'Hibernate de Spring, le template Hibernate. Le code suivant illustre la structure de la classe ArticlesHibernateDaoImpl :

```
public class ArticlesHibernateDaoImpl
                    extends HibernateDaoSupport
                    implements ArticlesDao {

    public List<Article> getListeArticles() {
        (..)
    }

    (..)
}
```

Les entités principales des supports d'accès aux données de Spring sont les templates. Ces derniers encapsulent toute la tuyauterie technique, répétitive et souvent source d'erreurs des technologies d'accès aux données. Ils s'appuient sur les fabriques des technologies supportées afin de créer les ressources pour interagir avec la base de données.

Dans le cas d'Hibernate, cette fabrique correspond à l'interface SessionFactory, entité permettant de masquer l'utilisation de JDBC tout en mettant en oeuvre la correspondance entre les objets et les tables de la base de données (Object Relational Mapping).

Ces templates ont donc la responsabilité de gérer l'utilisation correcte des ressources (acquisition et relâchement) afin de garantir notamment la libération de ces ressources après leur utilisation. Les développeurs de composants d'accès aux données peuvent alors se concentrer sur les traitements spécifiques à leur application, traitements correspondant dans le cas d'Hibernate à la persistance des objets et à des requêtes objet avec le langage HQL.

Les templates exposent généralement les différentes méthodes offertes par les API sous-jacentes, celles de la session Hibernate dans notre cas.

Le code suivant décrit un exemple de mise en oeuvre des principales méthodes de la classe HibernateTemplate, le template dédié au framework Hibernate, dans la classe d'implémentation du DAO du composant :

```
public List<Article> getListeArticles() {
    return getHibernateTemplate().find(
                        "from Article article");
}

public void ajouterArticle(Article article) {
    getHibernateTemplate().saveOrUpdate(article);
}

public Article chargerArticle(long id) {
    return (Article)getHibernateTemplate().load(
                        Article.class, new Long(id));
}
```

Si toutefois un traitement non prévu doit être implémenté, les templates permettent la mise à disposition, par le biais d'interfaces de rappel, des ressources sous-jacentes, comme l'illustre le code suivant qui permet d'utiliser une session Hibernate par le biais de l'interface Hibernate Callback :

```
getHibernateTemplate().execute(new HibernateCallback() {
    public Object doInHibernate(Session session)
            throws HibernateException, SQLException {
        // Utilisation de la session Hibernate
        return null;
    }
});
```

D'une manière générale, les développeurs doivent bien faire attention de ne pas utiliser les ressources mises à disposition en dehors des méthodes de rappel car celles-ci peuvent être relâchées implicitement par Spring, le framework ayant en effet la responsabilité de gérer ces ressources.

Au niveau des exceptions techniques, Spring suit la tendance actuelle de définir les exceptions techniques sous forme d'exceptions non vérifiées (exceptions héritant de RuntimeException).

Afin de découpler les exceptions de la technologie utilisée, Spring met à disposition un ensemble d'exceptions DAO génériques dont la classe mère est DataAccessException.

Ces exceptions sont intégrées directement aux traitements des templates qui ont la responsabilité de les créer et de les remonter tout en conservant les exceptions d'origine par l'intermédiaire d'un mécanisme de chaînage d'exceptions. Ainsi l'accès à l'exception d'origine de l'erreur est toujours possible.

Assemblage et configuration

Au moment de l'assemblage des constituants du composant, nous devons configurer les dépendances entre les POJO au moyen de l'injection de dépendances, comme l'illustre le code suivant :

```
<bean id="articlesService" class="ArticlesServiceImpl">
    <property name="articlesDao" ref="articlesHibernateDao"/>
</bean>

<bean id="articlesHibernateDao" class="ArticlesDaoHibernateImpl">
    <property name="sessionFactory" ref="sessionFactory"/>
</bean>
```

La classe ArticlesHibernateDaoImpl se fonde sur une SessionFactory, la fabrique d'Hibernate permettant de mettre à disposition des ressources afin d'accéder à une base de données relationnelle. Spring offre la classe LocalSessionFactoryBean afin de la configurer simplement comme l'illustre le code suivant :

```
<bean id="sessionFactory"
    class="org.springframework.orm.hibernate3.LocalSessionFactoryBe
    <property name="mappingResources">
        <list>
            <value>Article.hbm.xml</value>
        </list>
    </property>
    <property name="hibernateProperties">
        <props>
            <prop key="hibernate.dialect">
                org.hibernate.dialect.HSQLDialect</prop>
            <prop key="hibernate.show_sql">true</prop>
        </props>
    </property>
    <property name="dataSource" ref="dataSourceJndi"/>
</bean>
```

Vous pouvez remarquer que cette entité permet de spécifier différents éléments de configuration d'Hibernate tels que le dialecte et les fichiers de mapping. Cette entité se fonde elle-même sur une DataSource JDBC qui peut correspondre notamment à un pool de connexion embarqué ou accessible via JNDI. Le code suivant illustre la configuration de cette dernière approche avec l'espace de nommage jee de Spring :

```
<jee:jndi-lookup id="dataSourceJndi" jndi-name="jdbc/ds"/>
```

Lors de la mise en oeuvre de l'accès aux données, les transactions correspondent à un point essentiel puisqu'elles permettent de garantir l'atomicité de traitements ainsi que la cohérence des données lors de modifications. Pour le moment, aucun traitement relatif à cet aspect n'a été mis en oeuvre puisque Spring offre la possibilité d'ajouter des comportements transactionnels au moment de l'assemblage des composants par l'intermédiaire de la programmation orientée aspect. Cet aspect s'appuie sur les mécanismes intégrés au sein des templates d'accès aux données.

Nous pouvons ainsi spécifier que les méthodes de modification (ajouter Article, modifierArticle et supprimerArticle) du composant sont transactionnelles, tandis que les méthodes de lecture (getListeArticles et chargerArticle) ne le sont pas. Le code suivant illustre la configuration du gestionnaire de transactions de Spring, puis des comportements transactionnels des méthodes et enfin de l'application des transactions sur le service métier :

```
<bean id="hibernateTransactionManager"
    class="org.springframework.orm.hibernate3.HibernateTransactionManager">
    <property name="sessionFactory" ref="sessionFactory"/>
</bean>

<tx:advice id="txAdvice"
    transaction-manager="hibernateTransactionManager">
    <tx:attributes>
        <tx:method name="ajouter*"/>
        <tx:method name="modifier*"/>
        <tx:method name="supprimer*"/>
        <tx:method name="*" read-only="true"/>
    </tx:attributes>
</tx:advice>

<aop:config>
    <aop:advisor pointcut="execution(* *..*ServiceImpl.*(..))"
                 advice-ref="txAdvice" />
</aop:config>
```

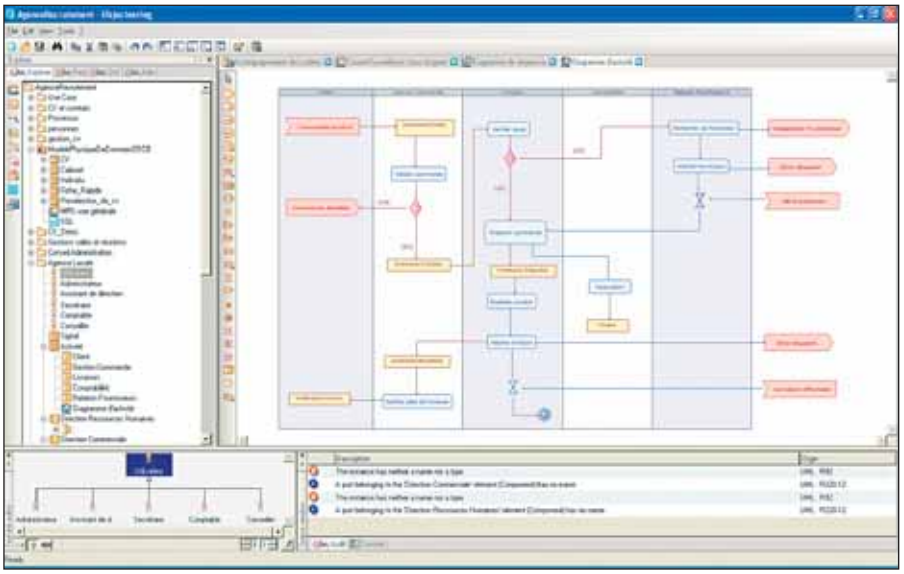
Objecteering 6.1

NOUVELLE VERSION

Le développement guidé par le modèle

Objecteering 6.1 optimise MDA et UML2 pour générer un code d'un haut niveau d'expertise : il maximise la productivité et la qualité des développements en Java, C++ ou C#.

Comment tirer parti au mieux de la modélisation UML à des fins de production automatisée d'un code de qualité, maintenu en cohérence avec le modèle ? Comment guider les développeurs dans leur modélisation et optimiser la production de code pour des architectures orientées services (SOA) s'appuyant sur des frameworks complexes ? L'approche MDA qui consiste à exploiter le modèle par des mécanismes de transformation répond précisément à ces problématiques en assurant également la traçabilité entre le code généré, le modèle dont il est issu et les exigences qui le justifient. Avec Objecteering 6.1, Objecteering Software met à disposition des développeurs une nouvelle génération d'outils de développement guidés par le modèle, en s'appuyant sur les dernières avancées de MDA et de UML2.



Objecteering 6.1 : Diagramme d'activité.

L'expression des besoins intégrée à la modélisation UML

Réussir l'expression des besoins de votre application est le point de départ fondamental pour le succès du développement. Objecteering 6.1 intègre la gestion des exigences avec la modélisation UML. Vous démarrez dès la phase de définition des exigences, et poursuivez sans rupture jusqu'aux modèles d'analyse et conception. Vous pouvez ainsi obtenir un modèle des exigences complet qui vous permettra d'aborder les étapes d'élaboration de votre application sur des bases solides et justifiées par les besoins.

Le maintien en cohérence exigences / modèle / diagrammes / code / documentation

La génération automatique de code pour les cibles Java/J2EE, C# .Net, C++, Corba, Fortran ou SQL, supportant des frameworks tels que Spring, JSF, Struts et Hibernate ou des frameworks spécifiques comme les architectures SOA apporte des gains substantiels en qualité et en productivité. La génération de documentation permet de fournir à chaque type d'acteurs des documents pour les exigences, le modèle et le code. Le référentiel unique est garant de la cohérence, de la traçabilité et de la non redondance des informations.

Une réelle assistance à la construction des modèles

Avec UML2.1 le standard de l'OMG est devenu un langage riche et complet pour couvrir le besoin en modélisation d'entreprise et de systèmes techniques.

Il est de ce fait aisé de commettre des erreurs de modélisation, et d'obtenir des modèles inconsistants.

Objecteering 6 est doté d'un éditeur graphique UML sensitif qui assiste l'utilisateur à construire des modèles corrects dès le début.

L'audit de modèle qui vérifie en temps réel 282 règles sémantiques permet en outre d'assurer la cohérence du modèle dans sa globalité, y compris dans le cadre d'un travail collaboratif sur un modèle partagé.

Nouveau ! Le support BPMN et l'architecture d'entreprise

Objecteering 6.1 intègre le support de BPMN pour modéliser les processus métier en liaison avec le modèle UML. Avec l'extension Objecteering EA, l'architecture d'entreprise, la cartographie des processus, l'urbanisation du SI sont intégrés en continuité dans la modélisation UML, pour adresser l'ensemble des participants à la définition du SI de l'entreprise.

Nouveau ! Une API Java de paramétrage MDA

Avec Objecteering MDA Modeler, vous modélisez vos extensions UML, vous définissez vos transformations de modèle, vos générateurs de code et vous implémentez en Java des comportements dédiés. L'API Java très riche permet de naviguer dans le modèle, de le transformer et de piloter Objecteering 6.1, son IHM, ses diagrammes pour que l'atelier soit dédié à votre environnement, qu'il génère un code spécifique et guide les modélisateurs selon votre méthodologie. Des wizards dédiés permettent de minimiser la programmation Java, et réduisent l'apprentissage du mode de paramétrage de l'outil. La puissance de MDA combinée à la richesse de Java et aux capacités de Objecteering 6.1 décupleront la productivité et la qualité de vos développements.

Objecteering SOFTWARE

Venez découvrir

Objecteering 6.1

Nouvelle Version !

Télécharger sur

www.objecteering.com

Pour plus d'information contactez-nous au

01 30 12 16 60

Notons que, par défaut, le comportement transactionnel est REQUIRED, comportement correspondant à une transaction dont le début se situe avant l'appel de la méthode et la fin après son retour. L'attribut read-only spécifie quant à lui l'absence de transactions tout en affectant à la méthode une session dédiée pour ses traitements.
Nous pouvons utiliser notre composant tel quel en nous fondant sur le bean Spring d'identifiant articlesService en l'injectant dans un autre bean. D'autres approches peuvent être néanmoins mises en oeuvre afin de mettre à disposition le composant de manière distribuée et ce uniquement par la configuration.
Nous pouvons citer notamment l'utilisation des technologies XFire (Web services) et RMI (appel d'objets distants), dont le code suivant décrit les configurations :

```
<!-- Accès via RMI -->
<bean
    class="org.springframework.remoting.rmi.RmiServiceExporter">
    <property name="serviceName" value="ArticlesService"/>
    <property name="service" ref="articlesService"/>
    <property name="serviceInterface"
              value="service.ArticlesService"/>
    <property name="registryPort" value="1099"/>
</bean>

<!-- Accès via un service Web -->
<bean name="/articlesService"
    class="org.codehaus.xfire.spring.remoting.XFireExporter">
    <property name="serviceInterface"
              value="org.codehaus.xfire.spring.Echo"/>
    <property name="serviceBean" ref="articlesService"/>
    <property name="xfire" ref="xfire"/>
</bean>
```

Dans le cas d'XFire, une ressource de l'outil doit être importée au début du fichier de configuration de Spring, comme l'illustre le code suivant :

```
<import resource="classpath:org/codehaus/xfire/spring/xfire.xml"/>
```

Packaging et utilisation

Afin de packager le composant d'accès aux données, une archive Java (fichier JAR) peut être mise en oeuvre. Cette dernière contiendra aussi bien le code compilé du composant que son fichier de configuration Spring. Nous placerons ce dernier dans le répertoire META-INF/spring de l'archive.
Pour utiliser le composant, il suffit simplement de placer l'archive dans le classpath de l'application et de préciser le chemin d'accès à son fichier de configuration au niveau de la création du contexte applicatif de Spring.
Le code suivant illustre une utilisation simple du composant dans une classe d'une application Java autonome :

```
ClassPathXmlApplicationContext contexte = null;
try {
    contexte = new ClassPathXmlApplicationContext(
        "classpath:/META-INF/spring/applicationContext.xml");

    ArticlesService composant = (ArticlesService)
                        contexte.getBean("articlesServices");
    List articles = composant.getListeArticles();
} catch(Exception ex) {
    ex.printStackTrace();
} finally {
    if( contexte!=null ) {
        contexte.close();
    }
}
```

Dans le cas d'une application Web, les lignes suivantes peuvent être spécifiées dans le fichier web.xml localisé dans le répertoire WEB-INF :

```
<web-app>
    <display-name>ApplicationWeb</display-name>

    <context-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>
            classpath:/META-INF/applicationContext.xml
        </param-value>
    </context-param>

    <listener>
        <listener-class>
            org.springframework.web.context.ContextLoaderListener
        </listener-class>
    </listener>
</web-app>
```

Pour finir, notons que, dans le cas de la mise en oeuvre du composant de manière distribuée, d'autres mécanismes doivent être utilisés, mécanismes que nous ne détaillerons pas ici.

En résumé

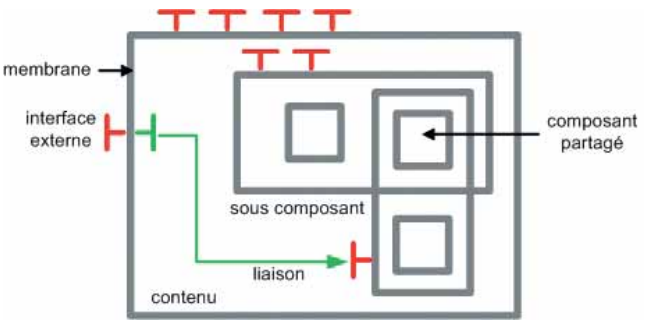
Dans cet article, après avoir rappelé les principes de la programmation orientée composant, nous avons décrit la manière de mettre en oeuvre un composant d'accès aux données en se fondant sur le framework Spring, composant dont l'objectif consiste en la manipulation de données d'articles stockées dans une base de données. Nous avons vu comment structurer ce composant avec le framework Spring aussi bien au niveau de son contrat que de son implémentation.
Cette dernière intègre des mécanismes afin de réaliser l'accès aux données, traitements basés sur le framework Hibernate et le support d'accès aux données relatif de Spring. Ce dernier support permet de réduire de manière significative la complexité ainsi que le nombre de lignes des composants DAO.
Les entités de base de ce support sont les templates, classes encapsulant l'utilisation des API d'accès aux données tout en gérant de manière optimale les ressources utilisées. Il est à noter également que les templates permettent aux composants de s'intégrer automatiquement dans des transactions déclaratives dirigées par Spring et configurées par l'intermédiaire de comportements transactionnels.
Pour finir, il est à noter que la programmation orientée composant offre un gain certain afin de modulariser et centraliser les traitements dans le but de les réutiliser au maximum. Cette approche nécessite néanmoins un important travail d'analyse afin de modulariser les applicatifs en composants et son bénéfice est essentiellement visible sur le long terme.
Une approche dirigée par les modèles peut être particulièrement utile dans ce cas afin de tirer parti de la modélisation et de générer les structures des classes du composant ainsi que les fichiers de configuration de Spring.

■ **Thierry Templier** *(templth@yahoo.fr)*
Architecte Java/J2EE – Argia-Engineering
Co-auteur des ouvrages Spring par la pratique et JavaScript pour le Web 2.0 parus aux éditions Eyrolles

Relecture technique assurée par Jérôme Benois.

La programmation par composants avec Fractal

Peut-être pensez-vous n'avoir jamais programmé par composants et être ainsi passé à côté de tout sans vous en rendre compte. Détrompez-vous si vous avez un jour réalisé des EJB, des composants COM, CCM ou encore .NET.



Ces modèles de composants, pas toujours bien réussis avaient en tout cas un point commun : proposer un modèle d'implémentation des interfaces exposées contractuellement, en communalisant certaines fonctions, qu'elles soient métiers ou techniques (sécurité, transaction, RMI, clustering…).

De la théorie…

Afin de démontrer ce que peut être la programmation par composants en évitant toute polémique ou tout effet de mode, j'ai choisi d'utiliser le modèle de composants Fractal. Il s'agit d'un modèle défini par France Telecom R&D et l'INRIA dès 2000 et qui n'a cessé d'évoluer depuis. Il se présente sous la forme d'une spécification et d'implémentations dans différents langages de programmation comme Java, C, C++, SmallTalk ou les langages de la plate-forme .NET. Fractal est organisé comme un projet du consortium OW2 (anciennement ObjectWeb). Les caractéristiques principales du modèle Fractal sont motivées par l'objectif de pouvoir construire, déployer et administrer des systèmes complexes tels que des intergiciels ou des systèmes d'exploitation. Le modèle est ainsi basé sur les principes suivants :

- composants composites (composants qui contiennent des sous-composants) pour permettre d'avoir une vue uniforme des applications à différents niveaux d'abstraction.
- composants partagés (sous-composants de plusieurs composites englobants) pour permettre de modéliser les ressources et leur partage, tout en préservant l'encapsulation des composants.
- capacités d'introspection pour permettre d'observer l'exécution d'un système.
- capacités de (re)configuration pour permettre de déployer et de configurer dynamiquement un système.

Fractal est conçu pour être extensible et prouve cette capacité à travers la mise en œuvre de l'implémentation du modèle EJB3 pour JoNAS.

Un composant Fractal est une entité d'exécution qui possède une ou plusieurs interfaces. Une interface est un point d'accès au composant. Il existe deux catégories d'interfaces : les interfaces serveurs - qui correspondent aux services fournis par le composant -, et les interfaces clients qui correspondent aux services requis par le composant. Un composant Fractal est généralement composé de deux parties : une membrane - qui possède des interfaces fonctionnelles et des interfaces permettant l'introspection et la configuration (dynamique) du composant -, et un contenu qui est constitué d'un ensemble fini de sous-composants. La membrane d'un composant est constituée d'un ensemble de contrôleurs. Les contrôleurs peuvent être considérés comme des méta-objets. Chaque contrôleur a un rôle particulier : un contrôleur peut, par exemple, permettre de suspendre/reprendre l'exécution d'un composant. Les composants discrets, une fois écrits, doivent être assemblés afin de construire un édifice logiciel. Cet assemblage peut être fait programmatiquement ou plus simplement à l'aide du langage XML de Fractal ADL.

… A la pratique !

Afin de démontrer rapidement et simplement l'utilisation de Fractal, j'ai choisi de reprendre un exemple simple type HelloWorld, directement issu des exemples fournis par Fractal.

Tout d'abord téléchargez Fractal avec le zip fraclet-annotation-2.0 à cette URL : http://forge.objectweb.org/project/showfiles.php?group_id=22. Fraclet fournit un jeu d'annotations Java 5, évitant la création de code spécifique Fractal fastidieux ; il fournit l'ensemble des jars permettant de mettre en œuvre Fractal à savoir l'API, l'implémentation Java de Fractal (dénommée Julia) et Fractal ADL. Créez un nouveau projet Eclipse. Créez un répertoire lib et placez-y les librairies fournies dans fraclet-annotations-2.0 : les jars situés dans externals et ceux situés dans examples/common/lib. Ajoutez ces librairies au classpath du projet. Nous allons créer un composant qui contiendra 2 sous-composants : Créez d'abord l'interface du composant Service :

```
@Interface(name="service")
public interface Service {
void print();
}
```

L'annotation utilisée comme les suivantes proviennent de fraclet qui permet à l'aide de l'outil Spoon de générer le code final du composant et les descripteurs de composants XML (de Fractal ADL). Ici, l'annotation permet de décrire une interface de composant. Créez-en maintenant une implémentation de Service qui sera le sous-composant fractal Server :

```
@FractalComponent
public class ServerA implements Service {
   @Attribute(value=">>")
   private String header ;

   @Service
private Component ref;

   @Attribute(argument="msg")
private String message ;
```

```
public void print() {
        NameController nc=null;
        try {
            nc = (NameController)ref.getFcInterface("name-controller");
        } catch (NoSuchInterfaceException e) {
            e.printStackTrace();
        }
        System.out.println(nc.getFcName()+header+message);
}

   @LifeCycle(on=LifeCycleType.START)
   private void init() {
        System.out.println("Starting Server A ...");
   }
}
```

Cette fois, les annotations spécifient que le code correspond à un composant possédant 2 attributs (header et message) et qui utilise une référence à la membrane (ref). Créez maintenant le code du sous-composant Client

```
@FractalComponent
@Provides(interfaces = @Interface(name = "r", signature = Runnable.class))
public class Client implements Runnable {

   @Requires(name = "default")
   protected Service service;

   @LifeCycle
   public void init() {
        System.out.println("STARTING client component");
   }

   public void run() {
        System.out.println("Service says: ");
        this.service.print();
   }
}
```

Nous avons ici décrit un autre composant Fractal qui fournit une interface nommée r de type Runnable et qui nécessite un service de type Service. Assemblons à présent ces composants en créant un fichier Fractal ADL nommé HelloWorld.fractal et situé à la racine de votre répertoire src.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE definition PUBLIC "-//objectweb.org//DTD Fractal ADL 2.0
//EN" "classpath://org/objectweb/fractal/adl/xml/basic.dtd">

<definition name="HelloWorld">
   <interface name="r" role="server" signature="java.lang.Runnable" />

   <component name="client" definition="Client" />
   <component name="serverA" definition="ServerA('Hi!')" />

   <binding client="this.r" server="client.r" />
   <binding client="client.default" server="serverA.service" />
</definition>
```

Ce fichier se lit comme suit : le composant HelloWorld expose une interface r de type Runnable. Il contient deux composants (dont les définitions sont générées dans d'autres fichiers fractal ADL générés par Spoon). L'interface exposée par le composant HelloWorld est connectée à l'interface r du composant client qui est lui-même connecté via son interface default à l'interface nommée service du composant ServerA.
Il nous faut maintenant exécuter Spoon puis compiler le code généré afin de l'exécuter. Pour cela vous pouvez utiliser une tâche *ant* qui devrait ressembler à ceci :

```
<taskdef name="spoon" classname="spoon.SpoonTask" classpathref=
"classpath" />

<spoon classpathref="classpath"verbose="true" output="${gen.spooned}"
   spoonlet="lib/fraclet-annotation.jar">
   <sourceset dir="${src}" includes="**/*.java" />
   <sourceset file="lib/spoon/AVal.jar" />
   <sourceset file="lib/fraclet-annotation.jar" />
</spoon>

<javac srcdir="${gen.spooned}" destdir="${build}" source="1.5" classpath
ref="classpath" />
```

Il ne reste plus qu'à exécuter le code à l'aide de Fractal ADL qui masque le code de Bootstrap de Fractal :

```
<target name="execute" depends="compile">
   <java classname="org.objectweb.fractal.adl.Launcher" classpathref=
"classpath" fork="yes" failonerror="yes">
        <jvmarg line="${run.jvm.parameters}" />
        <arg line="-fractal HelloWorld r" />
   </java>
</target>
```

Et voici la sortie obtenue :

```
STARTING client component
Starting Server A ...
Service says:
serverA>>'Hi!'
```

L'exercice intéressant à réaliser est de voir à quel point il est alors facile de créer une nouvelle implémentation ServerB de l'interface Service et de recomposer l'application en liant cette fois le client à ServerB au lieu de ServerA, dans le fichier de description du composant HelloWorld. Cela illustre d'abord que la reconfiguration d'une application est rendue plus aisée ; mais il est intéressant de savoir que la liaison des composants peut être faite et modifiée programmatiquement et donc dynamiquement, ce qui permet ainsi de changer le mode d'implémentation de l'application au runtime ! J'espère avoir pu ici vous donner un aperçu simple de la programmation par composant avec Fractal. Pour plus d'informations, n'hésitez pas à visiter le site http://fractal.objectweb.org et http://sardes.inrialpes.fr/ecole/livre/pub/Chapters/Fractal/fractal.html
Les sources de cet exemple sont disponibles sur http://fdewasmes.free.fr.

Fabrice Dewasmes
Java & Open Source Dept. Manager - PragmaConsult SA
www.pragmaconsult.lu - http://fdewasmes.free.fr

Composants et programmation orientée objet en PHP

La Programmation Orientée Composant peut être vue comme un raffinement de la programmation par objets pour faire face aux problèmes de couplage. Au cours de cet article nous allons rentrer dans le détail de la POC, nous en verrons les avantages et désavantages avant de prendre un exemple basé sur PHP.

Une bonne approche pour bien conceptualiser ses développements consiste en l'utilisation de composants. Quand on parle de composants on parle de programmes indépendants réutilisables dans différents contextes. Au niveau technique on parle de simples fichiers contenant du code, de bibliothèques de code, de bibliothèques de classes, voire de fonctions. L'idée générale est de structurer votre code et de décomposer votre logiciel.

La conduite de projet

Pour tout architecte informatique la conception d'une architecture modulaire et maintenable est un défi de tous les jours. Un projet mal conçu sera rigide, fragile et à terme immobile. A l'inverse, un projet qui a été bien conçu sera robuste, extensible et réutilisable. Les avantages à utiliser une approche POC pour conduire un projet sont multiples :

Productivité

La réutilisabilité d'un composant permet un gain de productivité non négligeable car elle diminue le temps de développement. Le gain de temps est d'autant plus important que des composants tout faits peuvent être trouvés en Open Source.

Spécialisation

Le développement d'un logiciel peut être divisé en sous-groupes, chacun se spécialisant dans le développement d'un composant.

Sous-traitance

Le développement d'un composant peut-être sous-traité.

Langages de développement

Chaque composant peut être développé avec un langage de programmation différent. L'idée étant de profiter des fonctionnalités particulières d'un langage pour développer un composant.

Comme toujours, chaque avantage à son revers et la programmation orientée composant n'y échappe pas. Il convient de prendre en compte les aspects suivants :

Composants de confiance

Il est possible de trouver un grand nombre de composants Open Source sur Internet voire même d'en acheter des propriétaires. Dans les deux cas il convient de veiller à utiliser des composants de confiance, si possible couverts par des tests unitaires.

Compatibilité

Il faut veiller à ce que les composants soient compatibles entre eux. Ces composants possèdent la particularité d'être développés par des tierces personnes sans concertation et ils sont donc potentiellement non compatibles.

Complexité et performances

L'utilisation d'objets très complexes va forcément avoir un impact sur les performances générales du logiciel. Dans certains cas il peut être judicieux de tester la tenue à la charge des composants que l'on utilise.

La POC pour les applications web

Début 2000 on parlait de "site Internet", le contenu était généralement statique, les navigateurs n'offraient pas autant de fonctionnalités qu'aujourd'hui, et les serveurs et leurs technologies étaient bien moins robustes. Aujourd'hui le Web a bien changé. En l'espace de dix ans, l'apparition de langages dits dynamiques, l'avènement du web2.0 et des technologies SOA ont changé la donne des "sites Internet". Si bien qu'on parle actuellement d'architecture, de plate-forme, et plus généralement d'applications Web.

Aujourd'hui, la complexité d'une application Web est similaire à la complexité d'une application lourde, elle nécessite donc, en toute logique, toute la panoplie de méthodologies et d'outils que nécessitent les applications lourdes, afin d'êtres analysées, conçues, testées, et utilisées correctement. Alors qu'il s'affranchit de tout ce qui est système, noyau, espaces, comme dans le cas d'un programme client lourd, un applicatif Internet joue avec la complexité de la communication : Clients, serveurs, protocoles, XML, SOA, bases de données.... La complexité a amené les technologies ainsi que les manières de concevoir ces “programmes web” à évoluer drastiquement. Par exemple, PHP avait dès le départ affiché son but : Internet, et de manière plus spécifique : le Web. En douze ans PHP et son écosystème ont bien évolué, on trouve maintenant un modèle objet tout à fait sérieux dont découlent de nombreuses bibliothèques, composants, frameworks et logiciels dédiés aux applications Web modernes.

Quelques composants et briques utiles :

ez components - http://ez.no/ezcomponents

Zend Framework – http://framework.zend.com

Un site regroupe une liste des principales briques logicielles existantes pour PHP : http://www.guidephp.com/

Exemple avec le Zend Framework

Dans cet exemple, l'accès à notre application est soumis à mot de passe. Si un administrateur s'identifie, alors il accède à la partie “admin”, sinon, si l'identification réussit mais avec un identifiant “guest”, alors le site 'de base' est affiché.

Dans le cas où l'authentification échoue, aucun accès n'est admis.

Nous nous baserons sur une identification HTTP de type basic (RFC 2617 pour plus d'infos). Nous utiliserons beaucoup de composants, qui pourtant sont tous distincts les uns des autres, mais en les couplant correctement, on arrive à notre résultat, et de plus il demeure extrêmement flexible. Cette application utilise Zend Framework dans sa version 1.0.2, et PHP 5.2 :

```
<?php
/**
```

```
 * Assertion pour l'administrateur
 * Cette classe vérifie que la personne s'est identifiée en tant que "admin"
 */
class AclAssertion implements Zend_Acl_Assert_Interface
{
   const ADMINKEY = "admin";

  private $_aclResult;

  public function __construct(array $result)
    {
      $this->_aclResult = $result;
    }

/**
* Zend_Acl_Assert_Interface
*/
  public function assert(Zend_Acl $acl, Zend_Acl_Role_Interface $role =
null, Zend_Acl_Resource_Interface $resource = null, $privilege = null)
    {
     return (array_key_exists("username",$this->_aclResult) && $this->_acl
Result['username'] == self::ADMINKEY) ? true : false;
    }
}

/**
 *
 * Authentification HTTP
 *
 */
$config = array(
    'accept_schemes' => 'basic',
    'realm'          => 'anaska',
                    );

// identification HTTP
$adapter = new Zend_Auth_Adapter_Http($config);

// resolveur d'identification : fichier pour HTTP
$basicResolver = new Zend_Auth_Adapter_Http_Resolver_File();
$basicResolver->setFile('auth.txt');

$adapter->setBasicResolver($basicResolver);

// passage de la requête HTTP actuelle
$adapter->setRequest(new Zend_Controller_Request_Http());
// puis passage d'un objet de réponse HTTP
$adapter->setResponse($response = new Zend_Controller_Response_ Http());


/**
 *
 * Définition des ACL
 *
 */
$acl = new Zend_Acl();
$acl->add(new Zend_Acl_Resource('My private space'));


$acl->addRole(new Zend_Acl_Role('administrator'));


$acl->deny('administrator', 'My private space');

/**
 * autorise uniquement si l'assertion renvoie true ( est admin )
 *
 * La méthode authenticate lance le processus HTTP d'identification "basic".
 * L'objet dans $adapter peuple l'objet Réponse HTTP en conséquence
notamment grâce à un code 401
 */
$acl->allow('administrator', 'My private space', null, new AclAssertion
($result = $adapter->authenticate()->getIdentity()));




/**
 *
 * Définition des ACL
 *
 */
if ($acl->isAllowed('administrator', 'My private space') ) // si administrateur
{
  $response->setHttpResponseCode(200);
  $response->appendBody('welcome administrator ;)');
}elseif(!empty($result)){ // sinon si guest : il y a un résultat suite à authenticate()
  $response->setHttpResponseCode(200);
  $response->appendBody('welcome to the guest part of the website');
}else{ // enfin, si l'authentification a échoué
  $response->appendBody('Please, authenticate');
}

// pour finir, renvoie simplement la réponse HTTP au client.
$response->sendResponse();
?>
```

Le fichier auth.txt ressemble à ceci :

```
admin:anaska:admin
guest:anaska:guest
```

Conclusion

La programmation orientée composant n'est pas une nouveauté c'est la résultante de la montée en compétence de tout développeur : il factorise pour éviter de réinventer la roue. La nouveauté, si l'on peut dire, c'est l'ouverture qu'a amené Internet et la foison de composants disponibles. Le revers de la médaille réside dans la difficulté à choisir ses composants au point qu'il vaut parfois mieux les écrire soi-même !

Julien PAULI *est consultant formateur chez ANASKA. Il intervient sur les domaines relatifs au Web et s'est spécialisé sur les frameworks et composants PHP.*

Définir une architecture par composants avec UML2

Les limitations de la modélisation/programmation par objet sur la réutilisabilité et l'autonomie, ont renouvelé l'intérêt pour les composants. En effet, réutiliser une classe signifie souvent importer un ensemble de classes liées, mal identifié par l'utilisateur. Hériter d'une classe signifie créer une dépendance forte, qui sera excessivement sensible aux évolutions de la classe héritée. Les points d'utilisation, les modes d'assemblage, l'autonomie des éléments réutilisés, le contrat d'utilisation ne sont pas fournis par la programmation/modélisation objet.

La notion de composant a été un ajout majeur introduit dans le standard UML2. Le composant permet en effet d'isoler des parties de logiciel en unités autonomes, avec des points de connexion bien définis. Les notions de "structure interne" de classe, de port et de part sont des notions nouvellement introduites offrant des mécanismes utiles pour les architectures orientées composant. A travers ces mécanismes, UML2 apporte la modélisation par composant. Ces mécanismes d'assemblage de composants permettent de réaliser le vieux rêve du "lego logiciel", c'est à dire d'assembler des composants qui ne se connaissent pas obligatoirement, pour former un système englobant. Pour ceux qui ont connu l'enfer de l'intégration logicielle, ou pire encore l'intégration logiciel/matériel, ces mécanismes sont très prometteurs.

UML2 permet enfin de formaliser les contrats des différents composants, leurs points d'interconnexion, et ensuite d'assembler ceux-ci.

Prenons un exemple couramment pratiqué: une session vidéo est réalisée en prenant un portable PC, un vidéo projecteur, et en connectant les deux par un câble. La connexion s'effectuera en branchant le câble sur les prises VGA de ces deux composants.

Le miracle de la vie courante fait qu'un non initié à l'électronique et l'informatique peut simplement assembler ces deux composants complexes pour obtenir un système plus sophistiqué encore. Les figures Erreur! Argument de commutateur inconnu montrent comment UML2 permet de représenter ce scénario: ceci n'était pas possible avec UML1.4. Tout d'abord, dans la première Figure, on identifie les composants, leurs différents points d'interconnexion (ports), les interfaces fournies par les composants via ces ports ainsi que les interfaces requises. Ainsi, des ports de cet exemple déclarent qu'ils requièrent ou fournissent une interface "VGA". Il sera très important de spécifier l'interface VGA, avec des pré et post conditions déterminant précisément les conditions d'emploi.

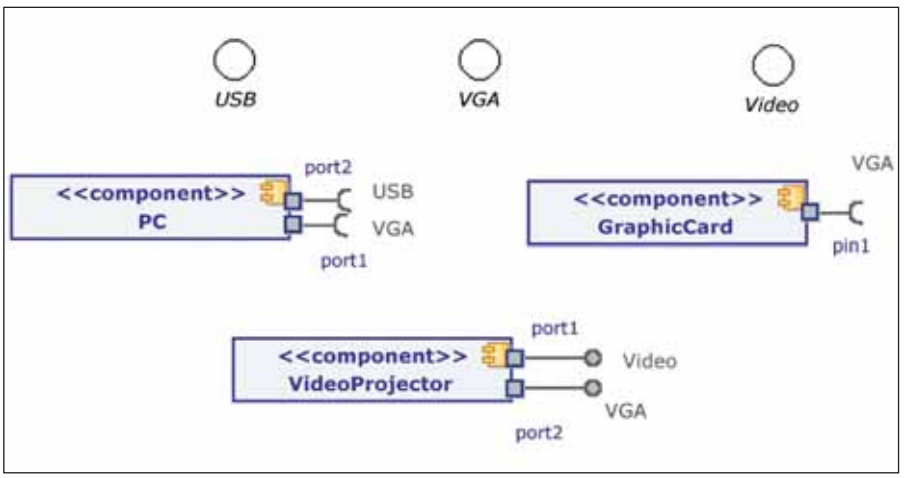


Figure Erreur! Argument de commutateur inconnu. – Les composants à assembler (modèle Objecteering)

Dans la deuxième figure, on détermine de quoi est constitué un composant assembleur (session vidéo) à travers ses "part" qui sont "myPC" et "myVideo", on visualise les prises de sortie (les "port") ainsi que le câble connectant les prises (le "connecteur").

Cet exemple simple illustre ce que l'on veut faire pour les architectures: l'architecture se représente sous la forme d'assemblage de composants pour constituer des composants plus complexes, via des modes de connexion devant être détaillés. On imagine aisément les types de composants souhaités (infrastructures, librairies, modules fonctionnels, etc.), et on connaît les types de connexion que l'on veut utiliser (réseaux, services web, accès directs, etc.).

Le lien avec la programmation

Les notions de part et de port existaient préalablement dans les domaines techniques et télécom, comme par exemple dans le langage SDL qui permet de générer du code. Avec des langages comme C++, Java ou C#, il est possible d'utiliser des " patterns " permettant de s'appuyer sur ces notions de part et de port, pour générer un code garantissant l'autonomie et la capacité d'assemblage des composants. Ainsi, dans l'exemple Figure Erreur! Argument de commutateur inconnu., C1 et C2 sont définis en totale indépendance : le code lié au composant C1 de I1 et le code lié au composant C2 de I2 ne connaissent que leurs ports respectifs (P1, P2), et ne s'adressent qu'à lui. Ces ports sont traduits comme des attributs, pointeurs sur le port connecté, fournisseur ou demandeur du service. L'assemblage I1/I2 n'est connu que du composant intégrateur C. Celui-ci instancie I1 et I2, leur fournit des valeurs initiales, et réalise la connexion I1/I2. On voit ainsi que cette démarche garantit l'autonomie C1/C2, et place la responsabilité de

l'assemblage I1/I2 au niveau de l'intégrateur. Cette approche permet de décomposer une application en composants, que l'on peut assembler sans effets de bord et interférences entre composants. Elle est très efficace pour les projets de grande taille, en répartissant bien les responsabilités entre les composants, et en identifiant clairement les points d'interaction.

Les composants pour l'architecture

C'est au niveau des architectures logiques et logicielles que l'approche par composant est la plus appliquée et la plus intéressante. Un standard tel que SysML (extension de UML2) permet par exemple de définir l'architecture des grands systèmes techniques (par exemple, un avion, une centrale nucléaire, un système d'arme) en s'appuyant sur cette notion.

Les approches dédiées aux architectures SOA insistent quant à elles sur la notion de composant de service, et utilisent les interfaces pour représenter les services requis/fournis. Dans l'exemple Figure Erreur! Argument de commutateur inconnu., l'architecture logique d'un système d'information d'une agence de voyage est présentée en s'appuyant sur cette notion. Ici, le composant est un élément de grande taille, constituant un " projet de développement " en soi, une application, bien plus large que par exemple un EJB ou une classe C#. Cette architecture ne préjuge pas du langage de programmation utilisé. Elle définit les composants à implémenter, les services à fournir, et en détaillant le modèle, les types de données échangées.

Par ailleurs, au-delà de la simple connexion technique des éléments (présence de connecteurs, et ports identifiés, signatures et types compatibles), la modélisation d'une architecture SOA nécessite une formalisation sémantique des services interconnectés. Le support des pré et post conditions de UML devient un moyen incontournable d'assurer la viabilité d'un modèle d'architecture SOA. Les démarches supportant la modélisation pour la SOA utilisent ce type de technique, comme par exemple l'atelier Objecteering EA supportant la méthode ouverte PRAXEME, SOMA de IBM, ou le standard UPMS de l'OMG.

La tendance actuelle conduit ainsi à une généralisation du développement par composants, tirée par la vague SOA.

Philippe DESFRAY - *SOFTEAM*

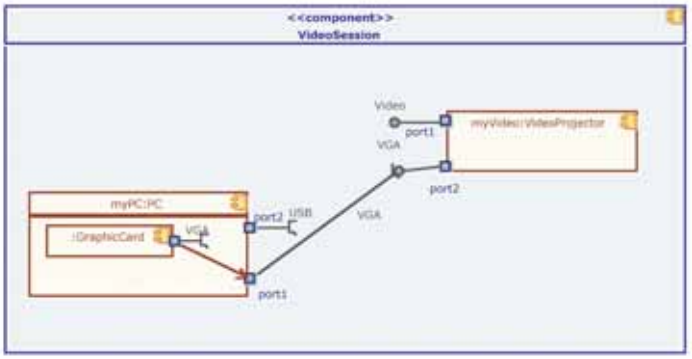


Figure Erreur! Argument de commutateur inconnu. – Assemblage d'un PC et d'un vidéo projecteur dans une session vidéo (modèle Objecteering)

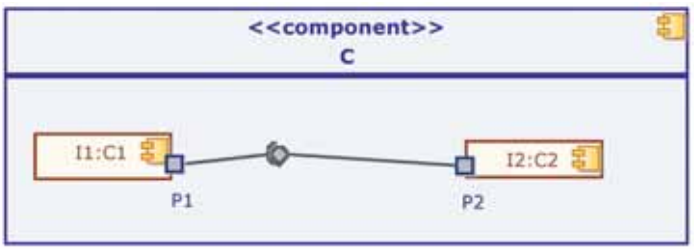


Figure Erreur! Argument de commutateur inconnu. Assemblage de I1 :C1 et I2 :C2 sous C (modèle Objecteering)

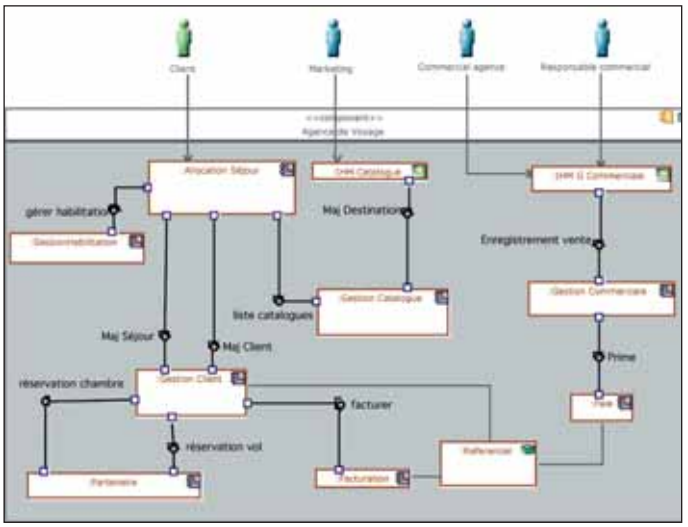


Figure Erreur! Argument de commutateur inconnu. Architecture logique du SI d'une agence de voyage (modèle Objecteering)

Petit rappel sur les composants

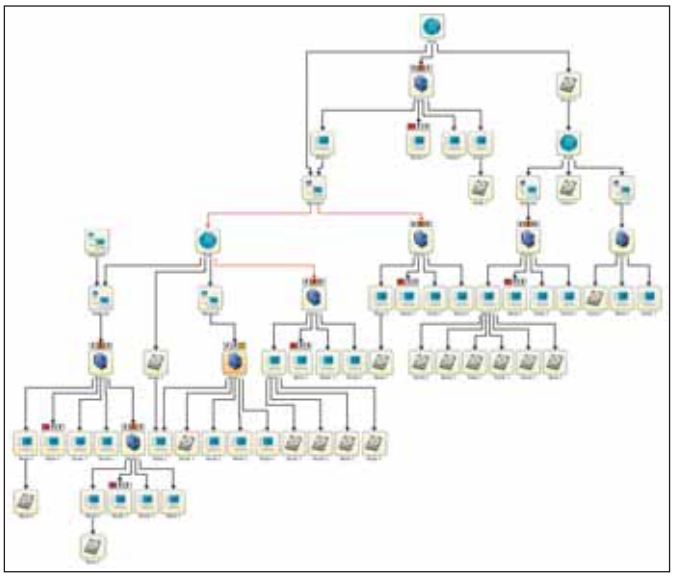
Un composant apporte un certain nombre de fonctionnalités qui peuvent être appelées depuis un programme externe. Il n'est pas nécessaire d'avoir accès au code du composant, mais il suffit de connaître ses fonctionnalités (son API) via une interface, c'est-à-dire un ensemble de fonctions lui permettant de communiquer avec le programme client. Par définition un composant est réutilisable, il n'est pas lié à un projet, il fournit un service bien précis. Les développeurs et architectes habitués à la programmation orientée objet trouveront des similitudes entre la programmation par composants et la programmation orientée objet. La programmation par composants permet de faire de la réutilisation, et de suivre les règles très connues "Ne réinventez pas la roue" et "Ne vous répétez pas". Un développeur se doit d'être fainéant ! En utilisant des bibliothèques architecturées, découplées, documentées et performantes, les développeurs ont non seulement un outil de travail commun, dans la mesure où ils utilisent tous les mêmes composants, ce qui leur permet de se comprendre plus facilement entre eux, mais en plus ils se focalisent sur leur problème, et non plus sur les multiples manières de le traiter. Il est un calcul qui ne trompe pas : plus on écrit de lignes de code, plus le risque de se tromper augmente. En utilisant et réutilisant des composants déjà prêts, on écrit moins de code, car celui-ci a déjà été rédigé et testé unitairement.

Julien Pauli

Création d'applications graphiques en Java pour le Desktop et le Web

Différents messages circulent chez les responsables informatiques. Certains pensent " logique applicative, intégration et Web services ", tandis que d'autres considèrent la fonctionnalité pure comme un élément secondaire face à la productivité, la facilité d'utilisation et la fiabilité. Les interfaces utilisateurs appartiennent clairement à cette seconde catégorie. Elles apportent aux utilisateurs un formalisme et un mode opératoire compatibles avec les données, la logique à manipuler et les contraintes d'exploitation.

Cependant, malgré l'importance attribuée aux interfaces de haute qualité, il faut rappeler que leur développement est souvent sous-estimé, tant en terme de complexité que d'intégration. Les difficultés sont plus sensibles encore lorsque ces interfaces doivent pouvoir être déployées sur plusieurs plates-formes. Comment faire pour qu'une même interface graphique puisse être livrée comme application riche et comme client léger ? Que peut-on réutiliser d'une interface Java existante pour en faire un plug-in Eclipse ou RCP ? Cet article présente les principes mis en œuvre dans JViews d'ILOG pour faciliter la définition et le déploiement d'applications graphiques riches en Java.



Un diagramme de réseau affiché automatiquement sous la forme d'une hiérarchie et dont le style des objets reflète l'état du système.

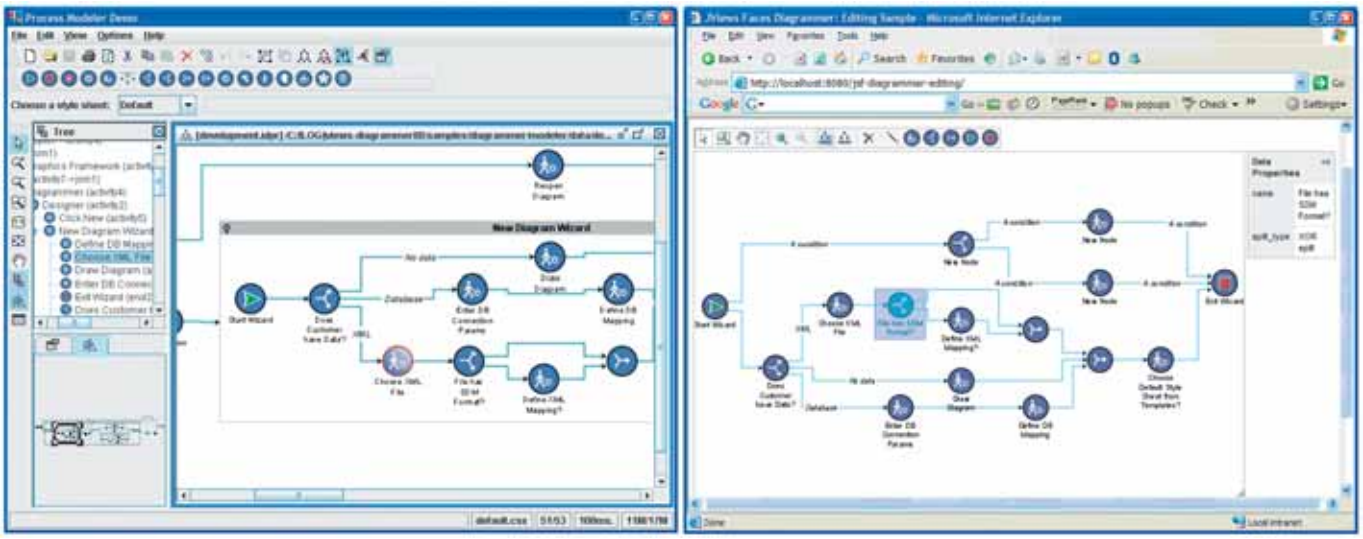
Les différents types de vues graphiques

La plupart des interfaces sont utilisées soit pour afficher de l'information, soit pour saisir et configurer des données. En se basant sur un standard ou avec l'aide d'ergonomes, les développeurs choisissent le type de vue qui correspond le mieux à leur application et à leurs utilisateurs. Ils peuvent aussi imaginer de nouvelles façons de représenter leurs données ou d'interagir avec les entités manipulées par l'application. Dans l'industrie, on retrouve sous différentes formes un certain nombre de formalismes assez standard et suffisamment riches pour s'adapter à différents métiers. On peut citer les diagrammes (par exemple entité-relation), les tableaux de bord, les cartes géographiques, les diagrammes de Gantt pour des plannings, les courbes, les calendriers, etc. En voici quelques exemples détaillés.

Les diagrammes sont très populaires car ils représentent des interconnexions entre objets. C'est le cas des schémas entité-relation, des réseaux de communications, des diagrammes de PERT ou des diagrammes de processus. Ces schémas sont caractérisés par un standard visuel propre (comme l'UML pour la modélisation, le BPMN pour les processus, des symboles métiers pour les réseaux). L'autre caractéristique est leur mode de fonctionnement. Les diagrammes peuvent être conçus manuellement, grâce à des éditeurs, ou bien générés et affichés automatiquement à partir de données structurées. Ces mêmes diagrammes peuvent également être animés dynamiquement pour superviser une activité en temps réel, comme dans le cas de réseaux, de processus de fabrication ou de schémas électriques.

Tableau de bord réaliste

D'autres types de représentation très répandus sont les tableaux de bord, car ils montrent de façon synthétique l'état d'un système parfois complexe. Ils sont utilisés pour des activités de supervision dans de nombreux domaines. Ils permettent de concevoir des consoles pour surveiller des données métier clé (des KPI, Key Performance Indicator) ou bien des synoptiques pour superviser des systèmes temps-réel, comme des chaînes de fabrication ou des usines. Ce type d'interface tente de reproduire la réalité et se compose souvent de jauges, compteurs et boutons de toutes sortes. Si les diagrammes évoqués plus haut peuvent être créés automatiquement, les tableaux de bords et les synoptiques sont souvent dessinés au fur et à mesure, en utilisant des palettes d'objets prêtes à l'emploi et connectées à des données dynamiques du système.



Exemple d'application disponible à la fois en client riche (à gauche) et en Web/Ajax (à droite)

Cibler la bonne plate-forme

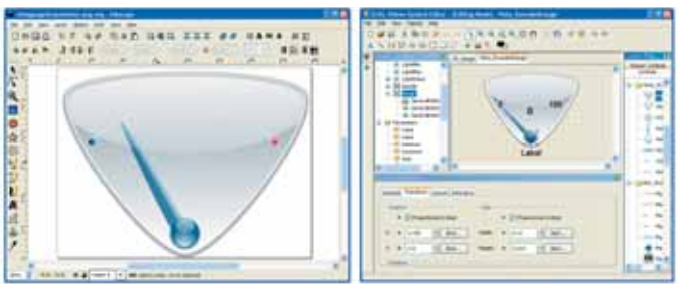
Une fois que le bon type d'affichage a été déterminé, encore faut-il choisir la technique de développement ainsi que les options de déploiement. Si certaines applications peuvent se contenter d'un seul type de déploiement (par exemple une application riche en Java) la tendance est clairement vers les applications supportant plusieurs plates-formes. Il est fréquent d'avoir un client riche pour les utilisateurs intensifs et une version client léger pour les itinérants ou les utilisateurs occasionnels. Si Java est la technologie phare pour les applications parfaitement portables, la conception d'une application à la fois pour le desktop et le web demande plus d'attention, et souvent deux versions distinctes de l'application. Et bien entendu, Ajax et la vague des applications web riches (RIA, Rich Internet Applications), ont relevé le niveau d'exigence sur la qualité des interfaces utilisateurs.

Plus de Design, moins de codage

Il y a deux options intéressantes pour concevoir des interfaces riches pouvant être déployées sur plusieurs technologies. La première consiste tout simplement à minimiser le volume de code nécessaire. Par exemple, si l'interface est principalement basée sur un contenu descriptif plutôt que du code, la tâche d'adaptation à une technologie particulière est grandement diminuée. L'autre option est d'utiliser une approche de type MDA (Model Driven Architecture) qui permet la création automatique d'interfaces à partir des données structurées de l'application. Si les composants utilisés sont portables d'une technologie à l'autre, les applications le seront également. De plus, ces approches sont bien adaptées à l'utilisation d'outils de design, qui simplifient le développement et prennent en compte les différents rôles dans la chaîne de production du logiciel. Par exemple, un expert graphique peut fabriquer du contenu visuel sans être lui-même un développeur, ou bien un administrateur de l'application doit pouvoir enrichir son contenu sans modifier le cœur du système ni même recompiler l'application. Pour la technologie Java, les produits JViews offrent un ensemble de composants et d'outils pour remplir les différents besoins graphiques cités plus haut, comme des diagrammes, des courbes, des tableaux de bord, des diagrammes de Gantt et des cartes. Des outils de design permettent de définir une grande partie de la symbologie et de la logique de ces vues dans un formalisme descriptif plutôt que par du code. Ces composants sont disponibles de façon à pouvoir être déployés sous plusieurs technologies, comme Swing, Eclipse, JSF/Ajax ou Portlets (JSR 168).

Création de Symbologie

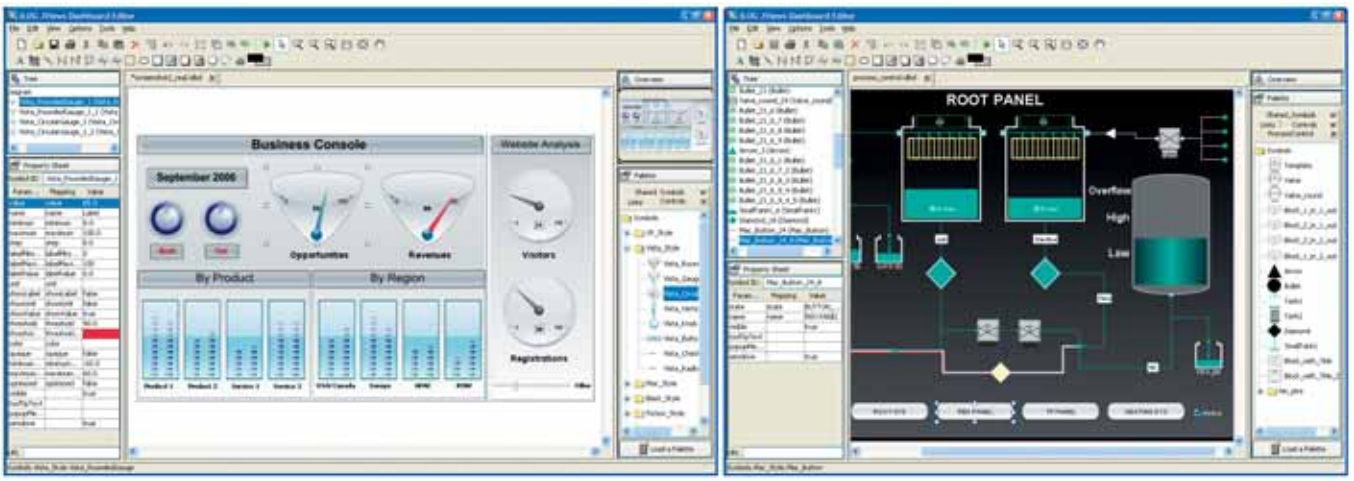
La définition des symboles manipulés par l'application est souvent le point de départ de la création des interfaces. Ces objets graphiques vont par exemple représenter des entités de diagramme de gestion de processus, des véhicules dans une application de logistique, des nœuds de réseau dans une application télécom, des compteurs pour des tableaux de bord, etc. JViews offre un éditeur de symboles grâce auquel il est possible de définir les aspects graphiques et fonctionnels des symboles. On peut importer du dessin vectoriel réalisé dans Illustrator ou dans InkScape au format SVG, puis retoucher la structure ou l'aspect visuel afin d'avoir un résultat aussi esthétique que possible. Il est ensuite possible de définir des paramètres et des règles de présentation afin de rendre ces symboles dynamiques et interactifs. Par exemple, un paramètre " Valeur " sera défini et pourra être associé à la rotation d'une aiguille pour réaliser une jauge. Egalement, une interaction avec la souris pourra modifier le paramètre " Valeur " et donc faire tourner l'aiguille. Tous les nouveaux symboles sont packagés ensemble dans des palettes, sous la forme d'un fichier JAR afin d'être réutilisés par les applications et d'autres outils de design.



Transformation d'un objet défini en SVG avec Inkscape (gauche) en un symbole intelligent (droite)

Un tel symbole et son comportement ne sont pas définis par du code, mais plutôt grâce au langage CSS (Cascading Style Sheet). Le fragment ci-dessous montre un cas simple où du code Java peut aisément être remplacé par des directives CSS contrôlant un composant JavaBean.

```
#Aiguille {
   class : "ilog.views.graphic.IlvGeneralPath" ;
   name : "Aiguille" ;
```



Design de tableaux de bord d'entreprise (gauche) et de supervision industrielle (droite)

```
    fillPaint : "grey" ;
    shape : "path \"M-30.0 -2.5L-25.0 2.5L-20.0 -2.5L-20.0 -15.0L
-25.0 -70.0L-30.0 -15.0z\" rule \"nonzero\" " ;
    strokeOn : "true" ;
}
```

A l'exécution, un moteur générique pourra instancier la bonne classe et configurer les objets créés en fonction de ces directives en CSS.

Interfaces Model-driven

Une fois les symboles définis, il faut pouvoir construire des interfaces les mettant en scène. Dans le cas de diagrammes, de réseaux, ou à chaque fois que l'on peut fabriquer un schéma automatiquement à partir de données structurées, l'approche Model-Driven doit être privilégiée. L'outil " Designer " de JViews permet de créer ce type d'interface.
En partant d'un modèle de données, des règles de présentation sont définies afin d'afficher des symboles et de configurer leurs paramètres en fonction des données.
D'autres réglages permettent de spécifier les caractéristiques générales de l'interface, comme un algorithme d'arrangement de graphe, l'apparence des liens, le type de fond de plan, le comportement de sous-graphes ou bien le contrôle du zoom.
Comme lors de la définition des symboles, l'association des graphiques au modèle de données peut se faire par l'intermédiaire de directives en CSS. Par exemple, voici une définition CSS qui indique comment représenter un nœud de diagramme :

```
node {
    / / Symbole graphique représentant un noeud
    class : "@|symbolResource(symbols/DiagramEntity.css)";  / / Nom du Symbole
  pris dans une palette
    / / Attributs du Symbole: valeurs par défaut et associations au modèle
    color : "yellow" ;
    borderColor : "black" ;
   label : "@name" ; / / Le label du symbole graphique est relié au nom 'name'
   du modèle de données
    showInfo : "true" ;
    type : "0" ;
    showLabel : "true" ;
}
```

L'exemple suivant illustre la directive CSS qui indique comment la valeur 'busy' d'un champ 'state' va altérer l'aspect d'un symbole :

```
node[state="busy"] {
    / / Attributs surchargeant les valeurs par défaut
    color : "red" ;
    borderColor : "white" ;
}
```

Le résultat final est un ensemble de palettes, de symboles, de fichiers CSS et XML. A l'exécution, des composants Java peuvent appliquer ces définitions à une source de données et ainsi créer dynamiquement les diagrammes correspondant, les éditer ou les animer. Ces composants sont des classes Java configurables, avec des versions basées sur Swing ou SWT pour les clients riches et JSP/JSF pour le déploiement en client léger Ajax.

Création de tableaux de bord

Le troisième outil de définition d'interfaces graphiques est l'éditeur de Dashboards de JViews. Il permet de créer des synoptiques, des tableaux de bord d'entreprise et des interfaces homme-machine génériques. Les symboles, pris dans des palettes prédéfinies, sont placés sur des fonds de plan et sont reliés à des valeurs fournies par l'application. Par exemple, on pourra indiquer que la valeur d'une jauge est reliée à une certaine variable ou à un champ de base de données. A l'exécution, cette définition est résolue lors de la connexion à la source de données. Ici encore, les écrans sont décrits en XML, sous une forme indépendante de la technologie utilisée à l'exécution. Des composants Java serviront à l'intégration au sein de l'application au travers de containers Swing, Eclipse ou JSP/JSF à déployer.

Conclusion

En généralisant l'approche par composants et en privilégiant la méthode descriptive et les outils de design, JViews 8.1 permet de créer rapidement des interfaces graphiques pouvant être déployées sur différentes technologies. Cet outil introduit aussi la possibilité de capitaliser sur la symbologie et sur des visuels réutilisables entre applications. La logique graphique étant dissociée du code, les interfaces sont plus simples à construire et à maintenir.

Jérôme Joubert

WebKit : Construire son propre navigateur

WebKit est un moteur de navigateur constituant le coeur de Safari, le navigateur de Mac OS X (et de plusieurs autres navigateurs). L'objet de cet article est de vous proposer une approche pratique en s'appuyant sur l'installation de l'environnement puis d'une mise en œuvre. Dans un premier temps, nous allons nous intéresser à l'origine ainsi qu'à la structure du projet.

Apple mise sur l'open source pour attirer de nouveaux utilisateurs sur ses systèmes. C'est un choix qui fût annoncé publiquement en juin 2005 pour WebKit. Le projet est depuis publié sous licence LGPL, ce qui lui apporte une protection de code ouvert tout en laissant la possibilité aux développeurs de créer leurs propres solutions, et de les distribuer indifféremment sous licence open source ou commerciale.

Le framework est développé en Objective C, un dérivé du langage C disposant d'une approche objet ainsi que de certains concepts empruntés à Smalltalk. Il n'y a rien d'étonnant à cela puisque ce langage est très utilisé sur la plate-forme Mac OS X. Le projet est organisé en deux parties distinctes mais intégrées : la première, le WebCore et la seconde JavascriptCore. WebCore est une refonte du moteur KHTML de Konqueror. Les tâches de parsing et de rendu des pages lui incombent. Le JavaScriptCore, quant à lui, est une évolution de KJS et assure l'exécution des scripts. WebKit forme un tout se basant sur deux technologies de KDE. Celles ci ont bien évidemment été modifiées en profondeur avec une nette amélioration des performances et du respect des standards. WebKit favorise ainsi l'interopérabilité et la compatibilité.

Linux dispose de deux environnements de bureaux principaux : Gnome et KDE. Chacun d'eux dispose de son propre navigateur. En ce qui concerne Gnome, son navigateur Epiphany a été conçu dans un design sobre qui le rend léger et performant dans toutes les tâches qu'il assure (onglets, ...). Epiphany est un descendant de Galeon, il est aussi disponible sous Mac OS X. Historiquement, il utilise Gecko, mais l'équipe de développement du projet porte ses efforts sur l'utilisation de la solution WebKit. Firefox a donc du souci à se faire dans les années à venir. Bien que sa survie financière soit maintenant assurée grâce à de nombreux partenariats, notamment Google.

KDE dispose quant à lui de son propre moteur KHTML qui n'est autre que l'origine du projet WebKit initié par Apple. KHTML était donc utilisé dans Konqueror, cependant le développeur de KDE, la société Trolltech, travaille actuellement à l'intégration de WebKit dans la version 4.4 de son environnement QT, utilisé entre autres pour développer KDE. KHTML va donc disparaître pour laisser place à WebKit au sein de Konqueror.

Environnement de développement

Le site officiel du projet met en avant deux environnements de développement : Mac OS X et Windows. Dans le premier cas, c'est Xcode qui fera office d'IDE. Il sera nécessaire de le coupler à un client subversion pour la gestion de configuration. Quant à Windows, Visual Studio est nécessaire. Si vous ne possédez pas de licence pour ce produit, il existe une version gratuite suffisante pour tester WebKit : Visual C++ 2005 Express. Cygwin est nécessaire afin de compiler les sources Webkit. La procédure en ligne étant particulièrement bien expliquée, je vous invite à la suivre puisque certaines informations comme les patchs pour Visual Studio ou les versions des logiciels sont destinées à évoluer. En ce qui concerne les développeurs Linux, c'est en naviguant sur différents sites web que l'on peut trouver des informations sur la procédure d'installation. La différence principale est que vous devez installer le framework QT et travailler avec l'IDE KDevelop.

Après avoir téléchargé la dernière version des sources, vous devrez compiler Webkit. La procédure étant spécifique à chaque plate-forme, il est nécessaire de se référer à la documentation présente sur le site. Après avoir lancé le script de compilation, vous obtenez un navigateur simple. Sous Linux, on le lancera via le script suivant :

```
WebKit/WebKitBuild/Release/WebKitQt/QtLauncher/QtLauncher about:blank
```

Mise en pratique

Cette partie de l'article montre comment créer son propre navigateur en utilisant les fonctionnalités de base de WebKit. Celui-ci sera composé d'une simple fenêtre comprenant une barre d'adresse et sept boutons : *précédent, suivant, arrêter, recharger, imprimer, agrandissement et réduction du texte.* Vous verrez qu'il n'y a que peu de code à écrire pour arriver à ce résultat.

Sous Mac OS, l'environnement de développement XCode intègre parfaitement le framework WebKit. Il suffit de créer un projet et de concevoir l'interface en mode graphique par drag and drop des composants. Il ne reste plus qu'à connecter les éléments de cette interface au framework. Là encore, il n'est pas nécessaire d'écrire du code. Finalement, il ne reste plus qu'à compiler et lancer le navigateur.

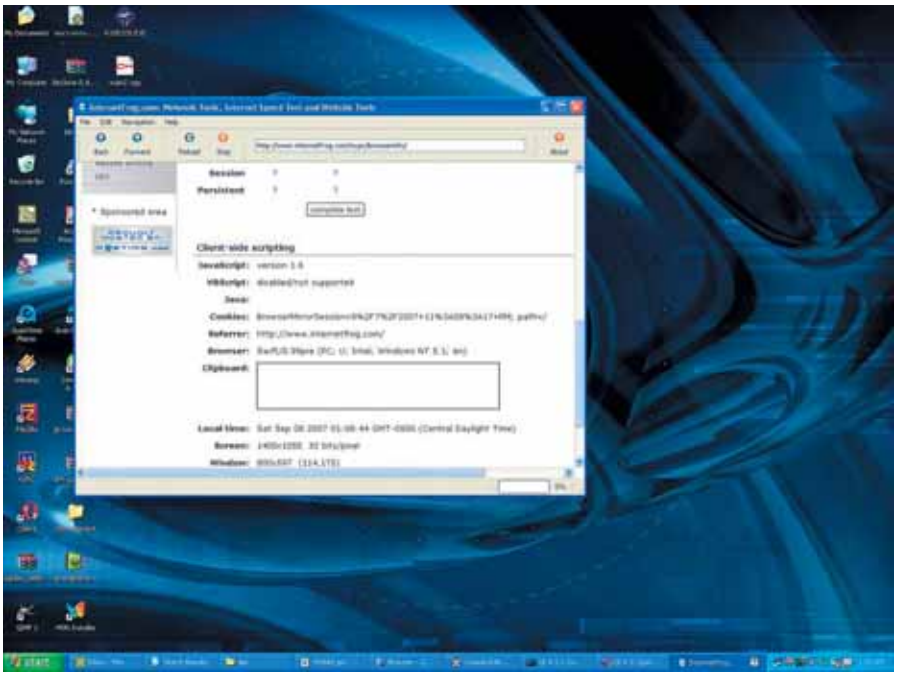
Sous Windows, une solution Visual Studio est

Web Inspector

Web Inspector est un outil pour le web designer qui permet la visualisation du code source et de la structure DOM des pages XHTML. Il est compilé par défaut avec Webkit. Une fois activé, on y accède directement par un clic droit sur la page via le menu contextuel " Inspect Element ". A tester !

iPhone & ipod touch

Les deux produits phares d'Apple que sont le téléphone mobile iPhone ainsi que le Ipod touch bénéficient d'une version modifiée de Safari. L'iPhone dispose ainsi de son navigateur web avec une forte expérience utilisateur. Les pages s'ajustent parfaitement aux proportions de l'écran et le zoom est d'une efficacité redoutable.
Webkit est d'ailleurs utilisé sur de nombreux mobiles via notamment Opera Mini et par Nokia pour son modèle S60.



générée lors de la compilation initiale. Vous n'avez plus qu'à l'ouvrir et à créer votre interface et la connecter.
Linux dispose quant à lui de l'environnement KDevelop. Il permettra de réaliser l'interface graphique de manière intuitive. La connexion des composants avec WebKit se fera par du code. Dans tous les cas, voici les actions à appeler pour chaque bouton :

```
Précédent goBack:
Suivant goForward:
Recharger reload:
Arrêter stopLoading:
Imprimer print:
Réduire le texte makeTextSmaller:
Agrandir le texte makeTextLarger:
```

La zone d'affichage du navigateur est un composant WebView. Pour changer l'url via la barre d'adresse, vous devrez appeler la fonction LoadRequest. Celle-ci prend en paramètre l'adresse de destination ainsi que l'objet WebView.
Il est possible d'aller bien plus loin, par exemple au niveau de la gestion de l'historique, des onglets ou encore en implémentant le spoofing. Je vous invite à utiliser le guide du développeur WebKit fourni par Apple. Vous y trouverez non seulement des tutoriels mais également la documentation de la bibliothèque.

Créer du contenu optimisé

Lorsque l'on navigue sur Internet il est fréquent de voir un logo précisant que le site est optimisé pour tel ou tel navigateur. Apple a souhaité privilégier la compatibilité, ainsi, en respectant les standards du consortium W3C, vous ne devriez jamais rencontrer de problème. Cependant, développer en respectant le standard W3C contraint les web designers à prendre quelques précautions. Pour commencer, la page web doit respecter la norme XHTML 1.0, pour cela vous devez ajouter l'entête suivant :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xh
tml1-transitional.dtd">
```

De cette façon, le navigateur connaît explicitement le langage que vous utilisez ainsi que le niveau de rigueur. De plus, il est possible de tester la validation de vos pages sur le site web du w3c : http://validator.w3.org. Il existe un certain nombre de différences entre le HTML et le XHTML, il serait trop long de toutes les énumérer ici. Sachez simplement que tout le code doit être écrit en minuscules et que le fichier doit être de type XML, c'est-à-dire que toute balise doit être fermée et que l'imbrication des balises doit être rigoureuse. Le style devant être centralisé dans un fichier CSS, vous bannirez donc les balises HTML du type <b>, <i>, <font> ou encore .
Web Kit propose des règles de style supplémentaires permettant d'augmenter l'expérience utilisateur. Cependant, ces règles ne seront pas visibles pour d'autres navigateurs comme Firefox et Internet Explorer. Tous les navigateurs proposent leur petit grain de sel à ce niveau là. WebKit supporte les CSS 3 en bêta.

Conclusion

Webkit est un moteur de navigateur en passe de devenir la référence et qui est déjà un concurrent sérieux pour Gecko, le moteur de Firefox, ainsi que pour le navigateur de Microsoft : Internet Explorer. L'avenir nous dira lequel des trois deviendra le numéro 1 dans cette guerre des navigateurs. Un combat dans lequel s'affrontent les plus grands : Apple, Google et Microsoft.

Liens :

http://developer.apple.com/documentation/Cocoa/Conceptual/DisplayWebContent : Guide du développeur Web Kit

http://www.w3.org/TR/xhtml1 : Site de référence sur le langage XHTML

http://trolltech.com/products/qt : Site officiel de la librairie QT

Loïc Guillois

Joli code avez-vous dit ?

Par essence, un joli code permet de donner un aperçu rapide de la structure du code sans avoir à le lire complètement. C'est ce que j'appelle une "analyse visuelle": discerner le flux et l'importance relative du code d'après sa forme. La conception d'un tel code exige une certaine quantité d'artifice pour transformer le code de travail en un code de travail lisible, tout en laissant à l'utilisateur et non au compilateur, la charge de laisser des signaux visuels.

Les sept piliers d'un joli code sont quelque peu entremêlés. Les cinq premiers portent sur la formule; les deux derniers font appel à l'intuition. Vous trouverez des cas concrets de ces sept piliers dans jam/make.c. Il s'agit d'un exemple avec le langage C, mais ces pratiques peuvent s'appliquer à n'importe quel langage de programmation de haut niveau.

Mélange

Les modifications apportées au code devraient se mélanger avec le style d'origine. Il ne devrait pas être possible de discerner les modifications précédentes apportées à un fichier sans voir les révisions. Rien ne vient davantage obscurcir les signaux visuels essentiels qu'un changement de style.

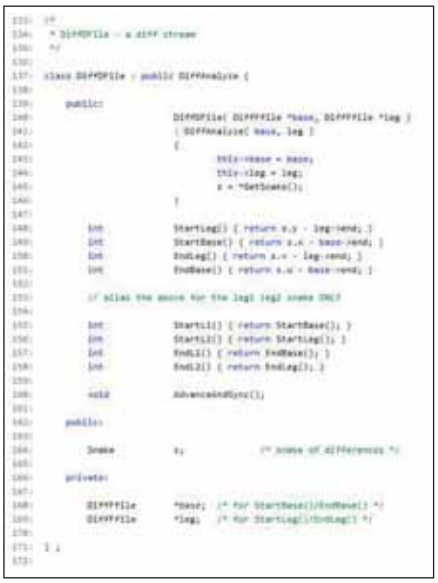
Cette pratique devrait être appliquée aussi largement que possible: de manière absolue à l'intérieur des fonctions, généralement à l'intérieur d'un fichier et si vous avez de la chance, sur le système.

Lorsque vous êtes en présence d'un code absolument hideux ou négligé, et que vous ne pouvez rien déduire de sa structure d'un simple coup d'œil, il peut être utile d'envisager de le reformater intégralement. La compréhension profonde qui découle d'un tel remodelage sera alors accessible à chaque lecteur.

make.c en est à sa vingt-septième révision, sans réécriture majeure.

Comme un livre

Les colonnes doivent rester étroites. Comme dans les livres et les magazines, le code doit être étroit pour capter le regard. Comme je l'explique dans le paragraphe *Surmonter les indentations,* le bord gauche du code maintient la structure tandis que le bord droit maintient le détail; les grandes lignes mélangent les zones de structure et de détail, ce qui induit une confusion du lecteur.



Il existe de nombreux remèdes aux lignes longues: utilisez des noms plus courts (voir : *Désencombrement*); si vous avez plusieurs arguments de fonction, alignez un argument par ligne (voir : *Faire ressembler ce qui est semblable*}; et appliquez une logique linéaire (voir : *Surmonter les indentations*).

À vue de nez, un format de 80 colonnes s'adapte partout; il faut toutefois admettre qu'il n'est pas physiquement possible de formater certains codes (comme les grandes tables) dans cette limite stricte.

make.c utilise des noms de variable courts tout en maintenant fermement l'indentation pour rester étroit.

Dénouer les blocs de code

Fragmentez le code en blocs logiques dans les fonctions et dénouez l'objectif des blocs séparés afin que chacun fasse une chose ou un type de chose unique. Un lecteur ne peut éviter une lecture totale que si une inspection superficielle peut révéler toute la nature du bloc.

Selon une approche, une fonction est en réalité une suite de mini-fonctions: chaque mini-fonction est un bloc et devrait être relativement autonome. Autrement dit, les informations transmises d'un bloc à l'autre devraient être attentivement envisagées.

Selon une autre approche, une fonction est une grande opération unique. Dans ce cas, des blocs distincts pourraient être organisés le long de lignes de type d'activité: par exemple, l'initialisation de variables, la vérification des paramètres, le calcul des résultats, le retour des résultats et l'impression d'un rapport de débogage.

Cette pratique est appliquée de manière récurrente pour les sous-blocs à l'intérieur de grands blocs (comme les grandes boucles).

make.c est un code hybride: il sépare un bloc de débogage/traçage, tout en étant une suite de mini-fonctions, dont chaque bloc a une fonction distincte.

Commenter les blocs de code

Compensez les blocs de code par des espaces blancs et des commentaires qui décrivent chaque bloc. Les gros blocs (avec des commentaires sur plusieurs lignes) peuvent imbriquer de petits blocs (avec des commentaires d'une seule ligne).

Les commentaires devraient reformuler le contenu du bloc et ne pas être une traduction littérale en français. De cette manière, même si votre code est impénétrable et vos commentaires repoussants, le lecteur peut au moins essayer de trianguler l'objectif réel.

Les commentaires volumineux sont nécessaires pour les blocs de code subtils ou problématiques, pas nécessairement pour les gros blocs de code.

Historiquement parlant, j'ai un rapport de 15% de blanc et de 25% de lignes de commentaire. make.c va aussi loin que le nombre de blocs et de sous-blocs pour simplifier l'identification.

```
137:  class DiffDFile : public DiffAnalyze (
138:  public:
139:      DiffDFile( DiffFfile *base, DiffFfile *leg )
140:      : DiffAnalyze( base, leg ) (
141:      this->base = base;
142:      this->leg = leg;
143:      s = *GetSnake();
144:      )
145:      int StartLeg() ( return s.y - leg->end; )
146:      int StartBase() ( return s.x - base->end; )
147:      int EndLeg() ( return s.v - leg->end; )
148:      int EndBase() ( return s.u - base->end; )
149:      // alias the above for the leg1 leg2 snake ONL
150:      int StartL1() ( return StartBase(); )
151:      int StartL2() ( return StartLeg(); )
152:      int EndL1() ( return EndBase(); )
153:      int EndL2() ( return EndLeg(); )
154:  );
```

Mauvais code

```
137:  class DiffDFile : public DiffAnalyze (
138:
139:      public:
140:                          DiffDFile( DiffFfile *base, DiffFfile *leg )
141:                          : DiffAnalyze( base, leg )
142:                          (
143:                                  this->base = base;
144:                                  this->leg = leg;
145:                                  s = *GetSnake();
146:                          )
147:
148:          int             StartLeg() ( return s.y - leg->end; )
149:          int             StartBase() ( return s.x - base->end; )
150:          int             EndLeg() ( return s.v - leg->end; )
151:          int             EndBase() ( return s.u - base->end; )
152:
153:          // alias the above for the leg1 leg2 snake ONLY
154:
155:          int             StartL1() ( return StartBase(); )
156:          int             StartL2() ( return StartLeg(); )
157:          int             EndL1() ( return EndBase(); )
158:          int             EndL2() ( return EndLeg(); )
159:
160:          void            AdvanceAndSync();
161:  ) ;
```

Bon code

Désencombrement

Réduire encore et toujours. Supprimez tout ce qui pourrait distraire le lecteur.

Utilisez des noms courts (comme i, x) pour les variables avec un spectre local court ou des noms ubiquistes. Utilisez des noms de longueur moyenne pour les membres. Réservez les noms plus longs au spectre international (comme les interfaces distantes ou les systèmes d'exploitation). D'une manière générale: plus le spectre est court, plus le nom doit être petit. Les longs noms descriptifs peuvent aider le lecteur la première fois, mais le gêner par la suite.

Éliminez la syntaxe superflue (comme '!= 0', les habillages inutiles et les parenthèses pesantes). Ces surcharges peuvent s'avérer utiles pour former un programmeur débutant, mais sont totalement inutiles pour toute personne procédant à un débogage sérieux et présentent une gêne pour les personnes qui essaient d'avoir une vision globale (ou intermédiaire).

Laissez tomber les 'ifdef notdef' et autres codes morts. Il est déjà assez difficile de lire un code vivant. Les systèmes de GCL maintiennent l'ancien code.

make.c utilise presque exclusivement des noms courts, ne s'encombre pratiquement pas de syntaxe inutile et ne comprend pas de code mort.

Faire ressembler ce qui est semblable

Deux parties ou plus d'un code qui font la même chose ou presque devraient se ressembler. Rien ne permet au lecteur d'aller plus vite que de voir un modèle.

De plus, ces parties de code qui se ressemblent devrait être alignées à la suite les unes des autres. Un regroupement réduit le nombre d'entités que le lecteur doit saisir et constitue une approche critique visant à simplifier l'apparente complexité du code.

Cette pratique est plus efficace si elle est utilisée conjointement à "Dénouer les blocs de code": un bloc de code distinct, composé d'un modèle de ligne ayant un seul objectif, est une entité simple pour le lecteur.

Malheureusement, cette pratique doit être appliquée partout et exige du doigté. Par chance, elle n'affecte que rarement le code généré. Voici quelques exemples :

Initialisez les variables conjointement.

- Utilisez 'this' de manière cohérente (ou ne l'utilisez pas).
- Alignez les paramètres sur un appel de fonction long.
- Utilisez {} de manière cohérente autour des clauses if/else: placez-les dans tous les blocs ou dans aucun.
- Mettez la { d'une clause if/for/while sur sa ligne (car la } de gauche est).
- Séparez les conditionnelles au niveau de &&'s ou ||'s et alignez-les.
- Le bloc "4d" de make.c est un exemple élaboré du nombre de lignes de code qui doivent être créées pour ressembler à une seule entité.

Surmonter les indentations

Le bord gauche du code définit sa structure, tandis que le bord droit maintient le détail. Vous devez lutter contre les indentations pour sauvegarder cette propriété. Un code qui passe trop rapidement de gauche à droite (et inversement) mélange le flux de contrôle majeur avec des détails mineurs.

Forcez l'alignement du flux de contrôle principal sur le coté gauche, avec un niveau d'indentation pour les instructions if/while/for/do/switch. Utilisez break, continue, return, voire goto pour forcer l'alignement gauche du code. Réorganisez les blocs conditionnels de sorte que le bloc ayant la sortie la plus rapide soit placé en premier, suivi du bloc return (ou break ou continue) de sorte que l'autre jambe puisse continuer au même niveau d'indentation.

Bien entendu, le vrai code exige des sous-blocs substantiels, nécessairement indentés, et ces sous-blocs devront ensuite lutter pour leur indentation. Vous devez vous assurer que vous n'opérez pas une indentation uniquement pour déplacer une structure vers le côté détail du code, ni même à cause d'un artéfact du langage de programmation.

C'est l'aspect le plus difficile de ces pratiques, car il exige le plus de stratagèmes et peut souvent influencer la mise en œuvre des fonctions distinctes.

make.c utilise rarement plus de deux niveaux d'indentation; cela n'est nullement un accident. Les sept piliers d'un joli code ne sont pas le dernier mot, ni même le premier, sur un bon code. Mais un bon code est à mon avis, ce qui distingue un code lisible et présentable des autres codes.

Christopher Seiwald

Extrait du livre **Beautiful code** *(Editions O'Reilly) dont Laura Wingerd, vice president of product technology de Perforce et Christopher Seiwald, président et CEO de Perforce, ont écrit le chapitre 32.*

site : www.perforce.com/beautifulcode/

Les outils Microsoft pour créer et développer des applications RIA

L'offre RIA de Microsoft est très large. De WPF en passant par Silverlight pour aboutir à ASP.NET AJAX. Je vous propose dans cet article de présenter cette offre ainsi que la plate-forme nécessaire à la création et au développement de RIA.

L'interactivité des sites internet est de plus en plus évoluée, le Web 2.0 ayant montré la voie des applications internet riche. Nous sommes donc passés des RWA pour Rich Web Application, qui sont interactives et multimédia, mais qui restent orientées Serveur, vers les RIA pour Rich Internet Application qui sont orientées clients. Ces dernières, basées sur des vidéos, des animations portées, par Flash et Flex initialement, se voient proposer de nouveaux outils et techniques par Microsoft et notamment Silverlight. Nous vous invitons dans cet article à faire le tour des technologies RIA proposées par Microsoft et à la présentation des plates-formes fournies pour leur création.

Les technologies proposées

Nombreuses sont les technologies Microsoft disponibles pour créer des RIA/RDA. On peut recenser les suivantes :

- **WPF :** Windows Presentation Foundation est la nouvelle technologie Microsoft pour la création d'interfaces Windows Vista. Celle-ci, via les applications XBAP, possède le moyen d'être publiée sur le Web. Nous pouvons alors développer des applications Windows pour le Web. La limitation qui demeure est le déploiement à destination d'internet Explorer uniquement.
- **ASP.NET AJAX :** Ajax, XHTML, JavaScript et CSS 2.0 sont les pierres fondatrices du Web 2.0. Si les trois derniers sont pris en charge depuis longtemps par Microsoft, Ajax manquait. Pour combler cela, Microsoft nous propose une extension Ajax à ASP.NET. celle-ci nous permet de développer des applications serveur bien plus interactives et évoluées.
- **Silverlight :** C'est la technologie Microsoft faite pour le RIA. Multi navigateur, multi-plateforme, elle permet la création d'applications multimédia dynamiques et très interactives. C'est une application Cliente s'exécutant dans le contexte du Navigateur.

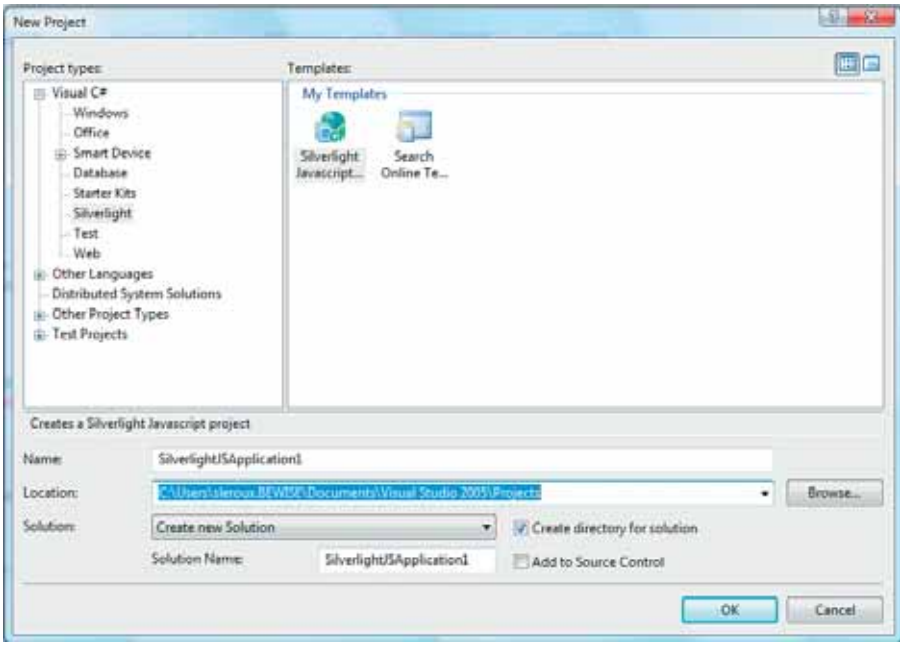


Plate-forme de Développement

La plate-forme de développement Microsoft quel que soit le langage est Visual Studio. Il existe aujourd'hui deux versions principales de l'éditeur dont une en release.

- Visual Studio 2005 : Outil de développement en release, il permet de développer à destination du Framework 2.0 et 3.0 (avec l'installation du SDK du Framework 3.0) disponible en téléchargement à cette adresse http://msdn2.microsoft.com/en-us/ie/bb188238.aspx
- Visual Studio 2008 : Remplaçant de la version 2005, la version 2008 est aujourd'hui est disponible en version finale. Cet éditeur permet de développer à destination des Framework 2.0 à 3.5 et est disponible à cette adresse http://msdn2 .microsoft.com/fr-fr/vstudio/aa700831.aspx

Regardons à présent, par technologie, les outils de développement disponibles et nécessaires.

WPF et XBAP

Vous trouverez ci-dessous les outils nécessaires pour le développement d'application WPF / XBAP.

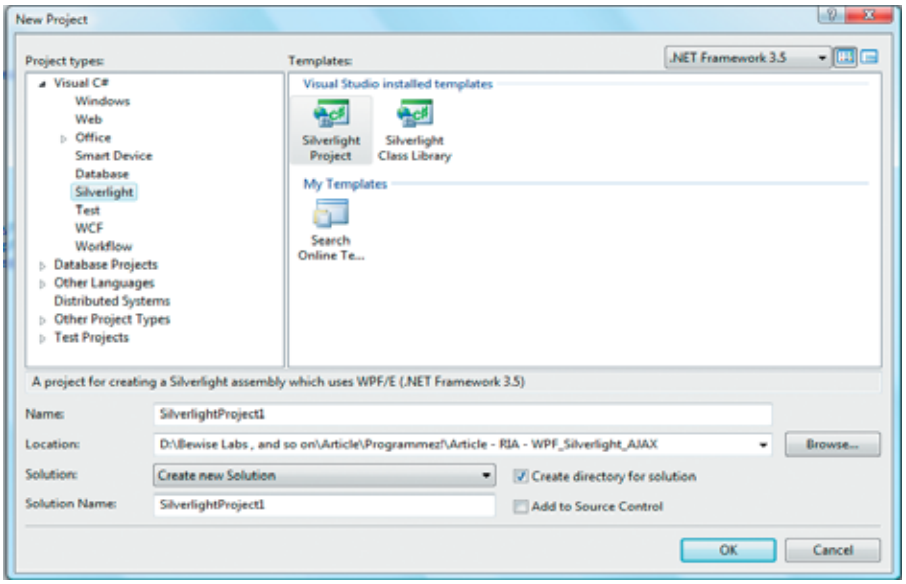
Visual studio 2005 :

Il est possible de développer des applications WPF à partir du Framework .NET 3.0. Pour Cela, il vous faut vous munir de Visual studio 2005. Vous devez aussi installer le SDK du Framework 3.0 accessible à cette adresse : http://www.microsoft.com/downloads/details.aspx?FamilyId=C2B1E300-F358-4523-B479-F53D234CDCCF&displaylang=en

Enfin, il faut installer les extensions WPF permettant d'avoir accès au modèle de projet WPF. http://www.microsoft.com/downloads/details.aspx?FamilyId=F54F5537-CC86-4BF5-AE44-F5A1E805680D&displaylang=en

Visual studio 2008 :

Ce dernier prend en charge directement le développement d'applications XBAP, soit avec le Framework 3.0, soit avec le Framework 3.5.



La version 2008 apporte une meilleure fiabilité et des fonctionnalités supplémentaires aux concepteurs d'applications WPF.

SILVERLIGHT 1.0

Vous trouverez ci-dessous les outils nécessaires pour le développement d'applications Silverlight 1.0.

Pour développer des applications Silverlight 1.0 il est nécessaire de posséder Visual studio 2005. La première étape pour le développement d'applications Silverlight est l'installation du SDK. Ce dernier est accessible sur le site officiel de silverlight :
http://silverlight.net/GetStarted/
Le SDK installera le modèle de projets Silverlight pour Visual studio. Ce dernier est accessible par la fenêtre de création de projet au nœud " Silverlight " comme montré ci-dessus :

SILVERLIGHT 1.1

Vous trouverez ci-dessous les outils nécessaires pour le développement d'applications Silverlight 1.1.

C'est le seul outil permettant la création d'un site Silverlight 1.1 Alpha. Il faut aussi installer le SDK de la version 1.1 disponible lui aussi sur le site officiel de Silverlight :
http://silverlight.net/GetStarted/
Les modèles de projet pour Visual Studio 2008 sont à installer séparément du SDK contrairement à la version 1.0. Ils sont disponibles à cette adresse : http://www.microsoft.com/downloads/details.aspx?FamilyId=B52AEB39-1F10-49A6-85FC-A0A19CAC99AF&displaylang=en. Une fois tout ceci installé, vous avez la possibilité de choisir les modèles de projets adéquats au sein de Visual studio 2008.

ASP.NET AJAX

Vous trouverez ci-dessous les outils nécessaires pour le développement d'applications ASP.NET AJAX.

Visual Studio 2005

Pour développer des applications ASP.NET AJAX, il est nécessaire de posséder Visual studio 2005. Il faut ensuite installer l'extension ASP.NET AJAX qui se trouve sur le site officiel :
http://asp.net/ajax/downloads/
Il est recommandé, mais pas obligatoire pour le développement, de récupérer aussi les contrôles AJAX fournis par Microsoft qui sont disponibles à cette adresse :
http://asp.net/ajax/downloads/

Visual Studio 2008

Visual studio 2008 intègre l'extension ASP.NET AJAX. Il n'est pas nécessaire d'installer autre chose (hormis les contrôles AJAX éventuellement).

Plate-forme de conception

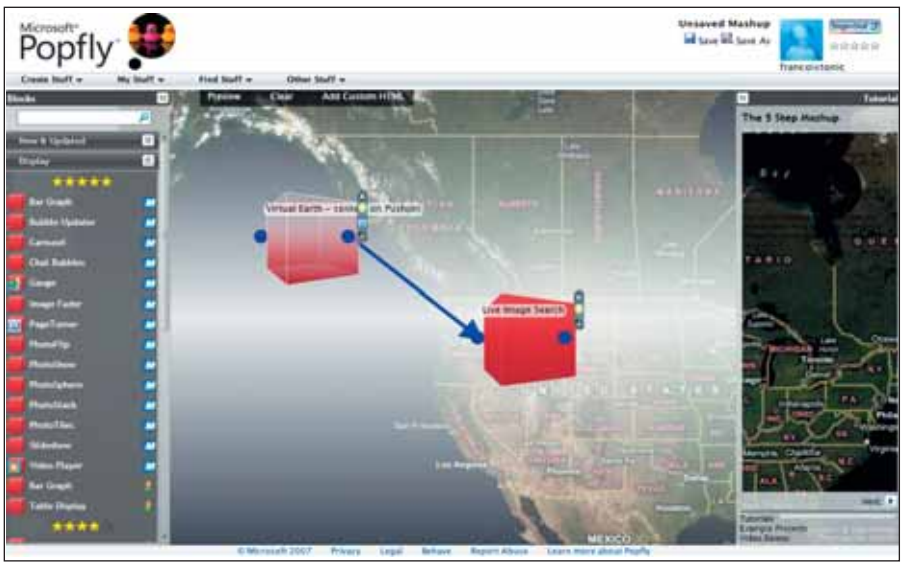
De nouveaux outils ont fait leur apparition dans le paysage Microsoft. Ces outils orientés Designer mettent à disposition tous les moyens nécessaires au Designer pour communiquer avec les équipes de développement. Mais bien sûr, leur but premier est de permettre au graphiste d'ajouter des effets visuels, via la mise en page ou encore l'ajout de multimédia, au travail du développeur. Ces outils de la gamme Expression sont les suivants :

- Expression Blend qui permet de créer et modifier des projets WPF ou Silverlight. Il est aujourd'hui en version 2 bêta et disponible à cette adresse : http://www.microsoft.com/ Expression/products/overview.aspx?key=blend
- Expression Design qui permet la retouche d'image et qui est disponible à cette adresse : http://www.microsoft.com/expression/products/overview.aspx?key=design

CONCLUSION

L'offre RIA de Microsoft est très large et elle est accompagnée d'outils d'aide à la création. Le développeur utilisera Visual Studio accompagné des SDK nécessaires. Les Designer, via la gamme Expression, pourront modifier ou créer des applications WPF ou Silverlight.

Sacha LEROUX
Sacha.leroux@bewise.fr - Bewise
Centre de compétences Team System.

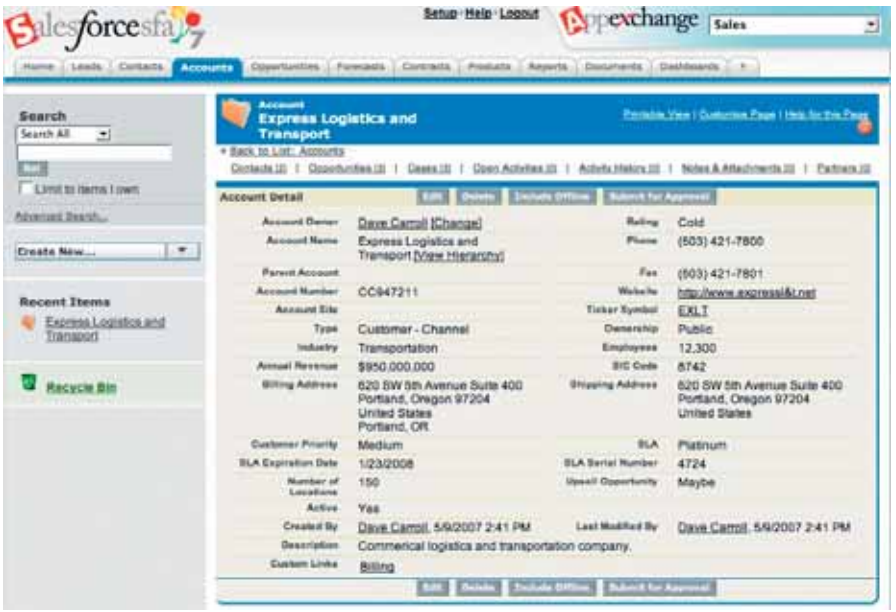


Popfly, l'outil de développement "tout public" pour le mashup et la RIA ?

Salesforce.com se dote d'une couche graphique basée sur MVC

Spécialiste de la gestion de la relation client à la demande, Salesforce.com a ouvert sa plate-forme technique - force.com - en 2005 pour permettre à ses clients et à des éditeurs de développer leurs propres applications métier. Il vient d'ajouter une couche graphique basée sur un modèle MVC.

En moins de deux ans, 335 éditeurs et 35 000 clients ont créé environ 45 000 nouvelles applications métier allant de la gestion d'une agence immobilière à la gestion des ressources humaines. Force.com est donc devenue une plate-forme de développement alternative à .NET et Java. Certains éditeurs comme Coda ont même choisi de développer nativement leurs outils pour force.com sans fournir de version pour Windows ou Linux. Une application basée sur Force.com s'exécute sur les serveurs de salesforce.com. Elle repose sur des objets métier créés par paramétrage dans une interface en ligne. Aucune manipulation de base de données n'est nécessaire puisque force.com génère automatiquement le mapping objet-relationnel. Chaque objet est également exposé automatiquement sous la forme d'un service web et possède une interface utilisateur par défaut. Comme la plate-forme permet à deux applications différentes de partager leurs objets, force.com propose le même modèle de données qu'un ERP.



Une application basée sur Apex

Un langage de tag basé sur XML

Jusqu'à présent, les applications basées sur force.com héritaient par défaut de l'interface graphique standard de l'éditeur et il était impossible de la changer. Lors de sa conférence annuelle qui se tenait le mois dernier à San Francisco, l'éditeur a présenté une nouvelle couche – Visualforce - qui complète sa plate-forme technique. Avec Visualforce, les développeurs peuvent enfin personnaliser l'interface de leurs applications à l'aide d'un langage de tag basé sur XML.

Visualforce s'appuie sur une architecture MVC (Modèle Vue Contrôleur) dont l'ensemble des composants sont générés par défaut par la plate-forme. Il suffit donc de les modifier pour personnaliser une application. Une vue peut s'appuyer sur du code HTML, DHTML ou même Flash puisque Adobe et Salesforce.com ont noué un partenariat. Pour adapter l'interface utilisateur à différents périphériques - PC, borne interactive, iPhone, etc. – il suffit de créer plusieurs vues. Bien entendu, il est possible de modifier le comportement des contrôleurs à l'aide du langage de bas niveau de force.com baptisé Apex. La syntaxe d'Apex est très proche de Java. Apex et sa composante graphique permettent d'appeler des API web publiques pour réaliser des mash-up. Pour l'instant, Salesforce.com fournit par défaut 50 composants graphiques pré-développés. Mais les partenaires de l'éditeur en proposeront bientôt sur le catalogue applicatif AppExchange. Encore en bêta, Visualforce ne sera disponible qu'au printemps prochain. Pour l'instant, l'édition des interfaces s'effectue dans un plug-in Eclipse ou dans n'importe quel éditeur de texte. Mais salesforce.com promet qu'un outil de design sera disponible dans les 6 mois. En revanche, la manipulation du modèle de données et le développement de fonctions de bas niveau sont assistés par deux outils, respectivement sforce Explorer pour le code SOQL (le SQL propriétaire de force.com) et un plug-in pour Eclipse pour le code Apex (auto complétion, etc.). Le débogage de l'application s'effectue en ligne, directement sur l'instance de pré-production et de test (sandbox) du serveur. force.com et visualforce s'adressent clairement aux PME et aux départements de grandes entreprises qui manquent de ressources informatiques. Ce sont d'excellents outils pour construire très rapidement des applications de gestion simples qui pourront se complexifier au fil du temps sans engendrer un coût de maintenance évolutive élevé. Force.com est également une bonne plate-forme pour créer des mash-up d'entreprises. Pour ces différents usages, force.com peut être considéré comme un concurrent sérieux de .Net et Java, bien plus complexes à mettre en œuvre. En revanche, pour des applications de gestion très spécifiques (notamment en terme de logique), le système de règles et de workflow de force.com (des formulaires) montre vite ses limites face aux possibilités de .NET et Java EE.

Frédéric Bordage

amen
IN WEB WE TRUST

serveurs dédiés DUO /

Vous n'avez pas à nous prier pour vous offrir deux fois plus de performance !

NOUVEAU
Serveurs dédiés DUO

AMD 64 Opteron™

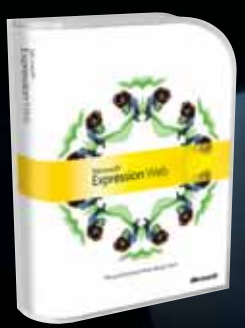
Pour les professionnels les plus exigeants, AMEN lance la nouvelle gamme de serveurs dédiés DUO basée sur des processeurs double coeur, disques durs en RAID, pour vous offrir 2 fois plus de puissance.

DUO 1000 ► 99 € ht/mois*
(118,40 € ttc/mois*)
AMD Opteron 1210 - 2x1,8GHz - RAM 1GB
Disque dur 2x160GB - Raid Soft
2 adresses IP - Interface Plesk 8 jusqu'à 100 domaines - Trafic illimité

DUO 2000 ► 149 € ht/mois*
(178,20 € ttc/mois*)
AMD Opteron 1212 - 2x2,0GHz - RAM 2GB
Disque dur 2x200GB - Raid 1 matériel
4 adresses IP - Interface Plesk 8 jusqu'à 300 domaines - Trafic illimité

DUO 4000 ► 199 € ht/mois*
(238,00 € ttc/mois*)
AMD Opteron 1214 - 2x2,2GHz - RAM 4GB
Disque dur 2x250GB - Raid 1 matériel
6 adresses IP - Interface Plesk 8 jusqu'à 300 domaines - Trafic illimité

Compatibles &



OFFERT ! **

Microsoft Expression Web vous offre tous les outils dont vous avez besoin pour créer des sites web dynamiques de qualité professionnelle : création CSS, prise en charge XML, ASP.NET 2.0, DHTML, XHTML, CSS, Javascript…

Nous avons foi en un idéal de services, surtout lorsqu'il vous permet de bénéficier des dernières avancées techniques : architecture réseau redondée, bande passante dédiée 2GB, haute disponibilité (99,9%), assistance technique par mail et téléphone 6j/7[1]. Quant à notre 'Garantie satisfait ou remboursé'[2], elle vous permettra d'atteindre la sérénité absolue. **Si vous croyez au web, vous croirez en nous.**

► Pour plus de renseignements **0 892 55 66 77** (0,34€ / min) ou **www.amen.fr**

Intégration du plug-in Google Gears dans une application desktop xulrunner

Les applications Web ont beaucoup évolué ces dernières années, tout particulièrement depuis l'avènement du concept Web 2.0. Nous disposons de plus en plus d'applications Web dynamiques, élégantes et riches en fonctionnalités se rapprochant de leurs homologues Desktop. Néanmoins, il reste encore beaucoup d'applications qui ne peuvent avoir leur équivalent en version web car les navigateurs sont encore limités par leur nature même. Comme à son habitude, Google se démarque par son dynamisme et son innovation en proposant une extension aux navigateurs afin de gérer ces nouvelles problématiques : Google Gears.

Notre article va plus loin en proposant l'intégration des mécanismes Gears dans un contexte RDA (Rich Desktop Application), développée sur le framework de Mozilla : xulrunner. Le but étant de vous initier au développement d'applications lourdes bénéficiant du meilleur des innovations web : les technologies du présent n'ont qu'à bien se tenir. Dans une première partie, nous présenterons les deux technologies et leur installation, puis un exemple d'application utilisant Google Gears dans un projet xulrunner.

Google Gears ouvre de nouvelles perspectives aux applications web

Gears se présente comme un module permettant d'étendre les fonctionnalités du navigateur Internet " bridé " dans son mode client serveur. L'idée derrière Gears est de proposer tout type de nouvelle fonctionnalité, en commençant par :

- L'installation d'une base de données locale pour gérer les données dynamiques
- Un serveur web interne de gestion de ressources statiques
- Des méthodes de conception d'applications AJAX standardisées

Les gains apportés par l'extension sont directs :

- Gestion de la perte de connexion réseau avec continuité des services proposés
- Nouvelles perspectives de développements d'applications Web
- Amélioration des performances car la boucle client serveur distant est brisée

Ce framework prend la forme d'une extension pour les navigateurs Mozilla Firefox (version 1.5 ou ultérieure) et Internet Explorer (version 6.0 ou ultérieure). À l'heure actuelle, plusieurs versions de Google Gears sont disponibles. La plus récente à l'écriture de ce document est la version 0.2 (0.2.2.0). Autre bonne nouvelle : Google a choisi d'ouvrir le code de l'application sous la licence Open Source BSD (http://www.opensource.org/licences/bsd-license.php). Cela signifie que tout le monde peut consulter le code source, participer aux nouvelles fonctionnalités, envoyer des correctifs… Ce mode de travail est avantageux pour tous les utilisateurs qui peuvent se reposer sur la communauté existante autour du projet.

Google Gears : un framework au sommet des technologies Web

La nouvelle " killer application " de Google est avant tout un mélange savamment composé de plusieurs technologies puissantes :

- Ajax : pour mettre à jour les interfaces en temps réel en allant chercher l'information nécessaire sur le serveur
- SQLite : base de données légère pour le stockage des données locales
- JSON : format JavaScript d'échange de données entre les différents mondes applicatifs

La gestion d'un site en mode déconnecté

Les sites Web d'aujourd'hui proposent des fonctionnalités uniques, intimement liées à la connexion permanente à l'Internet. Sans connexion, le site, et donc ses fonctionnalités, est tout simplement inaccessible. L'extension Google Gears, installée sur le navigateur du client, permet de proposer à l'utilisateur une continuité de certains de ses services. Ce dernier va pouvoir continuer à naviguer et utiliser tout ou partie du site qui sera copié sur son disque dur local. Cette continuité de services est à la fois intéressante pour l'utilisateur, car il peut continuer à utiliser le site en mode hors-ligne, et pour les propriétaires du site, car ces utilisateurs continuent à utiliser leur site et non pas un outil de type client lourd. Voici comment s'effectuent les transactions entre les modes " connecté " et " déconnecté " :

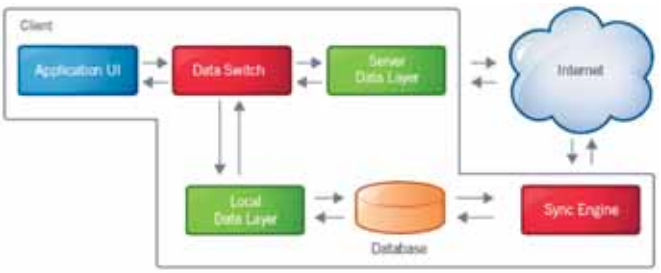


Figure 1 : Architecture des composants Google Gears.

Démonstration de l'utilisation de Google Gears dans une application concrète

Nous allons, au fil de l'article, construire ensemble une application Internet mettant en évidence la puissance de Google Gears dans une application web locale. Afin de démontrer l'utilité de la gestion du manque de connectivité, nous allons travailler sur une application de type RDA. Ce type d'application allie la robustesse d'une application cliente lourde, tout en se reposant sur les technologies Internet. Le fra-

mework utilisé est celui de Mozilla : XPFE (Cross Platform Front End), avec l'utilisation du projet xulrunner. L'application test est développée en XUL, qui est le langage utilisé par Mozilla pour mettre en forme les interfaces de ses applications. Soyez rassuré si vous ne connaissez pas encore le XUL, il s'agit d'un dialecte XML ressemblant au XHTML. Les fichiers sont hébergés sur un serveur web distant. L'application xulrunner va consulter ce site en mode connecté. Nous allons voir comment ces mêmes pages peuvent être affichées et utilisées lorsque la connexion au site n'existe plus.

Mettre en œuvre Gears

Pour Google Gears dans un environnement xulrunner, il faut procéder selon les étapes suivantes :

Installation de Gears dans le framework Mozilla :
Comme cité dans l'introduction, vous pouvez télécharger la dernière version (v0.2) de Gears (http://www.code.google.com/p/google-gears/downloads/list). Fermez tous les navigateurs, et lancez l'installation. Sous Windows, les fichiers sont copiés dans " C:\Program Files\Google\Google Gears ", ce répertoire contenant l'extension Firefox qui sera elle-même utilisée dans xulrunner.
Mise à disposition de l'extension Gears pour son application xulrunner :
Il faut éditer la base de registre pour ajouter une clé identifiant l'extension par rapport à l'application cible (" GearsRunner " dans l'exemple).

```
[HKEY_LOCAL_MACHINE\SOFTWARE\GearsRunner]
[HKEY_LOCAL_MACHINE\SOFTWARE\GearsRunner\Extensions]
"{000a9d1c-beef-4f90-9363-039d445309b8}"="C:\\Program
Files\\Google\\Google Gears\\Firefox\\"
```

Enregistrement dans Gears de l'application cible :
L'extension Google Gears est initialement développée pour IE et Firefox. Ce dernier étant basé sur les mêmes technologies que celles de xulrunner, il suffit de rajouter une nouvelle application cible dans le code l'extension Gears (fichier " C:\Program Files\Google\Google Gears\Firefox\install.rdf ") en concaténant le texte suivant :

```
<em:targetApplication>
  <Description>
    <em:id>gearsrunner@edis-consulting.com</em:id>
<em:minVersion>0.1</em:minVersion>
<em:maxVersion>1.0</em:maxVersion>
  </Description>
</em:targetApplication>
```

Activer le système de gestionnaire d'extension dans xulrunner : (tâche déjà réalisée dans les sources à télécharger)
Rajoutez dans le fichier application.ini de l'application xulrunner :

```
[XRE]
EnableExtensionManager=1
```

Si malgré tout, cela ne fonctionne pas, essayez de supprimer les fichiers de cache de xulrunner dans C:\Documents and Settings\[votre_utilisateur]\Application Data\GearsRunner
L'application GearsRunner se lance en double-cliquant sur le fichier " demo-gears.exe ". Au premier lancement, une fenêtre de sécurité va demander si l'on veut permettre l'accès à ces fichiers :

Initialisation du kit de développement Gears dans notre application Desktop

A partir de ce moment, nous allons développer dans le fichier principal de l'application : gears.xul. Pour commencer, il faut déclarer que l'on veut utiliser les fonctionnalités de Google Gears. Cela s'opère en incluant le fichier gears_init.js dans la page XUL. Ce fichier permet d'initialiser Google Gears afin de pouvoir l'utiliser depuis JavaScript.

```
<script type="text/javascript" src="gears_init.js"></script>
```

Note : ce fichier devrait à terme disparaître des applications utilisant Gears.

Une fois ce fichier chargé, nous pouvons utiliser les API du framework. Il est néanmoins conseillé de tester l'intégration du " plug-in " Google Gears avec notre application. Nous le faisons lors de l'appel à la fonction JavaScript " onload " qui est appelée une fois la page chargée.

gears.xul :

```
<window ...
  onload="onLoad()"
...>
```

index.js :

```
  var isGearsLoaded = false;
function onLoad() { ...
if (!window.google || !google.gears) {
  alert("merci d'installer l'extension google gears");
  window.open("http://gears.google.com/?action=install&message=
Mon+message+d%E2%80%99accueil  >&return=http://www.edis-con
sulting.com", "_blank");
   return;
}
isGearsLoaded = true;
...}
```

Grâce à ce test, nous allons prévoir de désactiver les fonctionnalités spécifiques à Google Gears du reste de l'application quand Gears s'est correctement chargé (variable isGearsLoaded).
Nous avons désormais le framework Gears initialisé dans notre application.

Création de la " factory "

Le framework étant disponible et initialisé, nous avons alors accès aux différents composants. Afin d'instancier les objets du modèle, il faut passer par un point central de création qui est l'objet usine ***google.gears.factory*** et sa méthode ***create.***

Les différentes briques du framework sont représentées dans ce tableau :

Module	Nom de la classe	Présent dans la version
Database	beta.database	0.1
HttpRequest	beta.httprequest	0.2
LocalServer	beta.localserver	0.1
Timer	beta.timer	0.2
WorkerPool	beta.workerpool	0.1

Zoom sur les différents modules proposés

Database

Présentation

Ce composant met à la disposition du code JavaScript une base de données de type relationnel, provenant du projet Open Source SQLite. L'intérêt est de pouvoir continuer à utiliser des données dynamiques quand bien même l'utilisateur est connecté ou déconnecté d'Internet. Afin d'obtenir une instance de cet objet on procède comme précédemment :

```
var db = google.gears.factory.create('beta.database', '1.1') ;
```

Nous pouvons maintenant établir une connexion à la base de données et exécuter des requêtes SQL :

```
db.open('demogears');
db.execute('create table if not exists twitmsg (data TEXT, timestamp INT)');
```

Utilisation

Pour l'exemple d'implémentation, nous allons faire saisir des données à l'utilisateur qu'il va falloir synchroniser ou non en fonction de l'état de connectivité réseau.

Voici le code XUL permettant d'afficher un champ texte et un bouton :

gears.xul :

```
<textbox id="message" multiline="true" value=""/>
<button label="envoyer" oncommand="onSubmit()" />
```

Dans le code JavaScript maintenant, nous allons étudier ce qui se passe quand on clique sur le bouton :

mygears.js :

```
function onSubmit() {

  if (!google.gears.factory || !db) {
      alert("Google Gears n'est pas chargé.");
      return;
  }
  var elm = document.getElementById('message');
  var message = elm.value;
  var currTime = new Date().getTime();
  db.execute('INSERT INTO twitmsg VALUES (?, ?)', [message, currTime]);

  // Reset the text message
  elm.value = '';
  showQueue();
}
```

On vérifie tout d'abord que Gears est installé, puis on récupère le contenu du champ texte pour l'insérer dans la base de données locale. Nous pourrions améliorer cette fonctionnalité en détectant la présence de la connexion en temps réel et le cas échant manipuler la base de données locale ou distante.

HttpRequest

Présentation

Ce module fournit au développeur la classe " HttpRequest " supportant le standard W3C. Cette dernière permet d'appeler des pages sur Internet sans avoir à recharger toute la page. Cette méthode est la base du fonctionnement AJAX. Elle a été ajoutée dans Google Gears afin d'unifier son utilisation pour le développeur.

Utilisation

Ce module de Google Gears va nous permettre de développer une nouvelle fonctionnalité : détecter en temps réel la présence de la connectivité réseau. Le principe retenu est de faire un appel AJAX sur notre serveur toutes les X secondes, et regarder s'il y a eu une erreur ou si le contenu nous est retourné. Afin de ne pas " trop " surcharger le serveur, nous allons faire des requêtes HTTP de type " HEAD ".

La fonction JavaScript ci-dessous va être appelée périodiquement (on verra plus loin que Gears nous propose une solution à ce besoin).

```
function detect() {
request.open('HEAD',  'http://maxhouse.no-ip.info/index.html?id='  +
Math.random(9999999));
   request.onreadystatechange = function() {
       try {
           if (request.readyState == 4) {
               if (request.status == 200) {
                   timer.setTimeout(onInternetOk, 0);
               }
               else {
                   timer.setTimeout(onInternetKo, 0);
               }
           }
       }
       catch (ex) {
           timer.setTimeout(onInternetKo, 0);
       }
   };
   request.send();
}
```

Ici nous appelons une certaine adresse, jugée unique pour chaque appel via la gestion d'un nombre aléatoire dans l'URL (sinon nous avons le système de cache qui prend le dessus et nous trompe). En cas de succès de la réponse du serveur, nous appelons la méthode globale " onInternetOk " ; cette dernière affiche une image de couleur verte. Dans le cas contraire, nous appelons la méthode globale " onInternetKo " qui affiche une image de couleur rouge.

Nous savons, à partir de maintenant, si l'utilisateur est connecté au serveur web distant, et nous pouvons mettre à jour les comportements de l'application corrélés à chacun de ces états.

LocalServer

Présentation

Le module " LocalServer " est la brique de Google Gears permettant à l'utilisateur de pouvoir continuer à naviguer sur le site une fois la connexion Internet perdue. Il s'agit tout simplement d'un système de cache sur les protocoles et HTTP et HTTPS, permettant de fournir localement des ressources statiques du site (pages HTML, JavaScript, feuilles de styles, images…)

L'instanciation de l'objet correspondant se fait comme suit :

```
// instanciation de la classe localserveur
var localSrv = google.gears.factory.create('beta.localeserver', '1.1') ;
```

```
// création d'un conteneur
var store = localServer.createStore('stockage');
```

Utilisation

Cette partie est la pièce maîtresse pour gérer le contenu hors ligne de l'application. Nous allons traiter le contenu de façon manuelle, via l'ajout de boutons :

```
<hbox>
<button oncommand="createStore()" label="Capture" />
<button oncommand="removeStore()" label="Erase" />
<image src="" id="connectivity" width="50" height="50" />
</hbox>

function createStore() {
  if (!window.google || !google.gears) {
    alert("merci d'installer l'extension google gears");
    return;
  }

  store.manifestUrl = "manifest.json";
  store.checkForUpdate();

  var timerId = window.setInterval(function() {
    if (store.currentVersion) {
      window.clearInterval(timerId);
    } else if (store.updateStatus == 3) {
      textOut("Erreur: " + store.lastErrorMessage);
    }
  }, 500);
}

function removeStore() {
   if (!window.google || !google.gears) {
   alert("You must install Google Gears first.");
   return;
}

localServer.removeManagedStore(STORE_NAME);
}
```

Comme vous pouvez le remarquer, le fichier " manifest.json " contient la liste des fichiers à mettre dans le cache. Notez qu'a chaque changement d'au moins un des fichiers du cache, il faut re-générer le nom de la version du cache.

Timer

Présentation

Google Gears propose dans ce module une classe " Timer " regroupant la gestion du temps. Il est à noter que ce module est identique à ce que propose par défaut votre navigateur, mis à part que l'implémentation respecte les spécifications du HTML version 5.

Utilisation

```
// instanciation de la classe timer
var timer = google.gears.factory.create("beta.timer", "1.0");
```

Nous utilisons le module de " timer " pour appeler le test de la connectivité vu plus haut. Il faut le relancer toutes les X secondes. Nous utilisons ce code au chargement de l'application :

```
function detectInternetConnectivity() {
   timer.setInterval("detect()", 5000);
}
```

Conclusion

Google fait un cadeau extraordinaire à tous les développeurs d'un framework léger et lève une des limitations critiques des applications Web. Bien que disponible en version bêta, cette boîte à outil bénéficie déjà d'une certaine maturité. Son avenir est très prometteur de par les potentialités offertes, comme nous avons pu le constater dans les exemples présentés.

L'intégration des fonctionnalités Gears avec les technologies RDA dépasse les limites d'Internet. Il s'agit d'une nouvelle révolution pour les applications lourdes, et le framework de Mozilla bénéficie d'un nouvel avantage dans la guerre des technologies RDA. Si vous développez des sites Internet, Google Gears devient incontournable pour ceux qui désirent toujours être de plus en plus disponibles pour leurs utilisateurs. Les applications qui trouvent le plus de potentiel dans la gestion du mode " offline " ne seraient-elles pas dans les machines à haute mobilité ? Quelque chose me dit que le nombre de sites et d'applications supportant Google Gears va vite augmenter dans les prochains mois. Êtes-vous prêts de votre côté à intégrer Google Gears dans les fonctionnalités que vous proposez sur votre site ?

Pour le mot de la fin, je vous poserai la question aventureuse : qu'aimeriez-vous que votre navigateur préféré puisse faire, sortant du cadre strict de l'application client serveur qui affiche des contenus Web ? Voilà le réel enjeu de Google Gears. Alors n'hésitez pas, participez et communiquez vos idées à la communauté.

Sources du projet : http://ressources.edis-consulting.com/articles/programmez/103-gears-files.rar

■ **Maxime ALEXANDRE**
Chef de Projet et Ingénieur d'étude EDIS Consulting (www.edis-consulting.com)
Spécialisé dans les technologies Web

1re partie

OpenLaszlo : une alternative à Flex

Beaucoup d'entre vous sont familiers aujourd'hui avec les Rich Internet Application (RIA) et connaissent même les différentes technologies qui permettent d'en réaliser. Parmi celles-ci, la plus connue est sans doute Flex, fortement plébiscitée par Adobe. Mais les plus technophiles d'entre vous auront aussi certainement eu vent d'une alternative sérieuse : OpenLaszlo.

Cet article en deux parties se propose de faire un tour d'horizon assez exhaustif d'OpenLaszlo, allant des concepts à la réalisation d'une petite, mais conviviale application : un front-end Google Reader.

Architecture et principes

La RIA permet de fournir aux utilisateurs finaux une interface aussi fonctionnellement riche et agréable à utiliser qu'un client lourd tout en gardant les modes de déploiement issus du web à savoir : pas besoin d'installer d'application sur le poste client, tout est téléchargé depuis le serveur. Ce mode de déploiement reste très prôné pour la simple raison qu'il permet des mises à jour continues et transparentes des programmes, mais aussi évite d'onéreuses installations poste à poste. Car dans le cas de clients lourds, il fallait graver les applications sur CD ou DVD, les envoyer par la poste mais aussi ... supporter les utilisateurs pour qu'ils réalisent les installations correctement, ce qui est très loin d'être gagné !

OpenLaszlo repose sur les mêmes principes que la plupart des technologies côté serveur, par exemple JSP : il repose sur un langage à base de tags XML, appelé LZX, et les fichiers de code LZX, placés sur le serveur, sont compilés lors de la première requête effectuée par un utilisateur. Cependant, ce qui est renvoyé par le serveur au client n'est pas comme pour les JSPs du HTML mais du SWF, le format de streaming inventé par Macromedia, plus connu sous le nom Flash. OpenLaszlo ne s'arrête pas là et permet également la génération d'un client DHTML/AJAX afin d'éviter les éventuelles levées de boucliers argumentant que le plug-in Flash n'est pas forcément déployé sur tous les postes clients. Par la suite, l'utilisateur interagira avec l'interface téléchargée dans le navigateur qui traitera la plupart des interactions liées uniquement à l'interface (comme déplacer ou redimensionner une fenêtre) et relaiera vers l'extérieur les demandes de rapatriement de données externes.

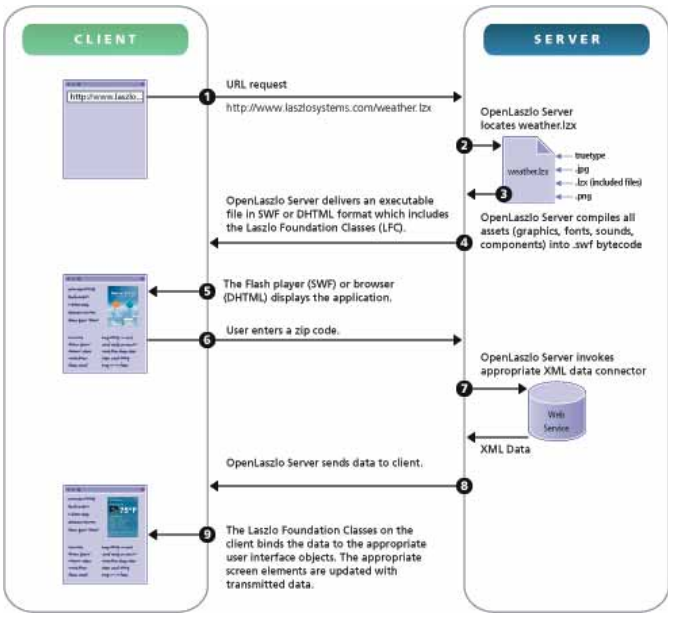
Avantages

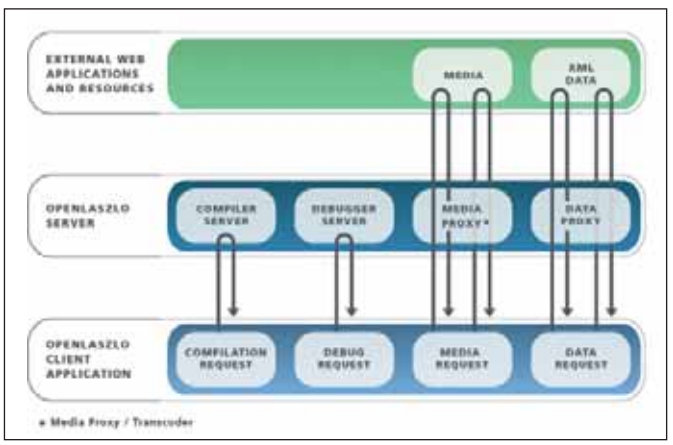
En soi, les principes exposés précédemment n'ont rien d'extraordinaire (si ce n'est bien sûr la génération de swf à la volée) et la force d'OpenLaszlo ne réside pas là. Voici une sélection des avantages liés à la technologie :

Les modes de déploiement

OpenLaszlo se distingue en permettant deux modes de déploiement différents : le premier, appelé 'proxied', installe le résultat de la compilation du LZX sur le serveur pour téléchargement par les clients. Par la suite, les requêtes émises par le client passeront par le serveur avant d'aller vers les URLs finales afin d'aller chercher soit des médias soit des données.

Le diagramme suivant illustre ce mode de fonctionnement :





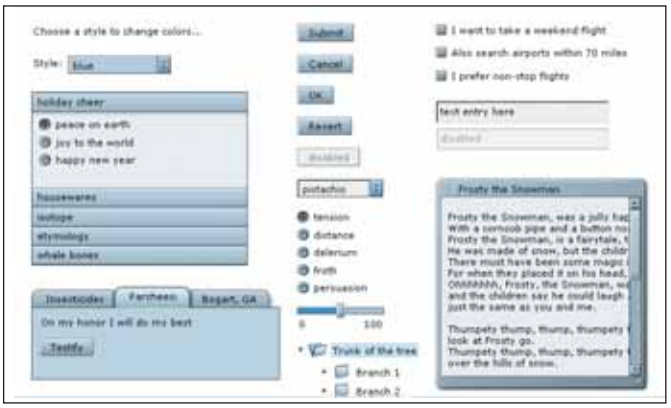
Il est intéressant de noter dans ce mode de déploiement que toutes les ressources de type média sont transcodées par le serveur OpenLaszlo afin de les faire passer dans le format final et notamment swf.

Le second mode de déploiement possible est appelé 'SOLO' et permet de générer une application "standalone" qui sera téléchargée une fois pour toute sur le client. Par la suite, les requêtes externes émises par l'application ne passeront pas par votre serveur mais iront directement attaquer les serveurs cibles.

La différence majeure entre les deux modes de déploiement est que le mode 'proxied' offre plus de possibilités mais peut entraîner des performances moindres au contraire du mode SOLO, plus contraint mais plus performant.

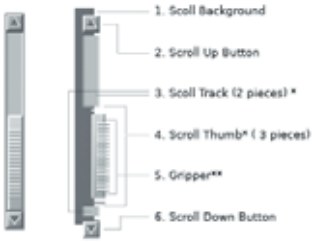
Les composants graphiques

L'avantage clair de la technologie est de proposer un jeu de composants graphiques permettant de construire de façon déclarative des interfaces utilisateurs (presque) identiques à une application type 'client lourd'. Ces composants graphiques (fenêtres, datagrid, calendriers, ...) illustrés ci-dessous reposent sur un ensemble de 'Foundation Classes' fournissant les services de base en exposant certaines fonctionnalités du kernel.



OpenLaszlo offre une extensibilité des composants à plusieurs niveaux. D'abord il est possible bien sûr d'utiliser les composants fournis tels quels mais aussi en modifiant certains attributs comme le type de police ou encore la teinte pour en modifier le style et l'adapter à la charte graphique de l'entreprise.

Mais il est aussi possible d'étendre les composants à tout niveau. En effet, les composants d'OpenLaszlo reposent sur le concept d'objets et respectent à ce titre la notion d'héritage. Ainsi, il est tout à fait possible d'étendre par exemple un bouton simple pour adjoindre à côté du label une icône.



Un cran au-dessus, les composants peuvent être "skinnés" pour en changer le look-and-feel. Ainsi, tout comme nous l'aurions fait pour WinAmp, nous pouvons assigner de nouvelles ressources graphiques au composant afin d'en changer l'apparence. Les ressources doivent cependant être de type vectorielles et non de type bitmap afin de permettre une mise à l'échelle sans effet d'escalier. Le seul format vectoriel supporté par OpenLaszlo est le swf.

Enfin, la dernière forme d'extensibilité est de créer son composant en partant de rien si ce n'est les classes du kernel.

Le Data Binding

Il s'agit là d'un grand classique des interfaces utilisateurs : comment synchroniser les données du modèle avec la couche de présentation ? OpenLaszlo apporte une solution véritablement simple en proposant de lier automatiquement et de façon transparente les données aux composants graphiques. OpenLaszlo repose sur un format de représentation XML des données. A chaque composant graphique il est possible d'associer un datapointer qui n'est autre qu'un chemin xpath sur une source de données XML. Nous verrons rapidement l'attrait de ceci lorsque nous commencerons la réalisation de notre application. Il faut savoir en complément de ceci que les données XML peuvent être récupérées de plusieurs façons : soit elles sont statiques et embarquées dans l'application, soit elles sont dynamiques et peuvent être récupérées en XML RPC, SOAP RPC ou encore Java RPC, ce qui offre une bonne souplesse.

Les contraintes

OpenLaszlo offre la possibilité de pouvoir établir la valeur de n'importe quel attribut à partir d'un autre attribut, contraignant ainsi le premier avec le second. Ceci facilite grandement la réalisation de l'application en permettant par exemple aux éléments de se placer de façon interdépendante et ce, sans avoir à placer d'écouteurs d'événements un peu partout :

```
x="${(parent.width / 2) - (width / 2)}"
```

Mais les possibilités offertes peuvent être bien plus grandes et les expressions de contraintes peuvent être évaluées à divers moment en précisant juste après, le signe $ once, always ou immediately. Par défaut, la contrainte est de type always signifiant qu'elle est en permanence réévaluée. En étant de type once, elle n'est évaluée qu'une seule fois, et pour immediately, elle n'est évaluée que si la contrainte ne dépend pas d'autres objets.

Les animations

Enfin, OpenLaszlo permet d'animer chaque élément de l'interface de manière à la rendre plus attractive et ergonomique. Comme pour le reste de l'interface ces animations sont décrites de façon déclarative. Chaque attribut de classe peut être animé permettant ainsi de déplacer une fenêtre, diminuer son opacité, changer sa teinte, etc. Pour éviter un travail fastidieux et répétitif si plusieurs éléments doivent être animés de la même façon, il est possible de les regrouper et d'assigner l'animation au groupe.

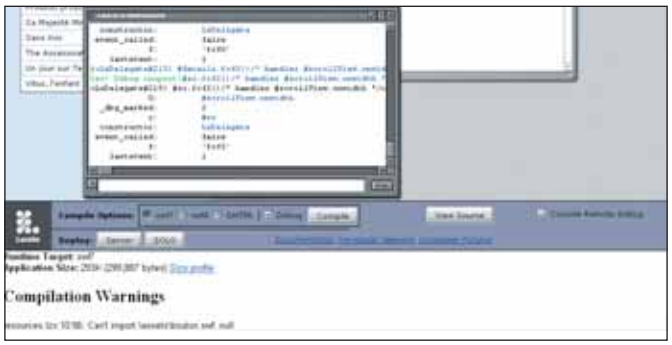
Les outils

Côté outil, OpenLaszlo offre une certaine palette mais il faut bien avouer que celle-ci reste relativement restreinte, même si elle rend quelques services. Il y a quelques temps de cela, IBM avait poussé pour la création d'un outil RAD pour OpenLaszlo, prenant la forme d'une série de plug-in Eclipse et dénommé IDE4Laszlo. Malheureusement, le peu d'activité et de soutien de la communauté ont entraîné la fermeture du projet qui est aujourd'hui quasiment introuvable au téléchargement. Il vous faudra donc vous munir d'outils plus basiques, à savoir un simple éditeur de texte. Si toutefois, cela devait être trop raide pour certains, OpenLaszlo fournit des fichiers de configuration vim et emacs permettant la coloration syntaxique. Mais pour les utilisateurs d'Eclipse il est possible d'utiliser spket (www.spket.com) qui fournit un éditeur de code LZX offrant la coloration syntaxique mais aussi la complétion de code, ce qui devient tout de même plus confortable. Il faut bien avouer que ceci est loin d'arriver à la cheville des outils que proposait IDE4Laszlo.

Par ailleurs, OpenLaszlo fournit un outil de debug et déploiement des applications, automatiquement appelé tant que votre application n'est pas déployée.

Les fonctionnalités offertes par cet outil sont :

- compilation et présentation des résultats de compilation (warnings et autres erreurs)
- activation/désactivation du débugger



- affichage des sources
- déploiement de l'application
- profilage de la taille de l'application

Le débugger fourni par OpenLaszlo est assez simple et permet d'afficher les traces de debug écrites dans l'application ou d'évaluer de nouvelles expressions à la volée. Si le contenu de la trace correspond à un objet de la machine virtuelle, celui-ci est présenté sous forme d'un lien permettant de naviguer dans les propriétés de l'objet et le reste du graphe d'objet. Avec beaucoup de patience, cela permet de résoudre certains soucis de l'application mais l'outil manque très clairement de convivialité.

Conclusion de cette première partie

OpenLaszlo est donc un framework de présentation web permettant un développement d'interfaces aussi riches, si ce n'est plus, qu'un client lourd. La programmation en XML est déclarative et s'accompagne de code Javascript pour décrire les comportements de l'interface. La richesse des composants et leur extensibilité, le binding des données et la possibilité d'animer les éléments de l'interface font de cette plateforme un choix intéressant pour rapidement monter des applications évoluées, même si l'outillage reste un frein potentiellement sérieux à l'adoption. Dans la prochaine partie de cet article, nous allons voir comment mettre en oeuvre les concepts évoqués ici. Mais afin de préparer le travail et d'avoir vu au moins tourner un petit quelque chose, installons OpenLaszlo et réalisons un petit programme HelloWorld !

Téléchargez OpenLaszlo 4.0.5 à l'adresse http://www.openlaszlo.org/download et installez-le. Pour Windows et MacOS un installeur est fourni. Pour Linux, il s'agit tout simplement de dézipper l'archive. Lancez le serveur en exécutant "OpenLaszlo Server 4.0.5\Server\lps-4.0.5\lps\utils\startTomcat.bat". Vérifiez que tout est OK en naviguant à l'URL : http://localhost:8080/lps-4.0.5. Profitez-en pour admirer les différentes démos et autres présentations de composants.

A présent, ouvrez Eclipse et installez spket en utilisant leur update site : http://www.spket.com/update/. Une fois Eclipse redémarré, créez un nouveau projet et créez-y un nouveau dossier en sélectionnant l'option 'avancé' afin de lier ce dossier à un endroit de votre système de fichiers. Liez le dossier au chemin "OpenLaszlo Server 4.0.3\Server\lps-4.0.3". Naviguez dans my-apps et créez-y un nouveau fichier nommé helloWorld.lzx. Ouvrez-le avec l'éditeur LZX de spket et entrez le code suivant :

```
<canvas width="300" height="200" bgcolor="white">
  <statictext>Hello, World!</statictext>
</canvas>
```

Sauvez et naviguez jusqu'à l'URL http://localhost:8080/lps-4.0.3/my-apps/helloWorld.lzx et admirez le résultat :)

Vous voilà maintenant fin prêt pour démarrer les choses sérieuses. Rendez-vous le mois prochain pour la suite.

Fabrice Dewasmes
Java & Open Source dept Manager
PragmaConsult SA
Fabrice.dewasmes@pragmaconsult.lu
Antonino Ricotta
Architecte Technique
Unilog

LANGAGE : OBJECTIVE C
DIFFICULTÉ : ●●●○○
APPLICATION : LEOPARD
SOURCE : non

Introduction à Core Animation

Parmi les nombreuses nouveautés de la version 10.5 Leopard de Mac OS X, Core Animation est certainement la technologie qui propose l'évolution la plus sensible pour l'utilisateur. Core Animation repose sur deux notions essentielles : le layer et l'animation.
Cet article commence par décrire ces deux notions puis illustre leur mise en œuvre au travers d'un exemple.

Figure 1 : Utilisation des layers dans la vue Cover Flow du Finder

Figure 2 : Animation de la position d'une vue

L'affichage de Mac OS X repose sur un puissant moteur de composition graphique qui se traduit par la superposition de couches organisées de manière arborescente :
- un layer peut contenir une liste de layers ;
- chaque layer est positionné dans le layer qui le contient selon un système de coordonnées (x, y, z) ;
- chaque layer peut, à son tour, contenir une liste de layers.

Cocoa propose la classe CALayer pour accéder à ses différentes propriétés et manipuler son contenu.
En plus de définir un espace de coordonnées, un layer définit aussi un espace temporel local permettant de jouer des animations. De manière similaire à la composition graphique de l'affichage, l'animation d'un layer se compose avec celle du layer contenant pour former une animation plus complète.
Les propriétés d'un CALayer peuvent être classées selon trois domaines principaux :
- les propriétés de géométrie ;
- les propriétés de contenu ;
- les attributs d'affichage.

Dans cet article, nous n'utilisons que la propriété de géométrie, nommée transform, qui est une matrice de transformation des coordonnées du layer. (Fig.1)

Les animations

Les animations sont des enchaînements graphiques automatiques liés aux modifications d'un layer. Une vue peut également bénéficier de ces animations si elle est associée à un layer. Mac OS X propose deux grandes catégories d'animations :
- celles liées aux modifications de propriétés : CABasicAnimation ;
- celles liées à l'apparition ou au masquage d'un layer : CATransition.

Une animation de type CABasicAnimation fonctionne avec une interpolation simple entre la valeur initiale et la valeur finale de la propriété modifiée. Ainsi, si l'on modifie la propriété position d'un layer, Core Animation crée, par défaut, une série régulière d'images représentant le layer se déplaçant linéairement entre le point défini par la valeur de position initiale et le point défini par la valeur de position finale.
L'animation de type CABasicAnimation dérive de la classe CAPropertyAnimation pour qui la variation de la valeur de la propriété n'est pas obligatoirement linéaire. En particulier, la classe sœur CAKeyframeAnimation est aussi une CAPropertyAnimation. Dans ce cas, la variation de la valeur est décrite par une série de valeurs clefs. L'interpolation est alors linéaire entre ces valeurs clefs. (Fig.2)

Animation d'une transformation géométrique

L'animation proposée ici comporte deux étapes successives : la réduction à 75% d'un layer, nommé frontLayer, puis son retournement autour de l'axe des y. Chaque étape correspond à la notion de key frame (instant clef) de Core Animation. L'animation utilisée est donc de type CAKeyFrameAnimation. Nous la nommons frontAnimation et elle anime la propriété "transform" du layer :

```
CAKeyframeAnimation *frontAnimation = [CAKeyframeAnimation
    animationWithKeyPath:@"transform"];
```

Une animation de ce type comporte deux éléments principaux : la liste des instants clefs correspondant à chaque key frame et la liste des valeurs prises par la propriété à chacun de ces instants clefs.
L'échelle temporelle d'une animation est définie par la valeur 1.0 pour la durée totale de l'animation. Si l'on souhaite découper l'animation en deux étapes de durées égales, la key frame intermédiaire sera donc définie pour l'instant 0.5. La liste keyTimes comprend ainsi trois valeurs : 0.0, 0.5 et 1.0 :

```
NSArray *keyTimes = [NSArray arrayWithObjects:
        [NSNumber numberWithFloat:0.0],
        [NSNumber numberWithFloat:0.5],
        [NSNumber numberWithFloat:1.0],
        nil];
frontAnimation.keyTimes = keyTimes;
```

La liste des valeurs associées sont des transformations que Core Animation utilisera successivement pour la propriété transform du layer :
- Pour l'instant initial, 0.0, la transformation initiale est perspectiveTransform. Cette transformation est définie de manière courante par Apple en utilisant la transformation neutre CATransform3DIdentity et un élément m34 de valeur -1.0/850.0. Cet élément indique le rapport à utiliser pour projeter l'axe z sur l'axe y et réaliser ainsi un effet de perspective lorsque z n'est pas nul :

```
perspectiveTransform = CATransform3DIdentity;
perspectiveTransform.m34 = -1.0/850.0;
```

- Pour l'instant intermédiaire, 0.5, la transformation frontOutTransform est définie pour réduire à 75 % l'échelle des coordonnées x et y et maintenir à 100 % la coordonnée z. Elle est composée à partir de perspectiveTransform pour conserver l'effet de perspective lors de l'étape suivante :

```
frontOutTransform = CATransform3DConcat(
    perspectiveTransform,
    CATransform3DMakeScale(.75,.75,1.0));
```

- Pour l'instant final, 1.0, la transformation frontOffTransform utilise la composition de la transformation précédente avec une rotation de pi radians autour d'un vecteur colinéaire avec l'axe des y :

```
frontOffTransform = CATransform3DConcat(
    frontOutTransform,
    CATransform3DMakeRotation(pi,0,1,0));
```

Ces transformations sont définies en tant que variables globales qui servent à construire la liste des valeurs de l'animation :

```
frontAnimation.values = [NSArray arrayWithObjects:
        [NSValue valueWithCATransform3D:perspectiveTransform],
        [NSValue valueWithCATransform3D:frontOutTransform],
        [NSValue valueWithCATransform3D:frontOffTransform],
        nil];
```

Pour terminer, l'animation est ajoutée au layer avec une clef quelconque, ayant ici pour valeur "frontAnimation" :

```
[frontLayer addAnimation:frontAnimation forKey:@"frontAnimation"];
```

Ces instructions sont extraites d'un exemple complet simulant le comportement d'un widget Dashboard. Le layer frontLayer est alors associé à une WebView réalisant l'affichage d'un document HTML. Ce document comporte un élément SVG et un bouton InfoButton décrit par les bibliothèques JavaScript d'Apple. Lorsque la souris survole le coin inférieur droit, une animation fait apparaître ce bouton. Un clic sur ce même bouton provoque l'animation décrite et révèle une vue de réglage.
Le code complet est téléchargeable sur le site : http://osx-prog.com/programmez.

Figure 3 : L'exemple d'animation complet

Étienne Vautherin
Auteur de l'ouvrage " Mac OS X Programmation " *chez Dunod.*

Directeurs, Responsables informatiques, recevez gratuitement le NUMERO 1

*Un nouveau mensuel paraît **début janvier** :*

- *Un magazine centré sur les logiciels en entreprise :*
- *Comment les choisir, les déployer, les exploiter…*
- *Toutes les questions que se posent les décideurs : directeurs informatiques et responsables.*

Car les entreprises, quelle que soit leur taille, reposent en partie sur les applications informatiques.

Cela méritait bien un magazine !

Si vous êtes Responsable ou Directeur Informatique, demandez GRATUITEMENT le NUMERO 1

Répondez en ligne sur www.solutions-logiciels.com

par fax : 01 55 56 70 20 ou en envoyant ce coupon à :
Solutions-Logiciels, service Diffusion, 22 rue rené Boulanger, 75472 PARIS

c'est aussi en ligne sur www.solutions-logiciels.com

☐ ***Oui**, envoyez-moi gratuitement le N°1 – réservé à la France métropolitaine (écrire en lettres capitales)*

NOM ..... Prénom .....

Fonction ☐ Directeur informatique ☐ responsable informatique

Titre ..... Société .....

N° ..... rue .....

Complément .....

Code postal : |_|_|_|_|_| Ville .....

Adresse mail ..... @ .....

Editeur : K-Now sarl au cap de 800€ siège social : 6 rue Bezout, 75014 Paris

Utiliser UDDI intelligemment

2e partie

Dans la première partie de cet article, nous avons expliqué les concepts de UDDI et réalisé l'installation d'une implémentation Open Source de UDDI : JUDDI. Dans cette seconde partie nous allons utiliser une API Java client de UDDI et Spring afin de construire un framework d'appels de services, simple et souple, se basant sur des POJOS.

A la fin de la première partie nous mentionnions l'existence de plusieurs APIs java pour attaquer l'annuaire UDDI. Nous avons fait le choix d'utiliser UDDI4J dont la librairie est disponible à l'URL : http://uddi4j.sourceforge.net.
Le but de UDDI est de permettre au client d'un web service de ne pas avoir à connaître l'emplacement de celui-ci, mais juste une information permettant de le retrouver de façon, si possible, unique.
Notre objectif est de permettre à une personne souhaitant utiliser un service déployé quelque part et publié dans l'annuaire UDDI de pouvoir y accéder en connaissant le minimum de choses sur le service et toute la mécanique web service. Ainsi, nous souhaitons donner la possibilité à n'importe quelle application de pouvoir rapidement tirer parti des fonctionnalités exposées par le service sans avoir à connaître de celui-ci autre chose que son nom et l'interface du service.

Création de l'interrogateur UDDI

Ecrivons donc d'abord une classe qui va nous permettre de faire les interrogations sur l'annuaire. Cette classe va rechercher l'emplacement du web service à partir du nom logique de celui-ci. En programmeurs consciencieux que nous sommes nous écrivons d'abord l'interface :

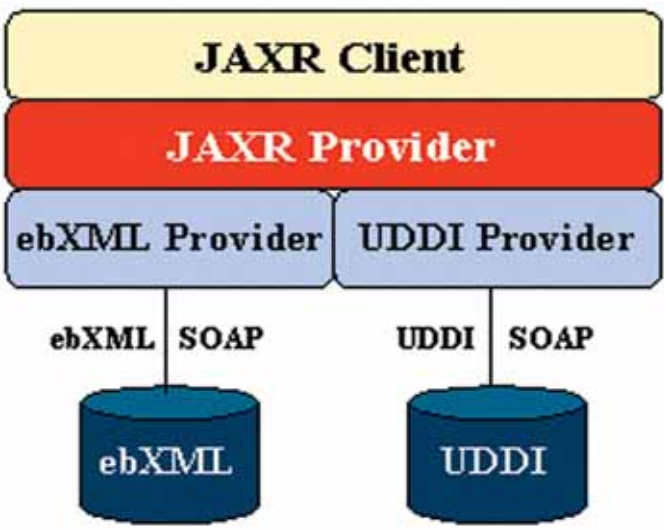
```
public interface IUddiManager {
   public ServiceParameters getWSParametersForServiceName(String
serviceName);
}
```

Maintenant, créons son implémentation. Il est tout d'abord nécessaire de se connecter à l'annuaire, ce qui peut être fait au niveau du constructeur :

```
System.setProperty(org.uddi4j.transport.TransportFactory.PROPERTY_NAME,
                   "org.uddi4j.transport.ApacheSOAPTransport");
proxy = new UDDIProxy();
try {
// Set the inquiry URL
   proxy.setInquiryURL(inquiryURL);
} catch (Exception e) {
   logger.error("JUDDI access problem", e);
   throw new UddiManagerAccessException("JUDDI access problem", e);
}
```

Seule l'inquiryURL est nécessaire ici. A présent nous pouvons implémenter le corps de la méthode getWSParametersForServiceName. Envoyons une première requête pour connaître la clé du service à partir de son nom :

```
// Search for the service
Vector serviceNames = new Vector();
```



Rappel de la 1ère partie

```
serviceNames.add(new Name(serviceName));
ServiceList serviceList = proxy.find_service(null, serviceNames, null, null, null, 0);
if (serviceList.getServiceInfos().size() == 0) {
logger.warn("No Services found for parameters (" + serviceName + ").");
return null;
}

ServiceInfo serviceInfo = serviceList.getServiceInfos().get(0);
String serviceKey = serviceInfo.getServiceKey();
```

Tout ce que nous récupérons est une simple clé. A présent, nous allons récupérer le bindingDetail, c'est-à-dire les informations du bindingTemplate, à partir de la clé du service :

```
BindingDetail bd = proxy.find_binding(null, serviceKey, null, 0);
Vector bdt = bd.getBindingTemplateVector();
if (bdt.size() == 0) {
logger.warn("Configuration service error : the service has no bindings
(" + serviceName + ").");
   return null;
}
BindingTemplate bindingTemplate = (BindingTemplate) bdt.elementAt(0);
```

Enfin, nous récupérons les informations du tModel :

```
tmodelInfos = (TModelInstanceInfo) bindingTemplate
                   .getTModelInstanceDetails().getTModelInstanceInfoVector()
                   .get(0);
```

```
detail = proxy.get_tModelDetail(tmodelInfos.getTModelKey());
detailTmodels = detail.getTModelVector();
aTmodel = (TModel) detailTmodels.get(0);
```

Il est intéressant de remarquer que pour récupérer les informations concernant le service, pas moins de 3 appels sur le serveur d'annuaire sont nécessaires (appels mis en gras dans les sections de code précédentes). Ceci n'est pas des plus efficace, et l'API UDDI est clairement perfectible à ce niveau. Le plus dur est fait car à présent il suffit juste aux projets désireux d'utiliser un service de se servir de Spring afin d'avoir un proxy sur le web service. Pour démontrer cela, il est nécessaire de créer un petit service web type HelloWorld.

Création d'un service web

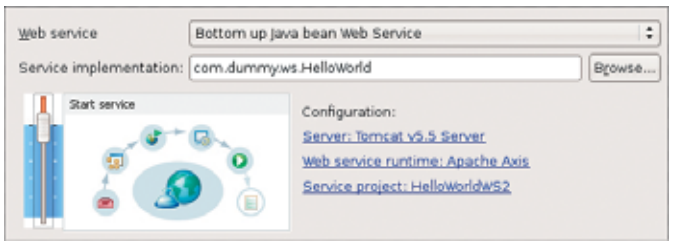
Le plus simple est d'utiliser encore une fois WTP, de créer un nouveau projet web dynamique HelloWorldWS. Créez ensuite une classe très simple implémentant cette interface de service :

```
package com.dummy.ws;
public interface IHelloWorld
{
   public String hello( String value );
}
```

Il vous faudra également créer une interface quasiment identique à celle-ci mais qui servira de endpoint pour le service :

```
package com.dummy.ws;
public interface IHelloWorldEndpoint extends java.rmi.Remote
{
   public String hello( String value ) throws java.rmi.RemoteException;
}
```

La seule différence avec l'interface précédente est que IHelloWorldEndpoint prend en compte l'aspect remoting en étendant Remote. Créez ensuite à l'aide des wizards web services de WTP un nouveau web service en bottom up en sélectionnant comme classe d'implémentation votre implémentation de l'interface de service IHelloWorld.



Si votre serveur tomcat n'est pas déclaré sur Eclipse il vous faudra sans doute le faire, tout comme choisir le runtime web service. Pour ce qui nous concerne, nous avons choisi de réutiliser notre installation de tomcat. Attention toutefois à éviter les conflits de ports : pour cela, cliquez droit sur la définition du serveur dans la vue 'servers' de la perspective 'JavaEE' et choisissez 'open'. Changez les ports http et ajp comme ci-contre :

Seul un port va continuer d'entrer en conflit avec le serveur tomcat qui fait tourner jusqu'à présent votre annuaire, et là aucun autre moyen que de changer le port d'écoute directement en modifiant le fichier conf/server.xml de votre installation de tomcat comme suit :

```
<?xml version="1.0" encoding="UTF-8"?>
<Server port="8205">
```

Vous pouvez à présent relancer le serveur tomcat contenant votre annuaire UDDI et lancer le serveur tomcat déclaré sur WTP, qui doit maintenant contenir votre service web HelloWorld si le déroulement du wizard s'est terminé correctement.
Publiez ce service dans l'annuaire UDDI comme nous avons pu le faire précédemment. L'URL du service web HelloWorld doit être : http://localhost:8280/HelloWorldWS/services/HelloWorld?wsdl.

Terminons notre mise en œuvre en assemblant ces morceaux. Pour cela, créez un nouveau projet et créez une nouvelle classe Client :

```
package com.dummy.uddi;

import com.dummy.ws.IHelloWorld;

public class Client {
   private IHelloWorld service;

   public IHelloWorld getService() {
      return service;
   }

   public void setService(IHelloWorld service) {
      this.service = service;
   }

   public void invokeService() {
      System.out.println(service.hello("Fabrice"));
   }
}
```

La clé de voûte de l'ensemble se résume ensuite à un simple fichier de configuration Spring qui va spécifier d'une part les paramètres de l'annuaire UDDI et d'autre part les éléments fondamentaux du service à attaquer :

```
   <bean id="uddiManager"
      class="com.dummy.uddi.impl.UddiManagerImpl">
      <constructor-arg type="java.lang.String">
         <value>http://localhost:8080/juddi/inquiry</value>
      </constructor-arg>
   </bean>

   <bean id="HelloWorldWS"
      class="com.dummy.spring.JaxRpcPortProxyFactoryBean">
      <property name="serviceFactoryClass">
         <value>org.apache.axis.client.ServiceFactory</value>
      </property>
```

```
        <property name="uddiManager">
            <ref bean="uddiManager"></ref>
        </property>
        <property name="serviceName">
            <value>HelloWorld</value>
        </property>
        <property name="serviceInterface">
            <value>com.dummy.ws.IHelloWorld</value>
        </property>
        <property name="portInterface">
            <value>com.dummy.ws.IHelloWorldEndpoint</value>
        </property>
    </bean>

    <bean id="client" class="com.dummy.uddi.Client">
        <property name="service">
            <ref bean="HelloWorldWS" />
        </property>
    </bean>
```

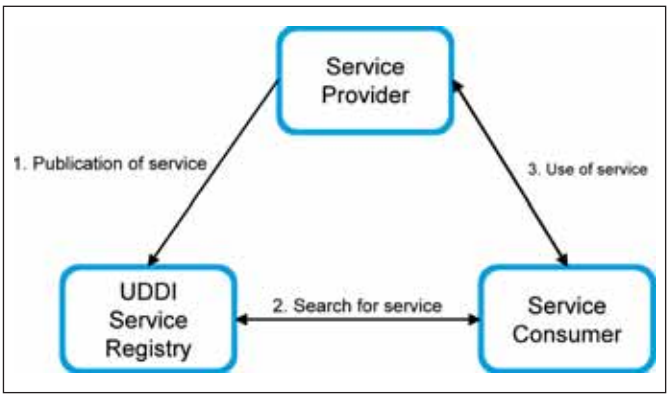
Afin de pouvoir utiliser les facilités de Spring pour atteindre des services web et en particulier le JaxRpcPortProxyFactoryBean, il a été nécessaire d'étendre cette classe afin de pouvoir injecter les valeurs issues de l'interrogation de l'annuaire. En sus de cela, nous parcourons le WSDL exposé par le service afin de récupérer les informations manquantes et éviter ainsi à l'utilisateur du service de s'embarrasser de détails un peu obscurs. Voici le code de notre JaxRpcPortProxyFactoryBean étendant celui de Spring :

```
public void analyzeWsdl() {
    if (logger.isInfoEnabled()) {
        logger.info("Retrieve of the wsdl informations.");
    }
    WSDLAnalyzer wsdl;
    wsdl = new WSDLAnalyzer(getWsdlDocumentUrl().toExternalForm());
    setNamespaceUri(wsdl.getNameSpaceURI());
    setServiceName(wsdl.getServiceName());
    setPortName(wsdl.getPortName());
}

public void prepare() throws ServiceException {
    ServiceParameters params = uddiManager.getWSParametersFor Service
Name(serviceName);
    try{
        setWsdlDocumentUrl(new URL(params.getWsdlUrl()));
    } catch (MalformedURLException e){
        e.printStackTrace();
    }
    analyzeWsdl();
    super.prepare();
}
```

Ajoutons une méthode main sur notre client afin de tester l'ensemble :

```
public static void main(String[] args) {
    try {
        ClassPathXmlApplicationContext context = new ClassPathXml
```




```
ApplicationContext(
                "test-context.xml");
        Client client = (Client) context.getBean("client");
        client.invokeService();
    } catch (Exception e) {
        e.printStackTrace();
    }
}
```

Lancez le client et vérifiez que vous obtenez bien cette sortie :

```
Retrieving document at 'http://localhost:8280/HelloWorldWS2/services
/HelloWorld?wsdl'.
Hello Fabrice
```

Voilà ! Il est important de noter dans cette réalisation d'abord que le client du service est un simple POJO n'ayant aucun lien de près ou de loin avec des web services ou du remoting. Ensuite la configuration de ce lien ne connaît que l'URL de l'annuaire, le nom du service et sa classe d'interface. Tout cela a le mérite de la simplicité pour le client.

Aller plus loin

Vous aurez noté que nous ne faisons ici qu'utiliser un binding de service de type 'web service'. Mais que faire si le service était accessible uniquement en RMI (par exemple via des EJBs) ? Tout ce que nous avons fait est réutilisable et adaptable pour requêter un nouveau type de binding template qui contiendrait cette fois non plus seulement l'URL du WSDL mais le nom JNDI de l'interface home de l'EJB, et les paramètres de connexion au serveur JNDI. En utilisant les proxy factory EJB de Spring, il est alors possible de garder un client type POJO n'ayant aucune conscience qu'il utilise un service distant et même mieux : ne sachant pas si le service est accédé en RMI ou en Web Services. Enfin, il est également possible d'étoffer les informations contenues dans l'annuaire UDDI comme les informations de version afin d'attaquer une version particulière d'un service, si toutefois ce service devait être amené à changer.

L'ensemble du code source de la solution décrite ici est disponible en libre téléchargement sur le site : http://fdewasmes.free.fr

Fabrice Dewasmes
Java & Open Source dept Manager - PragmaConsult SA
Fabrice.dewasmes@pragmaconsult.lu
Antonino Ricotta
Architecte Technique - Unilog

Utilisation des microformats dans vos sites

Après la vague marketing du web 2.0, les futurologues et autres visionnaires voient déjà l'arrivée du web 3.0 ou web sémantique. Basé sur RDF (Resource Description Framework) par exemple, mais pas seulement, le web sémantique est fortement soutenu par le W3C et Tim Berners Lee, l'un des fondateurs du World Wide Web.

Cependant et au-delà de cette notion de web sémantique et des langages associés, les microformats apportent dès aujourd'hui un moyen efficace d'ajouter des informations de sémantique au contenu de vos sites web sans grand effort.

Un microformat, souvent abrégé µF ou uF, est un moyen simple et rapide d'ajouter des informations sémantiques dans votre site, c'est-à-dire d'associer un sens au contenu que vous présentez à l'utilisateur. Les microformats permettent, par exemple, à des éléments d'un contenu HTML ou XHTML d'être identifié comme étant un contact, une localisation ou encore un évènement. Ces éléments pourront ensuite être extraits et interprétés par votre navigateur, des applications, ou être indexé de façon plus pertinente par les moteurs de recherche.

Principe

Techniquement, les microformats utilisent les technologies et principes usuels en termes de web, les éléments à identifier sont simplement marqués par des valeurs d'attribut spécifiques à un microformat donné. Les balises (X)HTML peuvent être enrichies grâce aux attributs class, *rel* et *rev.* Les attributs rel et rev sont peu connus du grand public et s'utilisent dans les balises de liens hypertexte *a* et *link.* Ils jouent des rôles complémentaires en exprimant le rôle et la relation entre plusieurs liens d'un document. L'attribut *rel* décrit la relation entre le document courant et le lien cible alors que l'attribut *rev* est utilisé pour décrire un lien inversé depuis le lien ciblé vers le document courant. L'exemple suivant illustre le principe de l'attribut *rel* :

```
<a rel="prev" href="introduction.html">
<a rel="next" href="chapitre2.html">
```

En revanche, tous les développeurs web connaissent l'attribut class qui permet, à l'instar de l'attribut *id,* d'ajouter des informations d'identifications aux balises. Cela s'illustre facilement avec l'exemple suivant :

```
<span id="msg1" class="info" lang="fr">Exemple d'utilisation de l'attribut
 id et class</span>
```

Les attributs *id* et class peuvent servir à plusieurs fins en s'interprétant comme étant :

- Un sélecteur de feuille de style,
- Une ancre cible pour les liens,
- Une référence pour un élément dans un script,
- Un nom d'un élément OBJECT
- Un identifiant pour traitement par certains agents (pour identifier des champs par exemple)

Ce dernier élément de la liste est important pour nous dans le contexte de cet article. Ne vous semblerait-il pas évident de pouvoir utiliser cette possibilité de l'attribut class pour spécifier des informations que l'on pourrait par la suite réexploiter ? C'est sur ce principe que la majorité des microformats fondent leur approche de la philosophie du " faire plus avec ce qui existe ".

Geo : le microformat de coordonnées géographiques

Pour commencer avec un exemple simple, voici comment il est possible de spécifier des coordonnées géographiques en utilisant le microformat *Geo* pour spécifier les coordonnées de la tour Eiffel (48.858° de latitude et 2.2945° de longitude) sur une carte :

```
<div class="geo">Tour Eiffel : <span class="latitude">48.858</span>,
<span class="longitude">2.2945</span></div>
```

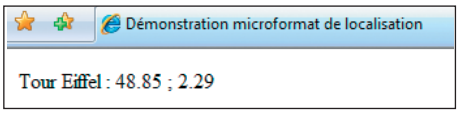


Figure 1

Chaque valeur des balises utilisant l'attribut class spécifie une valeur permettant d'identifier exactement les différentes parties d'une localisation, soit les coordonnées géographiques de latitude et de longitude, exploitables, par exemple, au travers de services de géolocalisation tels que Multimap, OpenStreetMap ou encore Flickr. A noter que vous pouvez annoter vos cartes hCard avec des coordonnées décrites en Geo. Voyons donc maintenant ce qu'est le microformat *h*Card.

hCard : vos contacts en microformat

Les coordonnées de vos contacts ou les vôtres sont souvent publiées sur un site web sous une forme similaire à celle-ci :

```
<p>
  Julien Chable<br/>
  Wygwam<br/>
  +33 (0)3 20 82 38 77<br/>
  <a href="http://www.wygwam.com/">http://www.wygwam.com/</a>
</p>
```

Cette forme simple ne présente a priori aucun problème à l'affichage, cependant il devient vite compliqué d'en extraire avec certitude les informations vous concernant ou celles de vos collaborateurs. L'utilisation du microformat *hCard* permet de pallier ce manque en marquant les différents éléments d'un contact de manière unique :

```
<div class="vcard">
```

```
    <div class="fn">Julien Chable</div>
    <div class="org">The Example Company</div>
    <div class="tel">604-555-1234</div>
    <a class="url" href="http://example.com/">http://example.com/</a>
  </div>
```

Ce contact au format *hCard* s'affiche sans modification dans un navigateur web (cf Figure 2) mais certaines applications ou extensions de navigateur (ici l'extension *Operator* de Firefox – Figure 3) permettent d'extraire les données exposées et de proposer des actions. Nous commençons donc à voir tout l'intérêt des microformats.

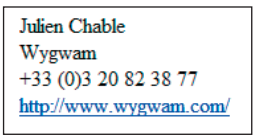


Figure 2

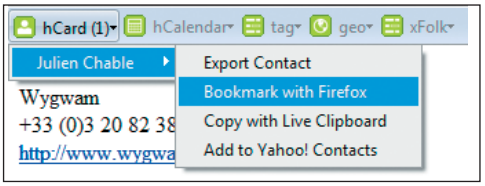


Figure 3

Un contact est constitué par des balises utilisant à sa racine, un élément avec un attribut class '*vCard*' (nous reviendrons sur ce point juste après) puis à l'intérieur des noms de class tel que *fn* (formal name) pour le nom formel, *org* pour le nom de l'organisation, tel pour le téléphone, etc. Un contact peut également être composé des noms de class suivants : *street-address* (adresse de rue), extended-address (information supplémentaire sur l'adresse), *email, locality* (ville), *postal-code* (code postal), etc. Le microformat hCard est une représentation de vCard en proposant des valeurs de ce dernier en sémantique XHTML/HTML.

hCalendar : le microformat des évènements

Tout comme hCard est une représentation en microformat de vCard, hCalendar est un microformat pour représenter des informations iCalendar (RFC 2245) comme des évènements. Voici l'exemple d'une invitation à une soirée de lancement de produit :

```
<p class="vevent">
  Nous sommes heureux de vous convier à la soirée de lancement de <span
 class="summary">notre produit Phare2007</span>
  le 15 Octobre 2007 avec une soirée qui se déroulera entre
   <abbr class="dtstart" title="2007-10-15T19:00:00+1:00">19</ab
br>h et <abbr class="dtend" title="2007-10-15T23:00:00+1:00">23
</abbr>h à
  <span class="location">la salle de la Mutualité (Paris 5ème)</span>
   (<a class="url" href="http://abc.myevents.com">Plus d'informations
</a>)
</p>
```

Cet exemple affiche le rendu de la figure 4 dans un navigateur. La figure 5 montre l'utilisation d'un traitement sur les informations identifiées avec hCalendar en proposant des actions d'ajout de l'évènement à différentes applications de calendrier.

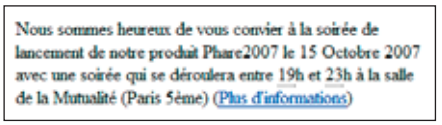


Figure 4

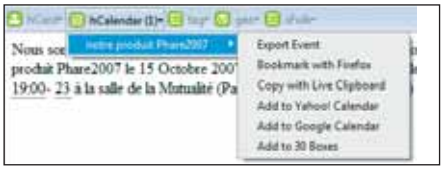


Figure 5

Les noms de class utilisés sont relativement simples à comprendre et caractérisent bien le concept d'évènement. Le nom *'vevent'* spécifie un évènement et possède des informations telles que la date de début (*dtstart*), la date de fin (*dtend*), un résumé (*summary*), un lieu (*location*) et une URL d'information (url). Cela n'a évidemment rien de compliqué et démontre bien la portée et la facilité de l'utilisation des microformats dans des pages web. A noter que les valeurs des attributs *'title'* pour les champs de date doivent respecter le format date ISO 8601.

hAtom : Atom au microformat

Il existe de nombreux microformats qui se basent sur tout ou partie des spécifications et de la sémantique d'autres formats ou standards. Ce principe est à appliquer au célèbre format XML de syndication qu'est Atom. Voici donc un exemple du format Atom en microformat :

```
<body class="hfeed" id="123456">
  <div id="content" class="hentry">
    <h1 class="entry-title"> Atom en microformat
      <span class="byline">Mis à jour
        <abbr class="published" title="2007-07-23T04:26:18Z">Mon,
23 Jul 2007 23:26:18 EST</abbr>
      </span>
    </h1>
    <p class="entry-content">Lorem ipsum</p>
  </div>
</body>
```

Conclusion

En attendant un hypothétique support natif des microformats par la prochaine génération de navigateur Internet, il est important pour le référencement du contenu de vos sites d'adopter dès à présent ces 'conventions' très simples à mettre en place. Cela n'accentuera que peu la complexité de votre code (X)HTML, et aucunement le rendu, pour une charge de travail minime qui en contrepartie pourra faire la différence entre vous et vos voisins.

Références

- Site microformats : http://microformats.org
- Propriétés des noms de class pour hCard : http://microformats.org/wiki/hcard-cheatsheet
- hAtom 0.1 : http://microformats.org/wiki/hatom
- Internet Mail Exchange (vCard et vCalendar) : http://www.imc. org/pdi/

Julien Chable – *Wygwam*

LANGAGE : JAVA
DIFFICULTÉ : ●●●○○
APPLICATION : TOUT
SOURCE : non

Du bon usage des exceptions en Java

Progrès incontestable dans la gestion des erreurs, les exceptions peuvent être elles aussi sources ... d'erreurs. Voici quelques rappels et bonnes pratiques pour aider à écrire un code plus robuste.

Cet article a pour but de mettre en évidence le bien fondé de quelques bonnes pratiques des exceptions en Java, en nous plaçant essentiellement du côté du code client. Nous supposons que le lecteur est déjà familiarisé avec les exceptions. En outre, si nous nous intéressons à Java, les principes directeurs donnés dans cet article sont, sur le fond, valables avec tous les langages à exceptions tels que Python, C++, C#, etc.

1 La vengeance du Goto masqué

L'expérience des programmeurs allant croissant, Les Goto des Fortran et autres Basic sont aujourd'hui considérés avec le plus grand mépris. A juste titre. Pour l'anecdote, il est arrivé à votre serviteur de porter du code Fortran vers C++. Le code Fortran, qui avait été écrit par un scientifique extrêmement compétent en son domaine, mais pas du tout informaticien, était une sorte de spaghetti à la sauce Goto. J'ai frôlé la folie à démêler l'imbroglio. Et qui sait si je ne conserve pas quelques séquelles finalement... ;) Il est clair que le Goto est dangereux comme la peste car il brise le flux d'exécution du code, le faisant sauter avec la plus grande brutalité d'un endroit à un autre. Et c'est ici que se situe la racine de tous les problèmes relatifs aux exceptions. Une exception n'est, fondamentalement, rien d'autre qu'un Goto, d'autant plus dangereux qu'il est masqué et que du coup on oublie qu'il risque de sévir à tout moment. Illustrons ceci par un peu d'horrible code:

```
public class DemoGoto  {
    public DemoGoto() {
        ma_methode();
    }

    private static void ma_methode() {
        int i = 1/0; // boum
    }

    public static void main(String[] args) {
        try {
            DemoGoto dg = new DemoGoto();
            System.out.println("Programmez!");
        }
        catch(Exception e) {
            System.out.println("Bing!");
            DemoGoto.ma_methode();
        }
        finally {
            System.out.println("Bang!");
            DemoGoto.ma_methode();
            System.out.println("Abonnez vous!");
    }
    System.out.println("A bientot");
    }
}
```

Ce code affichera-t-il ?

```
Programmez!
Abonnez vous!
A bientôt
```

Pas du tout. Il affichera :

```
Bing!
Bang!
Exception in thread "main" java.lang.ArithmeticException: / by zero
```

Car ma_methode tente une division par zéro, ce qui lève une exception, ou, si on veut, fait un goto sans le dire. Ainsi le flux d'exécution saute-t-il dans le bloc catch, ou, en termes politiquement corrects, la pile d'appels est remontée jusqu'au gestionnaire d'exception approprié :) Dans ce gestionnaire (le bloc catch) la méthode est appelée de nouveau. Saute-t-on pour autant à un autre éventuel gestionnaire d'exception (qui n'existe pas dans le code) ? Pas du tout. On saute dans le bloc finally qui, nous dit-on, est toujours exécuté à la suite d'un bloc try. Toujours exécuté ? Oui et non. Il est exécuté ici jusqu'à ce qu'on y appelle ma_méthode et à partir de ce point, le flux d'exécution saute à nouveau sans que le reste du bloc finally soit exécuté. Où saute-t-il ? Nulle part dans le code, bien qu'il reste quelque chose à exécuter derrière le bloc finally. L'exception non capturée met fin au programme avec un message d'erreur.

On remarquera que si on supprime DemoGoto.ma_methode(); du bloc catch, le code se comporte strictement de la même façon. Tout se passe comme si l'exception levée dans catch était purement et simplement escamotée par celle levée dans le bloc finally.

En effet, en Java c'est toujours la dernière exception levée qui compte et nous comprenons que ce mécanisme peut nous faire perdre des informations cruciales sur l'origine d'une erreur (remarque: C++ ne permet pas cela et termine immédiatement dans un tel cas).

Nous dégageons déjà un premier principe. Une méthode devrait toujours veiller à ne pas masquer la cause réelle d'un problème par une exception levée à posteriori dans un bloc catch ou finally. Une façon de procéder est de chaîner les exceptions en passant la première au constructeur de la suivante.

Voici un exemple de code possible.

```
package fred;

import java.util.Random;

class ExceptionUn extends Exception {
}

class ExceptionDeux extends Exception {
    ExceptionDeux() {
    }
    ExceptionDeux(Throwable t) {
        super(t);
```

```
    }
}

public class Enveloppe {

    Random random = new Random();

    public void ma_methode() throws ExceptionUn, ExceptionDeux {
        ExceptionUn memo = null;
        try {
            // ExceptionUn risque d'être levée'
            if((random.nextInt()%2) == 0) {
                memo = new ExceptionUn();
                throw memo;
            }

        } finally {
            // le traitement final risque
            // de lever ExceptionDeux
            if((random.nextInt() % 2) == 0 ) {
                if(memo != null)
                    throw new ExceptionDeux(memo);
                else
                    throw new ExceptionDeux();
            }
        }
    }

public static void main(String[] args) {
    Enveloppe ev = new Enveloppe();
    try {
        ev.ma_methode();
    } catch(Exception e) {
        System.out.println("La cause est: " + e.getCause());
        e.printStackTrace();
    }
    }
}
```

Si dans le premier code (celui qui est horrible :)) nous supprimons DemoGoto.ma_methode(); du bloc finally, le programme produit cette fois :

```
Bing!
Bang!
Abonnez vous!
Exception in thread "main" java.lang.ArithmeticException: / by zero
```

C'est-à-dire que le bloc finally s'exécute complètement, mais pas la ligne de code qui le suit. Car cette fois l'exception levée dans catch n'est plus supplantée et entre en action, mais seulement à la sortie du bloc finally. Le mécanisme des exceptions a en quelque sorte inventé le Goto à retardement :)
Tout ceci à de quoi donner mal à la tête, mais il est maintenant très clair que nous devons nous méfier des exceptions comme du feu, et nous déduisons immédiatement une seconde règle.

2 Ne jamais retourner de valeur depuis un bloc try.

C'est une erreur d'écrire une instruction return dans un bloc try.
Exemple :

```
public boolean noTryReturn() {
    try {
        int i = 1/0;
        return true;
    }
    catch(Exception e) {
        return true;
    }
    finally {
        return false;
    }
}
```

Il est absolument certain que notre méthode ne retournera jamais true, car qu'une exception soit levée ou pas lors de l'exécution du bloc try, il est garanti que le flux entrera dans le bloc finally (sauf et en toute rigueur si un appel à System.exit() est effectué avant, mais dans ce cas, parler de la valeur de retour n'a pas de sens non plus :)). Nouvelle facétie du Goto masqué, deux return sur trois n'en sont plus, et dans notre exemple c'est donc la valeur false qui sera retournée systématiquement. Pouvons nous écrire notre fonction en nous passant du bloc finally ? En théorie oui :

```
public boolean noTryReturn() {
    try {
        int i = 1/0;
        return true;
    }
    catch(Exception e) {
        // traitement de l'exception.
        return true;
    }
}
```

A supposer bien entendu que le traitement de l'exception n'en lève pas une lui-même, nous retournons le booléen souhaité. Mais c'est néanmoins une mauvaise solution car elle contient du code dupliqué, porteur d'erreurs et de problèmes de maintenance potentiels. Eliminer le bloc finally n'est pas une bonne idée. Il faut au contraire s'en servir:

```
public boolean noTryReturn() {
    try {
        int i = 1/0;
    }
    catch(Exception e) {
        // traitement de l'exception.
    }
    finally {
        // selon la situation.
        return true; // ou false
    }
}
```

Ce code fonctionne-t-il enfin ? Oui, à la condition qu'aucune exception non contrôlée telle qu'une division par zéro ne soit levée dans le bloc finally. Notons aussi le revers de la médaille: une exception contrôlée ou non, levée dans le bloc catch, passerait dans ce cas à la trappe, justement parce que l'exécution du return est garantie.

3 Toujours capturer l'exception la plus spécialisée.

Voici un conseil très avisé que nos exemples précédents n'appliquent pas toujours :) Le problème de fond se situe à nouveau dans le Goto masqué. Un Goto ne dit pas d'où il vient ni où il ira. Le type des exceptions résout une partie du problème, celui de l'origine. En Java toutes les exceptions dérivent de la classe Throwable. Pourtant on ne trouve jamais ce code, théoriquement possible.

```
try {
    // faire quelque chose
}
catch(Throwable) {
    // réagir
}
```

Car ce code n'informe pas sur l'origine de l'exception et donc pas sur le comportement à adopter dans le gestionnaire. Pour affiner les choses, Java définit une hiérarchie de classe d'exceptions assez compliquée. De Throwable la hiérarchie se sépare en 2 branches, Error et Exception. Les exceptions de types Error ne devraient jamais être capturées. En effet, elles signalent un dysfonctionnement de la JVM qui doit alors s'arrêter impérativement... ce qu'empêcherait le code ci-dessus. Nous aurions alors une JVM déglinguée continuant sa course folle :) La branche exception se divise ensuite en de multiples branches. Java, et contrairement à C# par exemple, est partisan de l'association de la ceinture et des bretelles pour tenir un pantalon. Ainsi le compilateur obligera à écrire un gestionnaire pour toutes ces exceptions dérivant de Exception et qui sont dites contrôlées,... à l'exception de celles qui dérivent de RuntimeException que Java considère comme des erreurs de programmation et sur lesquelles nous reviendrons. Il existe des multitudes d'exceptions contrôlées ce qui permet d'affiner le code. La paresse voudrait que l'on écrive ceci :

```
try {
    // travailler avec un fichier
}
catch(IOException) {
    // réagir
}
```

On trouve d'ailleurs souvent de tels codes dans des livres, par souci de concision et lorsque le sujet exposé n'est pas les exceptions lui-même. Mais en situation réelle la prudence impose de traiter tous les types d'exceptions possibles même si dans notre cas, tous dérivent de IOException:

```
try {
    // travailler avec un fichier
}
catch(FileNotFoundException) {
    // Fichier non trouvé
    // réagir
}
catch(IOException) {
    // Erreur de Lecture/Ecriture
    // réagir
}
```

4 Attention à la gestion des ressources

Les exceptions mal gérées peuvent conduire à l'épuisement de ressources. Voici le cas très classique de la connexion à une base de données :

```
Connection connect =  null;
try
{
    Class.forName("le_driver");
    connect = DriverManager.getConnection( // etc);
    // faire quelque chose.
    connect.close();
}
catch (Exception e)
{
    // écrire dans un log
}
```

Si pour une raison lambda "faire quelque chose", lève une exception, la connexion ne sera pas fermée immédiatement par l'appel à close du bloc try, mais ne sera fermée que lorsque le ramasse-miettes détruira l'objet, c'est-à-dire à un moment imprévisible et, surtout, différé. Si notre application est très sollicitée et que des exceptions sont fréquemment levées, nous atteindrons le nombre maximal de connexions supporté par la base de données et l'application ne fonctionnera plus convenablement. Pour cette raison, toutes les ressources rares (connexions base de données, socket, etc.) doivent être fermées/libérées dans un bloc finally comme ceci.

```
Connection connect = null;
try
{
    Class.forName("le_driver");
    connect = DriverManager.getConnection( // etc. );
    // faire quelque chose.
}
catch (Exception e)
{
    // écrire dans un log
}
finally {
    if(connect != null) {
        try {
            connect.close();
        }
        catch (Exception e){
            // réagir
        }
    }
}
```

5 Veiller au bon état des objets

Pour terminer cet article, examinons un point particulièrement délicat. Les exceptions peuvent laisser un objet dans un état instable ou incohérent. Supposons que nous écrivons une classe gérant un compte en banque. La plus grande rigueur s'impose. Notre classe, collaborant dans une application, utilise d'autres classes et est utilisée par d'autres classes. En aucun cas les données encapsulées par notre classe ne doivent être incohérentes. Voici un premier jet :

```
public class BadAccount {
    private int operations = 0;
    private double montant = 0.0;

    public BadAccount() {
    }

    public void update(double somme) {
        operations = new Incrementator().calcule(operations);
        // le Goto masqué nous menace ici :-(
        montant = new Creditor().calcule(montant, somme);
    }
}
```

Notre classe détient le nombre des opérations effectuées et le montant disponible sur le compte. Ces deux valeurs doivent toujours rester cohérentes. Pour l'exemple, nous supposons que l'incrémentation du nombre d'opérations est délégué à une classe Incrementator, et la mise à jour du montant est déléguée à une classe Creditor. Et bien ce code tout simple ne fonctionne tout simplement pas! Telle que nous présentons la chose les méthodes "calcule" de Incrementator et Creditor ne lèvent aucune exception contrôlée. Mais le problème est qu'elles peuvent toujours (insistons sur le toujours) lever une exception non contrôlée. Par exemple NullPointerException. Selon la philosophie de Java, il s'agirait d'une erreur de programmation. En tant que telle elle ne devrait pas arriver et l'exception ne doit donc pas être contrôlée. Le point de vue est discutable, mais admettons. Cependant, puisque nous instancions des classes, l'exception non contrôlée OutOfMemoryError peut elle aussi être levée. Si elle est levée à l'instanciation de Creditor, alors les valeurs de opérations et de montant ne seront plus cohérentes. Nous avons vu plus haut qu'il était bon de savoir d'où venait notre Goto masqué. La question est maintenant de savoir ce qui va se passer en amont de notre code. Si l'exception n'est jamais capturée la JVM s'arrête. Mais il est plus que probable que notre application capture l'exception en amont de notre code pour sérialiser et sauvegarder tous les comptes en banques.

En ce cas, notre compte sera sauvegardé dans un état incohérent car notre code n'est pas robuste aux exceptions. Pour remédier à cela les méthodes doivent avoir un comportement dit atomique. Autrement dit, une opération doit être faite complètement ou pas du tout. On appelle cela le "valider ou annuler", ou encore par analogie avec les bases de données, le Commit/Rollback. Pendant la phase de validation, on ne doit exécuter que du code dont on est absolument sûr qu'il ne lèvera aucune exception. En Java il s'agit de la manipulation de types primitifs (sauf pour la division par zéro...) et l'affectation. Bref, nous n'avons réellement que l'affectation. Mais c'est suffisant :) En ré-écrivant notre méthode update ains :

```
public void update(double somme) {
    // on travaille avec des variables temporaires
    int operations_ = new Incrementator().calcule(operations);
    double montant_ = new Creditor().calcule(montant, somme);
    // on valide le travail
    operations = operations_;
    montant = montant_;
}
```

nous sommes certains que l'état de notre objet sera toujours cohérent. Certains ? Maintenant que nous avons compris le principe, nous pouvons l'avouer, notre code n'est pas encore très bon. En effet en situation réelle, notre classe sera plus complexe et nous ne pourrions sans doute pas être sûrs d'être à l'abri d'effets de bord se produisant dans le corps d'update. Ou même, si nous devons écrire de nombreuses méthodes comme celle ci-dessus, nous sommes à la merci d'étourderies pouvant nous faire oublier de travailler avec une variable temporaire ici ou là. Le mieux est d'utiliser l'idiome dit de swap, très connu des programmeurs C++. Nous n'utilisons plus de variables temporaires, mais carrément une copie temporaire de notre classe :

```
public class BetterAccount implements Cloneable {
    private int operations = 0;
    private double montant = 0.0;

    private void swap(BetterAccount temp) {
        this.operations = temp.operations;
        this.montant =  temp.montant;
    }

    public void update(double somme) {
        try {
            BetterAccount temp = (BetterAccount)this.clone();
            temp.operations =
                new Incrementator().calcule(operations);
            temp.montant =
                new Creditor().calcule(montant, somme);
            // valider
            swap(temp);
        } catch (CloneNotSupportedException ex) {
            // cette exception ne doit
            // normalement pas se produire
        }
    }
}
```

Ici nous pouvons nous contenter de la méthode clone par défaut. Mais si notre classe contenait des références d'objets alors nous devrions implémenter notre propre méthode clone. Dernière remarque, notez bien que swap est private et non pas public, protected ou sans qualificateur de visibilité (visibilité dans tout le package), afin d'éviter les traquenards inhérents au polymorphisme. Tout n'est pas dit, loin s'en faut, sur ce vaste et difficile sujet. Nous disposons cependant de bonnes pratiques pour éviter de mauvais tours joués par les exceptions.

Frédéric Mazué - *fmazue@programmez.com*

Abonnez-vous

1 ÉCO

Recevez le magazine chaque mois

économisez 20 €

11 Numéros

45 €

Au lieu de 65,45 € (Prix au numéro)

(Prix France métropolitaine)

-40%

2 Étudiant

Vous devez justifier de votre statut d'étudiant

Economisez 26 €

11 Numéros

39 €

Au lieu de 65,45 €

(Prix au numéro)

(offre réservée France métropolitaine)

-30%

3 Numérique

Lisez chaque mois le magazine

Format PDF
(téléchargement)

11 Numéros

30 €

Tarif Monde entier

Inscription en ligne uniquement

2,73€ le numéro

www.programmez.com

+ Abonnement INTÉGRAL

NOUVEAU

ACCÈS ILLIMITÉ aux ARCHIVES du MAGAZINE pour 1€ par mois !

Prix de lancement

Cette option est réservée aux abonnés pour 1 an au magazine, quel que soit le type d'abonnement (Éco, Numérique, Etudiant). Le prix de leur abonnement normal est majoré de 12 € (Prix de lancement, identique pour toutes zones géographiques). Pendant la durée de leur abonnement, ils ont ainsi accès, en supplément, à tous les anciens numéros et articles /dossiers parus.

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 n°) : 39 € TTC. *Offre limitée à la France métropolitaine. Photocopie de la carte d'étudiant obligatoire*

☐ + SUPPLEMENT ABONNEMENT 1 an INTEGRAL au prix de lancement de 12 € TTC. (s'ajoute à une des formules d'abonnement)

MONTANT TOTAL DE L'ABONNEMENT : ☐ €

☐ 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 décembre 2007

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 05/01/78, 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.

PROG 103

ACTUS

StoneTrip : 3D et serveur temps réels

Evolutions intéressantes chez StoneTrip. Sa plate-forme de développement de jeux 3D Temps Réel Shive arrive en version 1.5.1, apportant la correction de quelques problèmes, mais surtout une interface enfin totalement francisée, et la fonction Multipack, qui permet " d'archiver " les applications les plus lourdes en plusieurs packs. Shive existe désormais en 3 versions : démo PLE, gratuite, Indie, à 199 $, ou Pro, à 1800 $. Seule la démo est limitée, la différence entre Indie et Pro se faisant juste sur les droits d'exploitation des jeux développés.

Les créateurs de jeux multijoueurs seront aussi contents d'apprendre que le Server Stone 3D existe désormais en PLE téléchargeable gratuitement : une bonne solution pour pouvoir prototyper en conditions réelles applications et jeux réseau, les limitations de performances n'étant guère gênantes dans cette optique, et celles de temps offrant une période de test acceptable. A découvrir sur www.stonetrip.com.

Nvidia : carte intermédiaire de transition

La nouvelle carte graphique de Nvidia, la GeForce 8800 GT, inaugure un nouveau type de carte doublement intermédiaire, à la fois sur le prix et sur la gestion des performances, versions successives de DirectX en tête. Aux alentours de 200 €, le produit s'adresse à des joueurs exigeants, mais pas fanatiques, avec une infrastructure pourtant très haut de gamme : 112 stream processeurs cadencés à 1,5 GHz, associés à une interface mémoire 256 bits, et une nouvelle technologie de lecture HD, c'est quand même du très lourd pour un prix franchement moyen ! Mais la carte fait aussi le pont entre deux générations : elle est d'un côté conçue pour exploiter DirectX 10, et ses nouvelles stars ludiques très gourmandes comme Hellgate : London, Gears of War, ou le très attendu Crisis. Et dans le même esprit, elle exploite le nouveau standard de bus PCI Express 2.0. Mais d'un autre côté, sa rétrocompatibilité est totale, aussi bien avec DirectX 9 qu'avec le standard " classique " PCI Express habituel. Bref, il s'agit d'une carte de transition, ouverte sur l'avenir immédiat (jeux et HD, avec une technologie spéciale de déco-

dage, imputant tout le travail à la carte pour décharger le CPU), mais aussi parfaitement adaptée aux " vieux " titres de 2006-2007. Un kit de développement spécifique (en fait, quelques aménagement seulement) devrait être disponible avant la fin de l'année.

Un monument pour les prototypes

Monumental, société anglaise spécialisée dans les middlewares ludiques, propose désormais son logiciel de création/gestion de jeu online multijoueur en utilisation gratuite, du moment qu'il s'agit d'un développement de prototype. La MTS (Monumental Technology Suite) est composée d'un ensemble de modules système, technologies serveur et client, moteur graphique, outils de développement et gestion du projet. Sa philosophie est clairement de suivre toutes les étapes d'un jeu multijoueur online, du codage initial à la gestion réseau en passant par le graphisme. La

critique qu'on peut évidemment faire à ce genre de package global est que chaque module peut sembler sommaire par rapport à son équivalent en middleware autonome. Le problème n'est cependant pas trop sensible avec MTS – seul le module graphique paraît assez limité, mais le moteur en lui –même est très suffisant pour un prototype. www.monumental.com

Wii vs PS3 : l'écart diminue

Les dernières statistiques du marché japonais montrent que la PS3 de Sony commence à réduire son retard sur la Wii de Nintendo. Ainsi, depuis le début de l'année, il se vendait 4 Wii contre 1 PS3, avec même un record de 8 contre 1 en novembre 2006 ! Mais depuis début octobre 2007, le rapport est tombé à 2 contre 1… La montée de l'offre logicielle avec l'annonce de multiples développements majeurs n'est, bien sûr, pas étrangère au redressement de la console de Sony. Simultanément, les possibilités graphiques de la Wii et son interface de commande par mouvements, très particulières, commencent à montrer leurs limites dans les types de jeux les plus vendeurs, tir, action, aventure. Enfin, les studios semblent commencer à maîtriser l'infrastructure complexe de la PS3 : la liste des développements en cours montre que même les studios " de deuxième catégorie " osent maintenant se lancer dans l'aventure.

Le phénomène ne concerne pour l'instant que le Japon, mais une autre étude dévoile un début de tendance similaire aux USA. D'ici que le marché et les développements des deux consoles suivent la même voie en Europe, il n'y a qu'un pas. A vérifier en janvier-février 2008.

A noter aussi que la PS2, malgré l'arrivée des consoles next-gen, a gardé un niveau de vente

remarquable tout au long de 2007. Si bien qu'une rumeur court maintenant avec insistance selon laquelle une nouvelle PS2 devrait voir le jour dans les mois qui viennent, dite " la Compacte " : un design relooké, quelques actualisations hardware, et en outre un prix de vente qui devrait tomber sous les 100 $ - soit 25% de moins… Sony n'a pas confirmé, mais n'a absolument pas démenti. Ce qui vaut généralement pour approbation.

Petits jeux sur le net : le gros appétit de Diner Dash

Au cas où il faudrait encore le démontrer, on peut faire de gros succès et beaucoup d'argent avec des petits jeux quasiment indies ET téléchargeables en ligne. C'est ainsi qu'un des grands spécialistes du genre, PlayFirst, vient d'annoncer le 200 millionnième téléchargement de son hit Diner Dash et un chiffre d'affaires en ligne sur ce seul jeu de plus de 35 millions de dollars ! Rappelons que Diner Dash est un jeu d'action et de réflexe où le joueur incarne une serveuse de restaurant qui doit agencer, servir, encaisser, et débarrasser toutes les tables dans les délais les plus serrés… stress garanti ! Ni 3D, ni shading, ni animatiques HD ou autres sophistications : Diner Dash, quoique techniquement performant et graphiquement très plaisant, est bel et bien un petit jeu, à ranger dans la catégorie du développement léger côté production et du casual gaming côté consommateur. Mais c'est la créativité et l'invention qui ont tout fait, avec d'abord un excellent principe ludique. Puis surtout avec le lancement d'une extension (45% de nouveaux clients), de niveaux supplémentaires (24% des achats en ligne) et d'un module de customisation (12%). On voit donc ici un modèle idéal de développement progressif par petites étapes à rendement immédiat ! En même temps, le concept marketing du site PlayFirst prouve pleinement son efficacité : " play first ", parce que chacun des jeux est proposé en téléchargement gratuit utilisable pendant près d'une heure, ou en applet jouable directement sur le site. Bref, le joueur teste pleinement avant d'acheter… et achète ! (www.playfirst.com)

Un bel exemple à suivre, qui devrait convaincre ceux qui pensent que World of Warcraft ou Halo 3 sont la seule voie ludique de succès et de rentabilité. Et redonner courage à tous les autres.

Fédération française de développeurs indies en ligne

Synapse, la fédération des créateurs indépendants de jeux en ligne alternatifs, officiellement née lors du festival du Jeu Vidéo de fin septembre 2007, est désormais pleinement opérationnelle. Le credo est simple : promouvoir ces jeux et leurs auteurs, mutualiser les ressources et parfois même les (petites) équipes, enfin défendre les droits de ce secteur peut-être réduit, mais très actif, et d'autant plus exposé qu'il est éparpillé en initiatives individuelles distinctes. Plusieur jeux et développements sont ainsi déjà mis en avant sur le site fédérateur (www.federation-synapse.fr), dont l'étonnant Active Fighting Club, ou le remarquable Archipels Online : preuves, sans discussion, du dynamisme et du talent de ces créateurs en France.

Maxon refait son Cinema 4D

Pas de grand jeu sans grandes cinématiques, vous connaissez la chanson ! Mais n'en abusez pas… La toute nouvelle version de Cinema 4D (la 10.5) pourrait cependant vous y inciter. Elle est franchement accessible à la plupart

des utilisateurs, même si talent graphique et sens de l'animation restent évidemment déterminants. Pour les amateurs de confort, une interface simplifiée permet de se borner aux fonctions essentielles, avec beaucoup de commandes graphiquement intuitives qui nécessitent peu d'apprentissage. En outre, ils disposent d'une belle bibliothèque d'animations et de déformations prêtes à l'emploi. Les développeurs plus spécialistes du secteur apprécieront le nouveau filtre d'import DWG, qui couvre les formats d'échange CAD les plus courants, et surtout l'intégralité d'AutoCAD, toutes versions confondues. Autre innovation très prometteuse, les Xrefs : ces références externes permettent de découper les scènes de grande taille en éléments distincts, traitables séparément. C'est précieux pour maintenir un flux parallèle du développement graphique, le même fichier pouvant être utilisé et modifié simultanément, avec contrôle de synchronisation des tâches évidemment.

On appréciera enfin la cohabitation de plusieurs modes de gestion des lumières, dont un module Eclairage qui simplifie grandement les choses pour une illumination globale d'une scène, d'un groupe de scènes, ou d'un sous-ensemble de lieux. Très complet et accessible.

Introduction à WCF

- ***Difficulté : *******
- ***Editeur : O'Reilly***
- ***Auteur : Michele Leroux Bustamante***
- ***Prix : 45 €***

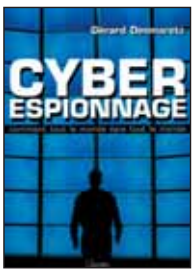
Les premières références pour les nouvelles briques .Net 3.0 arrivent enfin en français. O'Reilly propose avec " introduction à WCF " un ouvrage phare pour .Net 3.0 et 3.5. Avant tout technique, l'ouvrage s'adresse aux développeurs .Net et aux architectes, chefs de projet travaillant sur la SOA. L'auteur a misé sur une centaine d'exemples parsemant chaque chapitre. L'ouvrage se segmente simplement : contrats, liaisons, hébergement, instanciation et concurrence, fiabilité, sécurité et erreurs et exceptions. On débute par un rappel sur l'origine de WCF avec le projet Indigo et sur la définition de la SOA qui constitue un axe central du livre. Incontournable !

The security development lifecycle

- ***Difficulté : *******
- ***Editeur : Microsoft Press***
- ***Auteur : collectif***
- ***Prix : 34,99 $ (USA)***

Après le codage sécurisé, Microsoft Press propose un autre ouvrage sur la sécurité dans le développement en expliquant la démarche SDL (Security Development Lifecycle ou cycle de vie du développement sécurisé). SDL se compose de 13 étapes, décrites par les auteurs. Ce n'est pas une théorisation de plus mais de bonnes pratiques réelles, applicables sur le terrain. SDL a l'ambition de réduire les vulnérabilités, les failles dans les projets. Dès les premières pages, les auteurs sensibilisent les lecteurs sur l'impossibilité, avec les modèles et outils actuels, de fournir du code, des applications sécurisées et au fait qu'il ne faut pas confondre les méthodes de type CMMi et méthodes agiles avec le SDL, elles n'ont pas pour but de fournir une sécurisation du code. La réussite du SDL dépend du soutien des responsables, de la compétence des personnes la mettant en œuvre et de la bonne éducation des développeurs et des intervenants aux projets. Il y a une structure à mettre en place pour garantir la réussite du SDL dans les équipes et les projets. Passionnant et instructif. De nombreuses références et sources complètent le texte. Livré avec un CD. En Anglais.

Cyber espionnage



- ***Difficulté : *****
- ***Editeur : Chiron éditeur***
- ***Auteur : Gérard Desmaretz***
- ***Prix : 26 €***

L'informatique est une arme incontournable dans l'espionnage industriel, d'État à État, les dernières affaires qui ont défrayé la chronique, notamment en Allemagne, en sont la preuve. L'auteur revient sur l'affaire de la NSA et de son espionnage envers les pays amis, mais n'oublions pas que la France possède aussi son système " Echelon ". Puis on passe au fameux Big Brother et au respect de la vie privée et des données privées, de plus en plus difficiles à protéger, à tracer. L'auteur dresse la liste noire des menaces, des problèmes liés aux nouvelles technologies.

Architectures réparties en Java

- ***Difficulté : *****
- ***Editeur : Dunod***
- ***Auteur : Annick Fron***
- ***Prix : 25 €***

On parle beaucoup d'applications distribuées, de développement distribué, mais finalement qu'est-ce qu'une architecture répartie, ici en Java ? L'auteur prend appui sur les grandes technologies disponibles, RMI, Corba, SOAP, les services web ou encore JMS. Chaque architecture est présentée et expliquée avec des exemples de mises en œuvre, de code.

C++ pour les programmeurs C

- ***Difficulté : *******
- ***Editeur : Eyrolles***
- ***Auteur : Claude Delannoy***
- ***Prix : 32 €***

Nouvelle édition du " Programmez en C++ " du même auteur, devenu une référence pour les

développeurs. L'objectif du livre est de proposer une migration, une compréhension du C++ et de ses concepts aux développeurs C. S'appuyant sur la norme ANSI/ISO, l'auteur prend en compte la bibliothèque standard et l'ensemble des aspects du C++. De très nombreux codes permettent de mieux comprendre les concepts expliqués.

VBA pour Office 2007

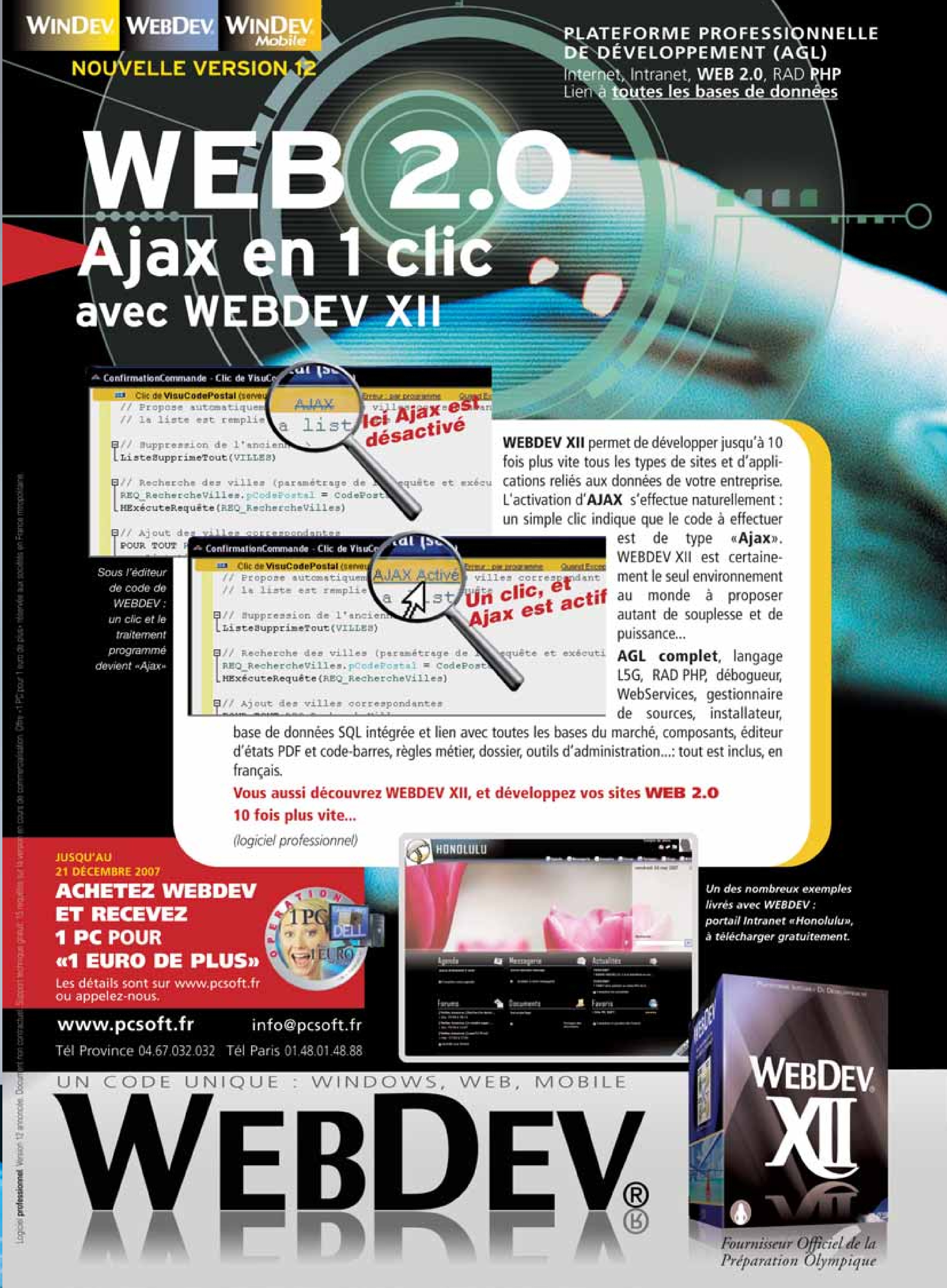
- ***Difficulté : ** / ******
- ***Editeur : Micro Application***
- ***Auteur : Collectif***
- ***Prix : 57,20 €***

Eh bien non, VBA n'est toujours pas mort ! Pour preuve, cet impressionnant livre sur la programmation VBA sous Office 2007. L'épaisseur du livre est à la hauteur de l'ambition de l'ouvrage : donner une vue, la plus complète possible, selon les différents outils (Word, Excel, PowerPoint, Access). La partie sur les techniques avancées intéressera sans doute nombre de développeurs Office comme, pouvoir mettre en œuvre l'API Windows dans une application VBA, ou encore, gérer les interactions entre les applications Office. Dommage que VSTO soit oublié. Malgré tout, c'est très complet et accessible quel que soit son niveau.

Algorithmique

- ***Difficulté : ******
- ***Editeur : éditions Eni***
- ***Auteur : Sébastien Rohaut***
- ***Prix : 27,14 €***

On ne le répétera jamais assez, une des bases de la programmation est la maîtrise des algorithmes. Une fois les notions de base acquises, le lecteur trouvera dans ce livre de quoi évoluer vers des notions plus avancées : deux chapitres, l'un sur les pointeurs et les références, l'autre sur les objets, ouvrent les portes de la programmation dans des langages évolués et puissants comme le C, le C++ et surtout Java. Une grande partie des algorithmes de ce livre sont réécrits en Java et les sources.


WINDEV WEBDEV WINDEV Mobile
NOUVELLE VERSION 12
PLATEFORME PROFESSIONNELLE DE DÉVELOPPEMENT (AGL)
Internet, Intranet, WEB 2.0, RAD PHP
Lien à toutes les bases de données
WEB 2.0
Ajax en 1 clic
avec WEBDEV XII
Ici Ajax est désactivé
Un clic, et Ajax est actif
Sous l'éditeur de code de WEBDEV : un clic et le traitement programmé devient «Ajax»
WEBDEV XII permet de développer jusqu'à 10 fois plus vite tous les types de sites et d'applications reliés aux données de votre entreprise. L'activation d'AJAX s'effectue naturellement : un simple clic indique que le code à effectuer est de type «Ajax». WEBDEV XII est certainement le seul environnement au monde à proposer autant de souplesse et de puissance...
AGL complet, langage L5G, RAD PHP, débogueur, WebServices, gestionnaire de sources, installateur, base de données SQL intégrée et lien avec toutes les bases du marché, composants, éditeur d'états PDF et code-barres, règles métier, dossier, outils d'administration...: tout est inclus, en français.
Vous aussi découvrez WEBDEV XII, et développez vos sites WEB 2.0 10 fois plus vite...
(logiciel professionnel)
JUSQU'AU 21 DÉCEMBRE 2007
ACHETEZ WEBDEV ET RECEVEZ 1 PC POUR «1 EURO DE PLUS»
OPÉRATION 1 PC 1 EURO
Les détails sont sur www.pcsoft.fr ou appelez-nous.
www.pcsoft.fr info@pcsoft.fr
Tél Province 04.67.032.032 Tél Paris 01.48.01.48.88
HONOLULU
Un des nombreux exemples livrés avec WEBDEV : portail Intranet «Honolulu», à télécharger gratuitement.
UN CODE UNIQUE : WINDOWS, WEB, MOBILE
WEBDEV®
WEBDEV XII
Fournisseur Officiel de la Préparation Olympique


Votre potentiel, notre passion.™
Microsoft®
LES OGRES SONT PARTOUT. MAIS SI VOTRE APPLICATION EST EN AVANCE SUR LE PLANNING, ILS VOUS LAISSERONT TRANQUILLE.
Votre défi : créer des applications riches, beaucoup plus vite.
Votre arme : Visual Studio® pour Microsoft® Office 2007.
Plus d'informations sur www.releveztouslesdefis.com
RELEVEZ TOUS LES DÉFIS
Microsoft
Visual Studio
© 2007 Microsoft Corporation. Tous droits réservés. Microsoft, Visual Studio, le logo Visual Studio, et « Votre potentiel, notre passion. » sont des marques de Microsoft déposées et/ou utilisées aux États-Unis et/ou dans d'autres pays.