Programmez !
LE MAGAZINE DU DÉVELOPPEMENT
Mensuel - Mai 2008 - N°108
www.programmez.com

Exclusif
DVD
les outils
ADOBE

WEB DevSign

Quand le développeur et le designer se rencontrent !

Collaboration et nouveaux outils chez Adobe et Microsoft. Témoignages

Illustration : WeAreFantastic

Exploiter toute la puissance de XML

Valider un document XML. Coder avec StAX, PHP et XML.

Ruby on Rails 2.0

Migrer de Rails 1.x à 2.0
Intégrer Rails à NetBeans
Rails dans vos applications .Net

WINDOWS
Vidéo avec Picture Acquisition SDK

.NET
Comment valider des données en ASP.Net ?

SGBD
Découvrir l'ETL de Talend

MÉTHODOLOGIE
Découvrir l'eXtreme Programming
Model Driven & Open Source : ça marche !

ECLIPSE
Toutes les annonces d'EclipseCon 2008

M 04319 - 108 - F: 5,95 €

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

DÉVELOPPEZ 10 FOIS PLUS VITE

WINDEV®
DÉVELOPPEMENT RICHE

INCLUS EN STANDARD DANS WINDEV 12 :

Environnement intégré (IDE & ALM)

- Tout est en **français**
- Hot Line personnalisée gratuite*
- Déploiement libre et gratuit
- Crée des .EXE sécurisés, des Web Services, des applications .NET, des applications Java...
- 32 bits, **64 bits**
- **Compilation** JIT
- Code multi-plateforme compatible **Mobile** et **Internet**
- Génération d'application **Java** à la demande
- Fonctionne en TSE et Citrix
- Générateur automatique d'**IHM**, avec charte graphique automatique. Création d'IHM «Vista» & «Vista Like» par utilisation de gabarits fournis
- Tous les Windows : 98, 2000, NT, XP , Vista, 2008...
- Générateur d'**Etats et Requêtes diffusable**, création de **PDF**, code-barres, étiquettes. Fond de page PDF
- Automatique dans chaque application: menu d'export vers Word, Excel, OpenOffice, XML, PDF; Graphiques 3D; Historique de saisie,... Envoi de mail, Macros utilisateur
- Real-**RAD** : Générateur d'applications complètes
- AAA (Architecture Automatisée d'Application): créez votre propre RAD (Patterns)
- **Hyper File SQL**, Base de Données Client/Serveur, Locale et Mobile sous Windows et Linux (version illimitée, **libre** et gratuite); Gère 4 millions de Téras
- Accès à toutes les Bases de Données tierces: **Oracle, AS/400, SQL Server**, DB2, **MySQL**, Access, xBase, etc...
- Réplication multibases assistée
- XML natif
- Accès natif à **SAP R/3**, Lotus Notes, LDAP, Outlook,...
- Centre de suivi du **planning** d'équipes
- Centre de suivi des retours utilisateurs
- Tableau de Bord de **suivi de projet**
- Centre de Modélisation **UML**, Merise et Souple; code généré depuis l'analyse, reverse engineering
- Dossier automatique : analyse & programmation
- Création et utilisation de **composants** ; 3-tiers
- Règles métier; Gestion des exigences
- Langage de 5° Génération **L5G**, qui élimine 90% du code
- **Ouverture** aux L4G et L3G: C++, C#, Java, VB, Cobol...
- Import d'applications Access et VB
- Fonctions Domotique (norme X10)
- Gestion liaison série RS 232, parallèle et USB
- Fonctions Bluetooth,
- Fonctions réseau **SNMP**
- Fonctions TAPI, OPC, FTP, HTTP, Socket, Twain, API, DLL,...
- Editeur de code intelligent, avec test immédiat sans recompilation
- **Tests unitaires** de code et d'IHM automatisés, Editeur visuel de tests de non-régression
- Refactoring
- Outil de **versionning** sophistiqué (gestion des sources)
- Débogueur puissant: threads, composants,... Débogage à distance
- **Profiler**, pour optimiser la vitesse du code
- Multilingue automatique: jusqu'à 20 langues
- Générateur d'aide CHM
- Fonctions Multimédia (image, son, vidéo)
- Générateur d'**Installations en 1 clic,** et mises à jour automatiques (local, à distance, via Internet)
- Autoformation en 1 semaine (**manuel livré**)

PC SOFT 25 ANNÉE Au service des développeurs

Développez de superbes IHM sans compétences graphiques grâce au générateur d'IHM (avec correcteur d'IHM) et aux gabarits fournis.

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr

Demandez le dossier gratuit (200 pages + 1 DVD) Inclut la version Express (gratuite) et 112 Témoignages détaillés. Tél: 04.67.032.032 ou info@pcsoft.fr

XII

Logiciel professionnel. Document non contractuel. Support technique gratuit: 15 requêtes sur la version en cours de commercialisation.

+ Tutoriel sur
www.programmez.com/tutoriel.php

SOMMAIRE N°108

> Actus
L'actualité en bref 6

> SGDB
MySQL User conference 2008 12
Agenda 12

> Evénement
EclipseCon 2008 : en route vers Eclipse 3.4 et 4.0 14

> Projets
Appréhender l'eXtreme Programming 18
Model Driven pragmatique et Open Source 24

> Gros Plan : RUBY on Rails (2e partie)
Migrer en douceur de Rails 1.2 à 2.0 30
Intégration de Rails dans NetBeans 6.0 32
Une introduction à Dynamic Silverlight et IronRuby 36

> Dossier : XML et le développeur
XML : les fondamentaux ! 41
Validation des documents XML (1re partie) 44
StAX : une API XML pour Java 46
Réutilisation de contenu dans Altova XMLSpy avec OOXML et XSLT 50
Résoudre le problème des performances de parsing 52

> Développement Web
Web Devsign :
quand le développeur et le designer se rencontrent...
Trouver un langage commun 55
Les bonnes pratiques... 57
Silverlight 1.0/2.0 58
Du designer CS3 au développeur Flex 61
Témoignage 64

> Code
La validation des données en ASP.Net 66
Extraction, Transformation et Chargement de données avec Talend Open Studio 71
Visualisation de sécurité en open source 75
Capturer des vidéos et des photos depuis une webcam 78

> Temps libre
Les livres du mois 82

DVD-Rom 108 PROGRAMMEZ !

ADOBE FLEX 3

Flex 3 SDK
Fournit un framework open source pour concevoir des applications Flex. Supporte le développement d'applications bureautiques avec Adobe AIR.

Flex Builder 3 Professional
Outil de développement professionnel, basé sur Eclipse pour le développement d'applications Flex. Prend également en charge le développement d'applications bureautiques avec Adobe AIR.

Plug-in Eclipse pour Flex Builder 3 Professional
Téléchargez le module externe Eclipse pour Flex Builder 3 Professional, si Eclipse est déjà installé sur votre ordinateur.

ADOBE AIR

AIR 1.0.1 Installer
Le moteur d'exécution Adobe AIR permettant d'exploiter des technologies web éprouvées pour la création d'applications Internet riches à déployer en local.

Kit SDK AIR
Les développeurs HTML et Ajax peuvent utiliser les outils en ligne de commande fournis dans le kit SDK Adobe AIR ou exploiter un environnement de développement intégré très complet compatible AIR.

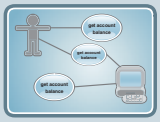
Extension AIR pour Dreamweaver
Fournie au format MXP elle peut être installée à l'aide d'Adobe Extension Manager. (attention : utilisez Extension Manager pour supprimer, avant l'installation, toute version antérieure éventuelle.)

Tous les jours : l'actu et le téléchargement.
Achetez les magazines, les articles en PDF et abonnez-vous en ligne
www.programmez.com

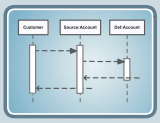
Modélisez vos logiciels à votre gré

ALTOVA® umodel® 2008

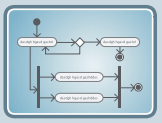
Faites l'expérience par vous-même, et découvrez Altova UModel® 2008, le plus simple et rentable des programmes de développement de logiciels graphiques en UML.



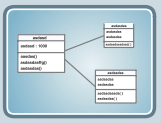
Use case



Sequence diagram



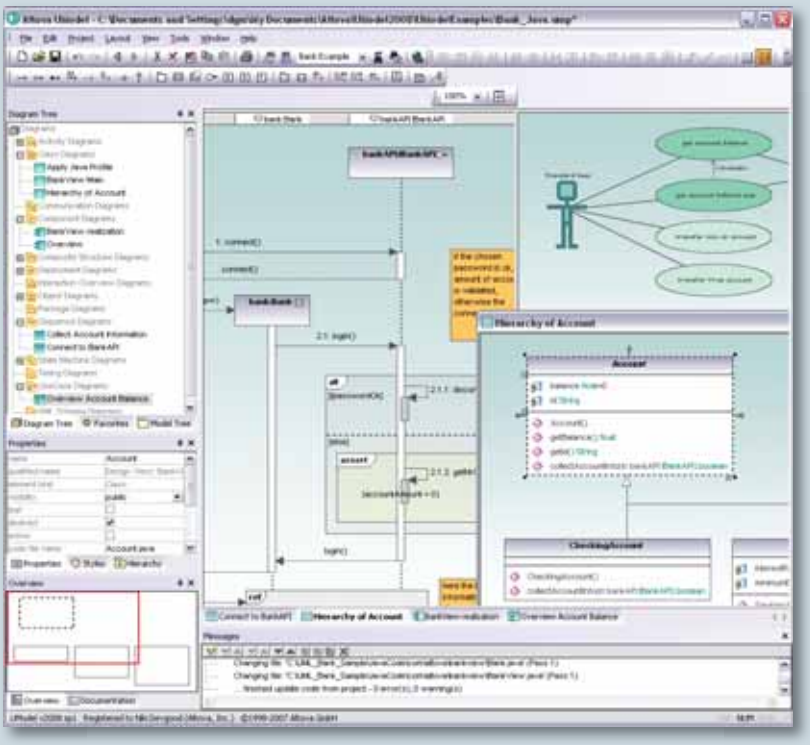
Activity diagram



Class diagram

- Prise en charge des 13 types de diagramme UML 2.1
- Génération de code d'application en Java, C#, et VB.NET
- Synchronisation du modèle et du code par ingénierie aller-retour
- Modélisation des schémas XML en UML
- Création automatique de documents personnalisables relatifs au projet
- Intégration poussée de Visual Studio® et Eclipse
- Liaison hypertexte des diagrammes, documents et pages Web
- Rétro-ingénierie du code Java, C#, et VB.NET
- Rétro-ingénierie des fichiers binaires Java, C#, et VB.NET
- Partage de sous-projets en vue d'une collaboration ou d'une réutilisation
- Echange de modèles via XMI 2.1
- Et bien plus encore . . .

UModel vous permet de concevoir votre application avec une prise en charge des 13 diagrammes UML 2.1, plus un diagramme de schéma XML de type UML pour organiser vos données XML dans votre projet. Générez du code en Java, C#, ou Visual Basic, puis améliorez votre design en modifiant le code ou bien le modèle, et synchronisez le tout par ingénierie aller-retour. Vous pouvez également procéder à une ingénierie aller-retour sur une application existante pour en faire une analyse visuelle. UModel s'adapte à votre style de développement : vous pouvez l'exécuter au sein de Visual Studio, Eclipse, ou comme outil indépendant. Avec Umodel, la conception de logiciels graphiques devient simple comme bonjour, quel que soit votre projet !



ALTOVA®

Testez UModel vous-même – Téléchargez gratuitement une version d'essai de 30 jours sur www.altova.com

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, J. Vidames, L. Guillois, C. Buret.

Experts : C. Brun, E. Chenu, B. Cabé, Y. Lugrin, T. Barrère, A. Dupuis, A. Brillant, A. Falk, E. Minneure, T. Lebrun, G. Lebrun, M. Chaize, J. Duprat, F. Bonan, P. Chifflier, S. Tricaud,

Dessin : Michel Piedoue

Illustration couverture et dossier Web : WeAreFantastic

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 1 DVD 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

Après le web 2, les applications 2.0, l'entreprise 2.0, si nous inventions le développeur 2.0 ? Vous allez sursauter. Est-ce une mutation du code source ? Un poisson d'avril dont la date de consommation serait dépassée ?
En observant le large mouvement des éditeurs vers le logiciel + service, je ne reviendrai pas ici sur l'affrontement stérile entre S+S et SaaS, on constate une vague de fond pour l'application qui dématérialise et se transforme en véritable informatique à la demande. Il ne se passe pas une journée sans une annonce d'un S+S par là, d'un nouveau service hébergé ici, etc. Ces annonces cachent une véritable révolution du modèle de développement que l'on pouvait, à peine je le reconnais, discerner avec les discours autour de RIA et du web 2.0 en général. On amorce avec le S+S un changement rapide dans la manière de penser l'application, tout autant que sa consommation par l'utilisateur. Et les annonces de Google et Microsoft ces dernières semaines, me confortent dans l'idée que l'avenir est à l'application éclatée, sous forme de services que l'on agrège. Ah ! Cela me rappelle OpenDoc, la préhistoire de l'informatique, il y a 15 ans !

Le développeur 2.0

Google, avec son App Engine, propose la possibilité d'héberger des applications sur son infrastructure et de consommer des applications déjà hébergées. Fini le déploiement et la gestion sur ses serveurs ! C'est aussi ce que propose Microsoft avec les services Online, et dans une moindre mesure les services Live. Bientôt (restons vague sur la date), on disposera d'un SQL Server sous forme de services Online. Le DBA n'aura plus de serveurs de données à surveiller *in situ* mais en ligne… Cela va être une révolution ! Et surtout, on pourra adapter les ressources à ses besoins. Bien sûr, le marché est encore anecdotique malgré les annonces. Mais en 2009, nous nous attendons à un véritable décollage. Et les éditeurs ayant eu le courage et le flair de s'y positionner rapidement, quitte à essayer des revers et le manque de clients, peuvent espérer être bien placés.

Là, vous vous dites " mais quel rapport avec le développeur ? ". C'est bien, quelques-uns d'entre- vous suivent ! Le développeur devra coder, adapter l'application au S+S : maîtriser le développement web, les protocoles, XML, les standards multiples, de nouveaux SDK, etc. La mouvance RIA prend ici un autre sens car on imagine très bien le mix entre la RIA et les services en ligne.

Cependant, il y a un léger problème. Quelle interopérabilité entre les briques d'un même éditeur et entre les services inter éditeurs ? Comme on le dit parfois en interview, " excellente question ! ". Et bien souvent, après cette superbe pirouette, la réponse ne vient pas. Aujourd'hui, nous sommes tellement en avance de phase que toutes ces questions attendent des réponses concrètes. Si on prend les services Live et Online de Microsoft, nous aurons bien du mal à les intégrer, à échanger. Il faut absolument des briques " middleware " pour assurer la glue entre les services. Et la situation est encore plus criante entre des services de différents éditeurs. Oui, il y a toujours la possibilité de passer par du XML, de contourner avec un web service, mais si l'interopérabilité n'est pas possible " nativement ", nous aurons un sérieux problème !
Car n'oublions pas une chose : S+S ne signifie nullement compatibilité, interopérabilité ! Et nous nous demandons même si le véritable défi n'est pas justement de créer un vaste monde hébergé, sachant communiquer, échanger. Espérons simplement que l'on saura concrétiser : " j'ai fait un rêve ".

François Tonic - ftonic@programmez.com

PROCHAIN NUMERO *N°109 - Juin - Parution : 30 mai 2008*

Web 2.0
Mashup
La nouvelle mode des applications web !
Comment et pourquoi agréger des services, des gadgets ? Quels outils utiliser ?

Projet
Tests
Êtes-vous prêts aux tests ? Quelles bonnes pratiques ? Quels outils ? Maîtriser et mettre en œuvre une vraie politique de tests. Votre nouveau métier ?

XML 2e partie
Pourquoi et comment utiliser le XSLT, les outils, validation des documents par les schémas. Tout savoir sur XML 2 !

EN BREF...

.Net sous Linux !

L'éditeur Mainsoft, connu pour ses outils permettant à des applications .Net de fonctionner en environnement JEE et Linux, récidive avec ASP.Net Ajax via ASP.Net 2.0 Ajax Extensions et Ajax Control Toolkit. ASP.Net 2.0 Ajax Extensions permet de créer des applications web Ajax et de porter les applications actuelles directement sur Tomcat ou Websphere. Il supporte intégralement le control toolkit de Microsoft. Le support de WebSphere Portal est déjà prévu prochainement ! Ces deux éléments sont intégrés dans Mainsoft for Java EE 2.2.

Journée française des tests logiciels !

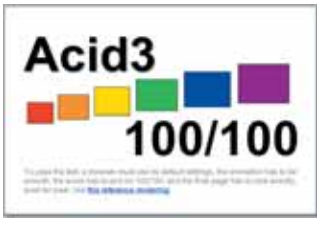
Le 17 juin prochain se tiendra à Paris un événement logiciel important : une journée entièrement consacrée aux tests logiciels. Organisée par le Comité Français des Tests Logiciels, cette journée proposera de nombreuses sessions techniques (autour du test) : la gestion des tests, les retours d'expérience, la génération automatique des tests, etc. Une dizaine d'exposants seront présents. L'accès à la journée est payant. Pour en savoir plus : www.cftl.fr

Adobe AIR 1.1 bientôt !

Adobe, durant son On Air 2008, a annoncé la disponibilité mi-2008 de AIR 1.1. Une des nouveautés sera le multi-langue, dont le français ! D'autre part, l'éditeur lance la pré-version d'AIR 1.0 pour Linux. L'expérience et le code seront identiques aux versions MacOS X et Windows. D'autre part, Adobe annonce la disponibilité de la v1.1 sur Linux pour la fin de l'année. Après la 1.1, les versions de AIR seront synchronisées sur les différents systèmes supportés.

NAVIGATEUR

ACID3 et Mozilla : polémique ou non ?



Il y a quelques semaines de cela, une petite polémique entre Opera et Webkit avait lieu autour d'ACID3. Mozilla est demeuré très discret. Tristan Nitot de Mozilla en donne la raison, toute simple : *" nous sommes focalisés sur la stabilité (du code) de Firefox. Disons que ACID3 tombe mal pour nous (au niveau agenda) car il est difficile de modifier le code de Firefox 3. On approche de la version finale. "* D'autre part, comment comprendre les résultats des navigateurs sur ACID3 ? Pour Mike Shaver, ACID3 est une opportunité ratée. Car finalement ACID est une batterie de tests sur des fonctions de navigateurs. Or la question est de savoir quelles fonctions il faut tester. ACID est fait pour que les navigateurs ne les passent pas immédiatement, comme nous l'a précisé Tristan Nitot. Il faut ensuite modifier le code pour les passer avec succès. L'autre question concerne Firefox Mobile. Aujourd'hui, avec les gros efforts des développeurs pour optimiser la gestion de la mémoire (et son empreinte) et les performances globales du navigateur, Mozilla peut réviser l'agenda mobile. *" J'espère une pré-version mobile d'ici la fin de l'année "* avance T. Nitot. Les progrès faits par le moteur Gecko sur la mémoire et les performances permettent d'espérer une version mobile dans les prochains mois.

ACID3 : http://shaver.off.net/diary/2008/03/27/the-missed-opportunity-of-acid-3/

Progrès en utilisation mémoire :
http://blog.pavlov.net/2008/03/11/firefox-3-memory-usage/

Progrès en rapidité JavaScript :
http://blogs.zdnet.com/hardware/?p=1536&page=2

Impact sur notre stratégie mobile :
http://www.0xdeadbeef.com/weblog/?p=349

PORTABLE

Un kit de développement pour l'Asus Eee PC

L'ultra portable d'Asus a droit à son kit de développement. Il doit permettre aux développeurs de développer des applications diverses et variées pour la gamme Eee PC tournant sous Linux. Le SDK est lui-même open source et disponible sur sourceforge. Le kit fournit donc un environnement complet, incluant un support de Xandros Desktop OS, un IDE Eclipse, un kit Qt 4, un guide du développeur, des exemples de code et une image VMware pour tester et déboguer. L'ensemble des outils et Xandros sont regroupés dans Xandros Desktop Open Circulation Edition. À la base, les applications se développent en C++. Les développeurs Qt ne seront pas trop dépaysés par la conception d'interfaces via Qt 4. Pour le packaging, la plate-forme utilise celui de Debian (la création se fait via Package Builder). Le guide du SDK est complet et montre les premières étapes pour créer et déployer son application Eee PC. Le SDK complet pèse environ 1,2 Go (fichier ISO). Attention ce SDK n'est valable que pour la version Linux de l'Eee PC. La version Windows XP n'est pas officiellement supportée par le kit.

Site : http://sourceforge.net/projects/eeecommunity/

WEB

Google héberge vos applications !

Les services hébergés sont désormais une pratique courante chez les éditeurs. Après Online de Microsoft et les prémices de Sharepoint et SQL Server en ligne, Google propose une nouvelle brique importante : Google App Engine.

App Engine permet de faire tourner vos applications web sur l'infrastructure Google! Mais il faut que ces applications soient des applications App Engine. Elles se veulent simples à construire, à maintenir et à déployer. La montée en charge et les ressources suivent vos besoins. Vous n'avez plus de serveur à maintenir, plus d'infrastructure interne ! D'autre part, on peut disposer d'un nom de domaine lié à l'application. Le coût est nul au départ et la création du compte, gratuite. On dispose par défaut de 500 Mo de stockage avec un trafic de 5 millions de pages

vues par mois. Cependant, le service deviendra payant lorsqu'il passera en version finale (actuellement en préversion). App Engine propose un environnement complet de développement : un serveur web dynamique (avec support des principales technologies web), un stockage persistant, un load balancing, des API d'authentification et de mails (pour utiliser les comptes Google) et enfin un environnement local pour simuler l'environnement App Engine. Pour le moment, seules les applications Python sont prises en charge ! Sur Python, il s'agit de la version 2.5.2 et elle inclut la librairie standard. Des API Python pour le datastore, Google Accounts, le mail… sont disponibles. App Engine expose un framework applicatif Python (webapp). On dispose aussi de Django, un framework web. Le développeur utilise le App Engine SDK incluant le serveur web, les émulateurs, les API et librairies. À cela se rajoute le Python SDK. Il fonctionne sous Windows et Mac OS X. Pour gérer ses applications et environnement App Engine, Google propose une console d'administration. Attention : le nombre de développeurs admis s'accroît au fur et à mesure chez Google. Voilà une initiative à surveiller de très près !
Site : http://code.google.com/appengine/

SÉCUTITÉ

Aladdin eToken NG-FLASH : authentification par mémoire flash et Java Card

Aladdin, spécialiste de l'authentification, de la gestion des droits numériques (DRM) et de la sécurité des contenus, lance eToken NG-FLASH 72K. Cette solution utilise à la fois les fonctions d'une mémoire flash (support USB) et la technologie Java Card. Le tout donnant une carte à puce sophistiquée qui assure une authentification à double facteur extrêmement sûre. Le token évite ainsi d'avoir recours à des jetons séparés pour l'accès et le stockage et associe jusqu'à 4 Go de stockage chiffré, ainsi qu'une mémoire flash importante, à une technologie d'authentification à carte à puce afin d'offrir une solution unifiée, portable et sûre. Les utilisateurs peuvent ainsi transporter des informations confidentielles, (leurs identifiants réseau, certificats numériques et clés de cryptage par exemple), authentifier et développer des fichiers et des applications, puis y accéder à partir de n'importe quel ordinateur. Toutes les cartes à puces eToken d'Aladdin sont maintenant proposées avec Java Card. Java Card offre une sécurité renforcée et une possibilité de personnalisation. D'autre part, un SDK Java, inclus, permet, aux développeurs, de créer et de charger leurs applets sur le jeton afin de personnaliser les applications et d'en enrichir leurs fonctionnalités. Site : www.Aladdin.com/eToken.

BREF...

Une nouvelle interface pour Perforce

L'éditeur Perforce lance une nouvelle interface pour son client Perforce Visual Client. Elle est unique, quel que soit le système utilisé et repose sur la librairie multi-plate-forme Qt 4. Cette nouvelle présentation doit apporter une meilleure visualisation des branches et un historique d'intégration, un meilleur confort d'utilisation, un outil d'administration amélioré.

4D lance 4D Web 2.0 Pack v11 Release 1

Le français 4D lance une nouvelle version de son pack 4D Web. Il est entièrement compatible avec 4D v11 SQL. Il permet de créer des applications riches 4D et d'utiliser les technologies Web. Le pack inclut un framework Ajax, 4D Live Windows et 4D Ajax for Dreamweaver. Le framework comprend une grille de données affinée, la gestion du glisser-déposer et d'autres nouveautés. On dispose aussi des formats SVG et PNG et du support de Google Chart API.

Sybase lance SQL Anywhere 11 bêta

L'éditeur vient de lancer la pré-version de SQL Anywhere 11, nom de code Panorama. Elle promet 200 nouveautés et améliorations. SGBD majeur dans l'embarqué et la mobilité, cette version doit renforcer son utilisation dans les applications et données critiques. Cela inclut une meilleure capacité dans les requêtes parallèles, une recherche plein texte, de nouvelles options de synchronisation de données ainsi que la possibilité de créer des procédures stockées en .Net et Perl ! Version finale prévue pour la fin de l'année.

Neomades lance NeoMAD 1.5

Le français Neomades propose la v1.5 de son framework NeoMAD. Il permet de développer des applications mobiles JavaME qui cachent une grande partie de la complexité de portabilité des applications d'un mobile à un autre. Cette version supporte la cartographie et la géolocalisation, les écrans tactiles, le streaming, et s'intègre aux principaux IDE actuels. Site : http://www.neomades.com

SYSTÈME

ExoPlatform lance eXo Portal 2.0

Le Français Exoplatform a lancé mi-avril dernier la v2 de son eXo entreprise webos. L'éditeur le positionne plutôt comme un portail 2.0 et non comme un webos, une question de sémantique commerciale… Avec cette version, l'environnement veut offrir une expérience utilisateur proche du desktop avec une bonne dose de RIA (notamment du Flex pour certaines fonctions), de web 2.0. Le tout en restant dans un navigateur. Nous sommes là dans un portail d'entreprise pour la collaboration et l'accès à ses applications (notamment métiers). On peut mettre en place des widgets, du glisser-déposer, personnaliser l'interface par des skins, utiliser des assistants pour simplifier la création de nouvelles pages. Le portail s'accompagne de la disponibilité de deux mises à jours pour la gestion de contenu et pour la collaboration. Il est possible d'intégrer des applications tierces dans le portail. Mais eXo est avant tout open source. Une édition communautaire est disponible. Le modèle économique de l'éditeur est de passer par des revendeurs, prestataires pour proposer ExoPlatform à leurs clients. La solution est accessible via une souscription annuelle. Très tôt, eXo a adopté les services Amazon S3 et EC2. À la base, l'environnement s'appuie sur Java et notamment sur des portlets et Java Content Repository. Parmi les spécifications retenues par les développeurs maison, on retrouve la JSR 286 pour les portlets ou encore la JSR311 pour tout ce qui est Jax-RS, RESTful en Java… Pour étendre les fonctions et applications d'eXo, on peut passer par des portlets ou encore des gadgets. Si dans ce dernier cas, l'intégration reste limitée, dans une future version, on pourrait intégrer tel quel les gadgets Google. Pour le développeur, on dispose d'API, notamment REST. L'intéropérabilité avec le monde Windows / .net n'est pas oubliée, notamment avec un add-on Office 2007 qui permet d'envoyer à eXo des données provenant de Word… L'avenir dira si le webos a réellement un avenir, surtout face aux Software + Services.

Site : http://www.exoplatform.com

SGBD

EnterpriseDB présente Postgres Plus

EnterpriseDB propose depuis quelques semaines une gamme complète de base de données : PostgreSQL Plus. Elle s'appuie sur le solide PostgreSQL, avec une connotation plus entreprise sur les performances et les capacités améliorées pour le DBA et le développeur : une installation simplifiée, la présence de Developer QuickStarts pour que le développeur puisse développer plus rapidement des applications données en Ruby, JBoss SEAM, REST. Les QuickStarts incluent de nombreux outils comme Drupal, MediaWiki, phpBB. On dispose aussi de Plus Advanced Server qui ajoute des fonctions comme la possibilité de faire tourner des applications conçues pour Oracle, faire des tuning de performances, une compatibilité accrue avec Oracle. Disponible sous Windows, Linux, MacOS X. À noter que Talend et Linagora ont signé des partenariats avec EnterpriseDB.

UN SIMPLE CONSTAT :

À travers le monde, dans les banques, compagnies d'assurances, sociétés financières, médias, organismes d'état, des dizaines de milliers d'applications critiques sont développées et maintenues en PowerBuilder.

NOTRE STRATEGIE

DEPUIS LE 18 AVRIL 2008 LA DERNIÈRE VERSION 11.2 EST DISPONIBLE.

***Avec PB11.2, configurez AJAX en quelques clics pour des applications web hautement intéractives et performantes.**

Prochaines étapes :

- support WPF (Windows Presentation Foundation)
- support WCF (Windows Communication Foundation)
- .NET 3.5/4
- Nouvel IDE basé sur la Visual Studio Shell 2008

SEMINAIRE POWERBUILDER

Paris, le 27 mai 2008

Coup de Projecteur sur les Evolutions Futures

Pavillon d'Armenonville Potel et Chabot

Détails et inscriptions:
+33 (0)1 41 91 96 80
europePB@sybase.com
http: www.sybase.fr

En conclusion avec PowerBuilder :

- Protégez votre investissement
- Accédez aux technologies qui seront les standards de demain

Informations produit et évaluation :
http://www.sybase.com/products/development/powerbuilder

SYBASE®

NOUVEAU

Une nouvelle rubrique démarre en mai sur www.programmez.com

les **TUTORIELS !**

Les pas à pas pour vous guider

www.programmez.com/tutoriels.php

RACHAT

Progress rachète Xcalia

Le français Xcalia spécialisé dans les technologies JDO et SDO et l'intermédiation a été racheté par Progress, qui renforce ainsi sa panoplie dans la gestion et la manipulation des données et notamment sur le rôle croissant de SDO dans les architectures SOA.

SYSTÈME

Windows Mobile 6.1 disponible

Microsoft propose la v6.1 de Windows Mobile. Cette mouture arrive avec une nouvelle version d'Internet Explorer Mobile ayant l'ambition d'offrir la même navigation que sur PC, et donc de rattraper son retard sur Safari et iPhone. Surtout, il supporte Flash, Silverlight. La 6.1 doit faciliter la configuration, améliorer l'utilisabilité sur de petits écrans.

SYSTÈME

Microsoft définit sa stratégie embarquée

L'éditeur a dévoilé mi-avril durant la Embedded Systems Conference Silicon Valley 2008, une annonce majeure sur le marché de l'embarqué. Il s'agit d'offrir aux clients, utilisateurs, une extrême modularité selon les matériels, les terminaux. La première initiative est Windows Embedded Standard, dont Windows XP Embedded (nouvelle version) sera le premier produit à en bénéficier. Tous les produits Windows Embedded actuellement disponibles seront commercialisés sous leur nom actuel jusqu'au prochain lancement prévu et resteront disponibles à l'achat, conformément à la politique standard de Microsoft pour le soutien du

cycle de vie. La gamme de produits Windows Embedded Ready pour les catégories d'appareils clés comprendra la prochaine génération de Windows Embedded pour point de service, Windows Embedded POSReady. L'activité Windows Embedded dévoilera sa stratégie plus en détail pour les produits Windows Embedded Ready plus tard dans l'année. L'agenda est le suivant :

- Windows Embedded Standard, la prochaine génération de Windows XP Embedded. Le lancement du produit est actuellement prévu pour le courant de l'année 2008.
- Windows Embedded Compact, la prochaine génération de Windows Embedded CE. La sortie du produit est actuellement prévue pour le courant de l'année 2009.
- Windows Embedded Enterprise, un système d'exploitation intégré compatible avec toute application, qui incorporera à l'avenir un ensemble plus large de fonctionnalités intégrées. Aujourd'hui, ce groupe de produits inclut Windows Vista et Windows XP for Embedded Systems et est offert exclusivement sous licence pour le développement de systèmes intégrés.
- Windows Embedded POSReady, la prochaine génération de Windows Embedded pour point de service. La sortie du produit est actuellement prévue pour le courant de l'année 2009.

LIBRAIRIE

Leonardi 4.0 disponible en open source

LYRIA, éditeur du logiciel LEONARDI, automatisant la production des IHM (Interfaces Homme-Machine), annonce la disponibilité de la V4.0 de son framework piloté par le modèle métier en open source. Cette version intègre de surcroît de nombreuses évolutions ergonomiques et fonctionnelles qui améliorent notablement l'atelier de conception Studio. Après avoir mis à disposition des développeurs depuis près de 2 ans une version gratuite de son framework, Lyria rejoint ainsi la liste grandissante des éditeurs adeptes de l'open source. Ce choix vise l'accélération de la diffusion de LEONARDI par l'usage et le renforcement de l'activité de services spécialisés autour de sa technologie pilotée par le modèle métier. LEONARDI V4.0 sera dorénavant diffusé selon le modèle de licence double, avec une version open source de type GPL et une offre commerciale plus complète et plus permissive pour la redistribution d'applications propriétaires. Cette version supporte Java 5, les tableaux de bord avec onglets, la gestion des champs File de type Image.

ARCHITECTURE

IBM mise aussi sur le nuage informatique

Le cloud computing est décidément à la mode. IBM vient d'ouvrir son premier laboratoire européen sur le sujet, à Dublin. Une des premières offres est le Idea Factory for Cloud Computing, un service capable de mettre les clients sur l'environnement cloud. Il utilise les possibilités du web 2. IBM compte étendre progressivement les services autour du nuage dans les prochains mois. De quoi offrir un nouveau champ de bataille !

OUTILS

IONA passe à WCF

Pour assurer l'interopérabilité avec le monde Microsoft, IONA implémente dans la gamme Artix : WCF (Windows Communication Foundation). L'outil se nomme Artix Connect pour WCF. Les entreprises peuvent à moindre coût intégrer leurs applications Microsoft avec des plates-formes non Microsoft telles que BEA, IBM, Oracle, TIBCO ainsi que les applications basées sur CORBA.

sEn tant que Gold Certified Partner, IONA est reconnu comme un fournisseur leader de solutions d'intégration et d'interopérabilité pour les entreprises utilisant des applications stratégiques sur des plates-formes Microsoft. L'initiative stratégique IONA pour l'interopérabilité Microsoft inclut Artix for Integration, Artix Mainframe, et désormais Artix Connect pour WCF, la dernière addition à Artix d'IONA, la famille de produits d'infrastructure SOA avancée.

PROCESSEUR

AMD joue aussi avec Eclipse

Le fondeur AMD vient de lancer un plug-in Eclipse (open source) pour améliorer la gestion des performances et la surveillance du code Java sur ses processeurs. Appelé CodeSleuth, ce module inclut des fonctions d'AMD CodeAnalyst Performance Analyzer. On peut ainsi auditer son code source Java sur du quadri cœur. C'est un outil indispensable pour le développeur souhaitant exploiter au mieux le multi cœur et comprendre finement le fonctionnement de son application multithreadée.

MySQL User Conference 2008

Sun va faire évoluer MySQL vers le " cloud computing "

Plus de 2000 personnes ont assisté à la grand-messe annuelle de MySQL – MySQL User Conference - la semaine dernière à Santa Clara en Californie. Désormais responsable de l'offre bases de données de Sun, Marten Mickos a ouvert la session inaugurale par un triomphant : " notre acquisition par Sun est un vote de confirmation de la plate-forme LAMP à 1 milliard de dollars ".

Session de la Tutorial Day...

Durant l'événement MySQL, la version 5.1 a été annoncée (disponibilité de la version finale d'ici fin juin). Cette version apporte plusieurs améliorations et nouveautés : partionnement étendu à 5 formes (dont index et table), deux nouveaux types de réplications (basés sur les rangées et la réplication hybride), un scheduler pour automatiser les tâches courantes du serveur de données. MySQL 5.1 sera disponible en édition communauty (gratuite et téléchargement), Enterprise Server et Embedded Server. Ces fonctionnalités ont pour objectif essentiel d'améliorer les performances et la disponibilité des bases.

Les DBA disposent également d'un nouvel outil de modélisation et de gestion de base de données : MySQL Workbench. Basé sur le projet open source DBDesigner 4, cet outil permet de modéliser, créer, et suivre les changements de structure de la base de données. *" Il génère en outre la documentation et possède une fonctionnalité très intéressante de synchronisation dans les deux sens "* ajoute Michael Zinner, développeur de l'outil.

Mais c'est surtout du côté des développeurs que les informations les plus excitantes circulaient. Marten Mickos a par exemple affirmé haut et fort que *" MySQL s'intéresse à l'accès aux données en dehors des requêtes SQL et à la vision (data) base as a service ".* Et comme par hasard, le CTO d'Amazon, Wermer Vogels, a passé la moitié de la session inaugurale à expliquer comment le premier site de commerce dans le monde – 4 milliards de requêtes SQL par jour - a construit et utilise S3 et SimpleDB. *" Nous utilisons S3 pour les accès basés sur des clés primaires et SimpleDB pour les requêtes plus complexes "* a-t-il dévoilé. À noter que Microsoft propose déjà un langage pour accéder à des données sans passer par du SQL : LINQ (en standard avec .Net 3.5).

Nul doute que la place accordée à Amazon lors du keynote vise à préparer les utilisateurs de MySQL à de nouveaux moteurs de stockage. *" Je vous conseille de jeter un coup d'œil à MySQL Storage Engine for AWS S3 "* a conseillé Marten Mickos. Peu de concret mis à part MySQL 5.1, mais une tendance forte : le développement d'une nouvelle couche intermédiaire d'accès aux données. Tout laisse donc à croire que MySQL proposera bientôt un moteur optimisé (gestion d'état, cache, etc.) pour le " cloud computing ". " Associé au model open source, le cloud computing est un changement majeur. L'informatique est sur le point de devenir une commodité au même titre que l'électricité " a expliqué Jonathan Schartz, CEO de Sun. On attend donc de pied ferme les nouvelles API et outils de MySQL dans ce domaine. La plate-forme LAMP est en effet en retard sur Microsoft qui propose désormais LINQ (Language Integrated Query) en version de production. Sun distribuant déjà son propre stack LAMP, on peut imaginer que ce type de fonctionnalités avancées feront partie d'un " premium ".

Carole Buret

MySQL sur une puce

L'appliance de Kickfire est un serveur bi-Xeon E5540 (2.8GHz - 4 cœurs) qui embarque 64 Go de RAM pour le moteur de requête et 8 disques SAS de 73 Go (15.000 tours minute). Elle a battu le benchmark international TPC-H à 0,7 dollars pour une heure de requêtes. Sun compte sur son partenaire pour faciliter la vie des développeurs : serveur pré-déployé, optimisé, paramétré, etc.

Agenda

MAI

Les 5 et 6 mai 2008, FIAP Jean Monnet - 30 rue Cabanis,75014 Paris - **XP Day France 2008** - La conférence agile sur les méthodes agiles ! http://www.xpday.fr/

13 mai, Paris : **Paris JUG** organise deux présentations : **Productivité des développements Java** - par Guillaume Duquesnay et **Maven 2** - par Arnaud Heritier site : http://www.parisjug.org/next Meetings.html

20 mai, Paris : **Microsoft Mobility Briefing 2008.** Journée consacré à la mobilité en entreprise autour des solutions Windows Mobile. Site : http://www.microsoft.com/france/windowsmobile/mobilitybriefing2008/

Le 21 Mai 2008, Paris, Salons Amphi Pereire, 75017. Intel et Micro Sigma organisent, dans le contexte de **la Tournée Européenne Intel Outils** pour le Développement sur Multi-Core, une **journée sur l'utilisation des outils de développement Intel**. www.microsigma.fr

Le 22 Mai, Lille, Euralille, Espace congrès - Seconde édition des **WygDay**, un évènement gratuit pour tous les professionnels des NTIC et du développement Informatique. http://wygday.wygwam.com/

Le 27 mai 2008, Paris, Porte Maillot, Pavillon d'Armenonville - **Séminaire Power Builder**, coup de projecteur sur les évolutions futures et les nouveautés de PowerBuilder 11.5 www.sybase.fr

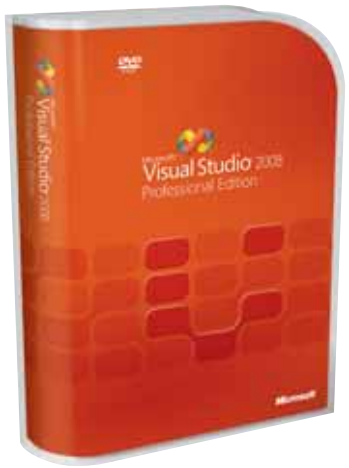
Les 28 & 29 mai, Paris Porte de Versailles : **l'événement de l'information numérique et de la gestion des connaissances** avec i-expo, KM Forum, i-medias et Online. www.i-expo.net

Recevez la Newsletter www.programmez.com/inscription.php

Microsoft®

COMSOFT direct
Bechtle's Software Specialist

SOS DEVELOPERS



Achetez un des produits de la gamme Microsoft Visual Studio 2008 Professional et recevez gratuitement un DVD d'auto-formation contenant plus de 280 heures de formation.

VALABLE POUR L'ACHAT DE :

- Visual Studio 2008 Professional
- Visual Studio 2008 Professional avec abonnement MSDN
- Visual Studio 2008 Team System (Architect, Developer, Database, Test)
- Visual Studio 2008 Team Suite

Microsoft GOLD CERTIFIED Partner

* Valable du 1er mai au 27 juin 2008. Achat en boîte ou licence, en produit complet ou mise à jour. 1 DVD OFFERT par licence

Plus d'informations sur www.sosdevelopers.com/msdn.htm

Tél. : 04 89 87 22 45 • msdn@sosdevelopers.com

EclipseCon 2008 : en route vers Eclipse 3.4 et 4.0

Comme chaque année, le grand événement Eclipse s'est tenu fin Mars à Santa Clara - en plein cœur de la Silicon Valley - et regroupait environ 1500 personnes. Cet évènement est la meilleure occasion pour prendre le pouls de cette communauté et appréhender les évènements majeurs autour d'Eclipse.

EclipseCon, c'est d'abord l'occasion de rencontrer les autres contributeurs et d'échanger avec eux. Ainsi les évènements "sociaux" se sont une nouvelle fois multipliés cette année avec des réceptions tous les soirs, des sprints de codage et d'expérimentation ou encore des tournois sportifs.

Les *commiters* issus des différents projets de haut niveau d'Eclipse, abordant des domaines tout à fait différents, peuvent échanger idées et expériences, créant ainsi une meilleure synergie entre les projets.

Les aspects liés à l'ouverture et à la communauté ont donné lieu cette année à des conférences, notamment celle sur la diversité au sein des projets, donnée par Ed Merks et Chris Aniszczyk, deux des contributeurs les plus reconnus au sein d'Eclipse (l'un leader du projet Modeling et l'autre membre de l'équipe du PDE).

Vers Eclipse 3.4

Chaque année depuis 3 ans, la fondation Eclipse définit une livraison simultanée d'un grand nombre de plug-in. ceci permet de s'assurer que ces plug in coopèrent bien, et facilite grandement la communication autour d'un projet, avec une date évènement tous les ans en Juin. Ganymède, nom de code de la prochaine version d'Eclipse, était évidemment très présent cette année à EclipseCon avec 24 projets qui seront livrés simultanément.

Ganymède proposera, par le biais des projets WTP et STP, un excellent support des applications web, des serveurs d'applications ainsi que des architectures à base de composants.

Le modeleur SCA agrégeant les concepts de services pour SOA et d'architectures à base de composants a été extrêmement bien accueilli lors de sa présentation par Stéphane Drapeau, leader du projet Eclipse SCA.

A noter également que Ganymède sera la première version d'Eclipse à fournir un support Subversion "out of the box" au même titre que pour CVS. Enfin, le système d'installation et de mise à jour des plug-in a été entièrement remis à plat et ré-écrit : il est désormais beaucoup plus simple pour une organisation de définir son propre dépôt de plug-in, partagé par les développeurs.

Plate-forme

L'an passé était profondément axé sur OSGi. Cet ensemble de spécifications au cœur de l'architecture d'Eclipse est désormais utilisé dans de nombreux autres environnements. Ainsi, l'année 2008 a vu de nombreuses initiatives de la part de la plupart des grands fournisseurs de serveurs d'applications pour proposer des conteneurs OSGi. EclipseCon a été l'occasion cette année encore de voir qu'OSGi est en excellente santé: son côté pragmatique et léger séduit et de nombreuses conférences traitaient de son utilisation au sein de projets d'informatique de gestion ou embarquée. Parmi les nouveautés à venir dans Eclipse 3.4, on peut citer par exemple l'"API Tooling" qui assiste à la gestion de l'API fournie par un plug-in. Cet outillage étant très prometteur, la communauté va soutenir l'effort en fournissant pour chaque composant d'Eclipse les fichiers de description d'API, si possible dès la sortie de Ganymède en Juin.

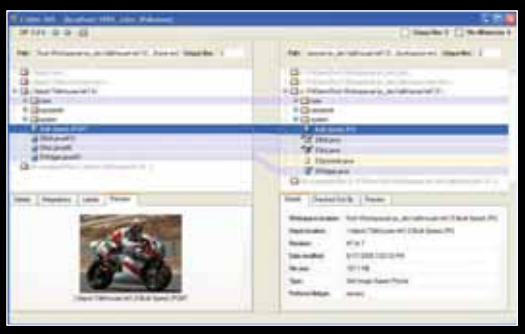
Modélisation

Avec 6 projets constitués de nombreux composants, le domaine de la modélisation est extrêmement bien représenté dans Ganymède. EclipseCon a été le reflet de ce dynamisme avec de nombreuses conférences traitant de sujets tels

Perforce | Gestion *rapide* de configuration logicielle (GCL)

La fonctionalité Folder Diff,

un atout de productivité du système de GCL Perforce.



Folder Diff de Perforce

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

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

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

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

PERFORCE
SOFTWARE

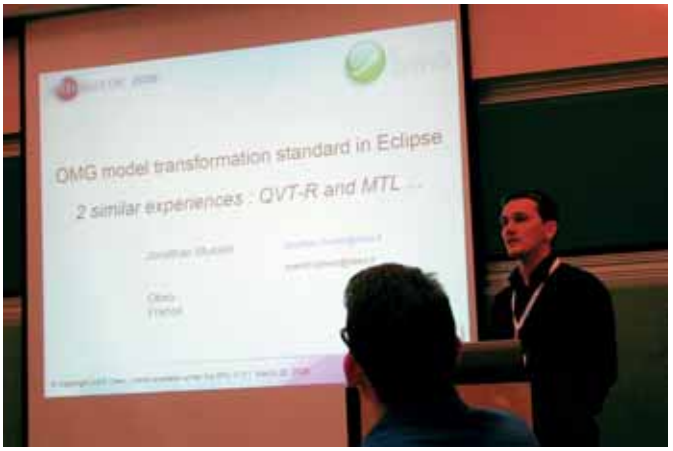
Téléchargez sans conditions une copie gratuite de Perforce sur www.perforce.com. Vous bénéficiez d'une assistance technique gratuite pendant votre période d'évaluation.

que la transformation modèle vers modèle ou modèle vers texte. Un nombre important de tutoriaux présentaient l'utilisation de ces outils; ATL par exemple a clairement démontré qu'il s'agissait de la technologie la plus efficace et simple d'accès pour transformer des modèles. L'effort d'industrialisation de cet outil par Obeo facilite grandement son accès et son utilisation grâce à la complétion ou aux assistants.

Les composants au cœur du "modeling" ne sont pas en reste avec pour EMF des nouveautés alléchantes telles que la gestion des content-types, des méta-modèles extensibles et un nouvel outillage dédié à la modélisation graphique des méta-modèles : "Ecore Tools". Ma propre conférence sur EMF Compare a été chaleureusement accueillie, ce composant comblant un manque important à l'heure actuelle dans le domaine de l'open source : la comparaison et le *merge* de tout type de modèle.

Eclipse et l'OMG

Pour la première fois, une séance de discussions sur les standards de l'OMG était organisée dans le cadre de cette conférence. Eclipse est un membre de l'OMG, mais l'objectif était d'identifier les domaines dans lesquels la collaboration pourrait encore être améliorée à l'avenir. La liste des projets Eclipse ayant un lien avec les spécifications OMG est longue. Les problèmes d'alignement entre les normes actuelles et les logiciels mis en œuvre ont été largement débattus, sur des domaines aussi variés que MOF, UML, QVT/MTL... Une partie du processus d'approbation d'une spécification OMG implique l'existence d'implémentations. L'objectif d'Eclipse est d'avoir des implémentations de référence des normes, et celui de l'OMG, d'accélérer leur adoption. A sa façon, Eclipse contribue grandement à la validation de l'intégrité des spécifications car les équipes projets mettent en évidence les problèmes. Lors de ce symposium, les intervenants ont appuyé leurs propos par de nombreux exemples d'inconsistance et d'ambiguïté.



Il a également été remarqué que faire modifier une spécification n'est pas chose aisée. Par exemple, dans le cas de l'implémentation de la norme de génération de code à base de template "MTL", le rapprochement des contributeurs des horizons Eclipse et OMG a été, est, et sera essentiel.

Microsoft et Eclipse

Les rumeurs étaient nombreuses suite à l'annonce faite par Sam Ramji, lors d'une keynote, le mercredi matin. En effet Eclipse est désormais un compétiteur important de Visual Studio avec le projet CDT, permettant le développement d'applications C et C++, qui gagne de plus en plus d'utilisateurs. Après avoir annoncé le rachat de la fondation Eclipse par Microsoft et la fusion d'Eclipse et de Visual Studio sous le nom de "Supernova", Sam Ramji a plus sérieusement évoqué les travaux menés par le "Open Source Lab" de Microsoft pour améliorer l'interopérabilité des outils open source avec les produits et les plates-formes de Microsoft.

Comment Eclipse et Microsoft vont pouvoir travailler ensemble ? Le projet Higgins au sein d'Eclipse, dédié à la gestion d'identité sur le web, supportera désormais la technologie CardSpace. Par ailleur, Microsoft souhaite affecter des ressources pour aider l'équipe "Platform" pour le support SWT sur "Windows Presentation Foundation" de Windows Vista. On est donc désormais en droit d'espérer une meilleure intégration d'Eclipse avec ce système d'exploitation. Malheureusement, Microsoft ne souhaite pas pour le moment affecter des ressources en tant que *commiter* Eclipse à proprement parler, mais plutôt collaborer avec les équipes existantes comme il a pu faire dans d'autres projets Open Source.

Disons qu'il ne s'agit que d'une première étape mais que cette intervention a été bien accueillie par la communauté des développeurs Eclipse.

En route vers Eclipse 4.0

Sujet sensible que celui de l'avenir d'Eclipse dans les 5 prochaines années. L'équipe "platform" a montré comment elle souhaitait adresser les problématiques liées à la convergence des applications vers le Web et des architectures REST. L'environnement de développement va offrir plus de dynamique en intégrant un support JavaScript et la prise en compte à chaud des changements effectués sur des objets d'interface utilisateur.

Les applications Eclipse et RCP pourront être déployées directement en tant qu'applications Web, fournissant la même richesse d'interaction avec l'utilisateur que lorsqu'elles sont déployées en tant que client riche à la manière de ce qu'a commencé à faire le projet RAP.

Une partie de la communauté reste cependant déçue par ces propositions car le "tout web" convainc difficilement.

Par exemple, pour les membres du projet CDT, qui vise à offrir un outillage pour le développement C et C++, Eclipse 4 est plus l'occasion de remettre à plat l'architecture pour améliorer les performances générales de l'IDE.

Les premières versions d'Eclipse 4 sont annoncées pour dans deux ans. A noter toutefois qu'Eclipse 3 existera encore pour au moins 5 ans, les grands contributeurs d'Eclipse (IBM par exemple) ayant entièrement fondé leurs produits commerciaux sur ces bases. Cette plate-forme restera donc pérenne et maintenue avec un grand nombre de backports des fonctionnalités d'Eclipse 4 vers Eclipse 3.x.

eclipseCON 2008

Le site de la conférence Eclipse Con 2008 : http://www.eclipsecon.org/2008

■ **Cédric Brun,** *leader du projet EMF Compare - Obeo (http://www.obeo.fr)*

NetAdvantage® for .NET

NetAdvantage pour Windows Forms

Des composants haute performance pour interfaces utilisateur. Une expérience supérieure pour vos utilisateurs.

Des composants interfaces utilisateur de haute performance
Utilisez une boite à outils complète pour Windows Forms et ASP.NET propulsant vos applications commerciales demandant une performance rapide et de qualité

Compatibilité Visual Studio®
Utilisez la version de Visual Studio qui vous convient – VS 2005 ou bien VS 2008 avec un support complet et continu pour NetAdvantage

Une expérience Microsoft Office® conséquente
D'Office 2007 au ruban en passant par les dernières innovations Windows® Vista® et Breadcrumb Navigation, nos composants transforment vos applications avec des expériences utilisateur prêtes à l'emploi

Une apparence Windows Vista
Les styles Vista et breadcrumb navigation offrent à vos applications Windows Forms le look and feel des expériences utilisateur récentes de Microsoft

Concevez des tableaux de bord professionnels
Donnez de la vie à vos Key Performance Indicators avec des graphiques et gauges de haute fidélité pour Windows Forms, renforcés par des grilles splendides

Design Once, Style Everywhere
Donnez un style professionnel instantanément à vos applications conçues avec NetAdvantage afin de leur donner un style Office® 2007, XP, Vista® ou bien créez et appliquez vos normes corporatives à vos composants et à vos applications

Pour de plus amples informations: infragistics.com/dotnet

Infragistics®
Powering The Presentation Layer

N° Vert 0800 667 307

sales-europe@infragistics.com

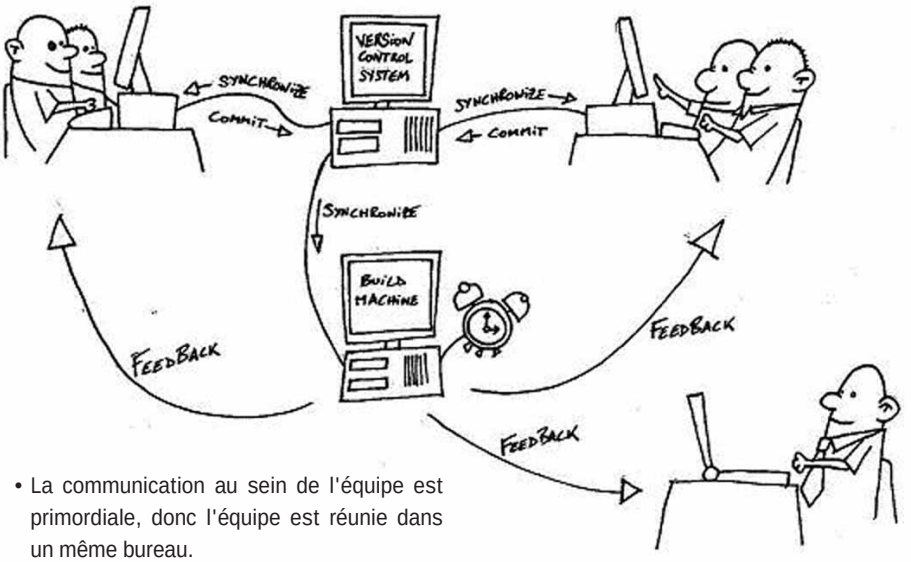
WINDOWS FORMS ASP.NET WPF JSF

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

Copyright 1996-2008 Infragistics, Inc. All rights reserved. Infragistics, the Infragistics logo and NetAdvantage are registered trademarks of Infragistics, Inc. Aikido is a trademark of Infragistics, Inc. All other trademarks or registered trademarks are the respective property of their owners.

Appréhender l'eXtreme Programming

L'eXtreme Programming (XP) est une méthode de développement logiciel mise au point par Kent Beck, Ward Cunningham et Ron Jeffries à la fin des années 90.



Elle recherche l'efficacité maximale en concentrant l'effort de travail sur l'objectif de développer vite et juste. "Développer juste", autrement dit, *bien développer* le *bon logiciel*. La démarche est légère, pragmatique, disciplinée, empirique et adaptative. XP fait partie de la famille des méthodes agiles de développement logiciel. La plupart des livres, articles et sites Internet font référence à la première édition de XP. C'est pourquoi cet article se base sur la seconde édition, décrite par Kent Beck en 2005. Cette dernière est surtout une reformulation complète de la première. Les pratiques sont découpées plus finement et complétées par de nouvelles. L'auteur justifie aussi la discipline en décrivant la philosophie et les motivations qui sont les racines de la démarche.

Pourquoi "eXtreme" ?

Afin de viser une efficacité maximale en développement, XP identifie un jeu de bonnes pratiques, les combine et pousse leur application à l'extrême. Par exemple,

- La revue de code est une bonne pratique, elle est donc menée en continu ;
- Les tests sont primordiaux, ils sont donc écrits systématiquement avant d'implémenter et rejoués en continu ;
- La conception est importante, elle est donc menée et remaniée tout au long du projet ;
- La simplicité permet d'avancer plus vite, ainsi la solution viable la plus simple est toujours retenue ;
- L'intégration des modifications est cruciale, elle est donc réalisée plusieurs fois par jour;
- Les besoins évoluent vite, donc l'équipe réalise des cycles de développement très rapides pour s'adapter au changement.
- La communication avec le client est essentielle, donc le client fait partie de l'équipe de développement.
- La communication au sein de l'équipe est primordiale, donc l'équipe est réunie dans un même bureau.

De plus, XP propose un pragmatisme extrême. Les pratiques qui apportent de la valeur au client doivent être appliquées. Le reste doit être amélioré et prouver son efficacité ou être supprimé en tant que gaspillage.

Mais où est l'originalité ?

Les pratiques d'XP sont connues et appliquées depuis longtemps par certains développeurs. L'originalité réside dans le fait de toutes les combiner et de pousser leur application à l'extrême. Aussi, la méthode accorde une très grande importance aux aspects humains du développement logiciel. Enfin, la méthode est ancrée sur une vraie philosophie de développement décrite en termes de valeurs, principes et pratiques.

Quelles sont les valeurs ?

XP affirme que l'efficacité maximale peut être atteinte si le développement respecte cinq valeurs: communication, retour d'information, simplicité, courage et respect. La **communication** est ce qui importe le plus au sein d'une équipe de développement. Les problèmes rencontrés lors d'un projet sont souvent dus à des manques d'échange. Alors, la méthode préconise une communication intense et en continu au sein de l'équipe y compris avec le client. La **simplicité** permet d'éviter les gaspillages et de rendre le logiciel souple, tolérant aux modifications. Ainsi, XP préconise de toujours choisir la solution la plus simple qui soit viable. Le changement est inévitable et continu lors d'un développement logiciel. Plutôt que de gaspiller de l'énergie à contenir le changement, XP préconise de s'y adapter. Un **retour d'information** rapide et continu sur le développement permet alors de piloter le travail de manière réactive et de l'adapter au changement. Il faut du **courage** pour surmonter la peur de développer, changer sa manière de travailler, communiquer de manière transparente et confronter ses points de vues au retour d'information. Enfin, le courage de l'équipe de développement mérite le **respect** de tous les intervenants du projet. De plus, un travail d'équipe efficace nécessite le respect mutuel des membres. Les valeurs sont universelles et donc difficiles à traduire en pratiques. Par contre, les principes sont spécifiques au domaine. Ils aident à déterminer les pratiques concrètes à appliquer en regard des valeurs.

Quels sont les principes ?

Humanité *(Humanity)*
Les logiciels sont développés par des personnes. Il faut en tenir compte dans l'intérêt de l'équipe et de l'entreprise.

Economie *(Economics)*
Tout travail doit apporter de la valeur au client et au fournisseur.

Bénéfice partagé *(Mutual benefit)*
Toute activité doit bénéficier à toute personne impliquée.

Similitude *(Self-similarity)*
Une solution efficace peut être adaptée à différents contextes, mais sans garantie de succès.

Amélioration *(Improvement)*
Il faut constamment rechercher à s'améliorer.

Diversité *(Diversity)*

Il n'existe pas une unique et meilleure manière de procéder. C'est pourquoi l'équipe doit regrouper toutes les différentes compétences permettant le succès du projet et les membres doivent confronter leurs points de vues.

Réflexion *(Reflection)*
Une équipe efficace réfléchit à sa manière de travailler et apprend de ses inévitables erreurs.

Flux *(Flow)*
Il faut livrer un logiciel opérationnel en flux continu et non en phases séquentielles.

Opportunité *(Opportunity)*
Les problèmes doivent être considérés comme des opportunités d'apprendre et de s'améliorer.

Redondance *(Redundancy)*
Les problèmes critiques et difficiles doivent être résolus simultanément de différentes manières afin d'assurer une qualité maximale.

Echec *(Failure)*
Il vaut mieux échouer, apprendre et s'améliorer que ne pas avancer.

Qualité *(Quality)*
On ne pilote pas un développement en jouant sur la qualité du logiciel. Les projets n'avancent pas plus rapidement en abaissant le niveau de qualité. Aussi, les projets n'avancent pas plus lentement en imposant un niveau de qualité exigeant.

Petits pas *(Baby steps)*
Il faut développer par petits pas, par simplicité et pour obtenir rapidement et continuellement des retours d'information.

Responsabilité acceptée *(Accepted Responsability)* On ne peut imposer la responsabilité. Elle doit être acceptée.

Pour respecter les cinq valeurs, XP propose d'appliquer un ensemble cohérent de pratiques guidées par les principes.

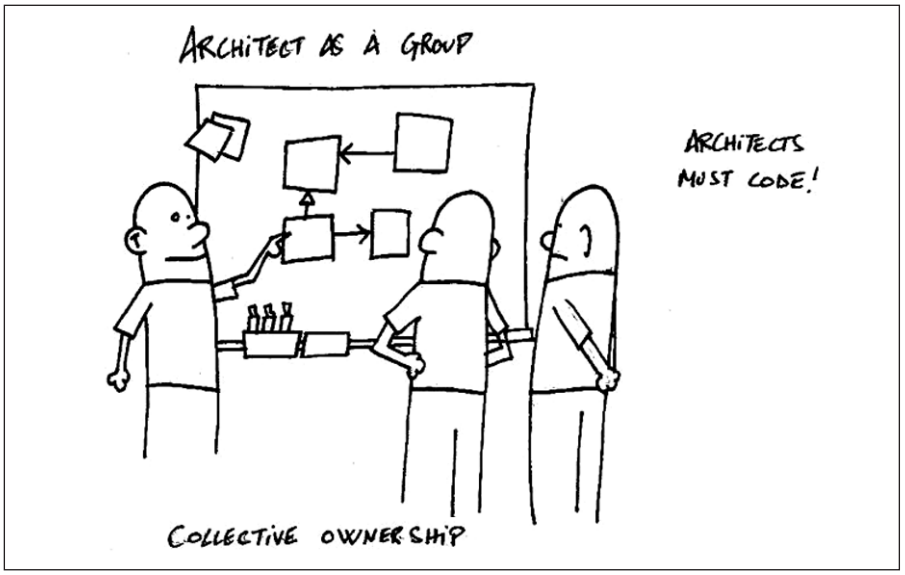
Quelles sont les pratiques ?

Il s'agit de la manière de travailler au jour le jour. Chaque pratique est applicable telle quelle. Néanmoins, elles peuvent aussi être considérées comme idéales et donc être adaptées à l'environnement concret du projet.

XP propose en premier un jeu de pratiques principales qui permettent de se mettre à XP.

Travail en binôme *(Pair programming)*
Tout code est écrit à deux développeurs devant un même poste de travail afin de revoir le code en continu et de privilégier la communication. Les paires permutent. Cette organisation favorise la communication, le retour d'information et le respect.

Espace commun de travail *(Sit together)*
L'équipe partage un même bureau afin de privilégier la communication orale face à face et le travail en binôme.



Equipe complète *(Whole team)*
Afin d'accélérer la communication, l'équipe regroupe toutes les personnes dont les compétences permettent le succès du projet.

Management visuel *(Informative workspace)*
L'avancement et les difficultés du projet sont affichés dans l'espace de travail afin de faciliter la communication et le retour d'information.

Travail dynamique *(Energized work)*
Le rythme de travail doit être durable à long terme. De plus, la discipline de travail doit se plier aux contraintes du bureau partagé, du travail d'équipe et en binôme tout en respectant un espace personnel.

Scenarios *(Stories)*
La planification du travail est basée sur la quantification de fonctionnalités. Les fonctionnalités sont décrites en scénarios d'utilisation pertinents pour le client. L'objectif est de faciliter la communication entre client et développeurs.

Cycle hebdomadaire *(Weekly cycle)*
Le travail est planifié à la semaine. Un incrément de fonctionnalité est produit et livré en une itération d'une semaine. Ce cycle court favorise le retour d'information rapide et continue.

Cycle trimestriel *(Quarterly cycle)*
Le travail est planifié au trimestre.
Ce cycle long permet de prendre du recul, de réfléchir à l'amélioration continue des pratiques et de traiter les goulots d'étranglement du flux de travail.

Du mou *(Slack)*
Il vaut mieux réduire le périmètre des engagements plutôt que de ne pas les tenir. Ceci permet de tenir un rythme durable et de maintenir un climat de confiance et de respect.

Construction en dix minutes *(Ten-minute build)*
Le système entier doit être automatiquement construit et testé en dix minutes. Cette pratique assure un retour rapide d'information.

Intégration continue (*Continuous integration*)
L'équipe entretient tôt et toujours une version testée et opérationnelle du système qui contient les toutes dernières modifications. Cette pratique assure également un retour rapide d'information.

Tester d'abord *(Test-first programming)*
Il faut écrire un test qui échoue avant toute modification de code. Ensuite, il faut changer le code pour faire passer le test. Ceci fournit un retour d'information très rapide sur la pertinence du code. Cela permet également d'écrire un code simple.

Conception incrémentale *(Incremental design)*
Il faut concevoir progressivement et continuellement améliorer et simplifier la conception. Ceci permet d'obtenir au plus vite un retour d'information sur les choix et de maintenir une conception simple et donc évolutive.

Ensuite, la méthode propose d'appliquer les pratiques corollaires une fois que les pratiques principales ont contribué à améliorer l'efficacité du développement.

Réelle implication du client *(Real customer involvement)*
Afin de favoriser la communication, le retour d'information et le respect mutuel, le client doit s'impliquer dans le développement, notamment en faisant partie de l'équipe, en partageant son espace de travail, en menant la planification, en fournissant des scénarios d'utilisation et des jeux de tests de recette.

Déploiement incrémental *(Incremental deployment)*
Afin d'obtenir un retour d'information et d'ap-

porter au plus tôt de la valeur au client, il faut déployer en production le système par incréments tout au long du projet.

Equipe stable *(Team continuity)*
Afin d'assurer la communication, il faut conserver les équipes efficaces.

Réduire progressivement les équipes *(Shrinking teams)*
Au fur et à mesure que l'équipe s'améliore et gagne en efficacité, on peut réduire l'effectif pour former d'autres équipes. Ceci permet de communiquer la méthode au sein d'une organisation.

Analyse cause-racine *(Root-cause analysis)*
Lorsqu'un défaut est détecté, il faut corriger la cause du problème afin que ce type d'erreur ne soit plus commis.

Code partagé *(Shared code)*
N'importe qui dans l'équipe peut améliorer n'importe quel code du projet. L'amélioration continue du code est toujours une priorité. Cette pratique permet d'entretenir une implémentation simple et de communiquer à tous l'intention de tout le code.

Code et tests *(Code and tests)*
Afin d'organiser le travail de la manière la plus simple possible, il ne faut maintenir que le code et les tests. Tous les autres produits du développement doivent être générés à partir du code et des tests.

Base de code unique *(Single code base)*
Afin de favoriser la simplicité, il faut à tout prix éviter de gérer plusieurs versions du logiciel.

Déploiement quotidien *(Daily deployment)*
Pour obtenir des retours d'information, il faut déployer en production le logiciel développé dans la journée.

Contrat à contour négocié *(Negociated scope contract)*
Il faut tenir les objectifs de délai, de coût et de qualité en négociant le contour fonctionnel du logiciel. Les contrats doivent être adaptés à cette démarche.

Paiement à l'utilisation *(Pay-per-use)*
Il faut privilégier un paiement du fournisseur à l'utilisation du logiciel plutôt qu'un paiement aux livraisons. Ceci permet d'organiser le développement de manière à maximiser la valeur apportée au client.

Quel est le cycle de développement ?

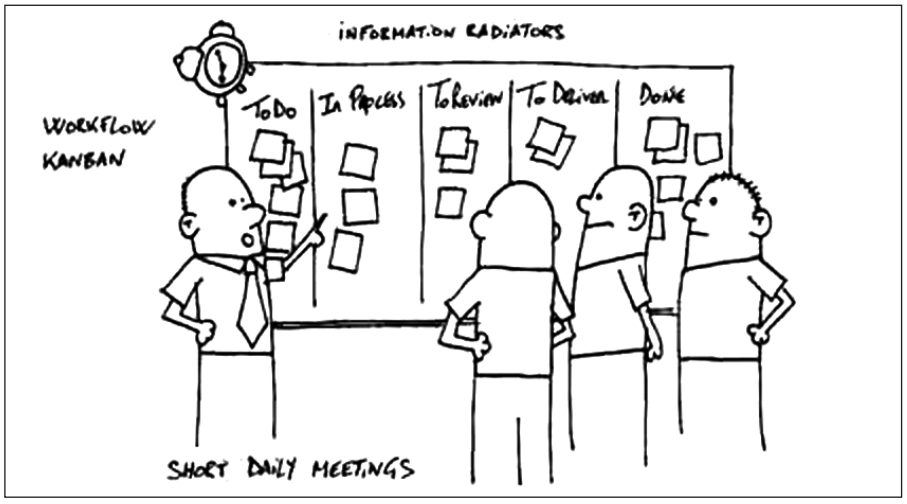
Le cycle de développement est itératif et incrémental. Une itération débute par une réunion de planification à laquelle participe le client et les développeurs. Le client décide des fonctionnalités prioritaires. Elles sont décrites sous la forme de scénarios d'utilisation. Les développeurs découpent les scénarios en tâches qu'ils quantifient. Avec cette mesure, l'équipe négocie l'incrément de fonctionnalité qu'elle s'engage à produire dans l'itération. Les tests automatisés d'acceptation sont écrits. Les binômes se forment et choisissent des tâches à implémenter. Lorsque les tests d'acception passent, le logiciel enrichi du nouvel incrément est livré.

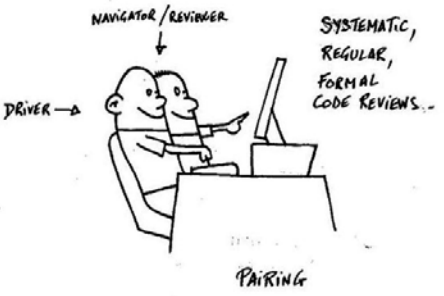
En quoi XP aide à développer le bon logiciel ?

Le client est impliqué tôt et régulièrement dans le développement. Les itérations fréquentes lui fournissent des versions fonctionnelles qui lui permettent d'affiner son besoin et de communiquer sa satisfaction. En quelque sorte, le développement est asservi sur le besoin et la satisfaction du client.

En quoi XP aide à bien développer le logiciel ?

Les relectures continues en binôme, les tests systématiques, le remaniement et le partage





du code assurent qu'un haut niveau de qualité est maintenu.

En quoi XP aide à développer vite ?

L'équipe ne réalise que des activités qui apportent de la valeur au client. L'intégration continue instaure un rythme soutenu de développement en équipe et permet de corriger tôt les dysfonctionnements liés au travail en parallèle. Enfin, le gaspillage est aussi et surtout évité en développant bien le bon logiciel.

Et l'homme dans tout cela ?

XP a l'originalité d'accorder une importance primordiale à l'homme dans le développement logiciel. En effet, la méthode

- accorde à l'équipe la fierté du travail accompli et bien fait en instaurant des livraisons fonctionnelles fréquentes et en exigeant un haut niveau de qualité ;
- stimule la solidarité et l'esprit d'équipe en regroupant les développeurs et systématisant le travail en binôme ;
- assure la formation continue et le partage des compétences dans le cadre du travail en binôme et les équipes pluridisciplinaires ;
- supprime la peur de développer et accorde le droit à l'erreur en systématisant les petits pas, le test, l'intégration continue et le retour rapide d'information ;
- exige un rythme soutenable pour l'équipe en limitant la durée de travail et en recommandant la réduction négociée de contour fonctionnel plutôt que le sprint ;
- instaure des rapports sains entre client et fournisseur en créant une équipe intégrée avec communication transparente ;
- accorde à l'équipe la liberté de s'organiser elle-même et d'améliorer ses propres pratiques ;
- respecte le travail en supprimant le gaspillage par un cycle de développement en flux tendu.

Comment se mettre à XP ?

La progression pour transiter vers XP dépend de l'appétit de changement de l'équipe. La méthode conseille de constater les bénéfices des pratiques principales avant de s'attaquer

Software Technology Resources

Les auteurs les plus réputés nous font confiance ...

Microsoft® Visual Studio® 2008

OFFRE SPECIALE *

- **l'outil de développement unique pour toute technologie** Microsoft (Web, Windows, Mobile ou basée sur Office)
- **Développez pour toutes les versions du framework .NET** du 2.0 au 3.5
- **Améliorez l'expérienceutilisateur sur vos développements** en tirant profit de nouvelles interfacesdisponibles (Ajax, WPF, Vista, Office 2007, Windows Mobile ...)

Et avec l'abonnement MSDN : disposez de toutes les versions de Visual Studio et des outils indispensables de prototypage et de développement.

En cadeau le DVD de ressources

280 heures d'auto-formation

Apprenez avec les Coachs MSDN

Le coach Développement Web

Le caoch Visual C++

Le coach Team System

Le coach Visual Basic

Le coach Visual C#

La saga .NET en détail par ses experts

Appeler pour toute information et pour commander

N° Indigo 0 820 209 096

La ligne Directe Microsoft, une exclusivité STR

STR renforce ses équipes techniques et commerciales Microsoft

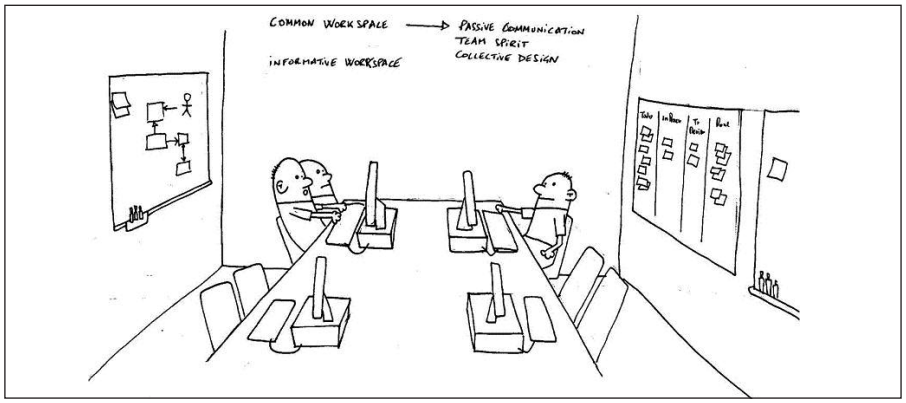
Consultants, Commerciaux, Techniciens, Débutants ou Confirmés
CDI et Stagiaires ***Prenez une place dans notre avenir***

S.T.R vous accompagne dans l'obtention des certifications MS techniques et commerciales

STR RECRUTE 01 3070 6161 ou sur www.str.fr

* Offre à durée limitée, à concurrence des quantités disponibles, non cummulable avec d'autres remises ou rabais - Valable pour toutes les licences Visual Studio sauf Standard

Informations et commandes Tél : 01 30 70 61 61 Fax : 01 39 46 38 64
www.str.fr



pleinement aux pratiques corollaires. On peut commencer par montrer l'exemple en appliquant soi-même certaines pratiques et en s'appuyant sur l'expérience de la communauté des praticiens. Certaines équipes choisissent d'être guidées par un " coach ".

Faut-il tout pratiquer ?

Il n'est pas nécessaire de tout appliquer à la lettre pour être un praticien. La méthode est un guide pour s'améliorer, gagner en efficacité et apporter de la valeur au client. Néanmoins, certains recommandent d'expérimenter une pratique extrême de XP pour ressentir l'alchimie qui naît et pouvoir ensuite optimiser les pratiques à son environnement en connaissance de cause.

XP est-il applicable pour tout type de développement ?

La méthode est appliquée pour du développement Internet comme pour élaborer des logiciels temps-réel critiques embarqués pour l'avionique. Il s'agit d'adapter les pratiques à son contexte. Cette démarche est facilitée par la nature adaptative d'XP.

XP se limite t-il au développement de logiciels ?

Même si les valeurs de XP sont universelles, les principes et les pratiques sont spécifiques au développement logiciel. Par contre, XP a des répercussions en dehors de ce cadre. En effet, l'amélioration continue de l'efficacité peut déplacer le goulot d'étranglement du flux de travail hors du périmètre du développement logiciel, en amont ou en aval. Pour étendre XP, on peut alors s'inspirer de méthodes proches, comme le *Lean*.

XP est-il lié à la démarche orientée-objet ?

Un développement XP n'est pas nécessairement orienté-objet. Néanmoins, la méthode se base sur la construction incrémentale du logiciel, la tolérance aux changements et la testabilité. Or, ces objectifs sont plus aisément atteints par une pratique de l'encapsulation, de l'héritage et du polymorphisme.

Est-ce compatible avec le développement offshore ? Est-ce applicable pour de grandes équipes ?

Toute méthode de travail d'équipe souffre de la perte de communication et réactivité qui accompagne les grandes équipes et les développements multisites. Néanmoins, comme XP met l'accent sur la communication, la réactivité et la synchronisation fréquente, elle demeure efficace dans ces cadres. Les valeurs et principes restent pertinents. Il s'agit alors d'adapter les pratiques.

XP est-il un retour en arrière ?

Tout comme certaines des premières démarches de développement logiciel, XP est une méthode adaptative et empirique, qui parallélise les activités. Cela s'oppose aux méthodes qui ont suivi et qui se veulent théoriques et prédictives, et qui sérialisent les activités, comme le cycle en V. Ce n'est pas un retour au développement chaotique, mais plutôt un développement qui s'adapte au chaos intrinsèque à toute démarche de développement logiciel.

Perd-t-on en visibilité et en prédictibilité ?

L'estimation, la prévision et la planification théorique et massive en début de projet n'offrent qu'une illusion de visibilité et de prédictibilité. En mode XP, ces activités sont menées progressivement au fil des itérations à partir de retours d'information de plus en plus pertinents. Ainsi, une vraie visibilité et prédictibilité s'affinent avec le temps de manière empirique et fiable. Mais, il y a plus important: la méthode assure que le développement est asservi sur un parcours qui minimise le gaspillage et donc réduit la durée du projet.

La conception a-t-elle disparu ?

La conception massive en phase amont de l'implémentation est remplacée par une conception progressive et continuellement améliorée au fil des itérations. L'idée est de ne pas prendre d'hypothèse en avance de phase mais de concevoir en flux tendu le minimum suffisant pour les besoins actuels. L'évolutivité de la conception est assurée par sa simplicité et l'échafaudage de tests construits autour du code.

XP est-il compatible avec des démarches de qualité certifiantes ?

XP privilégie la communication informelle et l'investissement dans les tests et le code à une production de preuves documentaires, recherchées par les démarches de qualité certifiantes telles que ISO, CMMi, la DO178B ...
Ainsi, si une certification devient un objectif qui apporte de la valeur, il convient d'adapter la démarche XP afin de produire les preuves auditées. Bien sûr cela représente un supplément d'activité.

XP est-il sectaire ?

Parce que les équipes XP sont soudées et changent radicalement leur manière de travailler et certainement aussi à cause du nom de la méthode, certains non-praticiens se sentent exclus et caricaturent ceux qui l'appliquent. D'ailleurs, il existe un débat intéressant sur ce sujet: XP partagerait certains attributs avec les sectes ou la méthode serait plutôt un mouvement en croissance.

Pour en savoir plus

- Livre en anglais sur la 2e édition: Kent Beck, Extreme Programming Explained (ISBN 0-321-27865-8).
- Livre en français sur la 1re édition: Jean-Louis Bénard, Laurent Bossavit, Régis Médina et Dominic Williams, Gestion de projet eXtreme Programming (ISBN 2-212-11561-X)
- Groupe de discussion en français: http://fr.groups.yahoo.com/group/xp-france/

Emmanuel CHENU
emmanuel.chenu@fr.thalesgroup.com
http://emmanuelchenu.blogspot.com/

La première fabrique logicielle entièrement pilotée par les modèles

CodeFluent est un produit de génie logiciel qui permet
d'industrialiser la fabrication d'applications professionnelles manipulant des données sur la plate-forme .NET
en automatisant la création des composants à partir d'une modélisation de votre métier.

CODEFLUENT PROCURE LES BÉNÉFICES SUIVANTS

UN GAIN TRÈS SIGNIFICATIF DE PRODUCTIVITÉ
par l'automatisation des tâches répétitives et la mise en œuvre du modèle métier selon une architecture orientée services.

UNE MAINTENABILITÉ ET UNE QUALITÉ ACCRUES
grâce à l'approche intrinsèque de génération qui évite les erreurs et permet d'effectuer des mises à jour sur toutes les couches à l'aide d'une modification centralisée.

LA LIMITATION DU RISQUE
par la structuration du travail des développeurs autour d'une modélisation objet évolutive qui garantit une mise en œuvre selon les meilleures pratiques d'implémentation SOA sous plate-forme Microsoft.

L'ÉVOLUTIVITÉ DE L'APPLICATION
car la prise en charge de nouvelles technologies et de nouvelles versions de plate-forme est assurée par la mise à disposition régulière de nouveaux producteurs (Windows Presentation Foundation, Office Business Applications, mobilité).

LES OFFRES CODEFLUENT

Pour commander votre licence, envoyez un email à **sales@softfluent.com**

Model Driven pragmatique et Open Source

Cet article vous présente la mise en œuvre d'un ensemble d'outils et technologies Open Source qui vous permettront de rapidement mettre en œuvre un site Web "CRUD" (Create-Retrieve-Update-Delete) tout simple mais ... généré ! :-)...

Installation de l'environnement

- Installer Eclipse 3.4M4 (à télécharger sur http://download.eclipse.org/eclipse/downloads/)
- Installer, via le mécanisme de mise à jour automatique d'Eclipse (menu Help > Software Updates > Find and install…) : EMF, GMF, Ecoretools, openArchitectureWare dans leurs versions les plus récentes
- Installer GWT 1.4.x (http://code.google.com/webtoolkit/download .html)

Les outils utilisés

EMF : Modélisation

EMF, pour Eclipse Modeling Framework, est un projet Eclipse visant à apporter un ensemble d'outils pour modéliser des données, et les manipuler de manière générique. Incontournable dans l'univers Eclipse —un très grand nombre de projets Eclipse utilisent et/ou gravitent autour—, EMF inclut son propre métamodèle, Ecore (voir notre encadré), à partir duquel on va pouvoir décrire ses propres données.

Ecore et la métamodélisation

Pour décrire des données, nous avons besoin d'un langage nous permettant de décrire le type de ces données, les liens (association, contenance, ...) qui peuvent exister entre elles, etc. Pour cela, nous utilisons un langage de métamodélisation : Ecore. Ecore, est très orienté “code” (contrairement à UML ou XML Schema, qui peuvent aussi être vus comme des langages de métamodélisation), et on va pouvoir générer, à partir d'un modèle Ecore, tout une “plomberie” Prenons l'exemple d'un modèle assez simple, des classes A et B, un “A” ayant des “B”. Après avoir configuré les propriétés de génération (grâce à un modèle de génération, stocké dans un fichier d'extension genmodel), on va pouvoir obtenir, automatiquement :

- Une fabrique, MyModelFactory, nous permettant de créer programmatiquement de nouvelles instances de A ou de B, initialisés avec les valeurs par défaut que nous avons renseignées dans notre modèle,

```
A a1 = MyModelFactory.eINSTANCE.createA();
                                    // la fabrique est accessible via un
                                    // singleton, comme souvent dans EMF
```

- Les classes d'implémentation (AImpl et BImpl) correspondant aux deux beans modélisés. Elles implémentent un pattern Observeur/Observable. On peut donc être notifié de la modification d'un attribut d'un objet, de l'ajout d'un objet à une référence, etc. en abonnant un Adapter.

```
a1.setAtt1("a1 initial att1");
a1.eAdapters().add(new AdapterImpl() {
   @Override
   public void notifyChanged(Notification msg) {
      System.out.println("old val : " + msg.getOldValue()
                  + " / new val : " + msg.getNewValue());
   }
});
```

- Enfin EMF nous permet, en standard, de sérialiser/désérialiser une instance donnée de notre modèle dans un fichier XML.

```
      ResourceSet resourceSet = new ResourceSetImpl();
      resourceSet.getResourceFactoryRegistry().getExtensionToFactory
Map()
            .put(Resource.Factory.Registry.DEFAULT_EXTENSION,
                  new XMLResourceFactoryImpl());

      URI fileURI = URI.createFileURI(new File("/tmp/mon_fichier.xml")
.getAbsolutePath());

      Resource resource = resourceSet.createResource(fileURI);
      resource.getContents().add(a1);

      try {
         resource.save(Collections.EMPTY_MAP);
      } catch (IOException ex) {
         // exception lors de la sauvegarde…
      }
```

- etc… :-)

openArchitectureWare : Génération de code

openArchitectureWare est un sous-projet du projet Eclipse GMT (Generative Modeling Tools). Il apporte un ensemble d'outils permettant de transformer des modèles :

- **Xpand :** C'est le langage de base de toutes les transformations écrites grâce à openArchitectureWare. La syntaxe est très simple, assez proche d'un langage de script type Ruby.
- **Xtend :** Couplé à Xpand, il permet de définir des librairies de transformations fréquemment utilisées (ex : générer le nom d'un setter). Ces librairies peuvent être écrites en Xpand mais peuvent aussi faire appel à du code écrit en Java, dans le cas où l'on souhaite réaliser des traitements un peu plus complexes.
- **Check :** Lorsque l'on réalise des transformations, on peut vouloir vérifier, avant de les appliquer, que notre modèle possède certaines propriétés. On peut par exemple écrire une règle Check pour vérifier que tous les attributs d'une EClass donnée ont bien un nom différent.
- **Workflow :** Grâce à un fichier XML, on vient configurer l'enchaînement des différents blocs de génération/vérification. On a la possibilité de faire passer des informations d'un bloc à l'autre. (un bloc peut par exemple être en charge de lire un modèle en mémoire)
- **xText :** Un peu à part, xText permet d'obtenir un éditeur Eclipse complet à partir d'une grammaire. Grâce à celle-ci pourront être générés, automatiquement : le parseur de ce langage, le colorateur syntaxique, le mécanisme d'autocomplétion, etc.

Il faut bien comprendre qu'une fois que l'on dispose d'un modèle (i.e. d'une abstraction de la réalité), on peut, mécaniquement, le transformer

BLUAGE™

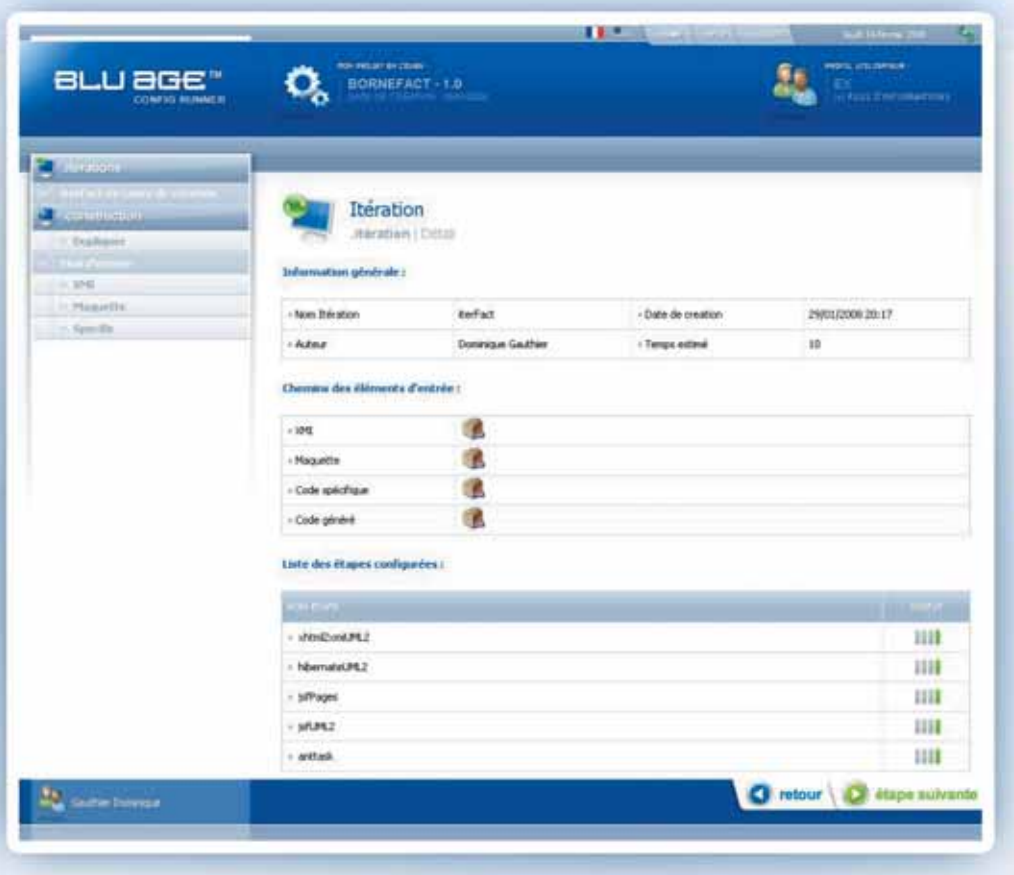
J2EE .NET APPLICATION GENERATOR

You Design, We Generate

www.bluage.com

Générateur MDD d'applications JAVA EE et .NET

BLU AGE™ est une suite logicielle MDA qui permet une transformation automatisée à 100% des diagrammes UML 2.0 en applications métiers Java EE et .Net. Grâce à BLU AGE™, les consultants analystes réalisent des applications SOA sans connaissance d'outils de développement, et réduisent de 30 à 50% les coûts et les délais de livraison de leurs projets.



Capitalisez sur le métier, pas sur la technique !

Sur la base de vos processus métiers modélisés en UML et des IHM maquettées en XHTML, BLU AGE™ génère automatiquement vos applications métiers dans le framework de votre choix. Un changement de framework ou une évolution fonctionnelle ? Vous modifiez les paramètres de génération (BSP) et/ou votre modélisation UML, et vous régénérez instantanément votre application !

Une solution ouverte, compatible avec les standards de l'Orienté Objet

Solution ouverte — aucun « run time » dans les applications générées - BLU AGE™ est proposé avec de nombreuses options de paramétrage préconfigurées (BSP - BLU AGE™ Shared Plugins), vous permettant de créer rapidement vos applications métiers dans des environnements professionnels (tels que ASPX, JSF, Struts, Spring, EJB, Hibernate, NHibernate…) pour les plateformes techniques telles que Websphere, Weblogic, IIS, Oracle, DB2. Vous utilisez des frameworks ou environnements de déploiement spécifiques ? Créez ou personnalisez vos propres BSP à l'aide des fonctionnalités de BLU AGE™ Software Factory (BSF), livrée nativement sous forme d'un Plugin Eclipse.

Travaillez suivant une méthode MDD, en mode itératif !

BLU AGE™ intègre Config-Runner, une interface web de gestion de projet, permettant aux consultants métiers (en charge de la modélisation UML) et aux architectes (en charge de la configuration des BSP) de travailler en parallèle et en mode itératif, jusqu'à la recette finale. Ces techniques accélèrent considérablement les délais de production des applications, tout en alignant les besoins fonctionnels avec les applications générées.

Toutes les marques citées sont la propriété de leurs propriétaires respectifs.

Des wébinars de démonstration de la solution BLU AGE™ sont disponibles en ligne.
Pour plus d'information, veuillez consulter notre site Web : http://www.bluage.com
Contactez-nous à l'adresse : **contact@bluage.com** ou par téléphone au : **01 56 05 88 00**

NETFECTIVE
Creative Technology

vers un autre modèle (i.e. une autre abstraction !), de n'importe quelle sorte. On peut donc générer un grand nombre de choses depuis notre modèle Posts/Comments :

- du code
  - scripts SQL de création de tables d'une base de données
  - code de création d'IHM
- de la documentation (pour chaque Classe, générer une page dont le titre est le nom de la classe, …)
- d'autres modèles (application de patterns de transformation)
- … tout ! (images SVG, cas de test, …)

GWT

GWT (Google Web Toolkit) est la plate-forme Open Source de développement de sites Ajax rendue publique par Google en mai 2006. Grâce à cette plate-forme, le développeur Web écrit principalement du code Java, qui sera compilé en Javascript par GWT pour être exécuté sur le navigateur Web du client. Pour plus de détails sur cet environnement, reportez-vous au dossier du n°101 de Programmez!. (Fig 1)

Le MDA grâce à un exemple simple : un moteur de blog

Modélisation avec EMF

Dans un premier temps, il convient de modéliser ce que va manipuler notre application. Nul besoin de penser au code de notre application, nous n'en sommes pas encore là ! Pour illustrer notre propos, prenons l'exemple d'une plate-forme simpliste de blog. Un blog, au minimum, c'est quoi ? Des articles et des commentaires, guère plus ! Pour créer et éditer notre modèle, deux solutions s'offrent à nous :

- utiliser l'éditeur arborescent de modèles, par forcément très sexy, fourni en standard par les plug-in EMF,
- ou mieux, utiliser un modeleur graphique. C'est ce que nous ferons, en utilisant l'éditeur graphique fourni par le projet Ecore Tools. (Fig 2)

Notre premier générateur openArchitectureWare

GWT 1.4 ne supportant pas encore le code Java 5 tel qu'il est généré par défaut par EMF, nous devons générer nos propres Javabeans, plus simples. C'est donc le moment pour nous de nous lancer dans l'écriture de notre premier générateur de code en Xpand. Ce template générera une classe Java pour chacune des EClass de notre modèle. Pour chacun des attributs (eAttributes) de la EClass, nous générons un getter, un setter et l'attribut.

```
"IMPORT ecore"

"DEFINE main FOR EClass"
   "FILE name+"Bean.java""
        package com.programmez.demo.client.core ;

        public class "name"Bean {
            "FOREACH eAttributes AS att"
                "LET att.eType.instanceTypeName AS type"
                private "type" _"att.name";

                public void set"att.name.toFirstUpper()"("type" "att.name") {
                    this._"att.name" = "att.name";
                }

                public "type" get"att.name.toFirstUpper()"() {
```

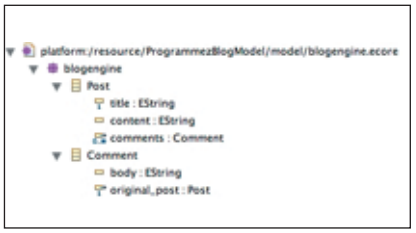


Fig.1

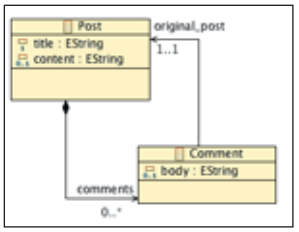


Fig.2

```
                    return _"att.name";
                }
                "ENDLET"
            "ENDFOREACH"
        }
    "ENDFILE"
"ENDDEFINE"
```

Le template doit être appelé depuis un fichier de workflow (fichier d'extension .oaw), dans lequel on enchaîne des appels à des composants oAW (parsing de modèle, transformation, mise en forme de code généré, …)

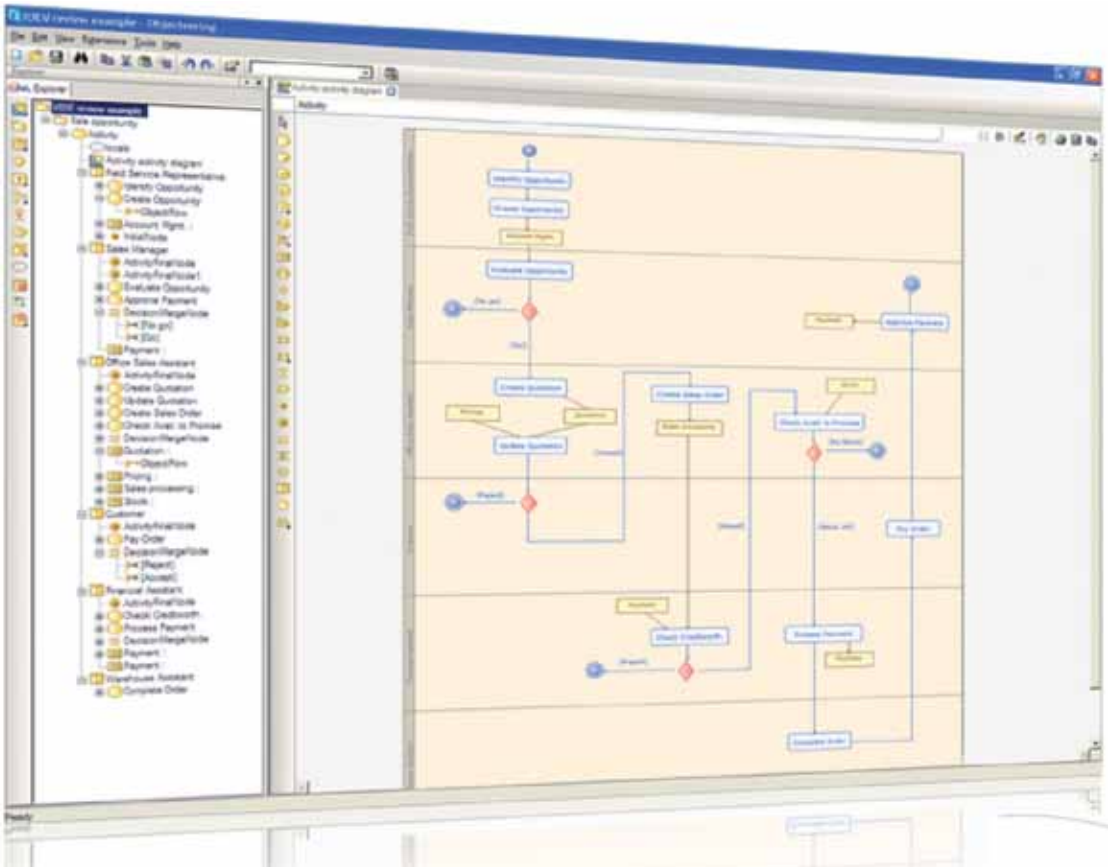
```
<?xml version="1.0"?>
<workflow>
   <property name="model" value="ProgrammezBlogModel/model/
blogengine.ecore" />
   <property name="src-gen" value="src-gen" />

   <!- lecture du modele et stockage dans la variable 'model' ->
   <component class="org.eclipse.mwe.emf.Reader">
        <uri value="../${model}" />
        <modelSlot value="model" />
   </component>

   <!- beans ->
   <component class="org.openarchitectureware.xpand2.Generator">
        <metaModel class="org.eclipse.m2t.type.emf.EmfRegistryMeta
Model" />
        <expand value="BeanTemplate::main FOREACH model.eClassifiers.
typeSelect(ecore::EClass)" />
        <outlet path="${src-gen}/com/programmez/demo/client/core">
            <postprocessor
                class="org.openarchitectureware.xpand2.output.Java
Beautifier" />
        </outlet>
   </component>

   <!- autres generateurs ->
   <!- … ->

</workflow>
```

Après avoir déclaré un certain nombre de variables (propriétés) que nous souhaitons manipuler dans notre workflow, nous faisons appel à deux composants :

- Le premier composant (component) lit le modèle Ecore situé dans le fichier blogengine.ecore et le charge dans le "slot" (variable locale au workflow) model.
- Le second composant fait appel au générateur Xpand pour le template

Your projects deserve a tool*

Objecteering Modeler



Objecteering 6.1

guide vos développements par les modèles sur toute la portée du système

Objecteering 6.1 est une solution de modélisation complète et ouverte supportant les standards OMG de modélisation UML 2.1, BPMN, et SysML, la modélisation de l'Architecture d'Entreprise et la modélisation d'une architecture SOA.
Il intègre le support de l'analyse des besoins et de la définition du dictionnaire, assurant ainsi une traçabilité complète sur tout le cycle de vie.
La technologie MDA associée à une ouverture de l'outillage en Java permet de guider le développement à chaque contexte utilisateur, chaque infrastructure technique cible. Ses générateurs sur étagère automatisent le développement d'applications Java, J2EE, .Net C#, C++, SQL...

Objecteering Software, éditeur de l'atelier Objecteering 6, est le spécialiste français UML/MDA pour le développement d'applications guidé par le modèle. Son offre modulaire couvre le cycle de vie de la gestion des exigences jusqu'au déploiement d'application.
Pour plus d'information sur Objecteering 6.1, pour télécharger Objecteering **Free Edition** ou Objecteering **Enterprise Edition**, rendez-vous sur : www.objecteering.com

Architecture d'Entreprise
UML2 BPMN
C# MDA SysML
Java SQL
J2EE SOA
C++

* Vos projets méritent un outil

www.objecteering.com - Tél. : 01 30 12 16 60 - sales@objecteering.com

Objecteering SOFTWARE
The model-driven development company

Quintre&sens • 4721

BeanTemplate. Le point d'entrée “main” de ce template est appelé pour chacun des éléments que contient notre modèle et qui sont des EClass (FOREACH model etc.). En outre, après l'étape de génération, on appelle le formateur de code Java afin d'avoir un code parfaitement mis en page.

Génération des pages GWT

Nul besoin d'être expert GWT pour cette partie, nous nous contenterons d'une génération de code assez simple, mais prouvant toute la puissance des outils mis à notre disposition dans l'écosystème Open Source Eclipse. Il est important de ne pas se lancer “à l'aveugle” dans l'écriture de templates de génération : il est toujours assez difficile de se représenter mentalement le bout de code ultragénérique qui, quasi-magiquement, fonctionnera dans tous les cas de figure. Il convient d'être pragmatique. Ainsi, il est généralement plus facile de conduire son développement de manière itérative. Nous commencerons donc par prototyper rapidement notre application Web directement en GWT. De là, nous déduirons les premiers fragments de templates génériques.

a) Squelette de l'application

Notre application, basique, se contentera d'afficher deux onglets (Posts et Comments) dans une page. Chacun de ces onglets affichera une table listant les éléments, et un bouton permettant d'en créer un nouveau. Un formulaire servira à éditer la fiche d'un item, ici un Post.

b) Génération des tables

On génère, pour chacune des classes *Post* et *Comment*, autant de colonnes qu'il y a de champs, et on écrit le code qui permet de rafraîchir le contenu de chaque cellule à partir d'une liste _objects. Le code de la classe AbstractTable (voir les sources), s'occupe d'ajouter un nouvel élément dans cette liste lorsque le bouton “Create a new XXX” est cliqué.

```
"IMPORT ecore"

"DEFINE main FOR Eclass"
   "FILE name+"Table.java""
        package com.programmez.demo.client.ui.tables;

        import java.util.*;

        import com.google.gwt.user.client.ui.*;
        import com.programmez.demo.client.core.*;
        import com.programmez.demo.client.ui.forms.*;

        public class "name"Table extends AbstractTable {
            public "name"Table() {
                super();

                _form = new "name"Form();

                refreshUi();
            }

            protected void createHeaders() {
                "FOREACH eAttributes AS att ITERATOR i-"
                    this.setWidget(0, "i.counter0",
                                new HTML("<b>"att.name.toFirstUpper
()"</b>"));
                "ENDFOREACH-"
            }

            public void refreshUi() {
                for (int i = 0; i < _objects.size(); i++) {
                    "name"Bean cb = ("name"Bean) _objects.get(i);
                    "FOREACH eAttributes AS att ITERATOR i-"
                        this.setWidget(i + 1, "i.counter0",
                                new Label(cb.get"att.name.toFirstUpper
()-"()));
                    "ENDFOREACH-"
                }
            }
        }
   "ENDFILE"
"ENDDEFINE"
```

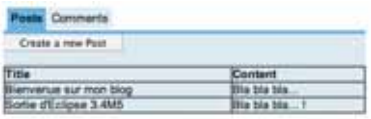


Fig.3

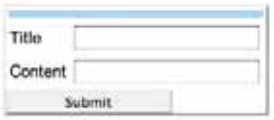


Fig.4

c) Génération des formulaires

Nous souhaitons générer des formulaires permettant d'éditer les champs d'un “bean”. On doit pouvoir passer un bean au formulaire, récupérer ce bean “modifié”, synchroniser l'IHM avec les valeurs des attributs du bean, et vice versa. Les formulaires générés hériteront tous d'une classe abstraite AbstractForm (voir le code de la classe abstraite et le template de génération sur le site) (Fig3)

Bilan

En quelques heures, et sans avoir eu à s'acquitter de la moindre licence, nous avons mis en place un moteur d'édition de "fiches" en écrivant très peu de code. Il est incomplet, certes. Mais, par raffinements successifs, nous allons pouvoir faire évoluer aussi bien notre modèle (si nécessaire) que nos templates de génération, et donc notre application finale.

Avoir une approche open-source nous a ainsi permis :

- de réutiliser des frameworks (EMF, oAW, ...) éprouvés, dans lesquels on peut venir plonger son nez, Open Source oblige. Il ne faut d'ailleurs pas hésiter à reporter sur le bugzilla Eclipse les bugs rencontrés ou bien les demandes de nouvelles fonctionnalités,
- de bénéficier d'un large support de la part de la communauté (les newsgroups Eclipse, sur news.eclipse.org, fourmillent de posts), (Fig4)

L'application générée est plutôt sobre : pas de gestion de la persistance des données, pas de possibilité de lier les fiches entre elles (le Post a des Commentaires)... Néanmoins, par itérations successives, nous pourrions intégrer Google Gears, d'autres modules de widgets GWT, etc. Enfin, dans la version 1.5 de GWT à venir, il sera possible d'intégrer les classes EMF automatiquement générées afin de bénéficier des classes utilitaires évoquées précédemment, sans avoir à écrire un template de génération de beans comme nous l'avons fait. Notre application reste donc ouverte et... évolutive ! À votre tour désormais de la faire évoluer, et devenir ainsi un des nombreux acteurs du monde de l'Open Source !

Benjamin Cabé - *Consultant Eclipse - Anyware Technologies. (benjamin.cabe@anyware-tech.com)* - www.anyware-tech.com

Anyware Technologies est une société spécialisée dans le développement d'applications basées sur des composants Open Source (Java, Eclipse, Apache...), intégrant les problématiques métier et les systèmes d'informations de ses clients.

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

LEONARDI est une marque de LYRIA S.A. Les autres marques citées dans ce document sont des marques déposées par leurs propriétaires respectifs

LEONARDI

...LYRIA REJOINT LE GROUPE W4 ! L'UNION DU BPM ET DE L'IHM MD... A SUIVRE...

L'ihm

EN TOUTE SIMPLICITÉ !

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*"...

NOUVELLE VERSION!

LEONARDI V4.0 OPEN SOURCE

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

2e partie

Ruby on Rails 2.0 : la nouvelle star du web !

Dans la première partie, Programmez ! n°107, nous avions parlé des nouveautés Rails 2.0, de la sécurité et de l'architecture REST. Dans cette seconde partie, nous allons techniquement plus loin avec l'intégration de Rails dans NetBeans, et la migration du code ancien vers Rails 2. Enfin, nous aborderons un sujet particulièrement important : utiliser IronRuby dans la DLR de Silverlight 2.0 ! À vous de jouer !

François Tonic

Migrer en douceur de Rails 1.2 à 2.0

La version 2.0 de Rails apporte son lot de nouveautés (Programmez! 107). En premier lieu, il vous faut migrer vers la version 1.2.6 si vous ne l'avez pas déjà fait, ce qui ne sera pas traité dans cet article. En effet, cette version, qui assure la compatibilité avec les précédentes, comporte de nombreux messages d'avertissement sur les modifications apportées à l'API de Rails. Ces messages sont affichés dans les logs ou durant les tests.

Vous l'aurez deviné, la première étape pour une migration réussie consiste à repérer les méthodes dépréciées, et à utiliser à la place celles de la nouvelle API. Pour cela, vous pouvez exécuter les tests de l'application et repérer tous les messages d'avertissements, afin d'apporter les modifications nécessaires. Vous pouvez également utiliser l'un des scripts [1][2] disponibles sur Internet, afin de vous aider dans cette recherche.

La suite de l'article est consacrée à la présentation des changements de manière assez exhaustive. Tous les problèmes soulevés ne s'appliqueront pas à votre application, et certains changements ont été initiés avant la version 1.2 de Rails. Ils vous sembleront donc peut-être superflus, mais la version 2.0 n'assurant pas la compatibilité avec les versions précédentes, il est tout de même nécessaire de revenir dessus.

- Les variables d'instance @params, @flash, @headers, @request, @response, @session, @cookie, et @env ne doivent plus être utilisées. Elles sont remplacées par des méthodes du même nom, respectivement *params, flash, headers, request, response, session, cookie,* et *env.* La méthode *keep_flash* est remplacée par une méthode d'instance *keep* dans la classe *Flash.*

```
User.create(params[:user])
flash[:notice] = "Migration réussie"
flash.keep
```

- Les méthodes *render_partial, render_action, render_without_layout,* ... n'existent plus, il faut désormais utiliser la méthode *render* avec l'option correspondante. Cette méthode attend maintenant une chaîne en argument.

```
render 'article/show'
render :partial => 'partial_view'
render :action => 'new'
render :action => 'new', :layout => false
render :action => 'new', :layout => 'admin'
```

- La méthode *expire_matched_fragments* n'est plus disponible, il faut maintenant utiliser la méthode *expire_fragment* avec une expression régulière en paramètre.

```
expire_fragment(/pages\/\d*\/notes/)
```

- Les routes nommées pour les ressources sont sous la forme *new_group_user_path* et non plus *group_new_user_path.*
- Le helper *human_size* a été renommé *number_to_human_size.*
- La méthode *content_for* dans un layout est remplacée par la méthode *yield.*
- Les méthodes *start_form_tag* et *end_form_tag* ne sont plus disponibles, il faut maintenant utiliser la méthode *form_tag.* L'option *:post => true* est remplacée par *:method => :post* (ceci est aussi valable pour la méthode *link_to*).

```
<%= form_tag('/articles') %>

<%= form_tag('/articles/1', :method => :put) %>
```

```
<% form_tag '/articles' do -%>
  <div><%= submit_tag 'Sauvegarder' %></div>
<% end -%>
```

- Les méthodes *link_to_image* et *link_image_to* ne sont plus disponibles, il faut maintenant utiliser conjointement les méthodes *image_tag* et *link_to*. La méthode *image_tag* ne peut plus recevoir en paramètre le nom d'une image sans son extension.

```
<%= link_to(image_tag('new.jpg'), :action => :new) %>
```

- Les méthodes *find_all* et *find_first* ne sont plus disponibles, il faut maintenant utiliser *find(:all)* et *find(:first).*
- Les méthodes *push_with_attributes* et *concat_with_attributes* pour une association *has_and_belongs_to_many* ne sont plus disponibles, il faut utiliser *has_many :through* avec un modèle de jointure.

```
# Il est possible d'avoir des informations dans le modèle de jointure
Subscriptions
# comme, par exemple, la date à laquelle l'utilisateur a rejoint le groupe, en y
# ajoutant le champ created_at qui sera automatiquement renseigné par
Rails.
class Subscriptions < ActiveRecord::Base
  belongs_to :group
  belongs_to :user
end

class Group < ActiveRecord::Base
  has_many :subscriptions has_many :users, :through => :subscriptions
end

class User < ActiveRecord::Base
  has_many :subscriptions
  has_many :groups, :through => :subscriptions
end
```

- Les options d'une association *:dependent => true* et *:exclusively_dependent => true* ont été remplacées par des valeurs distinctes pour l'option *:dependent.*

```
class User < ActiveRecord::Base
  has_many :profiles, :dependent => :destroy    # équivalent à :dependent
 => true
  has_many :articles, :dependent => :delete_all # équivalent à :exclusively
_dependent => true
end
```

- Si vous utilisez l'option *:class_name* sur une association *belongs_to*, le nom de la clé étrangère est maintenant déduite du nom de l'association, il faut donc préciser l'option *:foreign_key* ou modifier la table pour corriger ce problème.

```
class User < ActiveRecord::Base
  belongs_to :group, :class_name => 'UserGroup', :foreign_key => 'user
_group_id'
end
```

- La méthode *with_scope* de *ActiveRecord::Base* est maintenant privée, vous ne pouvez donc plus l'appeler directement. Si vous ne pouvez pas faire autrement, vous pouvez toujours l'utiliser indirectement avec la méthode *send* (attention, cette solution n'est pas compatible avec la version de dévelopemment 1.9 de Ruby).

```
Person.send(:with_scope, :find => { :conditions => :conditions => "age >
 26" }) { ... }
```

- Les conditions pour la méthode *count* sur un modèle doivent maintenant être transmises par l'option *:conditions*.

```
Person.count(:conditions => "age > 26")
```

- Le module *Reloadable* n'est plus disponible.
- La méthode *Hash.create_from_xml* a été renommée *Hash.from_xml.*
- La méthode *template_root* est remplacée par la méthode *view_paths* qui accepte en argument une chaîne ou une liste de chaînes.

```
view_paths = ''view/path"
view_paths = ["view/path"]
view_paths << ''view/path"
```

- Le ménage a été fait dans ActiveRecord et certains éléments n'en font maintenant plus partie. Ils ont été remplacés par des plug-in :

- acts_as_list - à installer avec la commande ruby script/plugin install acts_as_list.
- acts_as_nested_set - à installer avec la commande ruby script/plugin install acts_as_nested_set.
- acts_as_tree - à installer avec la commande ruby script/plugin install acts_as_tree.
- in_place_editing - à installer avec la commande ruby script/plugin install in_place_editing.
- auto_complete - à installer avec la commande ruby script/plugin install auto_complete.
- paginate - à installer avec la commande ruby script/plugin install svn://errtheblog.com/svn/plugins/classic_pagination. Le plugin classic_pagination assure la compatibilité avec l'implémentation de la pagination dans Rails 1.2, il est toutefois recommandé d'utiliser le plugin will_paginate, à installer avec la commande ruby script/plugin install svn://errtheblog.com/svn/plugins/will_paginate.
- à remplacer par le plugin will_paginate qui s'installe avec la commande ruby script/plugin install svn://errtheblog.com/svn/plugins/will_paginate. Attention, ça demande quelques adaptations dans le code qui ne sont pas décrites ici.

- Seuls les adaptateurs pour les bases de données SQLite, MySQL et Postgresql sont encore disponibles par défaut, les autres doivent être installés séparément si vous en avez besoin.

```
gem install activerecord-firebird-adapter --source http://gems.rubyonrails.org
gem install activerecord-frontbase-adapter --source http://gems.rubyonrails.org
gem install activerecord-openbase-adapter --source http://gems.rubyonrails.org
gem install activerecord-oracle-adapter --source http://gems.rubyonrails.org
gem install activerecord-sqlserver-adapter --source http://gems.rubyonrails.org
gem install activerecord-sybase-adapter --source http://gems.rubyonrails.org
```

- Les tâches rake ont été réorganisées pour profiter des espaces de nommage.

```
rake db:migrate
rake remote:deploy
```

Une fois que vous n'avez plus de messages vous indiquant des méthodes dépréciées, vérifiez que vos tests passent toujours, et, si ce n'est pas le cas, faites les corrections nécessaires. Tous les changements faits sur votre application, afin de préparer la migration, ne doivent pas l'empêcher de fonctionner avec la version 1.2.6.
Vous êtes maintenant prêt à changer de version, attention à bien choisir la dernière de la branche 2.0, afin de bénéficier d'éventuelles corrections. Vous devez ensuite également mettre à jour les fichiers javascript avec ta tâche rake adéquate.

```
rake rails:freeze:edge TAG=rel_2-0-2  # Mise à jour de Rails
rake rails:update  # Mise à jours des fichiers javascript
```

Il ne vous reste alors qu'à faire éventuellement deux changements avant d'exécuter à nouveau vos tests.

- Si vous avez une application REST et utilisez les ressources pour le routage, le nom des contrôleurs doit maintenant toujours être au pluriel, même pour les ressources au singulier. Il vous faut donc renommer vos contrôleurs ou utiliser l'option :controller dans votre fichier config/routes.rb.

```
map.resource :test, :controller => 'test'
```

Je vous conseille fortement de passer par la seconde solution dans un premier temps. Dans un second, une fois que tous vos tests passent, corrigez le nom d'un contrôleur après l'autre en n'oubliant pas de modifier le nom de la classe, du helper et du test associé, ainsi que de leurs fichiers. De cette façon, vous pourrez bénéficier de l'aide précieuse fournie par vos tests, afin de vous assurer que vous n'oubliez rien.

- Si vous le désirez, vous pouvez changer le nom des extensions de vos vues, afin de répondre à la nouvelle convention. Bien que cette modification ne soit pas obligatoire, je vous la conseille, afin de pouvoir profiter des nouvelles possibilités qu'elle apporte à la méthode *respond_to*. L'extension d'une vue se compose maintenant de deux parties, la première indiquant le format du contenu (html, js, xml, ...) et la seconde, le type de moteur de rendu utilisé (erb, rjs, builder, haml, liquid).

```
view.rhtml _ view.html.erb
view.liquid _ view.html.liquid
view.haml _ view.html.haml
view.rjs _ view.js.rjs
view.rxml _ view.xml.builder
```

Pour terminer, aidez-vous de vos tests, afin de vous assurer que la migration est réussie, et corrigez les problèmes liés aux particularités de votre application. Cet article avait pour but de migrer une application 1.2 vers 2.0, afin qu'elle fonctionne sur cette nouvelle version, mais il vous reste encore du travail si vous voulez modifier votre application pour lui permettre de bénéficier des nouveautés de Rails et de répondre aux nouvelles conventions en vigueur.

[1] http://pastie.caboo.se/private/krcevozww61drdeza13e3a
[2] http://www.slashdotdash.net/articles/2007/12/03/rails-2-upgrade-notes

Yann Lugrin

Intégration de Rails

Avec la sortie de sa version 6.0, Netbeans [1] fait son entrée dans l'univers de Ruby / Ruby on Rails. Cet IDE développé par Sun, disponible sur Linux, Mac et Windows, s'ouvre aujourd'hui à d'autres langages avec un certain succès. Voici un aperçu des avantages qui en font un très bon outil pour développer des applications Rails.

L'installation de Netbeans se fait très facilement, il suffit de télécharger la version pour Ruby (19 MB seulement) et vous aurez tout ce dont vous avez besoin pour développer avec ce langage, à noter qu'elle contient tous les éléments nécessaires pour Rails et qu'il est possible d'étendre l'nstallation avec d'autres fonctionnalités (UML, Java, C++, PHP, ...) via le panneau de gestion des plug-in [2]. Tout comme les plug-in, les gems (système de packaging Ruby) peuvent être gérés depuis le panneau adéquat qui liste ceux présents sur la machine, à mettre à jour ou disponibles sur le serveur.
La création d'un nouveau projet peut se faire à partir de rien, une nouvelle instance de Rails sera alors créée et apparaîtra dans la liste des projets. L'explorateur Projet représente une vue logique d'une application Rails en regroupant les fichiers dans des répertoires nommés *Controllers, Helpers, Models, Configurations, Database Migrations, Public,* etc. En cas de besoin il est aussi possible d'utiliser l'explorateur *Fichier* qui représente la structure physique de l'application. La création d'un nouveau projet depuis un dépôt Subversion, CVS ou encore Mercurial (disponible via un plug-in) est également disponible.
L'explorateur *Projet* [fig. 1] n'affiche que ceux ouverts ; il est possible de les classer dans des groupes afin de pouvoir les ouvrir et fermer facilement. Le menu contextuel sur le nom du projet permet d'accéder à de nombreuses fonctionnalités [fig. 1] : la

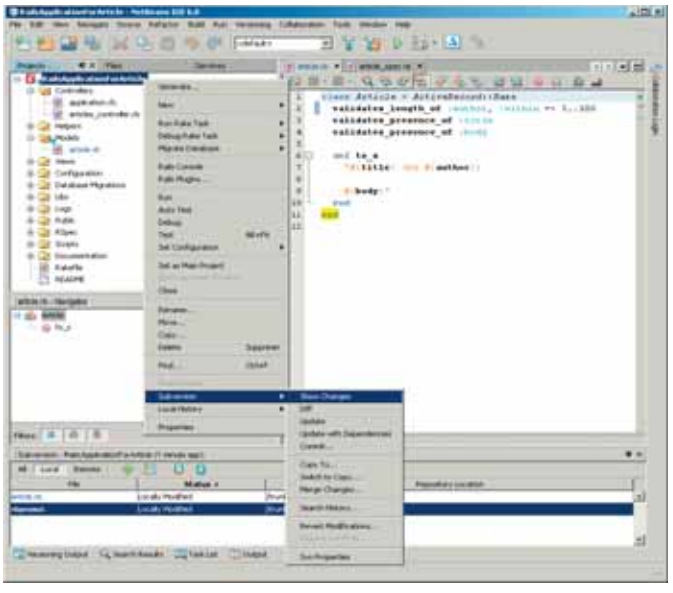


Fig.1

dans NetBeans 6.0

liste des tâches *rake* (mise à jour dynamiquement) qu'il est possible de lancer en mode normal ou debug, le générateur de Rails (également disponible par le menu contextuel de certains répertoires tel que *Controllers, Models ou Database Migrations*) et une liste des migrations classées par groupes de cinq, afin de trouver rapidement celle qu'on recherche, ainsi qu'un lien direct vers la version zéro et la dernière migration. On y trouve aussi la possibilité de lancer une console et la gestion des plug-in Rails. Il y a également un explorateur de classe [fig. 1] qui fonctionne très bien avec les fichiers Ruby, YAML et ERB ; dans le cas de ce dernier, il y a une séparation claire entre le code HTML et les blocs de code en Ruby. Il est bien entendu possible de réorganiser le placement de ces divers éléments et des autres disponibles (éditeur, fenêtre de sortie, ...) ou encore de les réduire.

L'intégration de Subversion (et des autres gestionnaires de sources) est une réussite, le statut d'un fichier est visible dans l'explorateur *Projet* et le menu contextuel permet d'accéder à tout ce dont nous avons besoin [fig. 1] : nous retiendrons tout particulièrement la possibilité de voir les changements effectués dans le projet et de sélectionner ces fichiers pour en voir les différences, faire un *commit* ou au contraire les en exclure, revenir sur les modifications, etc. La vue des différences entre deux versions [fig. 2] permet de revenir sur les modifications de façon très intuitive, une icône étant présente juste à côté du code modifié, afin de permettre de revenir à la version d'origine ou de replacer du code effacé. A noter aussi que Netbeans propose un historique des changements locaux, c'est-à-dire ceux intervenus entre le dernier *commit subversion* et l'instant présent. Cette fonction est extrêmement pratique pour revenir à une version antérieure de vos fichiers, même si aucun *commit* n'a eu lieu récemment.

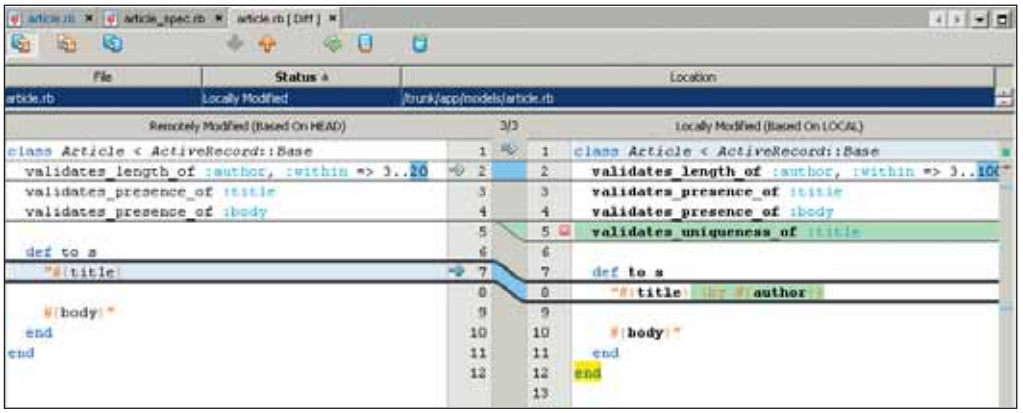


Fig.2

Le choix des couleurs pour la coloration syntaxique est judicieux et cela fonctionne très bien dans les fichiers ERB où le code HTML et Ruby n'entrent pas du tout en conflit. Il faut également souligner qu'elle colorise différemment les variables locales et les méthodes (ce qui permet de ne pas confondre l'utilisation de la première avec l'appel de la seconde) et qu'une variable *local* instanciée, mais jamais utilisée, est clairement indiquée. L'éditeur nous indique aussi les erreurs de syntaxe, permet de savoir où commence et se termine un bloc de code, une paire de parenthèses ou d'accolades. La sélection d'une variable, d'un mot clef ou d'une méthode met en avant les autres endroits où ce nom est utilisé. Il est également possible de voir où et quel code a été modifié ou effacé, et de revenir sur ces changements.

Le complétion automatique [fig. 3] du code permet de voir rapidement les méthodes disponibles dans l'API de Ruby, les librairies, l'API de Rails mais aussi le code de notre application (il ne manque que les méthodes d'accès aux attributs ou les relations générées implicitement par Rails dans les modèles). Cette complétion est accompagnée par la documentation des méthodes et une aide pour l'insertion des paramètres. Les blocs, parenthèses et accolades sont automatiquement fermés, il suffit de sélectionner un texte et d'utiliser la touche adéquate afin de le mettre entre guillemets, parenthèses ou accolades. Il existe un grand nombre de raccourcis [3][fig. 4] pour, par exemple, sélectionner tout un bloc, mettre du code en commentaire ou en changer l'indentation. Des patrons de code (template) [4] (facilement extensibles) sont disponibles afin

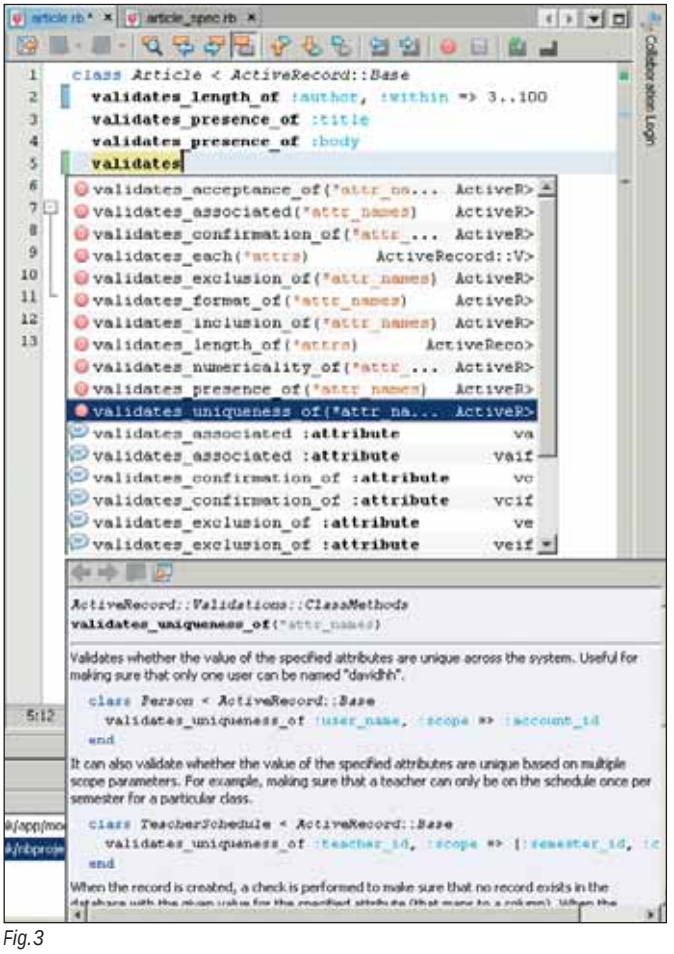


Fig.3

Recevez la Newsletter www.programmez.com/inscription.php

d'ajouter rapidement une paire clef/ valeur ou un bloc begin/ rescue, par exemple. [fig. 3]
TestUnit et *RSpec* (disponibles via un plug-in) sont intégrés, un raccourci [fig. 4] permet de passer d'un *Model* ou d'un *Controller* à son test. Associé à *autotest,* le résultat est directement indiqué en bas de l'éditeur du fichier testé après une sauvegarde de celui-ci. Le *traceback* des erreurs comprend des liens hypertextes, afin de rapidement retrouver le code impliqué. Le serveur Rails peut être lancé en mode debug, il est alors possible de profiter des *breakpoint* et *watch,* de voir toutes les variables de l'environnement, d'avancer pas à pas, et en laissant la souris sur une méthode, de connaître la valeur qu'elle retourne. Une interface permet d'accéder à la base de données et d'y faire des requêtes (malheureusement pas de détection automatique du fichier de configuration de Rails). Le système de recherche et de remplacement [fig. 5] est disponible pour le fichier courant avec mise en avant des résultats directement dans l'éditeur ou sur tout le projet. Il existe également un panneau afin de retrouver facilement un fichier d'après son nom. Les expressions rationnelles (regex) sont naturellement disponibles. A noter enfin qu'il existe un système de collaboration permettant de partager en direct

Action	Raccourcis	Raccourcis Mac
Montre les possibilités de complétion de code.	Ctrl+Espace	Ctrl+Espace
Montre la documentation pour la méthode, la classe ou le champ en dessous du curseur (ceci n'est pas toujours possible à cause de la nature dynamique de Ruby).	Ctrl+Maj+Espace	Commande+Maj+Espace
Montre le nom du paramètre courant lors de l'édition de la liste d'arguments pour l'appel à une méthode (ceci n'est pas toujours possible à cause de la nature dynamique de Ruby).	Ctrl+P	Commande+P
Passe entre l'action (une méthode dans un fichier controller) et la vue correspondante (un fichier .rhtml ou .erb).	Ctrl+Maj+A	Commande+Maj+A
Passe entre un fichier test et le fichier testé correspondant.	Ctrl+Maj+T	Commande+Maj+T
Renomme le symbole en dessous du curseur (dans tout le fichier).	Ctrl+R	Commande-R
Commente / Décommente la ligne sélectionnée ou la ligne contenant le curseur.	Ctrl+/	Commande+/
Reformate le code sélectionné ou le fichier entier s'il n'y en a pas.	Alt+Maj+F	Ctrl+Maj+F
Reformate le paragraphe du commentaire sélectionné (selon les conventions de RDoc).	Ctrl+Maj+P	Commande+Maj+P
Augmente / Diminue l'indentation des lignes sélectionnées.	Tab/Maj+Tab	Tab/Maj+Tab
Ajoute / Retire le breakpoint sur la ligne sélectionnée.	Ctrl+F8	Commande+F8
Complète le mot courant par le premier élément correspondant.	Ctrl+K	Commande+K
Ouvre un panneau permettant de retrouver la définition d'une classe dans le projet actuellement ouvert ou la librairie Ruby.	Ctrl+O	Commande+O
Ouvre un panneau permettant de retrouver un fichier par le contenu de son nom.	Alt+Maj+O	Ctrl+Maj+O
Exécute le fichier courant. Dans un projet Rails, un navigateur est ouvert sur l'URL correspondante si c'est possible.	Maj+F6	Maj+F6
Teste le fichier (exécute le fichier de test correspondant au fichier courant).	Ctrl+F6	Commande+F6
Maximise la fenêtre courante (souvent l'éditeur), utiliser une seconde fois pour revenir à l'état initial.	Maj+Espace	Maj+Espace
Ouvre le générateur de code Rails.	Ctrl+Insérer	Commande+I
Sélectionne le fichier courant dans l'explorateur Projet.	Ctrl+Maj+1	Commande+Maj+1
Sélectionne le fichier courant dans l'explorateur Fichier.	Ctrl+Maj+2	Commande+Maj+2
Ouvre le panneau de recherche pour le fichier courant.	Ctrl+F	Commande+F
Ouvre le panneau de recherche pour les projets ouverts.	Ctrl+Maj+F	Commande+Maj+F

[fig. 4]

son travail avec une personne distante, qu'un système de macro permet d'enregistrer une tâche, afin de la répéter à nouveau plus tard, et qu'il est possible de formater le code de tout un fichier ou simplement d'une partie de celui-ci. [fig. 5]
L'intégration de Ruby et Ruby on Rails dans NetBeans est donc une véritable réussite. Sun est parvenu à produire un IDE qui offre quasiment autant d'aide sur un langage fortement dynamique et non typé comme Ruby qu'il en offre sur Java ou C++. Allié à une grande stabilité et au fait qu'il est disponible sur plusieurs systèmes d'exploitation, la version 6.0 de Netbeans s'avère un outil de choix pour les développeurs Ruby et Ruby on Rails.

[1] http://www.netbeans.org
[2] http://plugins.netbeans.org/PluginPortal
[3] http://wiki.netbeans.org/wiki/view/RubyShortcuts
[4] http://wiki.netbeans.org/wiki/view/RubyCodeTemplates

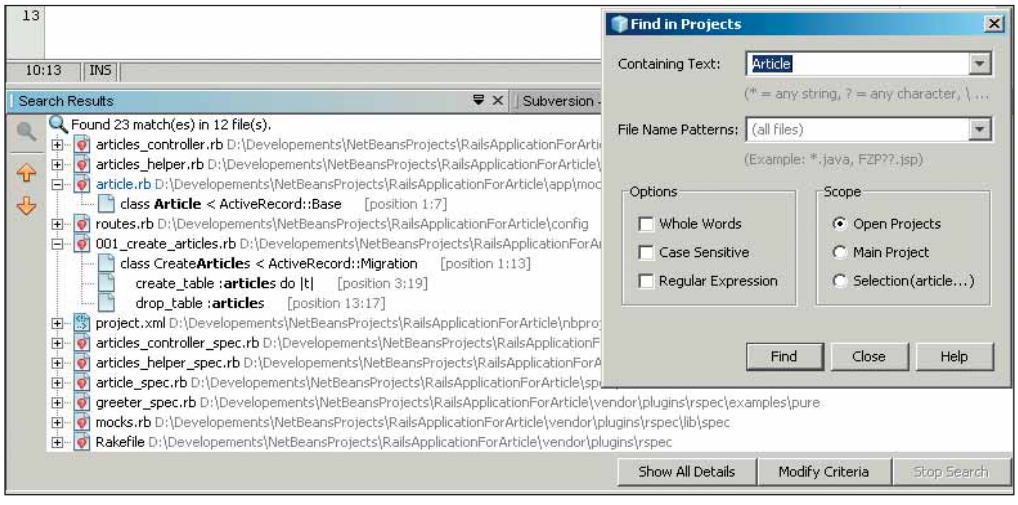


Fig.5

Yann Lugrin

co-fondateur de Liquid Concept (http://www.liquid-concept.ch), société suisse de développement web. Spécialiste Ruby et Ruby on Rails avec lequel il développe depuis 2005. il est également membre de l'équipe de développement de Globalize (http://www.globalize-rails.org : plugin Rails permettant de créer des applications multilingues). http://www.sans-savoir.net / yann.lugrin@sans-savoir.net.

Pour les .Net *addicted*

Hors-Série .Net

Programmez! HS

Programmez!

www.programmez.com

LE MAGAZINE DU DÉVELOPPEMENT

Avril / Mai 2008 - Hors-Série

Windows Live

Découvrez les API Live

Maîtrisez Virtual Earth, Live ID et Live Photo !

Web 2

Développez avec Silverlight 2.0 bêta !

Futur
F# : le prochain langage .Net ?

Multi-cœur
Optimisez vos codes avec Parallel FX

Architecture
Intéropérabilité .Net - Java 6 avec Workflow Foundation

Vista
Service Pack 1 et le développeur

.Net 3.5
Créez des documents OpenXML
Tout savoir ou presque sur LINQ

Serveur
Utiliser les nouvelles API de Windows Server 2008

M 02104 - 11 H - F: 4,95 € - RD

Printed in France - Imprimé en France - BELGIQUE 5,45 € - SUISSE 10 FS - LUXEMBOURG 5,45 € - Canada 7,45 $ CAN - DOM Surf 5,90 € - TOM 780 XPF - MAROC 50 DH

Demandez-le à votre marchand de journaux

Une introduction à Dynamic Silverlight et IronRuby

On appelle Dynamic Silverlight la technologie qui permet de travailler au sein de Silverlight tout en utilisant des langages dynamiques supportés par le Dynamic Language Runtime (DLR) comme IronRuby, IronPython ou encore le Managed Jscript.

Silverlight est pour mémoire un socle de librairies et un plug-in cross-plate-forme permettant de développer des applications riches tournant dans les navigateurs (on pourrait dire pour faire simple que Silverlight est le " Flash de Microsoft "). Le DLR est quant à lui une couche logicielle proposée par Microsoft, dont le but est de fournir un ensemble de services mutualisés facilitant le développement de langages dynamiques tels qu'IronPython, IronRuby et bien d'autres.

Comment travailler avec Dynamic Silverlight ?

Vous pouvez dès aujourd'hui télécharger les éléments nécessaires sur http://dynamicsilverlight.net. Voici ce dont vous avez besoin précisément (vous retrouverez les url correspondantes sur http://dynamicsilverlight.net) :

- le Dynamic Silverlight SDK (qui comprend notamment le DLR, IronRuby et IronPython)
- le Silverlight 2 Runtime (compatible Internet Explorer, Firefox et Safari)
- le Silverlight 2 SDK (ce SDK intègre également le support Ruby et Python, toutefois il est recommandé de télécharger la version fournie dans le Dynamic Silverlight SDK qui est plus récente)

Vous pourrez également télécharger le pack " Silverlight 2 Visual Studio Tools " qui, s'il ne supporte pas encore Ruby et Python en tant que templates projets, intègre toutefois la possibilité de débugger les applications dès aujourd'hui (nous approfondirons ce point plus loin dans l'article). Pour la suite de cet article nous travaillerons avec IronRuby. Vous pouvez transposer à IronPython ou autre selon votre préférence.

Chiron – le serveur web de développement pour Silverlight

Le SDK de Silverlight 2 intègre Chiron, un outil que nous utiliserons un peu comme Webrick pour ceux qui travaillent avec RubyOnRails. Chiron sert principalement à deux choses :

- il peut tourner comme serveur et sait alors servir à la demande le répertoire de votre application Silverlight en le transformant en fichier .XAP à la volée (format attendu par le plug-in Silverlight), très pratique pour la phase de développement,
- il peut également servir en ligne de commande à générer ce fichier .XAP, lorsque vous souhaiterez le déployer vers un serveur de production.

Pour la suite, le plus simple est d'ajouter le chemin qui contient Chiron (C:\Program Files\Microsoft SDKs\Silverlight\v2.0\Tools\Chiron) à votre PATH.

Créons un " Hello World " avec IronRuby et Silverlight

Nous allons créer un " Hello World " qui se contentera d'afficher un contrôle TextBlock, le rendra cliquable, et montrera comment modifier son libellé depuis IronRuby lorsqu'un clic intervient.

Structure de répertoire à adopter

La structure de répertoire et les fichiers à créer sont les suivants :

```
/ hello_world
   > Default.html
   / app
        > app.rb
        > app.xaml
```

Notez que les noms adoptés peuvent changer au besoin – il est toutefois alors nécessaire de prévenir Silverlight avec des paramètres spécifiques.

Le point d'entrée dans le navigateur : Default.html

Dans Default.html, placez le markup suivant (version raccourcie pour rester concis – la version longue comprend gestion d'erreurs, incitation à installer Silverlight s'il n'est pas installé etc.) :

```
<html>
   <body>
        <object data="data:application/x-silverlight," type="application
/x-silverlight-2-b1" width="100%" height="100%">
             <param name="source" value="app.xap"/>
        </object>
   </body>
</html>
```

Ce fichier fait référence à app.xap qui n'existe pas sur notre disque en tant que tel: il est généré à la demande par Chiron.

Note : si vous êtes amenés à faire le même développement sur votre machine, je vous invite à télécharger le code d'un Default.html avec gestion d'erreur intégrée dans les samples du SDK. Notre version n'affichera pas les messages d'erreurs – le développement sera difficile sans !

Le markup XAML : app.xaml

Ici aussi nous resterons plutôt concis – pour du XAML. Il faut savoir que ce XAML peut être généré à l'aide d'Expression Blend (outil de design d'interfaces riches fourni par Microsoft) ou de Visual Studio. Un certain nombre d'utilisateurs préfèrent déjà aujourd'hui générer les contrôles dynamiquement par le code plutôt que déclarativement comme ici, car cela devient rapidement verbeux :

```
<UserControl
   xmlns="http://schemas.microsoft.com/client/2007" xmlns:x="http://
schemas.microsoft.com/winfx/2006/xaml"
   x:Class="System.Windows.Controls.UserControl">
    <TextBlock x:Name="HelloLabel" x:Text="Cliquez moi !"/>
</UserControl>
```

NOUVEAU les TUTORIELS sur www.programmez.com

Ici nous déclarons un contrôle TextBlock dont le nom sera HelloLabel, avec un texte par défaut. Nous nous servirons de cet identifiant pour le retrouver et le manipuler depuis le code IronRuby.

Le code IronRuby: app.rb – Version 1

```
include System::Windows
include System::Windows::Controls

root = Application.Current.LoadRootVisual(UserControl.new(), "app.xaml")

root.find_name('HelloLabel').mouse_left_button_down do |sender,args|
  sender.text = "#{sender.text} - Hello from IronRuby !"
end
```

Démarrons l'application

Vous pouvez à présent démarrer une ligne de commande et vous placer dans le répertoire qui contient Default.html, puis lancer la commande suivante :

```
chiron /b
```

Cette commande démarrera le serveur et ouvrira un navigateur sur une liste de fichier. Choisissez Default.html et vous devriez voir notre application (si Silverlight est bien installé dans votre navigateur).

Le helper silverlight.rb à la rescousse

Le code ci-dessus ne sonne pas très Ruby. Nous allons le refactoriser en utilisant un petit helper dont voici le code (il est livré avec un des samples fournis dans le SDK Dynamic Silverlight). Ce helper va nous permettre de simplifier notre code applicatif tout de suite après :

```
include System::Windows
include System::Windows::Controls

class SilverlightApplication
  def application
    Application.current
  end

  def self.use_xaml(options = {})
    options = {:type => UserControl, :name => "app"}.merge(options)
    Application.current.load_root_visual(options[:type].new, "#{options
[:name]}.xaml")
  end

  def root
    application.root_visual
  end

  def method_missing(m)
    root.send(m)
  end
end

class FrameworkElement
  def method_missing(m)
    find_name(m.to_s.to_clr_string)
  end
end
```

Le helper est plus long que notre code d'origine. C'est vrai! Toutefois il va nous permettre de rendre plus simple la rédaction du code applicatif. Par ailleurs ce helper est réutilisable tel quel au sein d'autres projets (ce que je fais en pratique).
Il nous permet notamment de :

- rendre plus simple le chargement de la partie XAML (dont la syntaxe n'est pas évidente à retenir)
- simplifier l'accès aux éléments graphiques en faisant appel à method_missing (utilisation de mon_control au lieu de root.find_name('mon_control'))
- éviter d'avoir à spécifier les includes

Le code IronRuby : app.rb – Version 2

Avant d'utiliser ce nouveau code, veillez à renommer notre TextBlock en " hello_label " au lieu de " HelloLabel " (une version plus avancée permettrait de ne pas avoir à faire ce renommage, nous en resterons ici pour aujourd'hui). Veillez également à ajouter *silverlight.rb* au même niveau que *app.rb*. Voici le nouveau code :

```
require 'silverlight'

class HelloWorld < SilverlightApplication
  use_xaml

  def initialize
    hello_label.mouse_left_button_down do |sender,args|
      sender.text = "#{sender.text} - Hello from IronRuby !"
    end
  end

end

HelloWorld.new
```

Le refactoring nous a permis de rendre la syntaxe plus agréable à utiliser lorsqu'on attaque le développement de l'application.

Outillons-nous !

Utilisation du débuggage pas à pas

Ce premier outil va nous être très utile. Le débuggage pas à pas est déjà disponible, quoi qu'encore " jeune ". En voici une copie d'écran réalisée sur mon poste (Fig. 1).
Pour arriver à ce résultat aujourd'hui, vous aurez besoin des éléments suivants :

- Visual Studio 2008 RTM (la version d'évaluation suffit),
- Silverlight 2 Visual Studio Tools (disponible sur http://dynamicsilverlight.net/get – l'installeur vous demandera de désinstaller toute version de Silverlight ou de son SDK avant!),
- les dll fournies dans le Dynamic Silverlight SDK (qui sont les plus récentes et supportent le debuggage), à placer ou bien dans votre répertoire *app* ou bien dans le sous-répertoire */bin* du répertoire où est installé Chiron.

Il vous faudra ensuite ajouter le paramètre suivant dans votre Default.html :

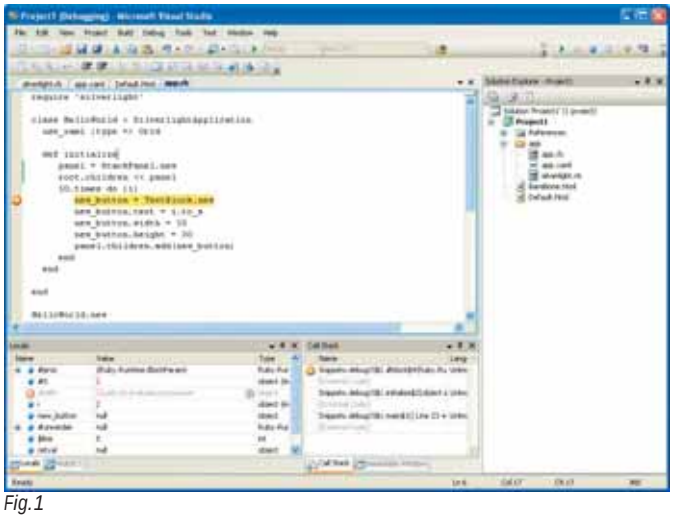


Fig.1

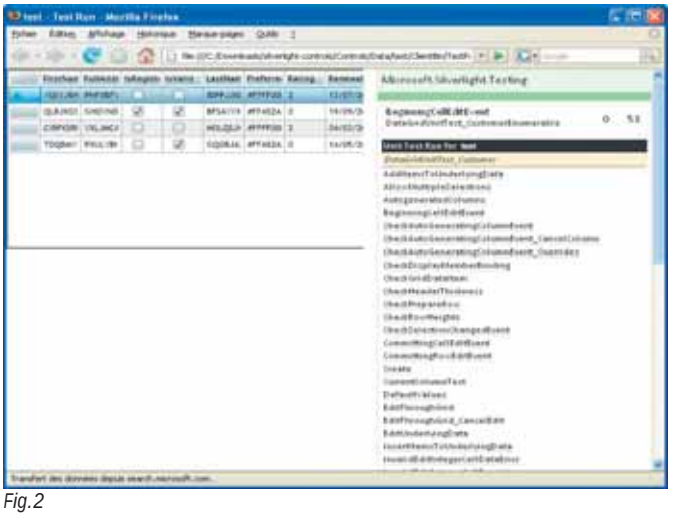


Fig.2

```
<param name="initParams" value="debug=true, reportErrors=error Location" />
```

Si vous êtes toujours là (!), voici maintenant les étapes pour déclencher le breakpoint :

- ouvrez le fichier ruby dans Visual Studio,
- placez le breakpoint,
- attachez le debugger à la fenêtre du navigateur qui fait tourner l'application Silverlight (menu Debug / Attach to Process)
- retournez dans le navigateur et faites F5

Le breakpoint devrait être activé lorsque le code est exécuté (cela fonctionne avec Firefox comme avec Internet Explorer). Vous pourrez alors observer les fenêtres habituelles de debuggage (Locals, Watch, Call Stack etc).

Comment tester le code et l'interface graphique ?

Pas de réponse toute faite aujourd'hui en ce qui concerne IronRuby, toutefois les pistes sont intéressantes – je choisis donc de les évoquer ici.

Le premier point est la livraison par Microsoft d'un framework de test UI et unitaire pour Silverlight. Pour en savoir plus, je vous recommande de regarder la vidéo réalisée par Jeff Wilcox (Test Lead sur les interfaces graphiques chez Microsoft) à l'adresse suivante : http://www.jeff.wilcox.name/2008/03/31/silverlight2-unit-testing/

Voici une copie d'écran des tests concernant la DataGrid (réalisée sur mon poste, voir fig.2).

Les tests inclus dans cet exemple sont codés en C#. Je ne serais pas étonné qu'apparaissent dans les jours ou semaines à venir des prototypes permettant de travailler de la même façon avec IronRuby si vous le souhaitez (c'est une tâche que j'entreprendrais probablement si personne ne le fait avant moi!). Par ailleurs, si vous téléchargez le code source d'IronRuby sur http://www.ironruby.net/, vous pourrez observer comment sont réalisés les tests d'IronRuby aujourd'hui.

Je pense qu'à terme nous observerons l'apparition de frameworks purement IronRuby permettant tant de tester le code unitairement que graphiquement lorsque c'est utile. J'apprécie particulièrement Ruby dans le contexte des tests car même sans framework perfectionné, il est aisé de refactoriser le code pour obtenir des tests faciles à comprendre et à maintenir d'une part, et la nature " ouverte " du langage simplifie la mise en place de *mocks objects* lorsque c'est nécessaire d'autre part.

Quelles perspectives pour Dynamic Silverlight ?

Dynamic Silverlight profite des perspectives offertes par Silverlight, en y ajoutant la productivité accrue apportée par les langages dynamiques tels qu'IronPython ou IronRuby entres autres. Dynamic Silverlight apporte donc la possibilité de générer des animations multimedia qui sont portables (Windows et Mac aujourd'hui, Linux en cours avec le projet MoonLight – http://www.mono-project.com/MoonlightShots), en nous appuyant sur un framework riche qui est un sous-ensemble de WPF, tout en continuant à travailler au sein des API DotNet que nous connaissons bien, et en profitant du support des langages dynamiques.

Je pense que ces possibilités seront largement exploitées cette année tant par les développeurs DotNet que par les développeurs Ruby, et que cette technologie nous prépare de beaux projets en perspective !

Pour en savoir plus

Les ressources sur Silverlight sont de plus en plus nombreuses. Je vous invite à consulter la page http://weblogs.asp.net/scottgu/pages/silverlight-posts.aspx où Scott Guthrie répertorie les astuces et tutoriels portant sur Silverlight de façon générale. Pour en savoir (beaucoup) plus sur les contrôles Silverlight, téléchargez leur code source qui a été rendu disponible sur http://blogs.msdn.com /sburke/archive/2008/03/05/silverlight-2-beta-1-controls-available-including-source-and-unit tests.aspx.

Un point important est que Scott Guthrie a annoncé sur son blog la possibilité de partir de ce code source, de le modifier selon vos besoins et de le relivrer dans vos applications. Le code source inclut environ 2000 tests unitaires qui permettent d'apprendre beaucoup de choses sur leur fonctionnement. Pour Dynamic Silverlight en particulier, vous trouverez un certain nombre d'exemples qui s'appuient sur IronPython sur le site de Michael Foord et dont vous pouvez vous inspirer pour explorer les API (http://www.voidspace.org.uk/ironpython/webide/webide.html).

Si vous êtes intéressés par IronRuby spécifiquement qui est encore assez jeune, je vous invite à rejoindre la liste ironruby-core (http://rubyforge.org/mailman/listinfo/ironruby-core).

Pour terminer, je vous incite à suivre le site http://dynamicsilverlight.net et à télécharger les exemples disponibles – c'est encore là qu'on peut en apprendre le plus pour le moment sur la partie dynamique.

■ **Thibaut Barrère** *est consultant indépendant et apprécie les projets innovants.* - http://www.logeek.fr.

Framework .Net

Ajax

JSF

GWT

Eclipse RCP

Flex

AIR

Extrême Java

valtech training

1 journée
3 ateliers
230 €HT

90 formations au développement logiciel

chez vous
ou à Paris, Toulouse, Lyon, Grenoble, Genève, Bruxelles, Luxembourg

Atelier clients riches Web : GWT, Silverlight et Flex

le 19 juin à Paris la Défense

Après une introduction rapide aux problématiques et solutions du client riche, vous participerez à des ateliers techniques successifs de 2 heures sur Flex, GWT puis Silverlight.

Vous écrirez votre première application dans chacune de ces technologies et serez ainsi amené à juger de leur facilité de mise en œuvre :

essayez-les, choisissez !

9h00 - 10h30 Théorie et démonstrations
par Xavier Paradon - Valtech Training (*)

Introduction

• Limites des applications Web
• Client riche : définition
- Richesse graphique
- Impact sur l'architecture
- Déploiement et portabilité
• Rich Desktop Application (RDA) et Rich Internet Application (RIA)
- Quelles différences et quel avenir ?

Tour d'horizon des technologies

• Flex, Adobe Integrated Runtime (AIR)
• Silverlight, Volta
• Ajax, Google Web Toolkit (GWT)

Démonstrations

10h45 - 12h45 atelier Flex
animé par Yann Chevalier - Baao (*)

14h15 - 16h15 atelier Silverlight
animé par Angelico Pacifico - Agile Partner (*)

16h30 - 18h30 atelier GWT
animé par Sami Jaber – DNG Consulting (*)

(*) Valtech Training se réserve la possibilité de changer d'intervenants.

www.valtech-training.fr
+33 (0)1 41 88 23 00 - +33 (0)5 62 47 52 00
info@valtech-training.fr

XML et le développeur

Le constat n'est plus à faire. XML est partout. Il sert à tout. Bref, il est incontournable, aussi bien dans les échanges de données que dans les applications, que ce soit sous la forme du XML pur ou par l'intermédiaire de dialectes, eux aussi très nombreux. Le monde XML sert de base au Web 2.0, aux applications internet riches ou encore aux architectures orientées services, jusqu'au cœur des bases de données.

XML est aujourd'hui mature. Il fête d'ailleurs ses dix années, la première spécification remonte, déjà, à 1998. La volonté initiale était d'offrir un langage générique et souple, adaptable selon les besoins. Comme à ses débuts, XML reste un langage capable de s'adapter à vos exigences, en créant un dérivé, grâce aux balises, car XML est un langage de balisage.

La plupart du temps, ce code XML est généré automatiquement, ou sinon les outils le cachent. On intervient en général pour décrire le contenant et le contenu, pour valider, pour transformer. En principe, le développeur n'a pas à écrire lui-même son code XML ou du code dialecte. Cependant, quand il y a un problème dans la génération, il faut bien rentrer dans le code et là, on constate que le langage demeure verbeux et lourd.

Pourquoi faire un grand dossier sur ce sujet dans Programmez ! ? Depuis longtemps, nous l'abordons, sans y avoir consacré un dossier spécial. C'est chose faite avec le présent numéro et le suivant, car le sujet est vaste. Nous allons ainsi aborder différents aspects : qualité et sécurité XML, XML et les langages de programmation, XML et les bases de données. Nous reviendrons aussi sur des notions fondamentales comme la sérialisation, la validation ou encore XSLT, sans oublier les outils. Nous nous interrogerons sur la qualité du code XML mais aussi sur la sécurité liée à ce même code. Et les questions sont nombreuses. Sans le savoir, les utilisateurs de suites bureautiques utilisent et sauvegardent des documents en XML, que ce soit avec OpenOffice ou Office 2007. Ce dernier point sera abordé dans les futurs numéros.

Actuellement, nous sommes en XML 1.x et le langage évolue lentement, contrairement aux nombreux dérivés. Ce n'est pas un mal, cela permet une stabilité et une pérennité. Depuis plusieurs années, on parle régulièrement de XML 2.0. Nous ferons aussi un point sur cette future mouture… dans la 2e partie.

Il est temps de se plonger dans le *eXtensible Markup Language.*

François Tonic

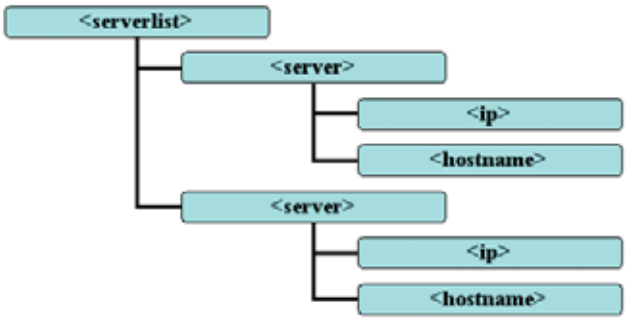
XML : les fondamentaux !

Aujourd'hui il est difficile de parler de programmation sans parler de XML. Que ce soit pour le stockage ou l'échange de données, pour la modélisation ou mille autres domaines, XML est omniprésent. Les développeurs en usent et en abusent autant pour sa simplicité d'utilisation, son format – relativement – lisible par un humain équipé d'un lecteur de texte, que pour sa pérennité. Nous allons donc détailler les causes d'un tel succès ainsi que les pièges dans lesquels ne pas tomber lorsque la question du XML se pose.

Le XML est un langage de balise, c'est-à-dire que, tout comme son ancêtre SGML, il s'agit d'un fichier texte contenant un marquage spécial (la " balise "). Ces balises délimitent des zones de données hiérarchisées dans le document. Par exemple, la représentation d'un parc de serveurs peut être faite de la façon (simpliste) suivante :

```
<serverlist>
    <server>
   <ip>192.168.0.1</ip>
   <hostname>firewall</hostname>
    </server>
    <server>
   <ip>192.168.0.2</ip>
   <hostname>web_server</hostname>
    </server>
</serverlist>
```

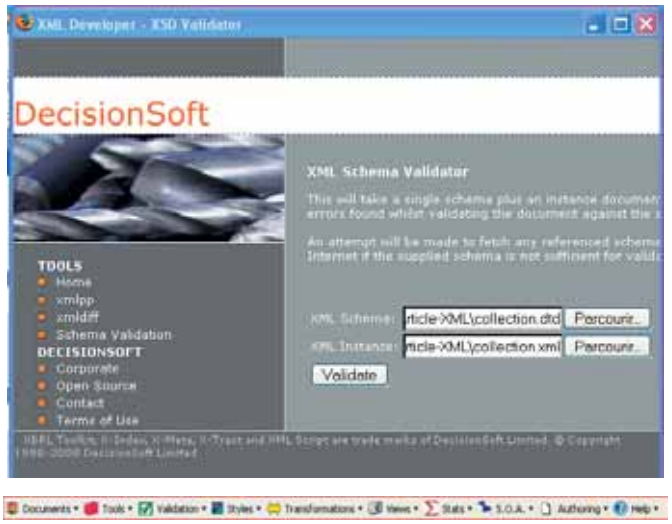
Chaque serveur est représenté dans la liste par une entité <server> qui contient elle-même une adresse IP ainsi qu'un nom d'hôte. Comme on le constate, aussi bien grâce à l'indentation qu'à la simple lecture du " code ", l'information est stockée sous une forme arborescente pouvant être représentée graphiquement de la façon suivante :



La syntaxe lisible et facilement compréhensible à la fois par un humain ou par un programme est simultanément une force et une faiblesse du XML. C'est d'ailleurs une des raisons qui a participé à son adoption massive dans le développement et particulièrement dans les applications ayant recours à des données complexes.

XML pour la gestion de données

Dans de très nombreux cas, le format XML est utilisé pour stocker des données pseudo-structurées dans un contexte potentiellement privé de base de données. Il est fréquent de rencontrer des applications proposant plusieurs méthodes de sauvegarde de données (souvent extensibles par plug-in), et parmi celles-ci, il est commun de trouver un format basé sur du XML en complément d'un autre s'appuyant, par exemple, sur du SQL. On peut citer Amarok et son moteur de sauvegarde de la collection de musique.



Il est vrai que XML ne manque pas d'atouts à faire valoir en ce qui concerne ce type d'utilisation. En effet, il est une notion qui est vitale en matière de stockage de données : la pérennité des données, (les violents débats autour de la normalisation du format OOXML en témoignent…) Et, de fait, XML est un format extrêmement adapté à ce genre d'utilisation :

- un format texte ;
- peut représenter la majorité des structures de données informatiques (listes, arbres, etc.) ;
- syntaxe stricte normalisée par un comité international ;
- une structure hiérarchique adaptée à une grande partie des données à stocker ;
- indépendant de la plate-forme ;
- possible validation via un schéma ;

Parmi les exemples " célèbre " de formats de fichiers basés sur du XML, on peut citer le sulfureux OOXML, l'UML, le MathML, les formats compressés de la suite Open Office (.odt en tête) ainsi qu'une pléthore de formats ouverts. Une des conséquences induites par les avantages cités est que les documents XML sont facilement maintenables. Ainsi la compatibilité avec les versions précédentes est relativement simple à mettre en place. Prenons l'exemple, original, d'une application multimédia gérant une collection de fichiers (musique, vidéo, images). Dans un premier temps cette application ne gère qu'un fichier d'index qui ne contient rien d'autre que le nom du fichier et son répertoire. Exemple :

```
<?xml version="1.0" encoding="UTF-8"?>
<collection>
    <item>
   <filename>zz_top-La_Grange.mp3</filename>
   <directory>/home/arnaud/Musique</directory>
    </item>
    <item>
   <filename>Eric_Johnson-Cliffs_of_Dover.mp3</filename>
   <directory>/home/arnaud/Musique</directory>
    </item>
    <item>
   <filename>slack-get_logo.png</filename>
   <directory>/home/arnaud/Images</directory>
    </item>
    <item>
   <filename>Gekijouban_Tengen_Toppa_Gurren_Lagann.mp4</filename>
   <directory>/home/arnaud/Video</directory>
    </item>
</collection>
```

Imaginons maintenant une autre application, qui n'a pas généré le fichier de collection, mais qui l'utilise. Prenons un script automatisé qui vérifie de façon journalière la date de dernier accès de chacun des fichiers de la collection, dans le but de déplacer ceux qui n'ont pas été utilisés depuis au moins un mois dans un répertoire spécial. En Perl, ce script serait le suivant :

```
#!/usr/bien/perl
use strict;
use warnings;
use XML::Simple;
use File::Copy;
my $unused= '/home/arnaud/UnusedMedias';
die "Fichier de collection illisible ou non spécifié.\n" unless( -e $ARGV[0]
&& -r $ARGV[0] );
my $collection = XML::Simple::XMLin( $ARGV[0] , ForceArray => ['item']
, ForceContent => 1 );
foreach my $item ( @{ $collection->{'item'} } ){
    my $file = $item->{'directory'}->{'content'}.'/'.$item->{'filename'}->
{'content'};
    if( -e $file){
   my @stats = stat($file);
   if( (time()-$stats[8]) >= (86400*30) ){
       move($file,$unused);
   }
    }
}
```

Certains passages de ce script méritent une petite explication. Tout d'abord, le fichier de collection est passé en argument du script, puis accédé via le tableau des options @ARGV. Nous avons ensuite recours au module XML::Simple (disponible sur le CPAN) pour analyser le fichier. Ce module retourne une structure de données de type " hashref " représentant le contenu du fichier de collection. Cependant, le comportement de ce module est parfois inadapté au traitement que nous désirons mettre en place, à savoir une boucle itérative. Nous forçons donc XML::Simple à toujours mettre les éléments <item> dans un " arrayref " (directive ForceArray => ['item']). Sans cela, s'il n'y a qu'un élément <item> dans le document, XML::Simple le dispose directement dans le hashref, ce qui ne manquera pas de provoquer un comportement désagréable lors du foreach. Cette boucle contient d'ailleurs une partie un peu ésotérique au premier abord : @{ $collection->{'item'} }. Rappelons, pour ceux qui ne sont pas coutumiers du langage Perl, que c'est une des façons possibles de déréférencer une référence sur une liste (arrayref). La fonction XMLin() se voit aussi attribué le paramètre ForceContent => 1, qui permet l'ajout d'une balise virtuelle " content " désignant le contenu d'une balise (à différencier de ses attributs). Pour finir nous testons, pour chaque fichier existant, que l'écart de temps (en secondes depuis l'epoch) entre maintenant et le dernier accès est inférieur au nombre de secondes dans un mois.

Que ce passe-t-il maintenant lors de l'évolution de l'application multimédia, particulièrement si le fichier de collection se voit enrichi de quelques balises ou attributs ? La réponse est simple : rien. Le même script exécute exactement la même besogne de la même façon et même si, par exemple, la nouvelle application génère le XML suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<collection>
    <item type="music">
   <filename>zz_top-La_Grange.mp3</filename>
   <directory>/home/arnaud/Musique</directory>
   <artist>ZZ Top</artist>
   <title>La Grange</title>
    </item>
    <!- … ->
    <item type="movie">
   <filename video-codec="xvid" audio-codec="mp3">Gekijouban_
Tengen_Toppa_Gurren_Lagann.mp4</filename>
   <directory>/home/arnaud/Video</directory>
   <style>Japan Anime</style>
    </item>
</collection>
```

Ceci dit, tordons d'emblée le cou à une idée préconçue très répandue : ceci n'est en rien magique et le format XML a autant sa part dans ce succès que le fait que le script soit bien codé ! En effet, nous avons pris deux précautions dans l'appel au parser :

1. conditionnement de la structure de données via la directive ForceArray ;
2. différenciation explicite du contenu d'une balise et de ses attributs.

Tentez l'expérience de supprimer ces directives d'analyse pour voir…

Cette compatibilité implicite est très rassurante pour le développement d'architectures lourdes où les développements sont potentiellement longs et coûteux. Une certaine sérénité s'impose naturellement quant à la pérennité des données, autant grâce à cette hiérarchisation rassurante qu'a la lisibilité de ce format textuel.

Cependant, ces possibilités seraient bien vite anéanties s'il n'était pas possible de vérifier qu'un fichier se conforme bien à un standard. C'est le rôle des fichiers de description.

Description d'un fichier XML

Un fichier XML en soi est un fichier texte, aucune force suprême ou métaphysique ne le prédispose à l'ordre et à la rigueur. Ce rôle est dévo-

lu aux schémas de description du document. Ceux-ci spécifient les éléments autorisés dans le document, leur organisation hiérarchique, leur type de données, etc. Il existe de nombreux " langages " de description tels que l'historique DTD, les plus récents XML Schema et Relax NG (prononcez " relaxing ") ou encore Schematron. Nous allons nous attarder sur deux de ceux-ci : DTD et Relax NG.
En premier lieu, intéressons nous à l'ancêtre DTD (pour Document Type Definition). Langage de description lui aussi dérivé du SGML, il possède l'avantage de pouvoir être embarqué en ligne dans un document XML. Il est en outre standardisé depuis la norme XML 1.0. De plus, il est très ancré par tradition dans les développements XML. Ces menus avantages sont, d'un point de vue technique, bien loin de compenser les monstrueux inconvénients qui s'y rattachent. Citons par exemple l'absence de support des namespaces, la difficulté de mise en place des choix conditionnels, ou encore, le fait que la syntaxe DTD ne soit pas XML, ce qui complique largement la lecture de tels documents.
Voyons la définition DTD de la dernière version de la collection :

```
<!ELEMENT collection (item+)>
<!ELEMENT item (filename,directory,artist?,title?,style?)>
<!ATTLIST item type (music|video|image|unknow) "unknow">
<!ELEMENT filename (#PCDATA)>
<!ATTLIST filename audio-codec CDATA #IMPLIED >
<!ATTLIST filename video-codec CDATA #IMPLIED >
<!ELEMENT directory (#PCDATA)>
<!ELEMENT artist (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT style (#PCDATA)>
```

Un élément est déclaré via <!ELEMENT> et un attribut via <!ATTLIST>. L'élément possède des enfants qui doivent apparaître dans l'ordre dans lequel ils sont déclarés (si l'ordre n'est pas important on remplacera alors les virgules par des " pipe ". Le mot clé #IMPLIED signifie l'aspect facultatif de l'élément ou attribut.
On remarque là que nous sommes assez loin de la syntaxe hiérarchique du XML. Adonnons nous maintenant au même exercice avec la syntaxe compacte de Relax NG :

```
element collection { item+ }
element item {
    attribute type ("music"|"video"|"image"|"unknow"),
    element filename {
   text,
   attribute video-codec {text},
   attribute audio-codec {text},
    },
    element directory {text},
    (
   (
       element artist {text},
       element title {text},
   )|(
       element style {text},
   )
    )?
}
```

Bien que plus verbeux que son ancestral homologue, la syntaxe RNC (Relax NG Compact) est bien plus compréhensible. On peut ensuite remarquer l'ajout d'une information de choix entre les balises <artist>, <title> et <style>.
En effet, le premier couple n'est jamais présent en même temps que <style> car ils sont relatifs à des types d'items différents. Il ne reste plus qu'à transformer ce fichier RNC en fichier RNG (la syntaxe étendue en XML) avec un outil comme Trang par exemple, puis de valider le document avec un outil comme Jing (ou la libxml2 qui fait ça très bien et qui dispose de nombreux binding vers tous types de langages de scripts). Cette transformation nous amène tout naturellement vers les feuilles de style XSLT…

XSLT : la vraie force du XML

Souvent négligées, les feuilles XSL sont pourtant LA raison qui devrait influencer le choix du XML. En effet, ces feuilles de styles sont de véritables filtres à tout faire pour un document XML. Capables de transformer n'importe quel document vers n'importe quel format le plus simplement du monde. C'est ce qui fait du XML le langage d'échange de données par excellence. Prenons notre fichier de collection par exemple, pour le transformer en SQL, il nous suffit d'écrire le fichier XSL suivant :

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL
/Transform">
<xsl:template match="/">
   <xsl:for-each select="collection/item">
INSERT INTO Collection VALUES("<xsl:value-of select="filename"/>","
<xsl:value-of select="directory"/>","<xsl:value-of select="./@type"/>");
   </xsl:for-each>
</xsl:template>
</xsl:stylesheet>
```

Ce qui signifie que pour chaque élément item (enfant de collection) le processeur génère une ligne SQL avec les valeurs des balises filles de l'item. La sortie générée est du type :

```
INSERT INTO Collection VALUES("zz_top-La_Grange.mp3" ,"/home/
arnaud/Musique", "music" );
```

Il y a suffisamment à dire sur les XSLT pour faire un article unique, je vous invite donc à vous documenter pour en savoir plus sur ce sujet qui mérite réellement tout votre attention.

Conclusion

Le XML est un format qui a de très nombreux avantages, cependant il est crucial de ne pas l'utiliser hors de son périmètre. La première question à poser au moment du choix est : *" comment peut-on faire sans XML ? "* suivi de très près par *" quel impact sur l'évolutivité et les performances ? ".* Typiquement, il est souvent possible de se passer de XML quand les données ne sont pas hiérarchisées. Par contre, quand il s'agit de trouver une solution d'échange/transformation de données, la question ne se pose même pas : foncez !

Arnaud Dupuis
Consultant Uperto (Open Source Business Unit) - Devoteam group

ABONNEMENT PDF : 2,73 € le numéro www.programmez.com

Validation des documents XML

1re partie

La validation va renforcer la qualité des échanges en contraignant l'émetteur de données et le consommateur de données à vérifier la cohérence des données structurées en XML. Par cohérence, il faut entendre à la fois le vocabulaire (éléments, attributs et espaces de noms) mais également, chose aussi importante, l'ordre et les quantités. En fin de compte, la validation revient à établir un visa sur le document XML.

La validation est donc importante. Il faut cependant en réserver l'usage à des documents qui présentent une certaine complexité et qui sont de taille raisonnable (moins de 100 Mo, par exemple). Lorsque la taille d'un document est trop élevée, la validation peut devenir très gourmande en ressources car, comme nous allons le voir, certaines règles nécessitent une vision globale du document et donc une présence en mémoire. L'idéal étant de vérifier les temps moyens, de contrôler si la réactivité dans un processus métier est suffisante. La validation, même si elle n'est pas toujours employée en production, peut toujours servir lors du développement à contrôler que les données XML sont correc-tement structurées. Dans un processus de développement en spirale (prototypages multiples qui tendent vers une version finale), les fluctuations des demandes client et les évolutions des programmes rendent indispensable la validation, ne serait ce que pour éviter d'éventuels problèmes de régression. La plupart des outils, et notamment les parseurs XML, proposent des outils de validation. Les parseurs courants supportent une ou plusieurs formes de grammaires. Les DTD (Document Type Definition), étant la forme la plus ancienne, sont présentes dans la plupart des outils. Viennent ensuite ce qu'on nomme les schémas W3C, une forme de grammaire plus moderne mais également plus complexe. Enfin, il existe d'autres alternatives dont l'avenir est encore incertain, même si certains développeurs semblent déjà conquis par la simplicité de RelaxNG, par exemple.

La première forme de validation par DTD

Une DTD (Document Type Definition) est une forme de grammaire relativement ancienne car issue de l'univers SGML (Standard Generalized Markup Language). Elle a l'avantage d'être rapide à écrire, tout en présentant l'inconvénient d'être pauvre en possibilités de contrôle (typage de données, par exemple). Autre point négatif, les espaces de noms sont difficilement gérables car il faut intégrer, dans la grammaire, les préfixes, ce qui est contraire à l'esprit des espaces de noms.

Une DTD peut être interne ou externe au document XML. L'usage voudra que l'on privilégie la forme externe pour des raisons de maintenance et de facilité d'accès. Dans cette dernière forme, le parseur XML trouvera une référence dans chaque document XML vers la DTD externe par l'instruction d'en-tête DOCTYPE (voir le chapitre 2).

Par exemple, un document XML ayant une DTD externe cours.dtd, située dans le même répertoire que notre document XML (accès relatif), se présente sous la forme :

```
<?xml version="1.0"?>
<!DOCTYPE cours SYSTEM "cours.dtd">
<cours> … </cours>
```

Commençons maintenant à analyser la syntaxe d'une DTD. Tout d'abord, il est important de comprendre que cette syntaxe, même si elle est liée à un usage XML, n'est pas à base de balise mais à base d'instructions, selon la syntaxe <!INSTRUCTION…> (conséquence de la parenté

La définition d'un élément

L'élément (ou balise) est exprimé par l'instruction ELEMENT suivie du nom de l'élément que l'on souhaite décrire et de son contenu. Ce dernier n'englobe que les éléments situés directement sous cet élément (les éléments fils). Voici une synthèse de cette syntaxe :

```
<!ELEMENT unNom DEF_CONTENU>
```

DEF_CONTENU peut contenir :

- EMPTY : l'élément n'a pas de contenu ; il est donc vide. Il peut cependant avoir des attributs.
- ANY : l'élément peut contenir n'importe quel élément présent dans la DTD.
- (#PCDATA) : l'élément contient du texte. Le caractère # est là pour éviter toute ambiguïté avec une balise et indique au parseur qu'il s'agit d'un mot-clé. PCDATA signifie *Parsable Character DATA*.
- Un élément placé entre parenthèses comme (nom_element). Le nom d'un élément désigne une référence vers un élément décrit dans une autre partie de la DTD
- Un ensemble d'éléments séparés par des opérateurs, le tout placé entre parenthèses. L'opérateur de choix, représenté par le caractère |, indique que l'un ou l'autre de deux éléments (ou deux ensembles d'éléments) doit être présent. L'opérateur de suite (ou séquence), représenté par le caractère, indique que les deux éléments (ou les deux ensembles d'éléments) doivent être présents. Des parenthèses supplémentaires peuvent être utilisées pour lever les ambiguïtés.

Quelques exemples :

```
<!ELEMENT personne (nom_prenom | nom)> <!ELEMENT nom_prenom
(#PCDATA)> <!ELEMENT nom (#PCDATA)>
```

Cela nous autorise deux documents XML, soit :

```
<personne>
 <nom_prenom>Brillant Alexandre</nom_prenom> </personne>
```

ou bien :

```
<personne> <nom>Brillant</nom> </personne>
```

Autre cas avec l'opérateur de séquence.

```
<!ELEMENT personne(prenom,nom)> <!ELEMENT prenom (#PCDATA)>
 <!ELEMENT nom (#PCDATA)>
```

Ici, l'opérateur de séquence limite les possibilités à un seul document XML valide :

```
<personne> <prenom>Alexandre</prenom> <nom>Brillant</nom>
</personne>
```

Les contenus (élément ou groupe d'éléments) peuvent être quantifiés par les opérateurs *, + et ?. Ces opérateurs sont liés au concept de cardinalité. Lorsqu'il n'y a pas d'opérateur, la quantification est de 1 (donc toujours présent). Voici le détail de ces opérateurs :

- * : 0 à n fois ;
- + : 1 à n fois ;
- ? : 0 ou 1 fois.

Quelques exemples :

<!ELEMENT plan (introduction?,chapitre+,conclusion?)> L'élément plan contient un élément introduction optionnel, suivi d'au moins un élément chapitre et se termine par un élément conclusion optionnel également. <!ELEMENT chapitre (auteur*,paragraphe+)> L'élément chapitre contient de 0 à n éléments auteur suivi d'au moins un élément paragraphe. <!ELEMENT livre (auteur?,chapitre)+> L'élément livre contient au moins un élément, chaque élément, étant un groupe d'éléments où l'élément auteur, est optionnel et l'élément chapitre est présent en un seul exemplaire.

La définition d'un attribut

Les attributs sont précisés dans l'instruction ATTLIST. Cette dernière, étant indépendante de l'instruction ELEMENT, on précise à nouveau le nom de l'élément sur lequel s'applique le ou les attributs. On peut considérer qu'il existe cette forme syntaxique :

```
nom TYPE OBLIGATION VALEUR_PAR_DEFAUT
```

Le TYPE peut être principalement :

- CDATA : du texte (Character Data) ;
- ID : un identifiant unique (combinaison de chiffres et de lettres) ;
- IDREF : une référence vers un ID ;
- IDREFS : une liste de références vers des ID (séparation par un blanc) ;
- NMTOKEN : un mot (donc pas de blanc) ;
- NMTOKENS : une liste de mots (séparation par un blanc) ;
- Une énumération de valeurs : chaque valeur est séparée par le caractère |.

L'OBLIGATION ne concerne pas les énumérations qui sont suivies d'une valeur par défaut. Dans les autres cas, on l'exprime ainsi :

- #REQUIRED : attribut obligatoire.
- #IMPLIED : attribut optionnel.
- #FIXED : attribut toujours présent avec une valeur. Cela peut servir, par exemple, à imposer la présence d'un espace de noms.

La VALEUR_PAR_DEFAUT est présente pour l'énumération ou lorsque la valeur est typée avec #IMPLIED ou #FIXED.

Quelques exemples :

```
<!ATTLIST chapitre titre CDATA #REQUIRED auteur CDATA #IMPLIED>
```

L'élément chapitre possède ici un attribut titre obligatoire et un attribut auteur optionnel.

```
<!ATTLIST crayon
 couleur (rouge|vert|bleu) "bleu">
```

L'élément crayon possède un attribut couleur dont les valeurs font partie de l'ensemble rouge, vert, bleu.

La définition d'une entité

Les entités sont déclarées par l'instruction ENTITY. Comme nous l'avons abordé dans le chapitre précédent, l'entité associe un nom à une valeur. Ce nom est employé dans le document XML comme une forme d'alias ou de raccourci vers la valeur suivant la syntaxe &nom;. La valeur d'une entité peut être interne ou externe.

Dans la forme interne la syntaxe pour déclarer une entité est simplement la suivante :

```
<!ENTITY nom "VALEUR">
```

Dans la forme externe, on se retrouve avec le même principe qu'avec l'instruction DOCTYPE en tête du document XML assurant le lien vers une DTD. Les mots-clés SYSTEM et PUBLIC servent donc à réaliser un lien vers une valeur présente dans un fichier.

Exemple :

```
<!ENTITY nom SYSTEM "unTexte.txt">
```

L'entité nom est ici liée au contenu du fichier unTexte.txt.

Les entités ne s'appliquent pas uniquement au document XML. Elles peuvent également servir à la réalisation de la DTD pour limiter les répétitions de blocs de définition (par exemple, un attribut présent dans plusieurs éléments). Cette forme d'entité est appelée entité paramétrique et doit être déclarée suivant la syntaxe : <!ENTITY % nom "VALEUR"> L'instruction %nom; sert à utiliser une entité paramétrique dans la DTD.

Exemple :

```
<!ENTITY % type_defaut "CDATA"> <!ATTLIST chapitre titre %type_defaut;
#REQUIRED>
```

Dans cet exemple, nous avons créé une entité paramétrique type_defaut qui est associée à un type (CDATA) pour un attribut. Cette valeur est ensuite employée pour définir le typage de l'attribut titre de l'élément chapitre. Grâce aux entités paramétriques, il est également possible d'activer ou de désactiver des blocs de définition. Ces blocs suivent la syntaxe suivante :

```
<![Valeur[ Partie de DTD ]]>
```

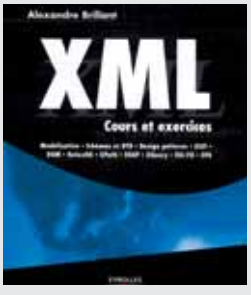
Si Valeur vaut INCLUDE alors la partie de DTD est activée. Si Valeur vaut IGNORE cette partie est ignorée.

Exemple :

```
<!ENTITY % anglais 'INCLUDE'>
<![%anglais;[
<!ATTLIST chapitre
 langue (anglais|français) "français">

]]>
```

Dans cet exemple, on définit un attribut langue pour un élément chapitre uniquement si l'entité paramétrique anglais a la valeur INCLUDE.

Dans la seconde partie nous verrons la validation par schéma.

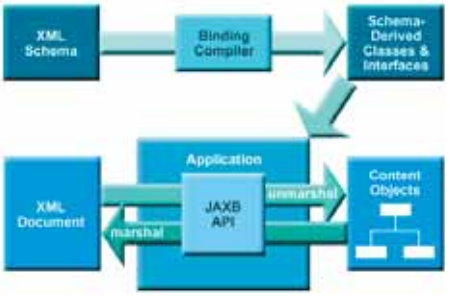


Extrait de l'ouvrage :
XML Cours et exercices
Editions Eyrolles

Alexandre Brillant
29,90 €
284 pages
www.editions-eyrolles.com

StAX : une API XML pour Java

Dans la sphère des applications d'entreprise, il existe un besoin croissant en applications Java capables de transformer des documents XML ainsi que des structures de données en arbre. Sans les bons outils, l'utilisation de documents structurés sera inefficace. De nombreuses solutions existent et dans la majorité des cas on se retrouve confronté à du code Java, des feuilles de styles XSLT ou encore des formats propriétaires.



Il existe donc un large panel d'entrées pour les analyseurs de documents : URL, flux XML, arbre DOM, événements SAX etc. On retrouve leurs équivalents pour les sorties. Le challenge principal d'une API de transformation est d'être à même de prendre en charge les différentes combinaisons d'entrées/sorties sans devenir un spécialiste d'une technologie particulière. StAX est une API qui comble les manques de SAX et de DOM. Elle se base sur le concept de flux XML contrôlé par l'application, comme nous allons le découvrir.

Origine du projet

BEA est bien connu pour son activité autour de la plate-forme Java et notamment via son offre commerciale de serveur d'application WebLogic. L'éditeur offre une gamme de produits destinée aux entreprises. Une offre essentiellement basée sur les technologies SOA et BPM. Cependant, la société propose également des outils tiers dont fait partie StAX. BEA a bénéficié du support de Sun, ce qui lui a permis d'obtenir l'approbation en 2004 pour la JSR du projet.

L'objectif premier de l'API StAX est d'offrir une solution de parsing plus efficace. Celle-ci est notamment facile d'utilisation grâce à un fonctionnement simple se basant sur un mécanisme d'itérations. Cela permet au développeur de contrôler le flux de l'analyse par une approche procédurale. StAX a donc été créé pour corriger les limitations des API déjà bien présentes : SAX et DOM. Simple d'utilisation et performante, elle ne supporte cependant pas la technologie XPath. Il existe différentes implémentations de l'API StAX. On y retrouve tous les acteurs du monde Java : BEA comme nous venons de le voir, mais également Oracle et Sun qui la proposent en standard depuis Java 6 au sein du JDK. Les éditeurs sont donc très actifs sur cette API, cependant la communauté du libre n'est pas en reste. Codehaus héberge de nombreux projets dont une implémentation de StAX et des outils complémentaires comme StaxMate que nous aborderons en fin d'article.

Principe de fonctionnement

StAX est puissant et permet de résoudre de nombreuses problématiques rencontrées en entreprise. Cependant son fonctionnement reste particulier et nécessite un effort d'apprentissage de la part des développeurs. Le gain en retour est bien là et l'approche itérative est très efficace une fois maîtrisée.

SAX suit le modèle de " push ", c'est-à-dire qu'il permet l'envoi d'événements avant même que le document soit analysé dans son intégralité. Cependant, cette approche comporte un défaut : c'est le parser qui contrôle le flux d'analyse. StAX propose donc une approche différente qui permet à l'application de reprendre le contrôle du flux. On parle donc de parsing selon le modèle " pull " : les événements sont ici générés par l'application. Cela revient à dire qu'avec StAX le développeur a la possibilité de suspendre le traitement, de sauter des éléments ou encore de traiter plusieurs documents en même temps.

En ce qui concerne l'API DOM, les documents XML sont chargés en mémoire sous la forme d'une structure DOM. Cela réduit l'efficacité des traitements et limite le parsing a des documents relativement simples du fait des contraintes de mémoire.

Pourquoi Stax

De par ses caractéristiques, StAX se présente comme un bon choix pour certains cas de figures. En revanche, son utilisation peut ne pas s'appliquer à certaines situations. Il est important d'avoir conscience de ses avantages et de ses inconvénients. Nous allons donc comparer les trois API de parsing XML que sont DOM, SAX et StAX.

Nous allons commencer par le cas où StAX se démarque le plus de ses concurrents. Dans certains cas complexes, le traitement de documents XML peut être décomposé en une série de transformations XML. Dans ce cas, il ne sera pas efficace de parser à nouveau le document pour chaque étape. Avec StAX, il est possible d'établir des pipelines qui permettront de communiquer entre chaque étape. StAX est la seule API JAXP à offrir cette possibilité qui permet d'optimiser largement les traitements de documents complexes. SAX est souvent utilisé pour réaliser des pipelines, cependant il n'est pas capable d'écrire du code XML. StAX est donc également un bon choix pour la transformation de docu-

TrAX, un projet Apache

TrAX est un projet développé par la communauté Apache. L'objectif de ses créateurs est d'étendre le spectre de fonctionnalités attribué aux API JAXP. TrAX se veut donc d'un usage généraliste et son point particulier est sa capacité en terme de manipulation de documents XML. L'API offre nativement une interface pour réaliser cette fonctionnalité. Voici un exemple :

```
TransformerFactory tf = TransformerFactory.newInstance() ;
Transformer t = tf.newTransformer(xsltSource);
```

TrAX n'est donc pas un concurrent des API StAX, SAX et DOM. Il constitue un pont entre ces différents outils parsing et les moteurs de transformation tels que XSLT.

ments XML en d'autres documents XML, cependant, sur ce point DOM est plus simple à utiliser.

SAX et StAX ont la possibilité de visualiser des structures de données arbitraires en tant que flux d'événement XML. Avec SAX, une classe qui implémente l'interface XMLReader est créée. Cette classe effectue l'analyse sur les données non structurées et les publie ensuite en tant qu'événements aux classes de traitements (handlers). Avec StAX, l'approche est différente. Une classe est écrite pour analyser le contenu non structuré et construit des événements XML en utilisant la classe XMLEventFactory. DOM peut également être utilisé mais il sera le moins efficace dans cette situation.

Le binding de données est un autre type de transformation XML que les développeurs rencontrent fréquemment. Le concept est simple : transformer un objet en document XML et vice versa. On entend aussi parler de sérialisation XML. SAX peut être utilisé pour instancier un objet Java à partir d'un document XML, cependant l'inverse n'est pas vrai et une autre approche devra être utilisée, DOM par exemple. StAX offre une solution consistante qui fonctionnera dans les deux cas et qui permettra ainsi de conserver une seule API, gage de bénéfice en terme de maintenance et d'évolution du code. Pour couronner le tout, StAX est également le plus performant grâce à son architecture de " pull " d'événements : moins de mémoire est nécessaire.

Plutôt que de transformer un document XML en objet, il est parfois utile de conserver la structure XML et de l'utiliser directement en tant que donnée métiers. L'exemple le plus connu à l'époque du Web 2.0 est bien évidemment Ajax. Par exemple, si vous récupérez les données depuis un service web et que vous souhaitez les transformer en code XHML, il n'y a aucun avantage à passer par le monde objet de Java. DOM reste donc le meilleur choix. C'est la seule API à permettre des actions CRUD (Create, Read, Update, Delete) et un accès aléatoire à un document XML.

Le tableau suivant résume les utilisations possibles de chacun de ces frameworks. StAX sort vainqueur dans la majorité des cas.

	Stax	SAX	DOM
XML vers XML	Choix acceptable pour les structures simples	N/A	Choix acceptable pour les structures modérément complexes
Structures de données vers XML	Bon choix	Bon choix	N/A
Data binding	Bon choix	Fonctionnera uniquement pour les fragments de documents XML	Devra être utilisé avec d'autres API pour bufferiser les structures complexes
Domain model	N/A	N/A	La meilleure solution
Pipelines XML	La meilleure solution	Fonctionnera en lecture	N/A

Cas pratique : Lecture

Ici, nous allons nous appuyer sur l'implémentation de Sun présente dans la version 6 du JDK. Celle-ci est de loin la plus répandue et offre l'avantage d'être disponible au sein du JDK standard. Nous allons commencer par un exercice de traitement en lecture d'un document XML. Pour cela, nous allons utiliser essentiellement trois classes : XMLInputFactory qui permet la création du flux XML ainsi que des classes qui prendront en charge les événements (XMLStreamReader). Enfin, l'interface XMLStreamConstants contient les constantes qui seront utilisées pour le parsing.

```
import java.io.FileReader;
import java.io.Reader;

import javax.xml.stream.XMLInputFactory;
import javax.xml.stream.XMLStreamConstants;
import javax.xml.stream.XMLStreamReader;

public class XMLStreamReaderDemo {
  public static void main(String[] args) throws Exception {
    XMLInputFactory factory = XMLInputFactory.newInstance();

    Reader fileReader = new FileReader("source.xml");
    XMLStreamReader reader = factory.createXMLStreamReader(fileReader);

    while (reader.hasNext()) {
      process(reader);
      reader.next();
    }
  }

  private static void process(XMLStreamReader reader) {
    int eventType = reader.getEventType();
    switch (eventType) {
    case XMLStreamConstants.START_ELEMENT:
      System.out.println("Start element: " + reader.getLocalName());

      int count = reader.getAttributeCount();
      for (int i = 0; i < count; i++) {
        String name = reader.getAttributeLocalName(i);
        String value = reader.getAttributeValue(i);
       System.out.println("\tAttribute name/value: " + name + "/" + value);
      }
      break;

    case XMLStreamConstants.END_ELEMENT:
      System.out.println("End element: " + reader.getLocalName());
      break;

    case XMLStreamConstants.CHARACTERS:
      System.out.println("Text: " + reader.getText());
      break;
    default:
      break;
    }
  }
}
```

Comme expliqué précédemment, on retrouve ici un déroulement itératif du traitement du fichier XML. Chaque nœud est affiché de manière assez simple et la complexité reste malgré tout limitée.

Cas pratique : Ecriture

L'écriture de document XML en StAX est ce qu'il y a de plus simple. Le développeur est cependant très proche de la structure XML de ses documents, ce qui peut constituer un défaut d'un certain point de vue. Ici, deux objets sont nécessaires. De manière analogue à la lecture de

documents, on retrouve la classe XMLOutputFactory qui permet d'ouvrir un flux de sortie et d'instancier l'objet qui va écrire sur le flux XML (XMLStreamWriter).

```
import javax.xml.stream.XMLOutputFactory;
import javax.xml.stream.XMLStreamWriter;

public class XMLStreamWriterDemo {
  public static void main(String[] args) throws Exception {
    XMLOutputFactory factory = XMLOutputFactory.newInstance();
    XMLStreamWriter writer = factory.createXMLStreamWriter(System.out);

    writer.writeStartDocument("1.0");

    writer.writeStartElement("catalog");

    writer.writeStartElement("book");

    writer.writeAttribute("id", "1");

    writer.writeStartElement("code");
    writer.writeCharacters("I01");
    writer.writeEndElement();

    writer.writeStartElement("title");
    writer.writeCharacters("This is the title");
    writer.writeEndElement();

    writer.writeStartElement("price");
    writer.writeCharacters("$2.95");
    writer.writeEndElement();

    writer.writeEndDocument();

    writer.flush();
    writer.close();
  }
}
```

StaxMate

StAX peut être complété par un framework très léger qui lui apporte cependant quelques améliorations pour le traitement de flux XML. StaxMate ajoute une couche d'abstraction à StAX qui permet notamment la gestion :

- Des curseurs, en s'appuyant sur des XMLStreamReaders
- Des objets de sorties, qui s'appuient quant à eux sur des XMLStreamWriters

Ces deux concepts correspondent respectivement aux StaxMateIterators et aux StaxMateOutputters et permettent de s'affranchir de la structure pure du document XML en travaillant à un niveau d'abstraction supplémentaire. Les conséquences en terme de performances restent limitées.

```
XMLInputFactory f = XMLInputFactory.newInstance();
XMLStreamReader sr = f.createStreamReader(new FileInputStream("my
doc.xml"));

SMInputCursor rootCrsr = SMInputFactory.rootElementCursor(sr);
rootCrsr.getNext();
assert(rootCrsr.getQName(), new QName("root"));
String id = rootCrsr.getAttrValue("id");
SMCursor childCrsr = rootCrsr.childElementCursor();
String value = crsr.collectDescendantText()
```

StaxMate est hébergé par la communauté Codehaus. Cet hébergeur de projets Open Source est particulièrement réputé pour héberger des projets agiles permettant la mise en œuvre de méthodes de développement RAD. On peut citer notamment les projets JRuby, Grails (framework mvc), Jetty (serveur web), ou encore Castor (framework de mapping objet/relationnel).

Conclusion

StAX est de plus en plus utilisé et enseigné dans les écoles d'informatique. Quelques projets l'utilisent et les développeurs commencent à en prendre connaissance. Dans les années à venir StAX devrait devenir la solution de référence en terme de manipulation de données XML. Cependant il ne couvre pas tous les domaines d'applications et c'est essentiellement à l'API SAX que celui-ci fait de l'ombre. DOM reste encore pour un bon moment la seule solution pour les applications web intégrant du contenu dynamique.

Loïc Guillois

TÉMOIGNAGE

"J'ai du mal à imaginer comment je pourrais me passer d'XML aujourd'hui."

Développeur et créateur de l'outil Jaxe, Damien Guillaume nous donne son point de vue sur XML et son rôle dans la programmation actuelle.

Programmez ! : XML est aujourd'hui au coeur des échanges, des applications. Quels sont les défis et problématiques pour le développeur, liés au XML ?

Damien Guillaume : *Je pense que le défi principal est, comme souvent en développement logiciel, la modélisation. L'intérêt d'utiliser XML vient de l'ajout de sémantique aux données formatées, mais cela pose de façon plus aiguë la question de la définition et de la structure des éléments du langage. Il est nécessaire de créer de nouveaux langages XML pour utiliser une sémantique précise par rapport à un métier, mais d'un autre côté, cela n'aurait pas de sens de créer un nouveau langage pour chaque application, l'intérêt d'XML venant du nombre de documents partageant le même langage. Un problème récurrent concerne donc la spécificité des éléments : pour chaque élément, une définition très large permet l'application à de nombreux cas, mais au détriment de la sémantique. Une solution parfois utilisée pour éviter le problème consiste à définir un langage très souple, presque dénué de séman-*

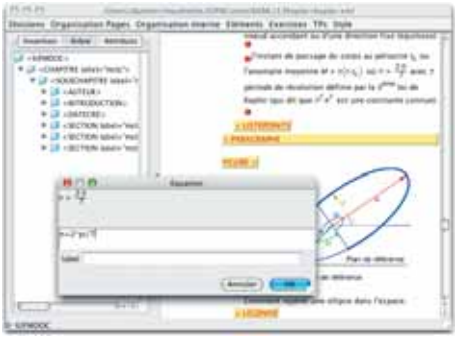
NOUVEAU les TUTORIELS sur www.programmez.com

tique. Celle-ci vient sous la forme d'un "vocabulaire" qui est utilisé, par exemple, avec des valeurs d'attributs dans le langage. Malheureusement, cela ne fait que reporter le problème du choix du langage XML au choix du vocabulaire... Avec le défaut qu'il n'existe pas d'outil pré-existant permettant de vérifier la validité d'un document par rapport à un vocabulaire donné.

P! : Bien souvent, des outils génèrent le code XML à la place du développeur. Mais comment faire quand le code XML généré est erroné ou ne répond pas aux attentes ?

D. G. : *Il faut distinguer plusieurs types de "code erroné". Primo, les documents qui ne sont pas "bien formés" (c'est-à-dire qui ne respectent pas la syntaxe de XML, ndlr) : ces documents ne sont pas plus utilisables que des informations griffonnées sur le coin d'une nappe en papier, il convient de les refuser automatiquement (ce qui est heureusement facile à faire). Secundo, les documents non valides (c'est-à-dire qui respectent la syntaxe, mais pas les règles du langage XML choisi, ndlr). Ces documents ne sont pas non plus utilisables automatiquement (du moins pas sans risque), mais il est facile d'obtenir une liste d'erreurs permettant de les corriger. Si la correction n'est pas automatisable, cela limite grandement l'intérêt d'utiliser XML. Tertio, les documents bien formés et valides, mais qui ne respectent pas les définitions des éléments du langage : par exemple, si j'écris "<auteur>26/3/2008</auteur>", cela n'est pas forcément invalide, mais il est clair que cela ne respecte pas le sens que l'on veut donner à l'élément "auteur". Ce type de problème est en général d'origine humaine. On peut l'éviter de diverses façons, par exemple en précisant le sens des éléments, ou en contraignant la syntaxe (par exemple en interdisant le caractère '/' dans l'élément "auteur"). Mais ce type d'erreur peut être grave lors d'une utilisation automatique, et il n'est pas toujours possible de l'éviter sans l'intervention d'un être humain pour vérifier le sens. Au final, cela peut donc être aussi coûteux à gérer que des problèmes purement syntaxiques, et il faut tout faire pour éviter ce type de problème (par exemple en améliorant l'interface de saisie).*

P! : On parle beaucoup de sécurité, de qualité. Or en XML, il paraît difficile de qualifier le code, par manque d'outils. Et la sécurité cause un souci car XML passe souvent à travers les pare-feux. Comment voyez-vous ces problématiques ?



D. G. : *la sécurité liée à la syntaxe XML s'est bien améliorée ces dernières années, grâce à l'utilisation généralisée d'analyseurs syntaxiques de plus en plus robustes. Ces analyseurs résistent aujourd'hui bien à des attaques par dépassement de taille. Le risque vient plus des attaques à l'intérieur des données véhiculées par les documents XML, mais cela n'est pas lié à XML : il faut donc utiliser les techniques habituelles pour traiter les données, qu'elles proviennent de documents XML ou pas. En fait, XML peut tout de même améliorer un peu la sécurité, avec l'utilisation de contraintes strictes pour les valeurs d'éléments et d'attributs dans les schémas XML : difficile d'ajouter une attaque avec un script dans un élément XML quand les seules valeurs autorisées sont les nombres de 10 à 20. En pratique, l'analyse de la qualité des documents est très difficile à réaliser automatiquement quand ceux-ci sont valides. Mais comme ils sont habituellement générés automatiquement, et en nombre avec la même méthode, il suffit souvent de vérifier la qualité d'un document "à la main" pour avoir une idée de la qualité d'un ensemble de documents produits de la même façon.*

P! : Des outils comme Altova, Oxygen... peuvent-ils aider efficacement le développeur ?

D. G. : *Bien sûr, les éditeurs graphiques permettent d'améliorer l'interface d'édition, et de mettre en valeur les informations importantes et les problèmes éventuels. Mais ils doivent être adaptés à chaque langage XML pour être vraiment utiles par rapport à de simples éditeurs de texte avec coloration syntaxique. Pour cette raison, nous avons lancé le développement de Jaxe à l'Observatoire de Paris. Jaxe ne s'utilise qu'avec un "fichier de configuration" pour chaque langage XML, qui définit une interface d'édition pour chaque élément. Il existe des méthodes d'affichages classiques (comme l'utilisation de balises de début et de fin), mais l'intérêt d'un tel éditeur vient de la possibilité de créer de nouvelles méthodes d'édition des éléments sous la forme de composants Java, facilement intégrables aux documents. On se rend bien compte qu'il est difficile de créer un grand tableau à la main en utilisant une interface sous forme de balises pour chaque ligne et chaque cellule : une visualisation sous forme de tableau permet d'éviter bien des erreurs. De même, l'utilisation d'interfaces graphiques appropriées pour chaque élément XML permet d'améliorer la qualité des documents. A mon avis, il faut cependant éviter l'utilisation d'interfaces "WYSIWYG", qui rendent difficiles l'édition sémantique des documents et sont souvent sources d'erreurs. Les développeurs préfèrent en général voir le code, ce qui n'empêche pas d'utiliser une interface graphique conviviale, mais n'est pas compatible avec une visualisation WYSIWYG.*

P! : Finalement, comment percevez-vous le XML ? Idéal pour les données mais peu maniable pour le développeur ?

D. G. : *J'ai du mal à imaginer comment je pourrais me passer d'XML aujourd'hui. Il est clair que c'est conçu pour le formatage de données plus que pour la programmation. XSLT, très beau conceptuellement et très interopérable mais un peu pénible à utiliser en pratique, me vient à l'esprit à ce sujet. Les développeurs peuvent aussi trouver une API comme DOM très lourde par rapport aux opérations réalisées sur les documents. Je pense que c'est essentiellement un problème d'adaptation : avec les bons outils logiciels, il est possible d'utiliser XML de façon très maniable. En effet, la plupart du temps, on n'utilise qu'une fraction des fonctionnalités d'XML, et il n'est pas souhaitable d'utiliser des outils qui gèrent 100% des fonctionnalités au prix d'une réduction considérable de la facilité d'utilisation. Il est parfois aussi utile de créer une API spécifique à un langage XML, de la même façon qu'on a intérêt à définir une interface graphique d'édition spécifique. Certains outils facilitent la création de telles API. Au final, on a tendance aujourd'hui à oublier les soucis que l'on avait avant l'apparition d'XML pour échanger des données, quand le problème de modélisation n'était pas le premier problème à résoudre. Un beau progrès, à mon avis.*

Sur Jaxe : http://jaxe.sourceforge.net/Jaxe.html

■ Propos recueillis par **François Tonic**

Réutilisation de contenu dans Altova XMLSpy avec OOXML & XSLT

Après avoir été normalisé par l'ECMA, le format de document Open XML (quelquefois appelé OOXML ou Office Open XML) a récemment été approuvé comme norme internationale par l'ISO.

En outre, tous les documents créés sous Microsoft Office 2007 et stockés en Open XML par défaut peuvent désormais être réutilisés plus facilement et profitablement que jamais. Les utilisateurs d'Altova XMLSpy trouveront ci-après quelques méthodes pour profiter pleinement de la prise en charge Open XML dès aujourd'hui. Jetons tout d'abord un oeil à un document Open XML tel qu'il s'affiche dans XMLSpy. Pour cet exemple, nous avons utilisé un document WordprocessingML (.docx) créé sous Microsoft Word 2007. A l'ouverture du fichier .docx, le contenu du package s'affiche immédiatement ; il est structuré selon l'Open Packaging Convention. Cela signifie tout simplement qu'il s'agit d'un fichier ZIP contenant des fichiers et des répertoires spécifiques qui définissent le contenu, la structure, les styles, les relations et d'autres parties du document. Grâce à la capacité intégrée de l'outil à ouvrir n'importe quelle archive au format ZIP, l'utilisateur peut naviguer directement dans la structure d'un répertoire du fichier ZIP, ajouter des nouveaux fichiers à ce package ou bien ouvrir un fichier XML compris dans le package (Fig.1). Afin de pouvoir réutiliser le contenu de ce fichier WordprocessingML, ouvrons le fichier 'document.xml' qui comprend le contenu du document. Dès que l'utilisateur double-clique sur le fichier dans l'archive ZIP, le code XML s'affiche dans une fenêtre séparée, comme tout autre document XML. L'utilisateur peut employer la vue Grille ou bien la vue Texte pour visualiser ou éditer les données XML (cela peut s'avérer utile s'il souhaite faire appel à la fonction Pretty Print en vue Texte pour rendre le fichier plus lisible) (Fig.2).

Il s'agit bien sûr d'une vue d'édition en direct. L'utilisateur peut donc non seulement visualiser les données Open XML, mais aussi modifier le code et le sauvegarder dans le package. Voyons maintenant comment l'utilisateur peut facilement réutiliser le contenu de ce document Open XML avec XSLT. L'environnement est d'ailleurs fourni avec plusieurs exemples de documents Open XML ainsi que des feuilles de style XSLT. Par exemple, penchons-nous sur la feuille de style 'docx2html.xslt', qui extrait tous les paragraphes d'un document WordprocessingML et les transforme en HTML. La feuille de style qui sert d'exemple ici n'est absolument pas un outil de conversion complet du .docx au HTML. C'est plutôt un modèle en ce qui concerne la réutilisation du contenu d'un fichier .docx, voire le point de départ des activités de développement de feuilles de style pour l'utilisateur.

Au sein de cette feuille de style XSLT, il faut une boucle <xsl:for-each> pour itérer sur tous les éléments <docx:p> qui seront ensuite transformés en simples paragraphes <p> HTML. Le texte contenu dans ces paragraphes est groupé par séries de caractères partageant des attributs communs ; c'est pourquoi une boucle secondaire <xsl:for-each> est éga-

Fig.1

Fig.2

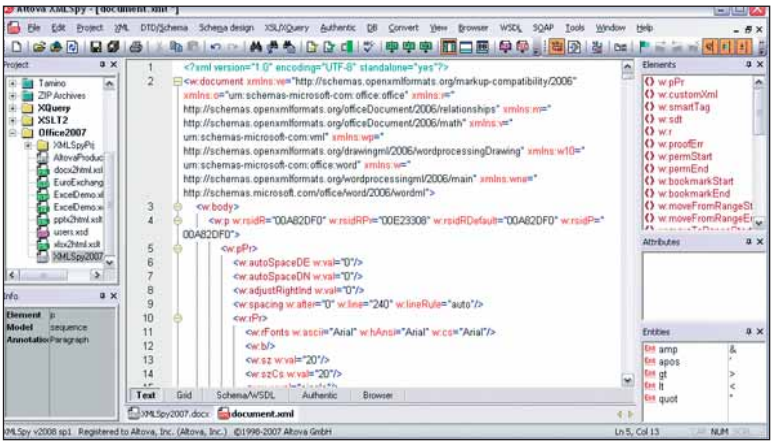


Fig.3

```
<xsl:for-each select="document($doc-uri)/docx:document/docx:body/docx:p">
  <p>
    <xsl:for-each select="docx:r">
      <xsl:value-of select="docx:t" />
    </xsl:for-each>
  </p>
</xsl:for-each>
```

Fig.4

```
<xsl:for-each select="document($doc-uri)/docx:document/docx:body/docx:p">
  <p>
    <xsl:for-each select="docx:r">
      <span>
        <xsl:attribute name="style">
          <xsl:apply-templates select="docx:rPr" />
        </xsl:attribute>
        <xsl:value-of select="docx:t" />
      </span>
    </xsl:for-each>
  </p>
</xsl:for-each>
```

Fig.5

```
<xsl:template match="docx:u">text-decoration:underline;</xsl:template>
<xsl:template match="docx:b">font-weight:bold;</xsl:template>
<xsl:template match="docx:i">font-style:italic;</xsl:template>
```

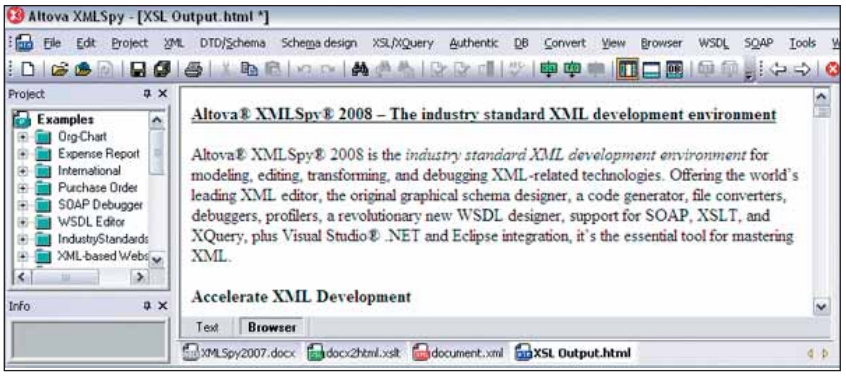
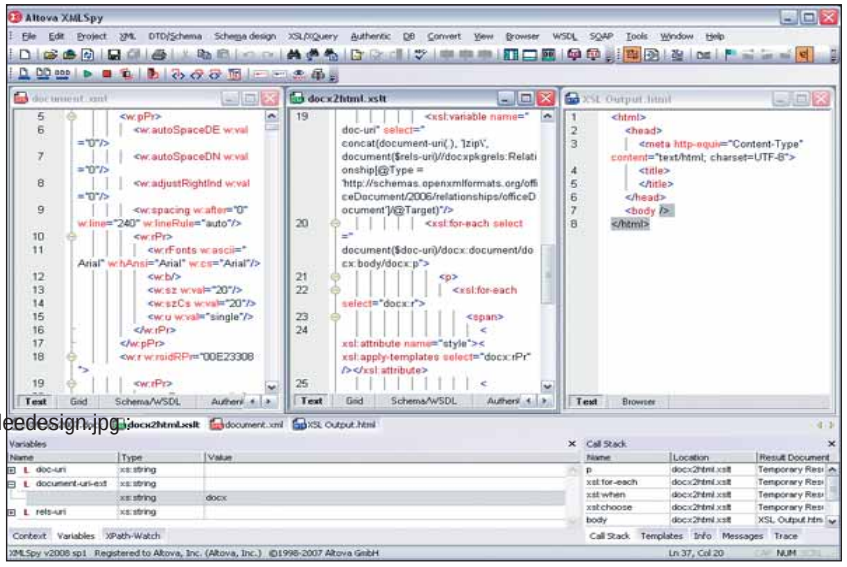


Fig.6



légende site-poleedesign.jpg :

Fig.7

lement nécessaire pour itérer sur tous les éléments <docx:r> et extraire le texte de leurs noeuds enfants de texte <docx:t>. Ainsi, la réutilisation de contenu la plus primaire, qui extrait uniquement le texte des paragraphes, ressemble à ceci (Fig.3).

Une fois les boucles construites, l'utilisateur peut éventuellement songer à extraire et réutiliser certaines informations relatives au style. Pour cela, il doit émettre un élément HTML pour chaque série de caractères <docx:r> et leur affecter un attribut de style dont la valeur dépend de l'élément <docx:rPr>. Dans cet exemple, nous avons utilisé <xsl:apply-templates> pour déterminer le style HTML à appliquer aux éléments (Fig.4). Les modèles correspondant aux trois styles les plus communs (gras, italique, souligné) sont extrêmement simples à construire et ressemblent à ceci (Fig.5).

Avec quelques lignes de XSLT et quelques modèles, nous avons écrit une feuille de style qui extrait les paragraphes de base et les principaux styles d'un document WordprocessingML, et les transforme en code HTML visualisable en vue Navigateur. Ci-dessous, vous pouvez voir le résultat de l'exécution de la feuille de style XSLT précitée sur le document WordprocessingML trouvé dans le répertoire d'exemples de XMLSpy (Fig.6).

De la même manière, il est facile de compléter cette feuille de style pour extraire des métadonnées, d'autres styles, ou bien des informations sur les images du document WordprocessingML et réutiliser ce contenu dans n'importe quelle application moderne, de la publication Web via HTML au RSS, en passant par les formats de média sociaux et les applications Web mobiles, et plus encore.

"Mais attendez ! Comment une feuille de style peut-elle être appliquée à un document XML stocké dans un fichier ZIP ?", vous demandez-vous peut-être. Naturellement, l'utilisateur peut extraire tous les fichiers XML à l'aide d'un décompresseur ZIP traditionnel, mais il existe une solution beaucoup plus pratique : lorsqu'il fait appel à la fonction document() en XSLT 2.0 dans XMLSpy ou dans AltovaXML, le moteur XML libre de droits développé par Altova, l'utilisateur peut accéder directement aux fichiers d'une archive ZIP en ajoutant l'opérateur de transfert de données "|zip" au nom du fichier. Ainsi, par exemple, "MyDocument.docx|zip_rels\.rels" associe le fichier de relation ".rels" du répertoire "_rels" situé dans l'archive ZIP au fichier "MyDocument.docx".

Les avantages du XSLT dans la réutilisation de contenu de documents Open XML sont évidents : le XSLT étant une pierre angulaire parmi les normes XML principales du W3C, l'utilisateur peut mettre en application ses connaissances en XML, Xpath et XSLT et profiter de l'excellente prise en charge d'outils dont disposent ces normes. Par exemple, l'utilisateur peut facilement développer et déboguer une feuille de style XSLT à l'aide du puissant débogueur XSLT intégré. Celui-ci permet à l'utilisateur de procéder à la transformation pas à pas, de placer des points d'arrêt sur des instructions XSLT et des noeuds de données dans le document Open XML, de visualiser le rendu partiellement généré, et d'inspecter l'état du processeur XSLT en détail au fur et à mesure de la construction du document (Fig.7).

Le débogueur XSLT épargne à l'utilisateur la plupart des difficultés souvent associées au développement de feuilles de style XSLT et offre une approche très itérative de la création et de l'optimisation de feuilles de style conçues pour réutiliser ou réaffecter du contenu. En résumé, il est très simple de réutiliser le contenu de documents Open XML pour toute une gamme d'applications Web, de scénarios mobiles, de média sociaux et de contextes Web 2.0, et ce grâce à des technologies XML standard, comme le XSLT.

Pour plus d'informations sur l'Open XML et sur tout le contenu désormais disponible dans ce format, consultez les sites Web suivants :

- XML Aficionado - http://www.xmlaficionado.com/
- Prise en charge d'Open XML dans XMLSpy - http://www.altova.com/features_office_2007.html
- Télécharger gratuitement une version d'évaluation de 30 jours de XMLSpy - http://www.altova.com/download/xmlspy/xml_editor_enterprise.html
- Centre de ressources sur les formats MSDN Open XML - http://msdn2.microsoft.com/en-us/office/bb265236.aspx
- OpenXMLDeveloper.org - http://openxmldeveloper.org/
- Brian Jones: les formats Open XML (blog) - http://blogs.msdn.com/brian_jones/

Alexander Falk - *Président et CEO Altova*

Résoudre le problème des performances de parsing

Avec l'omniprésence des fichiers XML pour les données et informations et surtout l'accroissement de leur volumétrie (ex. : dans les formats XML bureautique, dans les SGBD), un des problèmes concerne les performances dans leur manipulation. Comment éviter le goulot d'étranglement du parser ?

Les outils traditionnels s'appuient sur deux techniques habituelles de parsing: SAX (Simple API for XML) et DOM (Document Object Model). Elles ont chacune leurs inconvénients qui limitent leur intérêt dans le cas de très grands documents XML.

SAX déclenche des traitements au fur et à mesure de la lecture du fichier XML. Ne chargeant pas d'image du fichier en mémoire, il permet théoriquement de parser de très gros fichiers, mais avec des inconvénients qui limitent son utilisation. En particulier, il n'y a pas de possibilité de revenir en arrière lors du balayage du fichier. Cela nécessite que le fichier XML ait été structuré de manière à éviter ce cas de figure, ce qui n'est pas toujours possible.

DOM construit en mémoire une image du fichier XML, sous forme d'un graphe manipulable par une API normalisée. Les implémentations DOM traditionnelles deviennent très consommatrices en temps de parsing et en espace mémoire quand il s'agit de charger de gros fichiers. On note même des courbes de réponses qui deviennent exponentielles, atteignant la limite des 2 Go de mémoire adressable en 32 bits, ce qui rend impossible le parsing de gros fichiers

Technique de parsing SDO

En alternative à SAX et DOM, une nouvelle technique de parsing s'appuyant sur le standard SDO (Service Data Object) existe depuis plusieurs années. Ce standard, fruit d'une collaboration entre BEA et IBM, existe depuis 2004. Il est géré par le comité OASIS qui a récemment publié la version 2.1.1. Un parser SDO fonctionne de manière similaire à un parser DOM. En effet, l'opération de parsing avec SDO construit un graphe en mémoire, représentant le contenu de la source XML. Ce graphe est manipulable via l'API normalisée SDO, et on peut persister le graphe modifié dans un fichier XML.

Exemple de code

La spécification SDO définit l'API pour C++ et Java. L'API utilise des notations Xpath, avec des appels du genre Get() et Put().

Nous allons nous appuyer sur le schéma XSD suivant, représentant un portefeuille financier, pour montrer la simplicité de l'API SDO.

```
<?xml version="1.0" encoding="UTF-8"?>
<schema xmlns:tns="http://www.example.org/Portfolio/" xmlns="http://www.w3.org/2001/XMLSchema" targetNamespace="http://www.example.org/Portfolio/">
    <element name="Portfolio" type="tns:PortfolioType"></element>

    <complexType name="PortfolioType">
        <sequence>
            <element name="Name" type="string"></element>
            <element name="ID" type="int"></element>
            <element name="Position" type="tns:PositionType" minOccurs="0" maxOccurs="unbounded"></element>
        </sequence>

    <complexType name="PositionType">
        <sequence>
            <element name="Symbol" type="string"></element>
            <element name="PricePaid" type="double"></element>
            <element name="NumberOfShares" type="int"></element>
            <element name="Commission" type="double"></element>
        </sequence>
    </complexType>
</schema>
```

Parsing et validation

Le code pour réaliser ces opérations est particulièrement simple:

Première étape : créer une instance d'un Data Access Service (DAS). Le DAS gère le chargement et la persistance des données. Il charge le contenu de la source de donnée XML dans un graphe SDO (chargement), et vice versa (écriture). Seconde étape : charger le fichier XML. Une seule instruction suffit. Troisième étape, optionnelle : réaliser un "grammar checking" par rapport à un schéma XSD.

Navigation dans le graphe SDO

Le graphe SDO résultant du chargement a exactement la même structure arborescente que celle du fichier XML d'origine, avec simplement un noeud supplémentaire "root", parent de tous les noeuds.

Gros fichiers : l'implémentation SDO de RogueWave

Rogue Wave fournit le produit HydraSDO pour XML. C'est une implémentation de SDO, disponible en C++ et Java, sur plates-formes Windows, Unix et Linux. Aspect particulièrement intéressant, elle est spécialement dédiée à la gestion de gros fichiers XML, grâce à diverses caractéristiques techniques décrites ci-dessous.

- Implémentation native en C++, pour une efficacité maximale.
- Faible consommation mémoire, conséquence d'une technique d'indexation qui optimise la gestion des informations redondantes telles que les tags.
- Rapidité de parsing, obtenue par une stratégie dite de "pull parser", couplée avec un système de pagination qui accélère le processus.
- Algorithmes de parsing conçus pour avoir une courbe de réponse linéaire par rapport à la taille du fichier XML, à la fois en termes de consommation mémoire et temps de parsing.

Jean Vidames

SERVEUR ECO D'AMEN
UNE TECHNOLOGIE
EN HARMONIE
AVEC L'ECOSYSTÈME.
D'ÉMISSION
-55%
DE CO2*
amen
IN WEB WE TRUST
A partir de
49 € HT/mois **
soit 58,60 € TTC/mois
NE CHOISISSEZ PLUS ENTRE
PERFORMANCE ET ÉCOLOGIE.
SERVEURS ECO D'AMEN :
DE 1 À 3 Go DE RAM,
55 % D'ÉMISSION DE CO2
EN MOINS PAR AN.
http://eco.amen.fr
AMD Athlon X2 3400+ ou BE-2350
De 2x1,8GHz à 2x2,1GHz
Interface Plesk 8.2 – 10 domaines
OS : Fedora Core 7, Ubuntu 6, Debian 4, Windows Server 2003
De 1 à 3 Go de RAM
Disque dur de 80 Go à 160 Go
Trafic mensuel : 1 To
1 à 2 adresses IP
Inclus : Reboot, Restore et Recovery
En option : Amen DataBackup 10 Go (Inclus sur ECO 3000)
AMD 64 Athlon X2
PARTENAIRE INDUSTRIEL
entrepreneurs, faites le choix de l'économie numérique
www.econumerique.pme.gouv.fr
NUMÉRO GRATUIT : 0800 740 935 ou www.amen.fr
NOMS DE DOMAINE - EMAIL - HÉBERGEMENT - CRÉATION DE SITE - E-COMMERCE - RÉFÉRENCEMENT
* comparé à un serveur équivalent de la gamme confort - ** pour un engagement annuel - AMEN SAS PARIS - B 421 527 797 - IN WEB WE TRUST - Nous croyons au web. Conditions Générales de Vente sur le site www.amen.fr - REFLEXION FAITE

Web Devsign

Quand le développeur et le designer se rencontrent ...

Illustration : WeAreFantastic

Longtemps le développeur et le désigner web se sont côtoyés sans réellement s'aimer. Ils travaillaient ensemble, parfois contraints et forcés, parfois aussi, le désigner web faisait du développement et le développeur web de la conception graphique. Flash évitait d'avoir deux profils différents.

Aujourd'hui, avec l'évolution des outils et la complexité croissante des projets, des plates-formes, tout pousse vers plus de collaboration (active si possible) entre les deux profils : le développeur et le désigner. Ils sont contraints à travailler ensemble, voire, dans certains cas à fusionner. Mais un troisième profil apparaît aussi : l'intégrateur. On conserve alors une simple collaboration, avec à la charge de l'intégrateur le " build " du projet en intégrant les éléments graphiques et le code. Profil d'avenir ? Peut être. Car il ne sera pas toujours évident de faire intimement travailler le développeur et le designer. Le devsigner peut lui aussi connaître un brillant avenir, reste à définir son rôle exact et surtout les compétences qu'on lui demande. Il paraît difficile de maîtriser l'ensemble des plates-formes et langages.

Reconnaissons aussi que les plates-formes actuelles, principalement Adobe et Microsoft plus récemment, ne sont pas des modèles d'intéropérabilité, de compatibilité, même si au niveau graphique pur, et parfois du code, les échanges se feront. Ira-t-on alors vers du devsigner estampillé Adobe ou Microsoft ?

Le devsigner est né en grande partie de l'évolution des outils et des technologies. L'évolution du processus de production web va totalement dans ce sens. Désormais, on retrouve un cycle de développement incluant outils et profils différents. Cette évolution nécessite de bonnes pratiques, de nouveaux réflexes. Il faudra, pour le designer, acquérir une rigueur quand il crée une interface avec des objets d'interface. Il doit absolument nommer chaque objet, chaque élément. Nommage vital pour le développeur… Bref, le designer, même s'il demeure loin du code pur, touche à un élément sensible. Nécessitant une rigueur des process de développements et les bonnes pratiques.

Cela va être d'autant plus intéressant quand dans ces outils web, on disposera d'un véritable ALM (gestion du cycle de vie des applications), ALM que l'on commence un peu à voir dans la gamme Expression. À quand un Extreme Programming dédié au devsigner ? A n'en pas douter, nous arrivons à une évolution importante des métiers du web. Et les futures versions des éditeurs promettent d'aller (encore) plus loin. Mais attention à la qualité du code et au respect des standards.

Dans ce grand dossier, nous allons vous dévoiler les conseils, l'expérience, le témoignage de plusieurs grands designers web. Nous verrons aussi comment les plates-formes Adobe et Microsoft adressent le devsigner, avec des points communs, mais aussi des divergences.

François Tonic

“Trouver un langage commun”

TÉMOIGNAGE

Peter Tabor

Afin de bien clarifier certaines de mes positions, je voudrais signaler que je suis de formation designer graphique. Je voulais aussi signaler que pour un designer (et pour tout le monde en fait), la forme est signifiante. C'est-à-dire, que choisir une police de caractère, une forme, une couleur… va entraîner une esthétique particulière induisant du sens.

En effet, choisir pour un même texte de l'Arial ou du Garamond, visuellement ne donne pas du tout le même effet et donc son sens change (pour schématiser, un texte en Arial va plutôt donner l'impression d'un texte scientifique, alors que l'emploi du Garamond va donner une impression beaucoup plus littéraire au lecteur, sans parler du confort de lecture). Changer la police de caractère, mais aussi les couleurs, et la mise en page d'un texte (sur écran ou papier) n'est pas du tout anodin et neutre, cela entraîne au niveau des perceptions du lecteur de nombreuses conséquences. Voilà pourquoi, nous sommes toujours très pointilleux sur le respect de la forme sur le Net, alors qu'un langage comme le HTML, dissocie totalement la forme du contenu (ce qui est très pertinent dans certains cas et permet une grande accessibilité). Cette courte explication préalable, juste pour essayer de faire comprendre la vision d'un designer par rapport à un développeur, travaillant lui plutôt sur des structures abstraites assez éloignées de la forme finale de l'objet interactif que l'usager va voir sur son écran (que ce soit une application, un site web, une application sur téléphone portable…).

“Devsigner” ?

Les outils et les projets devenant de plus en plus complexes actuellement, je ne pense pas qu'une seule et même personne puisse tout faire. Les métiers de designer graphique et le métier de développeur demandent des compétences très spécifiques, et à part de très rares exceptions (John Maeda), il est impossible de cumuler les deux. Les éditeurs essayent de nous faire croire que tout est facile avec leurs logiciels (et c'est de bonne guerre) mais c'est totalement faux (ils font la même chose avec les softs de PAO, essayant de faire croire que tout le monde peut être graphiste ou typographe). Prenez l'exemple de Flex et de l'ActionScript3 d'Adobe, pour un designer comme moi, cela va demander un investissement énorme afin de me mettre à niveau, les langages et les outils sont de plus en plus perfectionnés (et efficaces), mais demandent un temps d'apprentissage très important pour un non spécialiste.

Incandescence, studio de création dans lequel travaille Étienne Mineur.

On nous raconte qu'avec Adobe AIR / Flex (par exemple, mais nous avons aussi Microsoft Silverlight) nous allons enfin pouvoir faire de petites applications facilement, mais je me rappelle qu'en sortant de mon école d'art (il y a quinze ans), j'avais pu développer un petit programme sur mon Mac, avec menu déroulant, gestion des fenêtres, gestion des boutons… avec un logiciel du nom de SuperCard (qui existe encore à ma grande surprise) avec son langage SuperTalk basé sur HyperCard. Sans avoir de notions très sérieuses de programmation j'étais capable de développer par moi-même une application validant certaines de mes pistes et permettant de présenter à un client un prototype digne de ce nom.

Sans dire, “c'était mieux avant” ;-) je pense que c'était aussi complexe de faire une application. En quinze ans, nous avons eu des améliorations et de nouvelles possibilités incroyables (le web avant tout), mais aussi une complexification exponentielle et nous arrivons donc au même résultat. Développer une application, un site web, une application pour téléphone… par soi-même demande toujours autant de temps. La vitesse accrue de nos processeurs et le développement de nouveaux outils informatiques n'ont, à ma grande surprise, pas autant simplifié la vie des développeurs que je le pensais.

Des tentatives de normalisation

Il existe, bien sûr, de nombreuses tentatives de normalisation, de simplifications… mais malheureusement les choses ne bougent que très lentement. Observez les tentatives de consortium comme le W3C, qui sont très importantes, mais extrêmement lentes à faire bouger (ou normaliser) les choses sur le net. Imaginez, nous sommes en 2008, nous avons le web, Youtube et Google Earth mais… en typographie, qui est tout de même la base la plus importante de la transmission de l'information (même actuellement, le texte est le pilier du Web et des moteurs de recherche…), c'est réellement la préhistoire !

Lors de la dernière présentation du nouveau player Adobe Flash version 10 (3D, axe des Z, des filtres graphiques… qui sortira sûrement l'année prochaine), on nous annonce une gestion améliorée du texte qui permettra de faire du multicolonnage. Et oui enfin, le multicolonnage, après plus de dix ans d'attente on croit rêver tellement c'est beau ; -)

De son côté le W3C devrait intégrer lui aussi le support du multicolonnage dans les spécifications CSS3. Le jour ou ces spécifications seront implémentées et respectées, donc vers 2010, nous auront peut-être des possibilités s'approchant vaguement des premiers softs de PAO datant de 1985 (par exemple Aldus PageMaker). Pour information, le multicolonnage existe depuis presque 2 000 ans dans notre culture, et c'est la base de la mise en page permettant de lire des textes avec un confort

visuel très satisfaisant (essayez de lire un journal sur une seule colonne, vous verrez le problème). Je ne parle même pas des césures, de l'interlettrage, des ligatures ou des approches de paires correctement affichées. Ces finesses typographiques, ne sont pas de la décoration ou une quelconque obsession de typographes pervers, mais permettent une lecture plus fluide, aisée et rapide (ce qui n'est vraiment pas du superflu sur écran). Malgré des progrès absolument incroyables dans le web durant ces dernières années, nous ne sommes toujours pas encore capables d'afficher des lettres à l'écran correctement, c'est totalement surréaliste, et démontre bien la manière dont ces différentes technologies avancent. On met avant tout les progrès " Bling Bling " en avant et les choses vraiment importantes et utiles aux gens en dernier (même si c'est moins sexy de parler d'interlettrage que de particules en 3D; -). Je suis donc assez pessimiste concernant la qualité typographique sur le Web (sauf si le PDF interactif devient la norme sur le Web, on verra bien avec Air), mais comme d'habitude ces grosses lacunes seront compensées par une interactivité accrue et des prouesses techniques bluffantes.

Un dialogue s'impose

Par contre, il est essentiel pour un développeur et un designer de trouver un langage commun, il faut donc que chacun s'intéresse à la culture et aux problématiques de l'autre. Les écoles ont une responsabilité très importante en ce moment, afin de former des gens assez ouverts vers d'autres cultures (faire une initiation à la programmation dans les écoles d'arts et une initiation à la lecture et à la conception des images dans les écoles d'ingénieurs ou d'informatique). Dans la pratique, je vois au contraire les métiers se spécialiser et le métier de designer de site Web s'industrialiser rapidement (avec des Frame Work, CMS…). D'un autre côté, il est possible de prototyper presque seul une maquette interactive permettant de valider avec le client certaines options (avec Flash ou même Processing que je vois beaucoup dans les universités). Une fois cette maquette de principe développée et validée, une équipe de développeurs professionnels va pouvoir se pencher sur la faisabilité technique (surtout la faisabilité dans le temps imparti), donner de nouvelles idées et refaire la plupart du temps la totalité du système (parfois en choisissant un autre langage plus adapté au projet).

■ **Étienne Mineur** - *Incandescence*

" Je pense qu'il est toujours bon de garder une spécialité ou spécificité "

***Kevin Gallot** est chef de projet web chez Limousin Expansion. Il a débuté comme développeur puis est venu à l'infographie grâce au Webdesign International Festival. Il nous donne sa vision sur l'évolution du métier et l'émergence de la collaboration designer – développeur.*

Programmez : on emploie le terme de "devsigner", fusion entre le développeur web et le designer web. Cette fusion se traduit dans la fonction, les processus, les outils, la collaboration, faut-il y croire et comment cela se concrétise ?

Kevin Gallot : Bien que les métiers soient différents, on voit effectivement des convergences, liées notamment au support ; ici le web. Cette évolution est possible grâce à l'évolution des outils de développement web, où le design a pris une plus grande importance (par rapport aux " vieux " outils de codage HTML/PHP). Et réciproquement, le webdesigner est aujourd'hui familier avec le codage (ActionScript, PHP…). D'autres métiers ont également vu des évolutions dans ce sens, notamment le métier de webmaster (initialement technique) touchant aujourd'hui les aspects techniques, marketing et éditoriaux d'un site web.

Le nouveau site du Pôle edesign a été développé de manière collaborative entre développeurs web et webdesigner (flasheurs). Techniquement, l'interface full flash récupère par flux xml les différents contenus, et propose une navigation par mots-clefs (navigation sémantique). C'est un exemple de collaboration de dev/designer pour une interface finale originale sortant des concepts classiques de navigation.

P ! : est-ce une bonne ou une mauvaise chose, et pourquoi ?

KG : Je pense qu'il est toujours bon de garder une spécialité ou une spécificité afin de pouvoir aller en profondeur. Personnellement, j'aurais du mal à parler de "devsigner"… mais plutôt d'un webdesigner qui fait du codage, ou d'un développeur web touchant au design…

P ! : les outils de type CS3 ou encore la gamme Expression Microsoft vont vers plus de collaboration entre les deux fonctions, comment comprendre cela, quelles bonnes pratiques mettre en place ?

KG : C'est en effet un des vecteurs de cette fusion… La prise en main de ces plates-formes demande de plus en plus de temps, ce qui explique des spécialisations sur ces outils. On voit déjà des " spécialistes CS3 ", et peut-être prochainement des " spécialistes Expression "… A propos de CS3, cet aspect collaboratif est venu doucement et avec le temps, ce qui a permis aux utilisateurs de s'adapter progressivement, sans changer leurs habitudes de travail. En sera-t-il de même pour les autres suites plus récentes et " jeunes " ?

P ! : ce rapprochement oblige-t-il à redéfinir le rôle de ces deux personnes ? ou tout du moins à mieux les faire travailler ensemble ?

KG : Je pense qu'il n'est pas nécessaire de redéfinir les rôles, que ce soit au sein d'une même structure, d'un même projet. Chaque compétence s'adapte suivant sa maîtrise du sujet et dois savoir gérer les relations avec l'équipe, en complément ou en support.

P ! : qu'en pense le " terrain "?

KG : Aujourd'hui, je constate que beaucoup de sociétés distinguent encore les deux rôles : le développeur et le webdesigner. Cependant, chacun dans son domaine maîtrise les bases de l'autre, offrant ainsi une compréhension globale d'un projet ou d'une architecture web. Enfin, il y a aussi des raisons propres au métier : un webdesigner ou un développeur web est curieux de nature !

■ Propos recueillis par **Jean Vidames**

Les bonnes pratiques pour améliorer la coordination entre designer et développeur

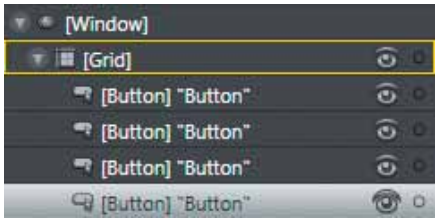
Cet article donnera aux développeurs et designers quelques conseils pour mieux collaborer entre eux, afin de leur permettre d'accroître leur efficacité.

Lors du développement d'une application WPF (Windows Presentation Foundation), le développeur se trouve souvent amené à travailler avec un designer. Certes, rien ne l'y oblige mais s'il s'en dispense, le rendu visuel ne sera absolument pas le même : il ne faut pas oublier que designer, c'est un métier à part entière.

Bref, ce tandem peut fonctionner à merveille mais avant d'arriver à un résultat correct, il va falloir apprendre quelques techniques au designer, afin que le travail du développeur/intégrateur soit plus simple et donc, plus rapide.

Nommage de contrôles

La première chose à enseigner à votre designer, c'est de nommer les différents éléments qu'il dépose sur la surface de dessin d'Expression Blend. Cela peut paraître simpliste comme raisonnement, mais il ne faut pas oublier que du point de vue du designer, peu importe le fait que les contrôles soient nommés ou non : il continuera de les manipuler en design. Mais vous, en tant que développeur, vous serez amené à les contrôler par le code, ce qui implique de les nommer explicitement. Voici donc le genre de chose qu'il faut éviter absolument, si vous ne voulez pas perdre de temps :

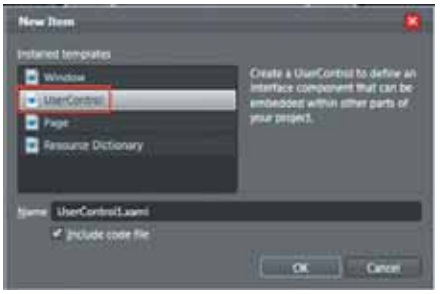


Inversement, voici un projet où tous les contrôles sont correctement nommés. Il sera ainsi plus aisé pour le développeur de travailler avec cette application :

La création de contrôle utilisateur

Voici un autre point auquel les designers ne sont pas sensibles, de par leur formation. En tant que développeur, vous avez parfaitement connaissance du fait que si plusieurs fenêtres/formulaires utilisent les mêmes contrôles, il convient de les rassembler dans un contrôle utilisateur (UserControl), afin d'éviter de faire trop de copier/coller inutiles.

Dans Expression Blend, les designers ont la possibilité de créer des contrôles utilisateur. Pour cela, il leur faut juste cliquer sur "File" et sélectionner "New Item…".Là, la fenêtre suivante apparaît et leur permet de demander la création d'un UserControl :



En terme de design, qu'il travaille sur un contrôle utilisateur ou un objet de type Window, pour le designer, l'expérience est la même : il dispose de la même surface de dessin, des mêmes outils, etc.

Communiquer avec votre designer

La philosophie de Windows Presentation Foundation sous-entend que le développeur et le designer peuvent travailler chacun de leur côté (le designer travaillera sur le fichier XAML et le développeur s'occupera du fichier de code-behind). Au bout d'un certain temps, les deux parties se retrouvent et un intégrateur (ou un développeur) réunit leur travail pour produire l'application finale. Bien que fort prometteur, ce scénario n'est vrai que lorsque les 2 parties ont déjà l'habitude de travailler sur des projets WPF. Dans le cas contraire (et, d'une manière générale dans tous les cas), il vaut mieux que vous communiquiez un maximum avec le designer, afin de le guider dans sa découverte de la technologie et/ou des outils. N'oubliez pas que si vous le laissez se débrouiller tout seul, votre designer risque :

- De mettre plus de temps à réaliser son travail/son interface
- De ne pas nommer les différents contrôles déposés sur la surface de dessin
- De ne pas utiliser de contrôles utilisateur
- Etc.…

Certes, ces différents éléments ne représentent pas grand-chose, à première vue. Cependant, ils peuvent faire perdre énormément de temps au développeur (ou à l'intégrateur) qui sera obligé de repasser derrière le travail du designer afin de corriger ces erreurs qui l'empêchent d'avancer.

Conclusions

Cet article vous a présenté un ensemble de conseils, simples mais efficaces, pour vous permettre d'améliorer la collaboration que vous pouvez avoir avec votre designer. Pour votre information, sachez que les explications que je vous ai données proviennent de l'expérience que j'ai acquis lors de développements sur des projets de type WPF avec un designer. Avec l'habitude, bien sûr, le fait d'encadrer le designer devient de moins en moins nécessaire car celui-ci devient de plus en plus autonome.

Ressources

"The New Iteration", whitepaper sur la collaboration designer/développeur : http://windows-client.net/wpf/white-papers/thenewiteration.aspx

Le site d'Expression Blend : http://www.microsoft.com/expression/products/overview.aspx?key=blend

Mon blog : http://blogs.developpeur.org/tom

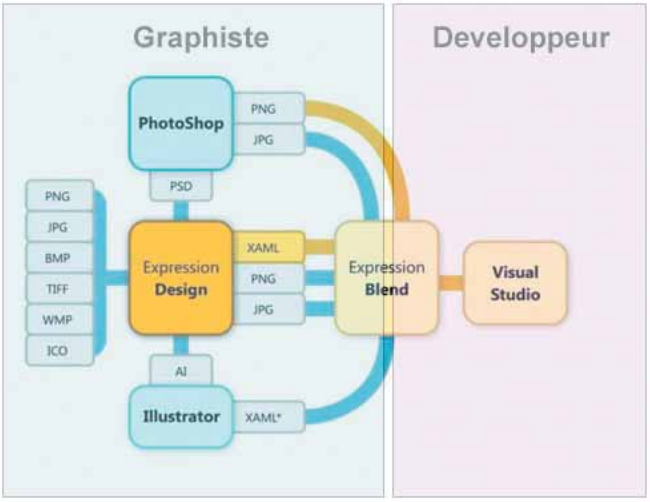
Thomas Lebrun
Consultant/Formateur – Winwise
Microsoft MVP C#
thomas.lebrun@winwise.fr

NOUVEAU les TUTORIELS sur www.programmez.com

Silverlight 1.0 / 2.0 :

le workflow graphiste - développeur, les styles, les templates et les ressources

Avec Silverlight, Microsoft apporte bien plus qu'une technologie "Rich Internet Application" aux entreprises. Il vient chambouler les méthodologies de création d'applications web classiques pour faire place à de nouvelles méthodologies propres aux RIA et donc indirectement à des méthodologies de performance et de rendement dans les projets clients. Tout cela, grâce aux outils Expression. Dans cet article, nous aborderons le nouveau workflow entre les graphistes et les développeurs pour la technologie RIA : Silverlight. Dans un premier temps nous décrirons ce nouveau concept et nous verrons ensuite plus en détail les différentes méthodes de réalisation offertes aux graphistes et aux développeurs. Quelles sont les " best practice " ? A contrario quelles méthodes sont à bannir c'est ce que nous allons tenter d'élucider.



Silverlight 1.0

Il est important de dissocier les méthodologies de développement d'un projet Silverlight 1.0 de celles d'un projet Silverlight 2. Silverlight est caractérisé par son langage XAML. Il s'agit d'un dialecte XML de description d'interface qui permet au plug-in Silverlight de réaliser un rendu graphique des interfaces spécifiées, de façon déclarative. Avec Silverlight 1.0 le développeur a la possibilité de communiquer avec ces " objets graphiques " grâce au langage de programmation JavaScript. Le graphiste quant à lui doit apprivoiser le langage XAML… ou utiliser les bons outils. Inutile de vous cacher qu'un graphiste par définition ne sera pas spécialement attiré par l'apprentissage d'un nouveau " langage de programmation ", c'est pour cela qu'il optera généralement pour laisser les outils effectuer la conversion de ses créations vers le langage XAML automatiquement.

Les outils de création graphique orientés XAML pour Silverlight 1.0

• Expression Studio 1.0 :

Fig.A *Fig.B*

- Expression Design version 1 : Cet outil permet de réaliser des créations vectorielles et de les exporter vers le format XAML. (Fig.A)
- Expression Blend version 1 : Cet outil plus orienté intégrateurs, permet d'avoir un aperçu de vos créations importées dans le projet final. Les solutions Expression Blend sont compatibles avec les solutions de projet Visual Studio, ce qui simplifie les allers-retours entre le graphiste et le développeur puisque les deux corps de métier communiquent sur le même langage. (Fig.B)

• Adobe Illustrator : Il est possible d'exporter vos fichiers AI au format XAML avec un plug-in que vous pourrez télécharger gratuitement à cette adresse : http://www.mikeswanson.com/xamlexport/. Ce plug-in est compatible avec les versions 1, 2 et 3 de Adobe CS

Le worflow développeur graphiste avec Silverlight 1.0

En Silverlight 1.0 le langage XAML n'est pas le même que celui que l'on trouve dans Windows Presentation Foundation ou Silverlight 2. Ainsi, le XAML Silverlight 1.0 est simplifié. La notion de ressource n'existe pas, il est donc pratiquement impossible de réutiliser des créations graphiques sans dupliquer physiquement le code, ceci entraînant des fichiers XAML conséquents.

Les différentes méthodologies

La seule méthodologie utilisable en Silverlight 1.0 va donc être la méthode de " style inline ", c'est-à-dire que les styles sont définis directement dans la source du document XAML et sur les balises de chaque objet graphique. Exemple :

```
<Rectangle Width="170" Height="77" Canvas.Left="100" Canvas.Top="117"
RadiusY="9.5" RadiusX="9.5" RenderTransformOrigin="0.5,0.5">
    <Rectangle.RenderTransform>
        <TransformGroup>
            <ScaleTransform ScaleX="1" ScaleY="1"/>
            <SkewTransform AngleX="0" AngleY="0"/>
            <RotateTransform Angle="-8.283"/>
            <TranslateTransform X="0" Y="0"/>
```

```
            </TransformGroup>
        </Rectangle.RenderTransform>
        <Rectangle.Fill>
            <LinearGradientBrush EndPoint="0.5,1" StartPoint="0.5,0">
                <GradientStop Color="#FF000000" Offset="0"/>
                <GradientStop Color="#FFFFFFFF" Offset="1"/>
            </LinearGradientBrush>
        </Rectangle.Fill>
    </Rectangle>
```

Amener la logique de template dans Silverlight 1.0 devient très vite drastique, cela n'est pas impossible mais vous serez obligé d'utiliser des frameworks JavaScript propriétaires contraignants afin de simuler les comportements attendus et de pouvoir optimiser vos effets, ressources, etc. Une solution simple, passer à Silverlight 2.

Silverlight 2.0

Vous l'avez compris, le langage XAML embarqué dans Silverlight 2 est très riche en terme de fonctionnalités. Celui-ci va donc permettre au graphiste et au développeur d'intégrer les design de différentes façons afin d'optimiser l'application : Style, Template, Contrôle, Ressource... Mais avant d'énumérer les différentes méthodologies, regardons les outils que nous avons à notre disposition.

Les outils de création graphique orientés XAML pour Silverlight 2

- Expression Studio 2
  - Expression Design version 2
  - Expression Blend version 2 preview March 2008
- Adobe Illustrator et son plug-in. (cf. Silverlight 1.0)

Le worflow développeur graphiste avec Silverlight 2.0

La nouvelle version de Blend permet de faire le lien entre le métier de graphiste et le métier de développeur il s'agit de la pièce maîtresse du workflow. Il va permettre d'harmoniser les deux corps de métiers et d'optimiser la partie design d'une application. Dans les projets conséquents il sera d'ailleurs conseillé d'attribuer cette tâche à une personne ayant pour fonction l'intégration des deux mondes (= l'intégrateur). Notons qu'en Silverlight 2 le langage utilisé par le développeur sera généralement le C# ou VB, bien que d'autres lui sont offerts.

Les différentes méthodologies (Silverlight 2.0)

La méthode de style local

Cette méthode permet de définir une ressource utilisable seulement au niveau d'un contrôle et de ses enfants. Avec cette méthode, la maintenance devient très vite compliquée dans le cas des gros projets étant donné que les styles seront dispatchés dans toute l'application. Dans l'exemple suivant, Ellipse qui est un enfant de Button a accès au style ayant pour clé "MyBrush1" au même titre que le Button lui-même :

```
<Button Style="{StaticResource MyBrush1}" Content="Button">
    <Button.Resources>
        <Style x:Key="MyBrush1" TargetType="Button">
            <Setter Property="Width" Value="100" />
            <Setter Property="Height" Value="20" />
        </Style>
        <LinearGradientBrush x:Key=" MyBrush2" EndPoint="0.5,1" StartPoint
="0.5,0">
            <GradientStop Color="Red "/>
            <GradientStop Color="Black" Offset="1"/>
        </LinearGradientBrush>
    </Button.Resources>
    <Button.Background>
        <StaticResource ResourceKey=" MyBrush2"/>
    </Button.Background>
    <Ellipse Style="{StaticResource MyBrush1}" Fill="#FFFFFFFF" Stroke=
"#FF000000"/>
</Button>
```

N.B : On notera d'ailleurs qu'il est possible pour le graphiste de venir modifier le template d'un contrôle et plus spécifiquement le ControlTemplate, ceci étant possible avec n'importe quelle méthode d'intégration graphique. Cette propriété permet de modifier totalement le rendu graphique d'un contrôle. De ce fait, un bouton peut prendre n'importe quelle apparence. Dans notre exemple, notre bouton deviendra un Rectangle rouge mais aura les " propriétés " d'un bouton :

```
<Button Style="{StaticResource MyBrush1}">
    <Button.Resources>
        <Style x:Key="MyBrush1" TargetType="Button">
            <Setter Property="Width" Value="100" />
            <Setter Property="Height" Value="20" />
            <Setter Property="Template">
                <Setter.Value>
                    <ControlTemplate TargetType="Button">
                        <Rectangle Width="100" Height="20" RadiusX
="5" RadiusY="5" Fill="Red" />
                    </ControlTemplate>
                </Setter.Value>
            </Setter>
        </Style>
    </Button.Resources>
</Button>
```

La méthode de style global

Cette méthode permet de définir une ressource utilisable au niveau de l'application. Ce type de ressource est renseigné dans le fichier " App.xaml ". Ainsi, n'importe où dans l'application il est possible d'accéder à la ressource. Exemple :

```
<Application
  xmlns="http://schemas.microsoft.com/client/2007"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  x:Class="SilverlightApplication1.App">
  <Application.Resources>
<!—Insérer les ressources ici -->
  </Application.Resources>
</Application>
```

Contrairement à WPF, la notion de dictionnaire externe n'existe pas, avec cette méthode il est donc risqué de venir saturer ce fichier. Cepen-

dant, ce type de méthode permet des mises à jours rapides étant donné que tous les styles, ressources, sont situés à un seul endroit dans l'application.

La méthode d'approche par composant

Il est possible pour le développeur de délimiter des parties " skinnable " à ses contrôles. Cette notion se nomme " TemplatePart " mais malheureusement celle-ci n'est pas encore implémentée dans la version preview Mars 2008 de Blend, ce qui pousse le développeur à faire une grosse part du travail. Cette méthode consiste à exposer, par exemple, des Storyboard définis. Dans ce cas précis, c'est le développeur qui délimite les possibilités du graphiste à skinner tel ou tel " objet ". Exemple :

```
[TemplatePart(Name = "RootElement", Type = typeof(FrameworkElement))]
[TemplatePart(Name = "FocusVisualElement", Type = typeof(Framework
Element))]

[TemplatePart(Name = "Normal State", Type = typeof(Storyboard))]
[TemplatePart(Name = "MouseOver State", Type = typeof(Storyboard))]
[TemplatePart(Name = "Pressed State", Type = typeof(Storyboard))]
```

```
public class MyButton : Control
{
    public static readonly DependencyProperty TextProperty;
    public static readonly DependencyProperty ForegroundProperty;
    public static readonly DependencyProperty FontSizeProperty;

    public string Text { get; set; }
    public Brush Foreground { get; set; }
    public double FontSize { get; set; }

}
```

Le graphiste n'aura plus qu'à renseigner les Storyboards imposés dans un fichier XAML externe. Exemple :

```
<Storyboard x :Name="Normal State">
<DoubleAnimationUsingKeyFrames …/>
</Storyboard>
```

Cette méthodologie est je pense, la plus aboutie et la plus exploitable dans un contexte de production. Plus de détails sur cette méthodologie ? http://msdn2.microsoft.com/en-us/library/cc278064(VS.95).aspx

■ **Guillaume André** - *Intégrateur .NET & Silverlight*
http://www.wygwam.com/ - http://blogs.codes-sources.com/guillaume

L'information du DÉCIDEUR

Choisir, déployer, exploiter les logiciels en entreprise

Lisez le seul magazine offrant aux responsables informatiques une information et des témoignages focalisés sur le logiciel en entreprise.

A lire dans le numéro 2 - Avril/Mai

- **Dossier Communication Unifiée**
- **Déployer de l'Open Source dans l'entreprise**
- **Mesure d'audience, installer un outil de «Web Analytique»**
- **Virtualisation : attention à l'administration**
- **Administration : Windows serveur 2008, Microsoft joue gros !**
- **Sustainable IT Architecture : refonte progressive des systèmes d'information avec le SOA**

www.solutions-logiciels.com

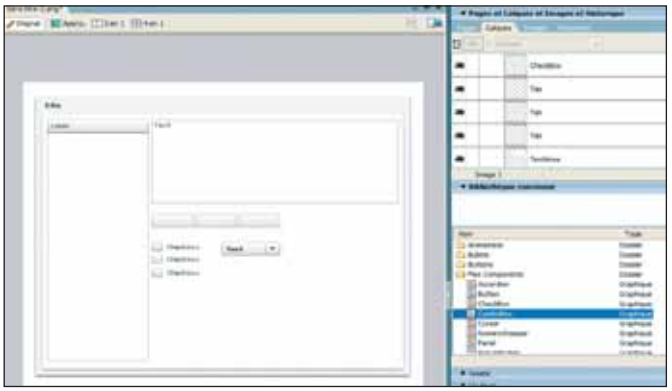
Du designer CS3 au développeur Flex

“L’explosion des applications RIA n’est pas seulement due à une révolution technologique.”

En effet, la technologie Flex permet de déporter beaucoup de calculs côté client (grâce à la nouvelle machine virtuelle du Flash Player 9), de conserver la facilité de maintenance et de mise à jour d’une application web, de profiter de standards de communications courants (comme l’appel de WebService) ou plus évolués (échanges AMF par Remoting), de refaire du push de données en temps réel… Mais au delà des apports technologiques, la RIA nous apprend que le Design est l’une des clefs du succès d’une application. La technologie Flash laisse au concepteur d’applications riches une totale liberté en termes d’expression de données, de transitions entre les vues ou d’habillages de composants. Mais tous ces effets visuels doivent se justifier par rapport au même objectif : répondre au besoin de l’utilisateur, le " pourquoi " de son application riche. Des éditeurs comme Business Objects ou Ilog l’ont bien compris, l’expression graphique de l’information améliore la prise de décision, donc la productivité. Ainsi, des profils comme les ergonomes, les Ux (User Experience) et les designers de RIA prennent une place stratégique dans ces cycles projets, bien en amont du code. Pour le développeur de RIA, cela implique de respecter le mieux possible cette phase d’étude préliminaire et d’apprendre à intégrer efficacement tous les éléments de design produits par les designers. Pour avoir eu à gérer dans le passé ces interactions en agences interactives, je peux affirmer que le " workflow " entre les designers et les développeurs est une source d’optimisation sur un cycle de développement de RIA. De nombreuses itérations surgissent entre ces profils tout au long de la phase de développement, ce qui peut malheureusement remettre en cause certains développements. Nous allons découvrir dans cet article comment un designer qui utilise Adobe Creative Suite 3 peut préparer le travail du développeur de RIA Flex. Pour ces techniques, j’utilise les extensions " Flex Skin Design for CS3 " disponibles sur http://www.adobe.com/go/flex3_skinning.

Première technique : du WireFrame au Prototype

Le designer va rapidement s’attaquer à la réalisation de WireFrames. Ces descriptions d’écrans synthétiques, agencent les éléments de l’application, sans se soucier de l’aspect graphique. Cela permet de rapidement ressentir si l’équilibre des écrans est cohérent, si l’œil repérera intuitivement les éléments de navigation, etc. Pour un site web, un designer Adobe se sert de FireWorks CS3, un outil pivot qui optimise les éléments graphiques, issus de PhotoShop par exemple, pour générer une page web. Il peut faire de même avec un écran Flex. En effet, il suffit de lancer FireWorks, d’ouvrir la bibliothèque commune et le dossier " Composants Flex ". Tous les composants de base d’une application Flex sont présents : *Panel, Bouton, TextArea, Accordion, checkbox…* et le designer peut les agencer sur son espace de travail FireWorks comme s’il travaillait avec des images. Habitué à exporter son travail en HTML (vers DreamWeaver par exemple), il va pouvoir cette fois-ci directement exporter son travail en MXML (pour être interprété par le Flex Builder). C’est une première approche intéressante où l’on met à disposition du designer une librairie de composants applicatifs, et où on lui laisse faire l’agencement. Le projet Thermo reprend ce principe : le designer place des éléments, du code MXML est automatiquement généré en tâche de



fond. L’approche dans FireWorks est encore limitée car on ne peut pas reprendre ces composants pour les skinner (personnaliser l’habillage). Ainsi, ce workflow n’interviendra que dans une phase de prototypage, pas dans une phase de réalisation.
Voici le wireframe réalisé sous FireWorks et le résultat MXML :

```
<?xml version="1.0" encoding="utf-8"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
 width="600" height="600" layout="absolute" backgroundGradientColors
="[#FFFFFF, #FFFFFF]">
   <mx:Style>
      Panel {
         disabledColor:#AAB3B3;
         color:#0B333C;
      }
      Accordion {
         textSelectedColor:#000000;
         textRollOverColor:#2B333C;
         disabledColor:#AAB3B3;
         color:#0B333C;
      }
      TextArea {
         disabledColor:#AAB3B3;
         color:#0B333C;
      }
      CheckBox {
         textSelectedColor:#000000;
         textRollOverColor:#2B333C;
         disabledColor:#AAB3B3;
         color:#0B333C;
      }
      ComboBox {
         textSelectedColor:#2B333C;
         textRollOverColor:#2B333C;
         disabledColor:#AAB3B3;
         color:#0B333C;
      }
   </mx:Style>
```

```
    <mx:Panel id="panel" x="19" y="22" width="561" height="453"
layout="absolute" title="Title" enabled="true">
        <mx:Accordion id="accordion" x="7" y="7" width="152" height
="392" enabled="true">
            <mx:Canvas width="100%" height="100%" label="Label"/>
        </mx:Accordion>
        <mx:TextArea id="textArea" x="164" y="8" width="370" height
="136" enabled="true" text="Text"/>
        <mx:CheckBox id="checkBox" x="165" y="206" height="14"
selected="false" label="Checkbox" enabled="true"/>
        <mx:CheckBox id="checkBox2" x="165" y="226" height="14"
 selected="false" label="Checkbox" enabled="true"/>
        <mx:CheckBox id="checkBox3" x="165" y="248" height="14"
 selected="false" label="Checkbox" enabled="true"/>
        <mx:ComboBox id="comboBox" x="269" y="205" width="88"
 height="22" enabled="true" editable="false">
            <mx:dataProvider>
                <mx:Array>
                    <mx:String>Text</mx:String>
                </mx:Array>
            </mx:dataProvider>
        </mx:ComboBox>
    </mx:Panel>
</mx:Application>
```

Seconde technique : le Skinning par CSS

Le skinning de composant Flex par CSS est disponible depuis Flex 2 mais peu d'outils permettaient de le faire visuellement. Une application Flex permettait de manipuler le design par CSS : Le Flex Style Explorer (disponible en ligne). Pour comprendre le principe des CSS, voici un bouton par défaut :

Et un bouton sur lequel j'applique cette feuille de style :

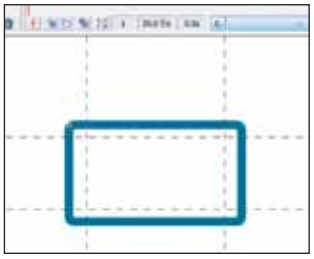
```
Button {
    cornerRadius: 10;
    paddingLeft: 8;
    paddingRight: 8;
    paddingTop: 6;
    paddingBottom: 6;
    letterSpacing: 2;
    fillAlphas: 0.75, 0.55, 0.75, 0.65;
    fillColors: #0099cc, #0033cc, #00cccc, #0099ff;
    color: #ffffff;
    themeColor: #00ff33;
    fontSize: 12;
    fontWeight: normal;
    fontStyle: italic;
}
```

Avant Flex 3, seuls les développeurs pouvaient skinner correctement les composants car tout se faisait au niveau du code. Le " CSS Design mode " de Flex Builder 3 permet de définir graphiquement une CSS sur un composant, sans avoir à coder. Pour utiliser ce nouveau mode,

démarrez un nouveau projet Flex et utilisez la commande File > New > CSS File. Passez en mode " design " et cliquez sur le bouton " New Style ". Vous pouvez alors sélectionner dans la liste le composant à habiller. Tous les états du composant s'affichent : Up, Over, Down, Disabled… et tout se paramètre dans le panneau Flex Properties. Si le skinning CSS ne suffit pas pour les composants, il faut utiliser les outils de la Creative Suite 3 et préparer des éléments de design pour le développeur.

Skinning de composants avec la Creative Suite 3.

Une fois les extensions " Flex Skinning for CS3 " installées, un designer Flash peut se rendre dans le menu Fichier > Nouveau > Modèle > Flex et sélectionner un composant à skinner. Un projet Flash complet va s'ouvrir avec tous les états du composant sur la ligne de temps. Il sera très simple de personnaliser le look de chaque état d'un composant, car tout est en vectoriel et respecte la règle du " 9-slices ". Cette règle de définition d'un composant permet de redimensionner un graphique sans sensation d'écrasement. Un composant (ou un sous-composant) est divisé en 9 parties. Les parties extérieures ne sont pas concernées par un redimensionnement. Seule les dimensions de la partie centrale seront impactées. Cette notion se retrouve dans tous les produits graphiques de la gamme Adobe : Flash, PhotoShop, FireWorks, Illustrator.



Pour tester le nouveau look de votre composant, rendez-vous dans le menu de Flash Fichier > Publier. Cela va automatiquement créer un SWF et un SWC. Dans Flex Builder 3, faites Import > Artwork et sélectionnez le fichier SWC. La feuille de style CSS sera automatiquement générée et appliquée à vos composants. Cette méthodologie d'export/import avec une génération automatique de CSS permet d'industrialiser les échanges du designer vers le développeur. En cas de modification du look de l'application, le développeur aura juste à importer le nouveau SWC. Cette technique est exactement la même sous Illustrator et PhotoShop. Sous Illustrator, les états se retrouvent dans la librairie des symboles ; sous Photoshop, un calque correspond à un état.

La skin DataGrid par défaut sous Flash : (Fig.A)

Ma skin Rock and Roll redesignée sour Flash : (Fig.B)

Le résultat trash sous Flex : (Fig.C)

Fig.A

Fig.B

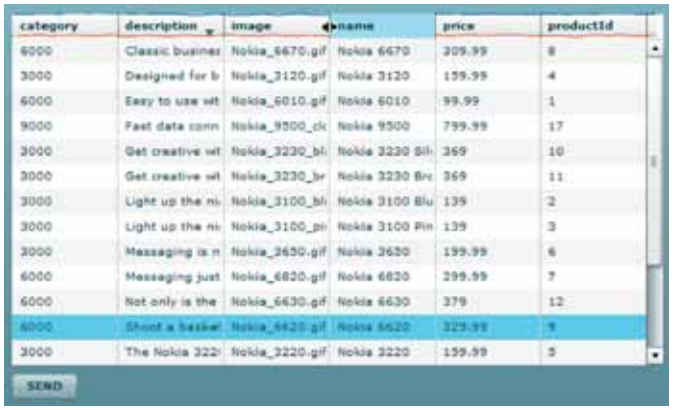


Fig.C

Création de composants Flex sous Flash CS3.

La technique la plus avancée aujourd'hui pour designer une RIA est selon moi de faire appel à un designer interactif Flash. Ces designers sont habitués à créer des animations expressives et parfois interactives à l'aide de code ActionScript. Un développeur Flex peut récupérer des éléments animés (le terme sous Flash est MovieClip) en provenance d'un Flasheur. Dans le menu Commande de Flash CS3 se trouve désormais une entrée " Convert symbol into Flex component ". Voici la démarche pour créer un composant Flex :

1) Créez un MovieClip dans lequel vous pouvez gérer des animations, du script AS3…
2) Donnez-lui un nom du type " monComposant "
3) Sélectionnez le MovieClip dans la librairie (CTRL+L) et lancez la commande " Convert symbol into Flex component "
4) Flash CS3 modifie les propriétés du MovieClip et votre projet. Placez le MovieClip sur la scène et publiez votre projet Flash. Un SWF et un SWC sont automatiquement créés.
5) Rendez-vous désormais sous Flex Builder 3 et allez dans les propriétés de votre projet Flex. Dans la rubrique Flex Build Path > Library Path, ajoutez le SWC.
6) Je vous conseille à cette étape de sauvegarder le projet Flex, de le fermer puis de le rouvrir (le rafraîchissement est parfois difficile).
7) Vous pouvez alors saisir le tag <local:monComposant/> et le placer sur la scène. Vous retrouvez toutes les variables déclarées dans le MovieClip Flash.

Conclusion

En attendant Thermo, de nombreuses connexions existent entre les outils des designers (Creative Suite 3) et le Flex Builder 3. L'approche par CSS devrait suffire à designer la grande majorité de vos applications. Si le design par CSS n'est pas suffisant, il faut passer par du skinning directement sur les composants. Les trois outils PhotoShop (pour du design bitmap avec des PNG), Illustrator ou Flash (pour du design vectoriel) sont désormais à jour pour préparer au mieux les éléments dont aura besoin le développeur Flex. Il faut récupérer et installer impérativement le kit " Flex Skinning for CS3 ", en libre téléchargement sur le site d'Adobe. La dernière méthode qui consiste à convertir des animations interactives Flash en composants Flex est de loin la plus évoluée et permet de rapidement définir des expériences riches et engageantes. Il est important de noter que Flex ne remplace pas Flash, et n'est pas une évolution de Flash. Flash reste un outil pour les designers et les concepteurs d'animations Flash. Flex s'adresse aux développeurs d'applications Flash. Ces deux profils, designers et développeurs, ont toujours eu des difficultés à échanger et adapter des éléments graphiques. Pour concevoir une RIA, Adobe propose désormais des ponts qui permettent un premier niveau d'industrialisation. Pour totalement fluidifier le workflow du design au coding, le projet Adobe Thermo apportera toutes les solutions. Des vidéos sont disponibles sur le web pour le découvrir en attendant la bêta…

Michaël Chaize

Thermo : l'environnement de design de RIA

Thermo est le nom de code d'un projet Adobe présenté dans sa version alpha lors du dernier Adobe MAX (en Octobre 2007). Ce nouveau logiciel se destinera aux designers de RIA. L'objectif sera de fluidifier intégralement les processus de travail et d'échange entre les designers et les développeurs de RIA. Trois acteurs vont donc à terme intervenir dans ce cycle de collaboration :

- le designer, qui continuera de designer les écrans de la RIA avec les outils qu'il connaît (PhotoShop, Illustrator…) ;
- l'intégrateur de design, qui importera les éléments de design et préparera toutes les vues (code Flex MXML compris) pour le développeur ;
- le développeur, qui récupèrera directement dans l'IDE Flex Builder les vues.

Sous Thermo, vous pourrez soit partir de zéro, et créer les wireframes (à partir de composants vierges, un peu comme sous FireWorks aujourd'hui), soit récupérer des éléments graphiques (depuis la Creative Suite 3). Une fois les écrans définis, vous pourrez alors indiquer le futur comportement des éléments graphiques. Si vous récupérez depuis PhotoShop deux flèches séparées d'un rectangle, vous pourrez indiquer à Thermo qu'il s'agit d'une barre de défilement pour naviguer verticalement à travers tel composant. Thermo pourra aussi simuler des connexions à des sources de données en générant du *Lorem Ipsum* structuré. Toutes les modifications dans l'interface génèrent en fond du code MXML. A tout moment, thermo pourra générer une prévisualisation en Flex. La définition de comportements sera aussi possible sous Thermo, sans avoir à saisir une ligne de code : action sur un click, un rollover… juste par paramétrage.

A mon sens, Thermo s'adressera à un public de designers, sensible à la RIA, qui connaît les codes de navigation de ces interfaces riches. Aucune date de sortie n'a été annoncée mais l'on sait déjà que Thermo génèrera du MXMLG, un langage XML déclaratif de design qui devrait logiquement être présent dans Flex 4. Il ne reste plus qu'à surveiller le site labs.adobe.com…

"L'erreur serait de croire qu'un seul profil est capable de maîtriser l'ensemble de la chaîne."

TÉMOIGNAGE

Rudy Jimenez, Le Studio Vert, nous dévoile sa vision, son quotidien avec les nouvelles technologies du web et la convergence collaborative proposée par les outils de dernière génération.

Programmez : on emploie le terme de "devsigner", fusion entre le développeur web et le designer web. Cette fusion se traduit dans la fonction, les processus, les outils, la collaboration, faut-il y croire et comment cela se concrétise ?

Rudy Jimenez : *Il me semble que cela dépend principalement de la typologie des projets, et du type de technologie mis en œuvre. S'il s'agit de développer des outils interactifs avec Flash par exemple, il est vrai que le designer est souvent formé à l'ActionScript pour avoir un maximum d'autonomie et de liberté dans la réalisation de son projet. En observant la mutation des profils au sein du Studio Vert, je dirais que ce sont surtout les designers/intégrateurs qui ont étendu leurs compétences techniques. En effet, la volonté de produire du XHTML propre et valide, de créer des feuilles de styles structurées, a impliqué progressivement une prise en compte de contraintes nouvelles, dont on se souciait moins en éditant des pages en tableau il y a quelques années. La démocratisation des standards a imposé une réflexion plus poussée dans la structuration des pages. La mise en forme, pour être efficace, a nécessité d'appliquer des concepts d'héritage qui sont issus de méthodes de développeurs. Cette prise de conscience et cette volonté de qualité ont été pour nous un déclencheur dans la convergence et l'ouverture entre designer et développeur. Chacun s'est donc intéressé aux contraintes de l'autre. Un autre facteur de cette évolution réside dans l'utilisation de CMS comme Spip, du Framework Ruby On Rails, ou de composants prêts à l'emploi. Les designers/Intégrateurs maîtrisent une plus grande partie de leur projet, exploitent du javascript (Scriptaculous...) intègrent des services pré-développés et sont donc beaucoup plus autonomes dans la mise en oeuvre d'un projet web. Le développeur "pur et dur" intervient donc plus tard dans la chaîne. Attention cependant, ceci est valable pour des projets de taille moyenne. Les projets lourds nécessitant une analyse pointue, la description de cas d'utilisation et une modélisation UML ne répondent pas à ces règles. Dans ce cas, l'équipe retrouve un schéma plus traditionnel où chacun se concentre sur ses compétences profondes pour optimiser les résultats et la gestion du projet.*

P ! : est-ce une bonne ou une mauvaise chose et pourquoi ?

R. J. : *Il me semble que repousser les frontières de ses compétences est toujours positif. Le gain d'autonomie des designers dans la production de sites web est bénéfique sur 2 plans : la maîtrise de certaines fonctionnalités, l'ex-*

ploitation de services dits "web2.0" ou d'Ajax, par exemple, permet au webdesigner d'élargir sa vision des interfaces, et donc d'élaborer une démarche créative différente en apportant de la valeur pour produire des sites plus riches et agréables à utiliser, et le gain d'autonomie permet de fluidifier la production, en se libérant partiellement de la dépendance à une compétence tierce. Cependant, il faut rester prudent. L'erreur serait de croire qu'un seul profil est capable de maîtriser l'ensemble de la chaîne. Il est important d'identifier les limites du champ d'action, pour que le designer soit en position de confort et ne se retrouve pas dans une situation d'échec. C'est alors à lui de repousser progressivement ses limites, en fonction de ses aspirations.

P ! : les outils poussent au " devsigner ". Qu'en est-il chez vous ?

R. J. : *Nous n'utilisons pas les services de workflow intégrés à ces solutions. Les designers utilisent un éditeur HTML pour la structure de leurs pages et un éditeur de feuilles de styles (CSS Edit) pour les mettre en forme. Tous les éléments graphiques sont, bien entendu, travaillés dans Illustrator, Photoshop et encore Fireworks. Les développeurs, quant à eux, travaillent dans un environnement de développement IDE. La fabrication des sites est sécurisée par un système de versionning (Subversion) pour bénéficier d'une traçabilité et permettre d'éventuels retours en arrière. Pour faciliter le travail collaboratif, il est primordial de définir une charte de nommage des fichiers, et de structurer leur organisation sur les serveurs pour limiter au maximum les erreurs d'interprétation et le sentiment désagréable de ne plus retrouver ses petits.*

P ! : ce rapprochement oblige-t-il à redéfinir le rôle de ces deux personnes ?

R. J. : *Il ne s'agit pas selon moi de redéfinir les rôles des acteurs, mais plutôt de bien baliser leur domaine d'intervention, et d'établir une méthodologie de travail claire et connue des deux parties. Il fut un temps où l'intégrateur préparait ses pages HTML et ses feuilles de styles, puis il passait la main au développeur pour qu'il puisse y intégrer son code et dynamiser les pages. Aujourd'hui, pour certains projets, l'utilisation de frameworks permet, par exemple, de modéliser la structure du site et de ses bases, de commencer à développer des fonctions métier avant même que les pages HTML soient prêtes. Cela impose un changement dans la chronologie du processus de fabrication, et une meilleure connaissance entre designer et développeur du métier de l'autre.*

P ! : et sur le terrain ?

R. J. : *Ce schéma a ses limites. Il existe une frontière entre les deux métiers. On ne choisit pas d'être designer ou développeur par hasard. Certes, selon la curiosité et la sensibilité de chacun cette frontière est plus ou moins nette, mais il y a toujours une limite pour laquelle l'un ou l'autre éprouve une réticence et se dit qu'il a bien fait de choisir son métier, et c'est cette complémentarité qui permet globalement d'aller plus loin, il y a tant à explorer dans chacun des domaines.*

■ propos recueillis par **Jean Vidames**

DEVELOPPEZ VOTRE SAVOIR-FAIRE

ABONNEMENT .NET

Programmez ! ***est le magazine du développement***

Langage et code, développement web, carrières et métier : Programmez, c'est votre outil de veille technologique.

Pour votre développement personnel et professionnel, abonnez-vous à Programmez !

PROGRAMMEZ

- **Abonnement 1 an au magazine : 45 €** (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*
- **1 an au magazine + 5 numéros HS .NET : 63 €** *Tarif France métropolitaine*
- **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 57 €** *Tarif France métropolitaine*
- **Abonnement PDF / 1 an : 30 €** *- Tarif unique* Inscription et paiement exclusivement en ligne www.programmez.com
- **Abonnement Etudiant : 1 an au magazine : 39 €** (au lieu de 65,45 € tarif au numéro) *Offre France métropolitaine*

PROGRAMMEZ HORS SERIE .NET *Seulement*

- **Abonnement 1 an aux Hors Série : 5 numéros : 20 €** (au lieu de 25 € tarif au numéro) *Tarif France métropolitaine*

Abonnez-vous :

Le magazine mensuel

11 numéros par an : **45 €***

soit **3** Numéros GRATUITS

**Tarifs France métropolitaine*

Abonnez-vous :

Les Hors-Série .Net

5 numéros par an : **20 €***

soit **1** Numéro GRATUIT

**Tarifs France métropolitaine*

+ 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*

Vous pouvez aussi vous abonner en ligne et trouver tous les tarifs www.programmez.com

PROGRAMMEZ

- ☐ **Abonnement 1 an au magazine : 45 €** (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*
- ☐ **1 an au magazine + 5 numéros HS .NET : 63 €** Seulement (au lieu de 90,45 € tarif au numéro) *Tarif France métropolitaine*
- ☐ **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 57 €** *Tarif France métropolitaine*
- ☐ **Abonnement Etudiant : 1 an au magazine : 39 €** (au lieu de 65,45 € tarif au numéro) *Offre France métropolitaine*

PROGRAMMEZ HORS SERIE .NET NOUVELLE OFFRE

- ☐ **Abonnement 1 an aux Hors Série : 5 numéros : 20 €** *(au lieu de 25 € tarif au numéro) Tarif France métropolitaine*

☐ M. ☐ Mme ☐ Mlle Entreprise : ..... Fonction : .....

Nom : ..... Prénom : .....

Adresse : .....

Code postal : ..... Ville : .....

Tél : ..... E-mail : .....

☐ Je joins mon règlement par chèque à l'ordre de Programmez ! ☐ Je souhaite régler à réception de facture

A remplir et retourner sous enveloppe affranchie à :

Programmez ! - Service Abonnements - 22 rue René Boulanger - 75472 Paris Cedex 10.

abonnements.programmez@groupe-gli.com

Programmez! LE MAGAZINE DU DÉVELOPPEMENT

Offre limitée, valable jusqu'au 30 mai 2008

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 108

La validation des données en ASP.NET

Depuis l'avènement de la plate-forme .NET, Microsoft à orienté sa politique sur l'ergonomie du développeur pour donner un modèle de programmation simple et complet. En effet, de nombreuses classes et interfaces ont été implémentées pour rendre votre code robuste et réutilisable. Tout cela dans le but de répondre à des besoins utilisateurs de plus en plus complexes.

Nous allons dans cet article mettre en avant un concept important dans vos développements, c'est la validation des données " Utilisateur ". En effet, une application robuste et ergonomique se doit de traiter de manière pragmatique les informations que l'utilisateur va fournir. Nous allons aborder ensemble les validateurs en ASP.NET.

Pourquoi valider les données utilisateurs ?

Tout d'abord, avant de démarrer avec des exemples concrets, il est nécessaire de présenter la validation de manière " théorique ". En ce qui concerne une application Web, la validation concerne deux parties bien distinctes : la validation côté client et côté serveur.

Pourquoi la validation côté client ?

La validation côté client est utile pour le contrôle des informations sur " la forme ", c'est-à-dire vérifier le format des données. En effet, il n'est pas nécessaire de faire un postback (retour serveur), pour vérifier qu'un type saisi dans une textbox est valide ou non. Dans un formulaire de saisie, le type de chaque donnée est connu à l'avance, par conséquent, une validation côté client permet de notifier directement à l'utilisateur que l'information qu'il vient de saisir n'est pas valide sans manipulation côté serveur.

Pourquoi la validation côté Serveur ?

La validation côté Serveur est dans vos applications, en effet, les validations côté client sont effectuées par du code javascript. Dans le cas où le ce code est désactivé sur le navigateur, la validation côté client n'a pas lieu, il est donc nécessaire d'effectuer une validation côté serveur pour déterminer si les données saisies sont valides. Si votre code-behind (code serveur) a été désactivé, ne lancez pas la validation de la page, vous ne maîtriserez pas les données que l'utilisateur a posté, via le formulaire. Nous verrons pour chaque validateur comment effectuer la validation côté client et côté serveur.

Côté serveur, le fonctionnement est similaire quel que soit le validateur employé, il suffit d'appeler la méthode " Page.Validate() ", et tous les validateurs de la page effectueront leur test de validité. Nous verrons également, qu'il est possible de réaliser une validation conditionnelle, c'est-à-dire en précisant le groupe de validation à vérifier.

Les contrôles validables

Contrôle	Propriété à Valider
HtmlInputText	Value
HtmlTextArea	Value
HtmlSelect	Value
HtmlInputFile	Value
TextBox	Text
ListBox	SelectedItem.Value
DropDownList	SelectedItem.Value
RadioButtonList	SelectedItem.Value
FileUpload	FileName

RequiredFieldValidator : Testez un champ obligatoire

Le " RequiredFieldValidator " est associé à un contrôle utilisateur via la propriété " ControlToValidate ".

Propriété	Description
ControlToValidate	Identifiant du contrôle à valider.
Text	Texte affiché par le validateur en cas de contrôle associé vide.
Error Message	message explicite affiché dans un "ValidationSummary ", ce composant sera utilisé en fin d'article.
ValidationGroup	détermine le groupe de validation auquel le validateur est associé, utilisé pour la validation conditionnelle, nous verrons également ce point plus précisément au cours de l'article.

Cadres d'utilisation

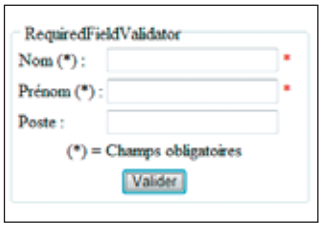
Dans un formulaire de saisie, si vous avez des champs obligatoires, vous pouvez associer un RequiredFieldValidator par contrôle. De plus, vous pourrez combiner l'utilisation d'un ValidationSummary, qui présentera toutes les erreurs du formulaire.

Code Client fichier " UseRequiredFieldValidator.aspx "

Les champs " Nom " et " Prénom " sont chacun associés à un RequiredFieldValidator.
Extrait de code

```
<asp:TextBox ID="txtNom" runat="server"></asp:TextBox>
<asp:RequiredFieldValidator ID="rfvTxtNom" ControlToValidate="txt
Nom" ErrorMessage="Nom Obligatoire" Text="*" ValidationGroup="pn
lNomPrenom" runat="server" />
```

Si nous tentons de faire un retour serveur avec le bouton " Valider " regardons ce qui se passe :



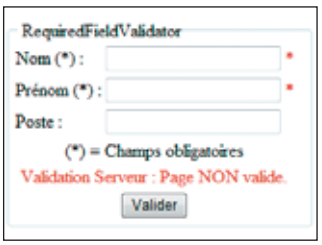
Lors de la demande de postback, les deux validateurs ont exécuté chacun leur code de validation côté client (via javascript). La validation côté client à échoué, donc il n'y a pas de validation côté serveur. Les erreurs sont affichées par les validateurs. Si nous renseignons des valeurs pour les champs textes " Nom " et " Prénom ", voici le résultat (ci-contre). Après saisie des champs obligatoires, nous cliquons sur le bouton Valider : Le contenu de la page

a changé, affichage du message : " Validation Serveur : Page valide ". Conséquence : La page a été postée, de plus, une validation côté serveur à été lancée, et le résultat est affichée en vert.

Code " Serveur " (Code-Behind) fichier " UseRequiredFieldValidator.aspx.cs "

Voici le code qui a été exécuté coté serveur pour valider le formulaire

```
protected void Page_Load(object sender, EventArgs e)
{
    // Test si la page est issue d'un premier postback
if (Page.IsPostBack)
        {
          // Appel de la méthode de validation de la Page coté serveur
                // Pourquoi appeler la validation coté serveur ?
                // Réponse : Si le javascript à été désactivé sur le navigateur client
                // le validateur n'a pu signaler si la TextBox était valide (non vide)
                Page.Validate();

                // Chaine de caractère générant le message de résultat de validation
                StringBuilder sbResultat = new StringBuilder("Validation Serveur : ");

            // Test si à l'issue de la validation via la méthode "Page.Validate()"
                // la page est bien valide.
                if (Page.IsValid)
                {
                    // la page est valide.
                    sbResultat.Append("Page valide.");
                }
                else
                {
                    // la page est NON valide.
                    sbResultat.Append("Page NON valide.");
                }

                // affichage à l'écran du résultat.
                this.ltlResultat.Text = sbResultat.ToString();
    }
}
```



Si le code javascript à été désactivé sur le navigateur client, c'est le code côté serveur qui va se charger de la validation de la page. L'exemple ci-contre illustre très bien l'obligation de tester la validité de la page côté serveur, cela permet de maîtriser les données envoyées par l'utilisateur.

CompareValidator : Testez le type et le contenu d'une valeur

Le CompareValidator permet d'effectuer deux types de comparaison :
- Vérifier le type saisi d'un contrôle.
- Comparer le contenu de deux contrôles.

A noter que les deux types de validation sont mutuellement exclusifs.

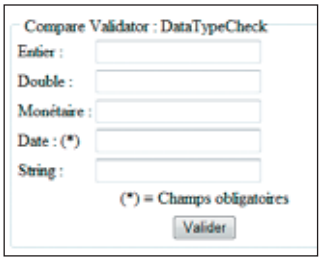
CompareValidator : DataTypeCheck

D'une part, le CompareValidator permet de vérifier que le type saisi dans un contrôle respecte le type de donnée défini.

Propriétés

Propriété	Description
ControlToValidate	Identifiant du contrôle à valider
Operator	DataTypeCheck
Type	type de donnée souhaité, valeurs possibles : " Integer, Double, Currency, Date, String "

Exemple

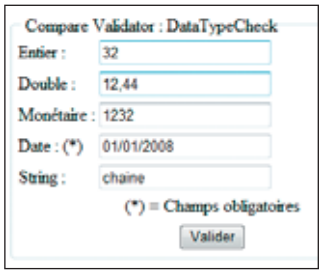


Dans cet exemple nous allons voir toutes les comparaisons de type possibles. A noter, le champ Date se voit attribué deux validateurs, un RequiredFieldValidator pour rendre obligatoire la date, et un CompareValidator pour tester la validité de la date.

```
<asp:Literal ID="ltlEntier" Text="Entier : " runat="server" />
<asp:TextBox ID="txtEntier" runat="server" />
<asp:CompareValidator ID="cvTxtEntier" runat="server" ErrorMessage=
"Format Non Valide" ControlToValidate="txtEntier" Operator="DataType
Check" Type="Integer" />
<asp:Literal ID="ltlDouble" Text="Double : " runat="server" />

<asp:TextBox ID="txtDouble" runat="server" />
<asp:CompareValidator ID="cvTxtDouble" runat="server" ErrorMessage
="Format Non Valide" ControlToValidate="txtDouble" Operator="Data
TypeCheck" Type="Double" />

<asp:Literal ID="ltlCurrency" Text="Monétaire :" runat="server" />
<asp:TextBox ID="txtCurrency" runat="server" />
<asp:CompareValidator ID="cvTxtCurrency" runat="server" ErrorMessage
="Format Non Valide" ControlToValidate="txtCurrency" Operator="Data
TypeCheck" Type="Integer" />

<asp:Literal ID="ltlDate" Text="Date : (*)" runat="server" />
<asp:TextBox ID="txtDate" runat="server" />
<asp:CompareValidator ID="cvTxtDate" ErrorMessage="Format Non Valide"
 ControlToValidate="txtDate" Operator="DataTypeCheck" Type="Date"
Display="Dynamic" runat="server" />
<asp:RequiredFieldValidator ID="rfvTxtDate" ControlToValidate="txtDate"
 Text="Date Obligatoire" runat="server" />
```

Saisie de données erronées

Le " CompareValidator " teste le contrôle associé dès que celui-ci perd le focus, de ce fait, il n'y pas besoin de tenter un postback avec le " bouton " " Valider " pour lancer la validation côté client.



Saisies de données correctes (Sans Postback)

Etant donné que chaque " CompareValidator " valide son contrôle à chaque fois que le contrôle associé perd le focus, la tentative de postback n'est pas nécessaire pour valider l'ensemble du formulaire côté client.

Si nous tentons de faire un postBack, voici la réponse du serveur en vert.

CompareValidator : " Comparaison de Contrôles "

D'autre part, le CompareValidator permet d'agréger le contenu de deux contrôles.

Propriétés

Propriété	Description
Type	Le type de la comparaison : String, Integer, Double, Date et Currency. Les valeurs sont contenues dans le type Enum :ValidationDataType.
Operator	Valeurs disponibles : NotEqual, GreaterThan, GreaterThanEqual, LessThan et LessThanEqual. Les valeurs sont contenues dans le type Enum : ValidationCompareOperator.
ControlToValidate	Contrôle utilisateur à valider
ControlToCompare	Contrôle tiers sur lequel est basé la comparaison

Remarque

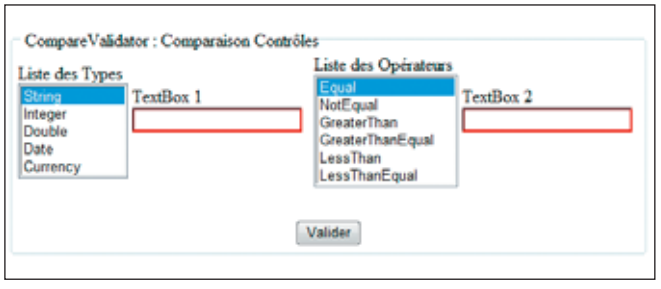
La propriété " Type " est très importante, si vous ne précisez pas le type de comparaison, c'est le type " String " qui sera utilisé par défaut… Attention aux mauvaises surprises…

Contexte

Cet exemple permet d'illustrer l'utilisation du CompareValidator en comparant deux contrôles :

- Le type de Comparaison : " Liste des Types " (tous les types acceptés par la propriété Type).
- L'opérateur de Comparaison : " Liste des Opérateurs " (Tous les opérateurs acceptés par la propriété Operator).

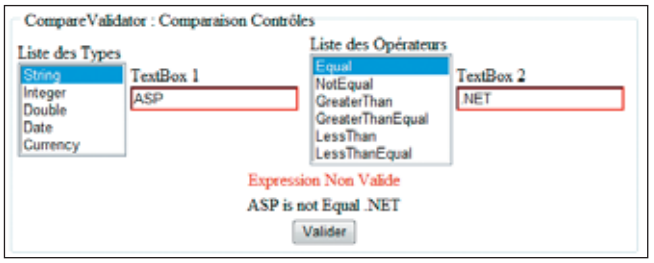
But



Le but de l'exemple est de générer des comparaisons entre les deux textbox en précisant le Type et l'Opérateur.

Première comparaison

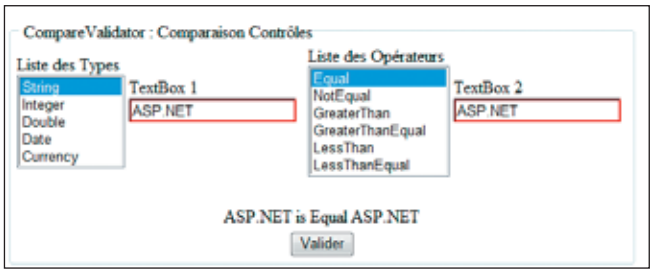
Type = " String " TextBox1 = " ASP " Operator = " Equal " TextBox2 = ".NET "



Le contenu des deux TextBox est bien différent, par conséquent le CompareValidator affiche un message en rouge : " Expression Non Valide ".

Deuxième comparaison

Type = " String " TextBox1 = " ASP.NET " Operator = " Equal " TextBox2 = ".NET "



Le contenu des deux textbox est identique, par conséquent, le " CompareValidator " n'a pas retourné d'erreur.

Pourquoi ces exemples

Cet exemple à deux objectifs, le premier est de vous montrer la puissance du " CompareValidator " en ce qui concerne la comparaison, en effet, avec quelques lignes de code il est possible d'effectuer une comparaison, et si le besoin fonctionnel évolue, et que le type doit changer, il suffit simplement de modifier les propriétés : " Operator " et " Type ". Le deuxième objectif, est de vous montrer qu'il est possible de changer au cours de l'exécution les propriétés de votre " CompareValidator ", par conséquent, dans vos développements, il est tout à fait possible d'utiliser un seul " CompareValidator " pour effectuer la validation conditionnelle. C'est-à-dire, adapter le comportement du " CompareValidator " en fonction du contexte. A noter : c'est exactement ce qui a été réalisé dans l'exemple ci-dessus. A chaque changement d'index dans une ListBox, les propriétés " Type " ou " Operator " sont changées côté serveur.

RangeValidator : Vérifiez l'existence d'une valeur dans un Intervalle

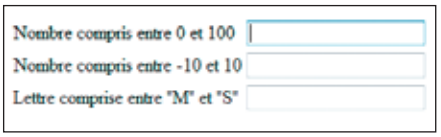
Nous allons dans ce troisième exemple, découvrir le fonctionnement du " RangeValidator ", qui permet de vérifier si la valeur d'un contrôle est bien définie dans un intervalle donné.

Propriété	Description
MinimumValue	borne minimale de l'intervalle.
MaximumValue	borne maximale de l'intervalle.
Type	définit le type de la valeur saisie. A noter que cette propriété est très importante, en effet, par défaut, le type défini est " String ", donc si vous souhaitez comparer des entiers, il faut bien préciser la propriété Type = Integer.

Remarque

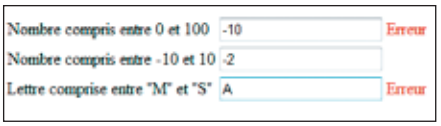
L'intervalle défini par la borne minimale et maximale est un intervalle ouvert.

Exemple



Trois Textbox associées à trois RangeValidator.

Première saisie



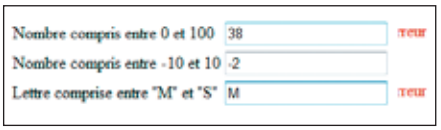
Résultats

Des erreurs ont relevées par les RangeValidator :

Nombre compris entre 0 et 100 => Erreur, le nombre " -10 " n'appartient pas à la borne spécifiée.

Nombre compris entre -10 et 10 => OK, le nombre " -2 " appartient bien à la borne [-10,10].

Deuxième saisie



Après correction les valeurs saisies sont valides.

Remarque

Dans cet exemple les validations ont été effectuées sur le client via le code javascript des " RangeValidator ", pour être exhaustif il est obligatoire d'effectuer une validation côté serveur avec la méthode " Page.Validate() ", comme présenté sur l'exemple du " RequiredFieldValidator ".

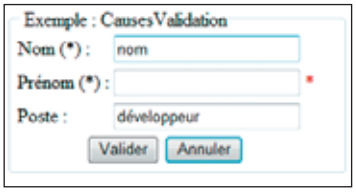
Utilisation du " CausesValidation " : Annulez une saisie

La propriété " CausesValidation " permet de spécifier le comportement d'un " Bouton " ou autre composant pouvant générer un postback. Dans votre application web, si vous utilisez des composants de " Validation ", lors d'une demande de postback, il se peut que les données saisies soient incorrectes. De fait, une problématique apparaît d'elle-même, *" Comment peut-on annuler la saisie d'un formulaire contenant des validateurs ? "*. Autrement dit, comment peut on " Annuler " l'action en cours, et demandez une autre page via un postback ?

La réponse est simple, le bouton " Annuler " ne doit pas déclencher les fonctions de validation. Pour ce faire, il faut mettre False à la propriété " CausesValidation " du contrôle demandant le postback sans validation de la page. Par conséquent, si une saisie est erronée (un ou plusieurs " Validateurs " retournent " False "), et que l'utilisateur souhaite " Annuler ", lors du clic sur le bouton " Annuler ", la validation de la page côté client n'est pas exécutée, par conséquent le postback est autorisé.

Exemple

Voici un formulaire simple mais montrant bien la problématique. Nous avons un formulaire avec trois " Textbox ", deux " RequiredFieldValidator ", et deux boutons.



Admettons qu'un utilisateur soit en cours de saisie, et ne souhaite pas valider les informations. L'utilisateur tente de cliquer sur le bouton " Annuler ".

Résultat

La page n'est pas renvoyée vers le serveur, en effet, le clic sur le bouton " Annuler " déclenche la validation côté client, et celle-ci retourne " False " car le " Validateur " associé au champ " Prénom " a relevé une erreur.

Comment passer outre la validation en cas d'annulation ?

Pour autoriser un postback, même si la page n'est pas valide, il faut directement intervenir sur le comportement du bouton " Annuler ", via la propriété " CausesValidation ".

Par défaut, lorsque vous créez un composant permettant de faire un postback, la propriété " CausesValidation " est à " True ", donc, si vous cliquez sur ce composant, la validation côté client va être exécutée, par conséquent, si un validateur retourne " False ", la demande de postback sera rejetée. Par contre, si vous affectez la valeur " False " à la Propriété " CausesValidation ", lors du clic sur le bouton annulé, la validation côté client ne sera pas exécutée et le postback est accepté.

```
<asp:Button ID="btnAnnulerCausesValidation" runat="server" Text="Annuler" CausesValidation="false" />
```

Si un de vos contrôles a besoin de faire un postback, même si la page est non valide, la Propriété " CausesValidation " doit être à " False " pour que votre postback soit accepté sans condition.

Groupes de Validation : Validez partiellement un formulaire

Nous allons maintenant parler des groupes de validation. Ceux-ci vous seront très utiles dans le cadre de la réalisation de formulaires complexes. En effet, si votre page est complexe, et qu'il y a différents éléments à valider, les groupes de validation vont vous permettre d'organiser votre processus de validation.

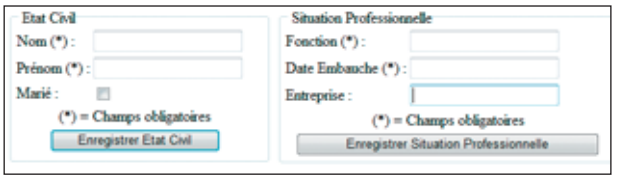
Propriété

Propriété	Description
ValidationGroup	méthode permettant de spécifier le groupe de validation d'un composant. Cette propriété existe pour tous les contrôles ASP.NET, elle permet de grouper la validation de plusieurs contrôles.

Exemple

Admettons que vous ayez un formulaire plus complexe avec deux groupes de validation :

- Etat Civil : 3 textbox et 2 " RequiredFieldValidator ". 1 bouton d'Enregistrement.
- Situation professionnelle : 3 textbox, 2 " RequiredFieldValidator " et 1 " CompareValidator " pour gérer la date + Un bouton d'Enregistrement.



Première tentative

Enregistrement des informations d'état civil en cliquant sur " Enregistrer Etat Civil "

Résultat

Seuls les " Validateurs " associés au ValidationGroup " EtatCivilValidationGroup " ont levé une erreur.

Pourquoi ?

Le composant qui a demandé le postback est le bouton " Enregistrer Etat Civil " qui appartient au " ValidationGroup " " EtatCivilValidationGroup ". Par conséquent, tous les validateurs de ce " ValidationGroup " ont lancé leurs méthodes " EvaluateIsValid ". Tous les autres " Validateurs " de la page n'ont pas été consultés. Donc, si nous remplissons les champs obligatoires dans le " ValidationGroup " " EtatCivilValidationGroup " le postback sera accepté.

ValidationSummary : Affichage Global des erreurs

Nous allons voir maintenant un composant omniprésent dans la validation ASP.NET, il s'agit du " ValidationSummary ". Ce composant permet comme son nom l'indique de faire une énumération des erreurs levées par les " Validateurs ".

Propriété	Description
HeaderText	Texte affiché en cas d'erreur.
ValidationGroup	permet d'associer le "ValidationSummary " à d'autres contrôles issus du même "ValidationGroup ".

Comment fonctionnent-ils ?

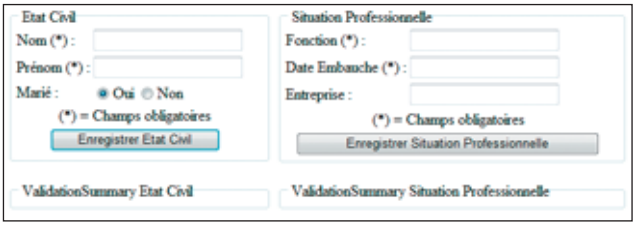
Un " ValidationGroup " permet d'afficher les erreurs de tous les validateurs associés au même " ValidationGroup ". Pour afficher le message d'erreur, le " ValidationSummary " affiche simplement la Propriété " ErrorMessage " du " Validateur " retournant " False ". Pour illustrer ce composant, nous allons réutiliser l'exemple précédent, et ajouter deux " ValidationSummary ", un pour chaque " ValidationGroup ".

Remarque

Les messages d'erreurs affichés à l'intérieur du " ValidationSummary " sont issus de la propriété " ErrorMessage " de chaque " Validateur " ayant retourné " False ".

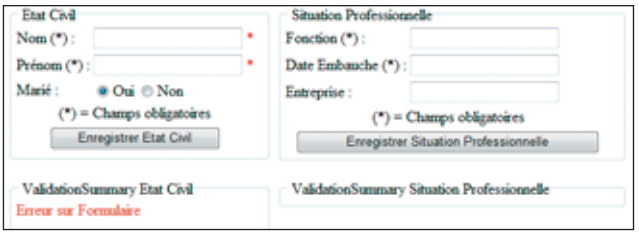
Contexte

Deux " ValidationSummary ", le premier nommé " ValidationSummaryEtatCivil " appartenant au " ValidationGroup " Etat Civil, et le deuxième nommé " ValidationSummarySituationProfessionnelle " appartenant au " ValidationGroup " Situation Professionnelle.



Première tentative

Enregistrement de l'état civil (avec des erreurs)

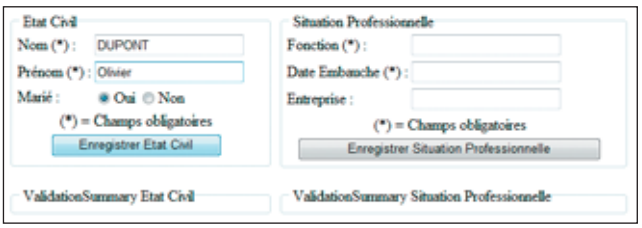


Résultat :

Le formulaire n'a pas été renvoyé vers le serveur, les " Validateurs " ont relevé des erreurs, par conséquent, le " ValidationSummary " affiche la propriété " ErrorMessage " de ces derniers. A noter, que le ValidationSummary lié à la situation professionnelle n'affiche aucune erreur.

Deuxième tentative

Tentative d'enregistrement de l'état civil (sans erreurs)



Résultat :

Tous les " Validateurs " dans le " ValidationGroup " Etat Civil, retournent " True " par conséquent, le " Validation Summary " n'affiche aucun message, et le formulaire est posté vers le serveur. A noter, que les deux parties du formulaire sont bien désynchronisées au niveau de la validation.

Conclusion

Nous avons vu au cours de cet article les différents principes de validation en ASP.NET, de plus, vous avez pu découvrir les différents composants offerts par le Framework .NET. A noter, que deux composants de validation n'ont pas été abordés au cours de l'article, d'une part le RegularExpressionValidator, dont le principe est de comparer une saisie utilisateur avec un pattern, d'autre part le CustomValidator qui de par sa puissance et sa complexité mérite à lui seul un article.

Julien Duprat - *Bewise* - *julien.duprat@bewise.fr*

Boutique Boutique Boutique
Achetez les magazines, les articles en PDF et abonnez-vous en ligne
www.programmez.com

Extraction, Transformation et Chargement de données avec Talend Open Studio

Même si ce n'est pas une évolution récente, les systèmes d'information des entreprises ont tendance à se complexifier au cours du temps, en raison de plusieurs facteurs, et notamment : l' " empilage de couches " (déploiement de nouvelles applications sans suppression des " vieux " systèmes), et l'ouverture, de plus en plus fréquente, des systèmes d'information de l'entreprise vers ceux de ses fournisseurs, de ses partenaires et de ses clients. Cette tendance ne fait que s'amplifier.

En parallèle, il faut gérer la multiplication des formats de stockage de données (fichiers : XML, plats positionnels, plats délimités multi-valués…), des protocoles (FTP, HTTP, SOAP, SCP…) et des technologies des bases de données.
Du côté des SGBD, même si on aurait pu croire il y a quelques années à une concentration du secteur, force est de constater que le nombre d'acteurs a plutôt tendance à augmenter : d'une part les acteurs historiques sont peu consolidés et perdurent, d'autre part de nouveaux entrants apparaissent (souvent issus de *fork* des acteurs historiques).
Ces constats nous amènent naturellement à l'intégration des données réparties dans le système d'information. L'intégration de données a plusieurs finalités : l'intégration décisionnelle (alimentation d'entrepôts de données) et l'intégration opérationnelle (migration de données, synchronisation de bases, échanges inter applicatifs…).

Outils commerciaux

La concentration dans ce domaine a été forte ces dernières années, de nombreux acteurs ont ainsi " disparu " :
- Génio (outil français) a été absorbé par Hummingbird (en 1999) elle-même absorbée par OpenText (en 2006)
- Acta par Business Objects (en 2002) puis par SAP (2007)
- Mercator par Ascential (2003) puis par IBM (en 2005)
- Sunopsis par Oracle (en 2006)
- DataMirror par IBM (en 2007)

Aujourd'hui, le seul acteur d'envergure sur ce marché à rester indépendant est Informatica.
Néanmoins, cela ne devrait pas durer longtemps ; en effet, cette société a publiquement déclaré que, même si elle n'est pas à vendre, elle ne refusera pas une offre généreuse de rachat (sic) !
Ces rachats et fusions posent de nombreux problèmes aux utilisateurs des solutions. D'une part, les offres techniques sont rarement conservées en l'état par le nouvel acquéreur, qui se doit de gérer ses priorités propres. D'autre part, les conditions commerciales changent du tout au tout, mettant en porte à faux les clients et compromettant des déploiements en cours. Qui plus est, la migration des processus d'intégration est particulièrement difficile, voire impossible.

La montée des Solutions Open Source

Le monde Open Source fourmille de projets liés de près ou de loin à l'intégration de données. La plupart restent au stade de projet, sont abandonnés ou développés par de très petites équipes.
Deux outils ont néanmoins émergé, portés par des sociétés commerciales, ils dominent le marché des outils d'intégration de données Open Source : Kettle et Talend.
Kettle a été, durant l'été 2006, racheté par la société Pentaho. Ce projet a alors été renommé *Pentaho Data Integrator* et intégré dans la ligne de produits décisionnels de Pentaho. L'ambition de Pentaho est de couvrir la totalité de la chaîne décisionnelle en procédant à des acquisitions dans chaque secteur et en intégrant les produits.
En plus de Kettle, Pentaho a notamment acquis les produits Mondrian (moteur ROLAP) et Weka (outil de Datamining). Talend reste, quant à lui, un éditeur indépendant, un " pure player " focalisé sur l'intégration de données pour tous types de projets – décisionnels ou non.
Ces deux produits sont actuellement matures, les offres portées par leurs éditeurs respectifs concurrencent sans problème celles des acteurs propriétaires et comprennent :
- cursus de formation
- support technique commercial (avec niveaux de services associés)
- et expertise (accompagnement)

Notre préférence va à l'outil Talend Open Studio que nous allons présenter ci-après.
Plus simple dans son approche et plus puissant, Talend Open Studio possède une étendue fonctionnelle très importante permettant de couvrir des besoins très variés :
- chargement de fichiers (positionnels, délimités…) dans des tables
- manipulation de fichiers XML (validation, transformation…)
- recherche de références croisées (lookup)
- transformations complexes
- tri, agrégation
- appel de Web Services à la volée
- dépôt et récupération de fichiers sur des serveurs FTP…
- qualité de données (déboulonnage, rapprochement flou…)

Prise en main

Téléchargez Talend Open Studio V2.3 à partir du site talend.com : http://www.talend.com
L'outil ne nécessite pas d'installation à proprement parler, une simple décompression suffit. Utilisez l'outil libre SevenZip, par exemple, car le décompresseur de Windows pose parfois quelques problèmes.
Ceci fait, lancez l'application en cliquant sur l'application TalendOpenStudio-win32-x86.exe

Scénario

Pour illustrer le fonctionnement de cet outil, nous allons nous baser sur un scénario réel. Dans ce scénario, nous devons charger un fichier clients dans une table MySQL en base, en appliquant des transformations à la volée. Pour compliquer un peu les choses nous avons choisi de ne charger que les données des clients de la région Île-de-France (départements 75, 77, 78, 91, 92, 93, 94, 95) en appliquant un filtre dynamique. Notre fichier clients d'entrée contient quatre colonnes :

- Prénom
- Nom
- Adresse1
- Département

La structure de notre table cible est différente puisqu'elle est composée par les quatre champs suivants :

- Key (clef, type entier)
- Name (type chaîne, longueur max 40)
- Address (type chaîne, longueur max 40)
- Region (type chaîne, longueur max 20)

Nous allons utiliser le Mapping suivant pour remplir notre table :

Le champ Key sera remplit avec un entier auto-incrémenté,

Le champ Name sera remplit avec la concaténation des champs Nom et Prénom,

Le champ Address reprendra le contenu du champ d'entrée Adresse1 et sera mis en majuscule,

Le champ Région sera rempli en mettant le nom du département correspondant au département du client.

Pour filtrer nos données et remplir correctement le champ Region, nous nous baserons sur le fichier de référence " IleDeFrance.txt " contenant les deux colonnes suivantes :

```
Departement;NomDepartement
75;Paris
77;Seine-et-Marne
...
95;Val-d'Oise
```

1 Création du job / définition du fichier d'entrée / lecture du fichier

Après avoir lancé Talend Open Studio, il faut créer une connexion au référentiel local ; pour cela, cliquer sur le bouton " … " à droite de Connection. Dans le champ **User E-mail,** entrez votre email (!) puis cliquez sur **Ok.** Cliquez ensuite sur le bouton Demo et sélectionnez Java pour importer les projets de Demo Java. Ceci prend un peu de temps, mais vous aurez accès à plusieurs dizaines d'exemples illustrant les principales fonctionnalités du Studio. Enfin, cliquez sur le bouton Ok pour lancer le Studio. Puis cliquez sur **Start using Talend Open Studio now!** La fenêtre de travail est divisée en plusieurs espaces :

A gauche : le référentiel ou sont référencés les jobs, les business models, les métadonnées, le code partagé, la documentation… Au centre : la zone de travail principale. En bas : les palettes de propriétés

A droite : les palettes de composants métier et techniques suivant l'outil utilisé.

Sur la gauche, au sein du référentiel sont disponibles les 3 logiciels principaux composants Talend Open Studio :

1) Le Business Modeler : interface de modélisation métier proche d'un outil comme MS Visio

2) Le Job Designer : interface de développement graphique des processus techniques

3) Le Metadata Manager : application permettant le stockage et la réutilisation des métadonnées.

Pour créer notre job nous allons commencer par cliquer sur l'Item **Job Designs** avec le bouton droit et choisir la première option du menu : **Create Job.** Dans la boîte de dialogue qui s'ouvre alors à l'écran, seul le premier champ **Name** est obligatoire. Entrez **Job1** et validez en cliquant sur le bouton Finish.

Un job vierge apparaît alors et la palette de composants techniques (sur la droite) se développe avec une dizaine de familles dans lesquelles sont regroupés les composants : Databases, Files, Internet, DataQuality… Cette version intègre environ 200 composants !

Pour lire notre fichier Clients, nous allons utiliser le composant **tFileInputDelimited.** Ce composant se trouve dans la famille **File / Input.** Cliquez sur ce composant puis cliquez sur la gauche de l'espace de travail pour créer le composant.

Il faut maintenant définir les propriétés de lecture de ce composant (nom du fichier, séparateur de colonnes, encodage…). Pour ce faire, nous allons utiliser le **Metadata Manager.** Cet outil va nous aider à spécifier les paramètres nécessaires grâce à des assistants et nous permettre de stocker ces propriétés pour les réutiliser dans de nombreux Jobs en un seul clic.

Puisque notre fichier d'entrée est un fichier plat délimité ; nous allons cliquer avec le bouton droit sur **Metadata / File Delimited** et sélectionner l'option **Create file delimited.** Ceci a pour effet de faire apparaître l'assistant spécifique aux fichiers délimités.

A l'étape 1, seul le champ **Name** est obligatoire : entrez simplement **" Clients "** et passez à l'étape suivante en cliquant sur le bouton **Next.**

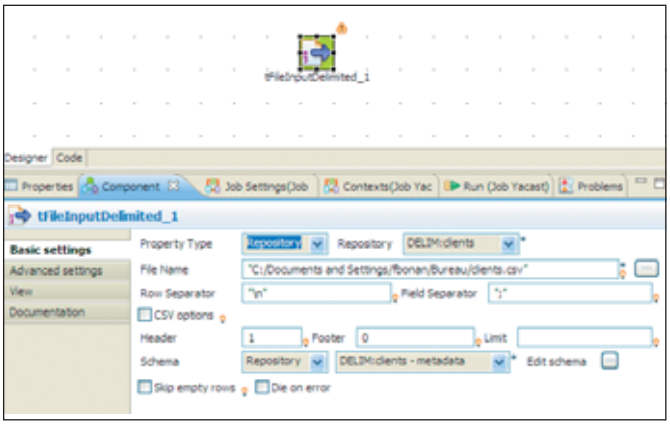
A cette étape, sélectionnez votre fichier d'entrée (clients.txt) avec le bouton **Browse...**

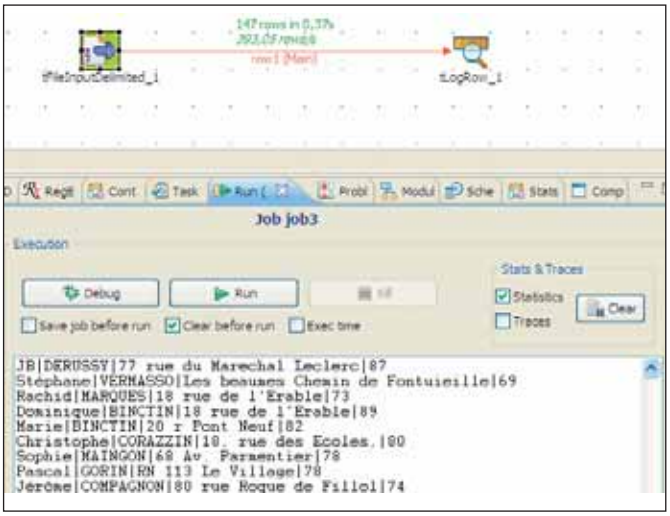
Ceci fait, un extrait du fichier apparaît en dessous, vous permettant d'en vérifier le contenu. Cliquez sur **Next.** A l'étape suivante, nous allons pouvoir spécifier l'encodage du fichier, le séparateur de ligne, de colonne… Puisque notre fichier d'entrée est standard, nous allons garder la plupart des valeurs par défaut. La première ligne de notre fichier est un header contenant le nom des colonnes. Pour récupérer automatiquement ces noms, cliquez sur la case à cocher **Set heading row as column name** puis sur le bouton **Refresh Preview.** Enfin, cliquez sur **Next** pour accéder à la dernière étape de cet assistant.

Dans cette étape, nous devons définir les types pour chacune des colonnes de notre fichier. Cet assistant intègre des algorithmes qui essayent de deviner ces types en analysant les données du fichier. Cette proposition reste modifiable ; néanmoins, dans notre cas ces informations sont cohérentes, nous allons donc les garder telles quelles.

La metadonnée " Clients " est définie ! Nous allons l'utiliser dans notre composant d'entrée. Sélectionnez votre composant **tFileInputDelimited,** puis cliquez sur le panneau **Properties.** Sélectionnez l'onglet vertical **Properties.** Dans cet onglet vous retrouvez les propriétés techniques nécessaires au composant. Au lieu de les saisir une à une, nous allons utiliser la Metadata que nous venons de définir. Dans la combobox **Property Type** sélectionnez **Repository,** ceci fait, une nouvelle combobox apparaît à droite : **Repository.**

Choisissez dans cette nouvelle liste la Metadata **Client.** Vous constatez alors que tous les paramètres sont automatiquement renseignés.



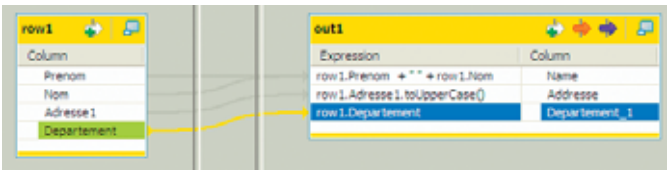


A cette étape, nous allons terminer notre flux en envoyant les données lues dans ce fichier vers la sortie standard (StdOut). Pour ce faire, ajoutez un composant **tLogRow** (il se trouve dans la famille **Logs & Errors**).
Pour lier ces composants entre eux, cliquez avec le bouton droit sur le composant de lecture et choisissez l'option **Row / Main.** Cliquez enfin vers votre composant de sortie : **tLogRow.**
Ce job est prêt à être exécuté. Pour le lancer, sélectionnez le panneau **Run (Job job1)** en bas.
Activez les statistiques en cochant la case **Statistics** et lancez le job en cliquant sur le bouton **Run.** Le contenu du fichier apparaît alors dans la console :

2 Mapping et application des transformations

Nous allons maintenant enrichir notre Job pour transformer nos données à la volée. Pour les transformations dont nous avons besoin dans notre scénario, nous allons utiliser le composant tMap (famille **Processing**). Ce composant est très puissant, il permet de gérer :
- entrées et sorties multiples
- recherche de références (simple, produit cartésien, premier, dernier…)
- les jointures (interne/externe)
- les transformations
- les rejets
- etc.

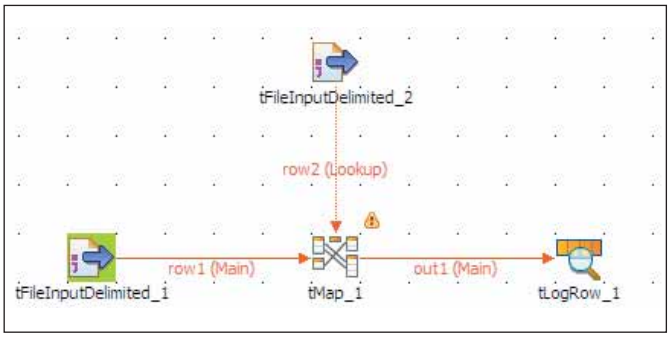
Supprimez le lien qui relie vos deux composants (bouton droit sur le lien puis option Delete). Placez le composant **tMap** entre vos deux composants. Liez votre composant d'entrée avec le **tMap** comme nous l'avons fait précédemment.
Enfin, pour lier votre **tMap** à la sortie standard, cliquez sur le composant **tMap** avec le bouton droit et sélectionnez **Row / *New Output* (Main)** ; entrez **out1** dans la boîte de dialogue et cliquez enfin sur votre composant de sortie tLogRow pour créer le lien. En toute logique, une boîte de message apparaît (retro-propagation des schémas), ni prêtez pas attention et cliquez simplement sur le bouton **No.**
Double cliquez maintenant sur le composant **tMap** pour ouvrir son interface. Sur la gauche, vous retrouvez le schéma (la description) de votre fichier d'entrée (row1). Sur la droite, votre sortie est (pour l'instant) vide

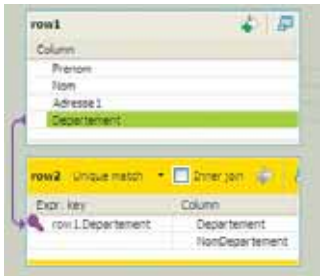


(out1). Glissez-déposez les colonnes Prenom et Nom de la gauche vers la droite comme le montre la copie d'écran suivante : ci-dessus.
Saisissez

3 Définition du fichier IleDeFrance.txt, mise en référence sur le tMap, sélection du mode de jointure interne

Définissez la Metadata correspondant au fichier **IleDeFrance.txt** comme nous l'avons fait précédemment pour le fichier **Clients** avec l'assistant. A l'étape 1 de l'assistant, nommez cette Metadata **IleDeFrance.** Glissez-déposez cette Metadata sur le haut de votre espace de travail pour créer automatiquement un composant de lecture pointant vers cette métadonnée.
Liez ensuite ce composant avec votre composant **tMap :**
Double-cliquez de nouveau sur le composant **tMap** pour ouvrir son Interface. Vous noterez sur la gauche en dessous de votre entrée principale (row1) que votre entrée de référence (correspondant à votre fichier Région) est apparue (row2). Reste à définir la jointure entre notre flux principal et notre flux de référence. Dans notre cas, la jointure est simple puisque dans nos deux flux, le code Département correspond exactement (2 chiffres). S'il en avait été autrement, il aurait été possible de faire des opérations directement à ce niveau pour rapprocher les données (padding, changement de casse…). Pour établir la jointure, glissez-déplacez la colonne Departement de votre première entrée sur la colonne Departement de votre entrée de référence. Un lient Violet apparaît, matérialisant cette jointure :





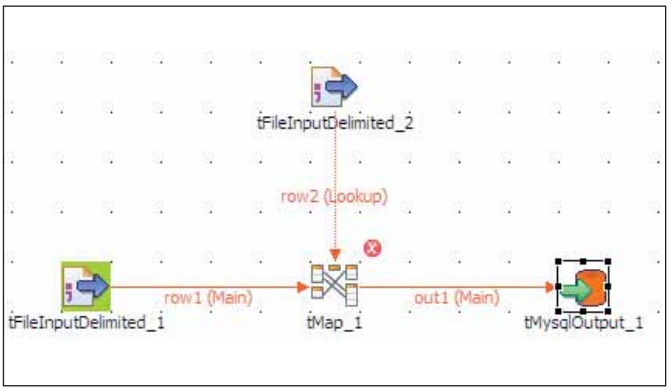
Nous pouvons maintenant utiliser la colonne NomDepartement de notre entrée de référence dans notre sortie. Pour ce faire, glissez/déposez cette colonne sur le bas de votre sortie à droite :
Enfin cliquez sur le bouton **Ok** pour valider vos modifications et lancez votre nouveau job.

Vous devriez obtenir la sortie console suivante : ci-dessus.
Comme vous le constatez, la dernière colonne n'est remplie que pour les lignes dont les départements font partie de notre fichier d'entrée. Pour les autres, cette colonne est vide.
Ceci s'explique par le fait, que, par défaut, le tMap travaille en mode de jointure interne. Si vous voulez filtrer vos données pour ne sortir que les lignes pour lesquelles le tMap arrive à faire la correspondance avec le fichier de référence, ouvrez le tMap et cochez la case *Inner join* sur votre entrée de référence (row2).

4 Redirection du résultat vers une table MySQL

Notre Job fonctionne à merveille, pour le finaliser, la dernière étape consiste à rediriger le flux de sortie vers une table MySQL.
Pour ce faire, la première étape est de créer la Métadonnée décrivant la connexion à notre base MySQL.

En double-cliquant sur Metadata / MySQL / DemoMysql dans le référentiel (sur la gauche), vous ouvrez l'assistant Metadata. A l'étape 2 de cet assistant, entrez les paramètres pour vous connecter à votre base de données. Vérifiez la validité de votre connexion en cliquant sur le bouton **Check** ; enfin, validez vos modifications en cliquant sur le bouton Finish.
Glissez-déposez sur la droite de votre espace de travail cette Metadonnée en maintenant la touche **Control** enfoncée pour créer automatiquement un composant tMysqlOutput. Supprimez le composant tLogRow.
Reconnectez votre sortie **out1** de votre tMap vers votre composant tMysqlOutput (clic droit / Row / out1) :
Dans l'onglet Properties de ce composant :



- entrez "idf_clients" dans le champ Table pour nommer votre table cible qui va être crée à la volée.
- sélectionnez également l'option **Drop and create** table pour le champ **Action on table.**

Lancez de nouveau votre Job. Votre table idf_client devrait être automatiquement créée et remplie en moins d'une seconde !
Nous avons dans ce scénario utilisé quatre composants différents, mais la palette en contient plus de 200 (bases de données, Webservice, Ftp…) ! D'autres composants, réalisés, eux, par la communauté, sont également disponibles sur le site communautaire de l'éditeur : talendforge.org.

Pour aller plus loin, vous pouvez télécharger la documentation (+ de 600 pages !) disponible gratuitement sur le site.
Et en cas de problème, n'hésitez pas à utiliser les outils communautaires de l'éditeur (Wiki, Forums, tutoriels…), les ressources offertes sont très nombreuses !

Fabrice Bonan

+ Tutoriel sur www.programmez.com/tutoriel.php

L'information permanente
• L'**actu** de Programmez.com : le fil d'info **quotidien**
• La **newsletter hebdo** : la synthèse des informations indispensables.
Abonnez-vous, c'est gratuit !
www.programmez.com

2e partie

Visualisation de sécurité en open source

Grâce aux éléments vus dans le précédent numéro, nous avons les clés pour réaliser la fonction qui nous intéresse : visualiser en temps réel une attaque.

Tout d'abord, cette visualisation se fera en 2D en utilisant Graphviz. Ensuite, nous allons voir comment modifier un programme existant pour adapter ce code Graphviz (langage dot) à un système de visualisation 3D.

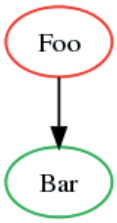
Graphviz

Graphviz est un système qui permet, à partir d'un langage de description, de produire une image dans des formats bitmaps (gif, png, jpg, ...) comme vectoriels (svg, ps, ...). Il est même possible de décrire la position des noeuds au format MAP compréhensible par du code HTML, afin de rendre les images cliquables précisément par rapport aux noeuds posés.

Le langage dot

Il s'agit du langage permettant de simplement décrire les graphes que vous souhaitez réaliser. Par exemple, le code dot suivant :

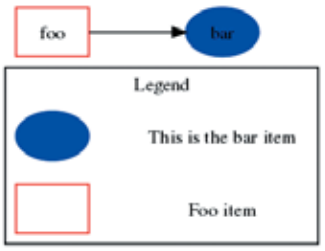
```
digraph G {
        node1 [label="Foo",color="red"] ;
        node2 [label="Bar",color="green"] ;
        node1 -> node2 ;
}
```



Produit, grâce à la commande **dot -Tpng graph.dot > graph.png** le graphe (ci-contre).

Le langage dot est relativement structuré : les attributs des noeuds comme ceux des liens sont placés devant leur déclaration. Afin de simplifier la création du code dot, nous utiliserons la classe Python **GvGen** (http://software.inl.fr/trac/wiki/GvGen). De cette façon, il est aisé de réaliser des graphes complexes : il suffit de déclarer les différents objets et de poser leur relations. Par exemple, si nous souhaitons créer un graphe avec une légende, le code dot devient compliqué parce que par défaut, graphviz ne sait pas gérer ce cas. Il faut en effet supprimer la flèche, tout en imposant une direction au graphe afin que le texte s'affiche à droite de l'élément que nous voulons mettre en légende, puis supprimer le lien et éviter d'entourer le texte de description. Voici le code GvGen permettant de générer un graphe simple avec une légende :

```
#! /usr/bin/python
import gvgen
 # Création d'un nouveau graphe avec une légende
graph = gvgen.GvGen("Legend")
 # Création des items "foo" et "bar"
a = graph.newItem("foo")
b = graph.newItem("bar")
 # Lier "foo" avec "bar"
graph.newLink(a,b)
 # Création et application des styles
graph.styleAppend("foostyle","color","red")
graph.styleAppend("foostyle","shape","rectangle")
graph.styleApply("foostyle", a)
 graph.styleAppend("barstyle","color","blue")
graph.styleAppend("barstyle","style","filled")
graph.styleApply("barstyle", b)
 # Création d'une zone de légende
graph.legendAppend("foostyle", "Foo item")
graph.legendAppend("barstyle", "This is the bar item")
 # Création du code dot
graph.dot()
```



On remarque de le code Python avec GvGen nous permet d'automatiser la création du langage dot, mais aussi de rendre sa création plus simple à l'aide de fonctions n'existant pas dans graphviz, comme les définitions de styles.

Ce code dot permet de créer l'image ci-dessus.

Si vous souhaitez voir un excellent exemple d'application utilisant GvGen, vous pouvez vous tourner vers Graphdep (http://haypo.hachoir.org/trac/wiki/graphdep), un outil d'affichage de dépendances entre fonctions dans du code C, C++ ou python.

int-gration-prelude [2]Intégration à Preludeint-gration-prelude

Intégration à Prelude

Prelude, comme vu au début de l'article, permet de récupérer grâce au système de polling toutes les alertes que les différents analyseurs ont pu lui envoyer. Grâce aux *binding* faciles, nous pouvons récupérer tous les objets IDMEF souhaités pour les passer à **GvGen.** Le but étant d'arriver au graphe 2D permettant de visualiser des attaques. Nous allons récupérer les cibles afin de regrouper les attaques avec des jeux de couleurs représentant la gravité de chacune des attaques. C'est en récupérant une branche spécifique de Prelude que nous allons pouvoir développer facilement l'affichage des attaques en langage dot.

Intégration à Rtgraph3d

Rtgraph3dà été développé par Philippe Biondi dans le but d'offrir une application équivalente à Graphviz, mais en 3D et en temps réel. DBus est utilisé pour pouvoir se connecter et interagir avec l'application en envoyant une simple chaîne de texte. Sachant que nous avons déjà réussi à générer le code dot, il serait fâcheux de reprendre tout le travail. Il est donc plus simple de modifier le code de rtgraph3d afin d'y ajouter l'import Graphviz. Pour cela, nous allons utiliser Pydot, qui permet de lire le code dot et de le transformer en objets Python. L'algorithme utilisé sera le suivant :

Pour chaque lien :
  Pour chaque noeud du lien :
    Si c'est la source :
      label_source = attribut label du noeud
      couleur_source = attribut color du noeud

```
    Si c'est la destination :
        label_dest = attribut label du noeud
        couleur_dest = attribut color du noeud
    Si nous n'avons aucun label pour les sources :
        Attribuer le nom du lien pour le noeud
    Création du noeud source nommé et avec la couleur normalisée
    Création du noeud destination nommé et avec la couleur normalisée
    Si la couleur est mise en paramètre, mettre à jour les noeuds
```

Et en appliquant le patch suivant, on peut maintenant utiliser la fonction d'import dot dans rtgraph3d :

```
-
-- rtg_cli.py.orig      2008-01-10 19 :20 :30.000000000 +0100
+++ rtg_cli.py  2008-01-10 19 :19 :47.000000000 +0100
@@ -94,6 +154,35 @@
          for n1,n2 in zip(e[ :-1],e[1 :]) :
              self.rtg.new_edge(n1,{},n2,{})
 +    def do_dot(self, args) :
+        graph = pydot.graph_from_dot_file(args)
+        for edge in graph.get_edge_list() :
+            edge_src = "\"" + edge.get_source() + "\""
+            edge_dst = "\"" + edge.get_destination() + "\""
+            for node in graph.get_node_list() :
+                if node.name == edge_src :
+                    src_label = node.get_label()
+                    src_color = node.get_color()
+                if node.name == edge_dst :
+                    dst_label = node.get_label()
+                    dst_color = node.get_color()
+
+            if not src_label :
+                src_label = edge.get_source()
+
+            if not dst_label :
+                dst_label = edge.get_destination()
+
+
+            edge_args = src_label + " " + dst_label
+            self.do_edge(edge_args)
+            if src_color :
+                src_color_s = src_label + " color=" + color_normalize(src_color)
+                self.do_update(src_color_s)
+            if dst_color :
+                dst_color_s = dst_label + " color=" + color_normalize(dst_color)
+                self.do_update(dst_color_s)
+
     def do_glow(self, args) :
         for n in args.split() :
             self.rtg.glow(n)
```

Pour la normalisation des couleurs, nous utilisons la fonction **color_normalize** qui est disponible à l'adresse suivante : http://www.wallinfire.net/files/color_calc.py

Maintenant que nous avons tout ce code, il nous reste à faire le code Prelude qui fait le lien entre :

- Une attaque que nous produisons volontairement : un scan de ports, un utilisateur qui se trompe de mot de passe, etc.
- Une sonde Prelude qui détecte cette attaque
- Notre code qui récupère les objets que le collecteur reçoit des sondes
- Graphviz, que nous utilisons avec GvGen pour générer du code dot
- RtGraph3d que nous utilisons avec le patch mis plus haut pour avoir une vue en 3D des attaques

Voici le code que nous allons utiliser pour produire le graphe. Il est montré ici par morceaux, mais il est disponible en libre téléchargement à l'adresse : http://svn.prelude-ids.org/libprelude/branches/libprelude-easy-bindings/bindings/tools/idmef-path-gvgen.py

Tout d'abord, on crée le client avec ce code :

```
import PreludeEasy
 client = PreludeEasy.Client("PreludeGvGen")
client.Init()
 client.PoolInit("192.168.33.215", 1)
```

Le nom de notre client est "PreludeGvGen", cela veut dire qu'il faut l'enregistrer avec les capacités IDMEF de lecture sur le collecteur. On le fait en tapant sur la machine qui sera utilisée pour lancer l'application de cette façon :

```
~
# prelude-admin register "PreludeGvGen" "idmef :r" 192.168.33.215 --uid
 1000 --gid 1000
```

Puis, sur le collecteur :

```
~
# prelude-admin registration-server prelude-manager
```

Une fois la sonde enregistrée pour l'utilisateur ayant le numéro 1000, on peut l'utiliser en tant que simple utilisateur. La fonction de lecture d'objets IDMEF est initialisée à l'aide de la fonction PoolInit.

Maintenant, nous pouvons faire une boucle infinie pour la lecture de chaque objet IDMEF que le collecteur verra :

```
while 1 :
    idmef = client.ReadIDMEF(1)
    if idmef :
        handle_
alert(idmef)
```

C'est la fonction ReadIDMEF qui permet de lire un message IDMEF que le collecteur de Prelude lui aura envoyé.

Nous définissons comme fonction de callback **handle_alert,** qui est définie comme suit :

```
def handle_
alert(idmef) :
     source = idmef.Get("alert.source(0).node.address(0).address")
    if not sourceslist.has_key(source) :
        sourceitem = graph.newItem(source)
        sourceslist[source] = sourceitem
    target = idmef.Get("alert.target(0).node.address(0).address")
    if not targetslist.has_key(target) :
        targetitem = graph.newItem(target)
        targetslist[target] = targetitem
    alert  = idmef.Get("alert.classification.text")
    alert = alert.replace(' ','_')
```

```
        if not alertslist.has_key(alert) :
                alertitem = graph.newItem(alert)
                alertslist[alert] = alertitem
                 severity = idmef.Get("alert.assessment.impact.severity")
                if severity :
                        if severity == "high" :
                          graph.propertyAppend(alertslist[alert],"color","red")
                         graph.propertyAppend(alertslist[alert],"style","filled")
                        if severity == "medium" :
                           graph.propertyAppend(alertslist[alert],"color","orange")
                         graph.propertyAppend(alertslist[alert],"style","filled")
                        if severity == "low" :
                           graph.propertyAppend(alertslist[alert],"color","green")
                         graph.propertyAppend(alertslist[alert],"style","filled")
                        if severity == "info" :
                         graph.propertyAppend(alertslist[alert],"color","blue")
                         graph.propertyAppend(alertslist[alert],"style","filled")
          graph.newLink(sourceslist[source],alertslist[alert])
        graph.newLink(alertslist[alert],targetslist[target])
```

Il s'agit tout simplement de la fonction principale, qui gère la récupération de la source, destination ainsi que le nom de l'alerte pour utiliser les fonctions de GvGen pour produire le graphe en langage dot.
Nous allons utiliser un signal de terminaison pour produire le graphe. Par exemple un simple kill ou Ctrl-C permettra de l'afficher :

```
def sighandler(sig, frame) :
        graph.dot()
        sys.exit(0)
 signal.signal(signal.SIGTERM, sighandler)
signal.signal(signal.SIGINT, sighandler)
```

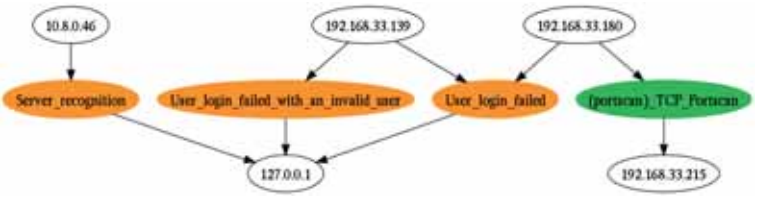
Maintenant, nous pouvons lancer notre programme :

```
$
 ./idmef-path-gvgen.py
```

Puis nous appuyons sur Ctrl+C, et nous voyons :

```
digraph G {
 compound=true ;
   node1 [label="192.168.33.180"] ;
   node2 [label="127.0.0.1"] ;
   node3 [color="orange",style="filled",label="User_login_failed"] ;
   node4 [label="192.168.33.215"] ;
   node5 [color="green",style="filled",label="(portscan)_TCP_Portscan"] ;
   node6 [label="10.8.0.46"] ;
   node7 [color="orange",style="filled",label="Server_recognition"] ;
   node8 [label="192.168.33.139"] ;
   node9 [color="orange",style="filled",label="User_login_failed_with_
an_invalid_user"] ;
node1->node3 ;
node1->node5 ;
node3->node2 ;
node5->node4 ;
node6->node7 ;
node7->node2 ;
node8->node9 ;
node8->node3 ;
node9->node2 ;
}
```

Ce qui nous donne, à l'aide d'un **dot -Tpng gvgen-2d.dot >** :



Le graphe se lit en regardant les noeuds en haut, qui sont les sources. Nous avons donc été attaqués par trois sources différentes. Au milieu, ce sont les attaques, avec un jeu de couleur en fonction de la gravité. Puis, en bas nous avons les cibles. Il y en a deux : 127.0.0.1 et 192.168.33.215.
192.168.33.215 est l'adresse IP réseau de la machine et 127.0.0.1 aussi, mais il s'agit de l'adresse IP locale. Si l'on voit cette adresse IP, c'est simplement parce que cela correspond à la cible vue par Prelude LML lors de l'analyse de log. Cette adresse peut bien sûr se configurer.
Maintenant, grâce au patch sur RTgraph3d, nous allons pouvoir visualiser en 3D ces mêmes évènements grâce à l'import dot.
Nous lançons donc la fenêtre OpenGL de rtgraph3d :

```
$
 ./rtgraph3d.py &
```

Puis son programme client :

```
$
 ./rtg_cli.py
```

Nous pouvons taper notre commande "dot", qui correspond à l'import dot de notre graphe :

```
RTG> dot gvgen-2d.png
```

Ce qui nous donne l'image suivante :

Conclusion

Le modèle de développement des logiciels libres permet à tous de pouvoir contribuer aux applications de manière très simple : par l'ajout de fonctionnalités, la documentation, les traductions etc. Nous tenons à remercier tous les acteurs de cette communauté qui nous ont permis d'obtenir des connaissances, qui nous servent depuis, quotidiennement, dans notre travail. La 3D offre une alternative intéressante aux représentations graphiques habituelles. Elle permet, par une disposition judicieuse des éléments, de présenter les informations différemment, de mieux pouvoir comprendre les relations entre chaque élément. Grâce aux modifications et améliorations apportées aux logiciels, la représentation graphique des alertes est maintenant accessible facilement. Ceci apporte des possibilités intéressantes, qui devront être explorées pour pouvoir représenter des informations pertinentes dans le but d'aider l'utilisateur à mieux comprendre les attaques détectées.

■ **Pierre Chifflier,** *Ingénieur-Chercheur - INL*
■ **Sébastien Tricaud,** *Ingénieur-Chercheur - INL*

ABONNEMENT PDF : 2,73 € le numéro www.programmez.com

+ Tutoriel sur
www.programmez.com/tutoriel.php

Capturer des Vidéos et des photos depuis une webcam

Après avoir vu, le mois dernier, comment acquérir des photos depuis un appareil numérique, nous étudions comment capturer des vidéos ou des photos depuis une webcam.

Le mois dernier, nous avons étudié l'API documentée dans la MSDN sous le nom de Picture Acquisition SDK pour réaliser des acquisitions d'images. Nous avons vu que cette API n'est apparue qu'avec Windows Vista et qu'elle est entièrement basée sur la technologie COM. Aujourd'hui, pour traiter des captures de vidéos, nous étudions une API beaucoup ancienne qui est documentée dans la MSDN sous le nom de Windows Multimedia SDK. Dans un souci de cohérence que l'on retrouve assez fréquemment chez Microsoft, cette API ne présente absolument aucun rapport avec COM, il s'agit d'une API asynchrone à base de messages comme le sont les API de gestions des fenêtres et destinée à collaborer avec elles.

1 Vue d'ensemble de l'API

La conception de l'API qui nous intéresse aujourd'hui impose une façon de travailler. Si le mois dernier rien ne nous empêchait de travailler avec une application purement "console" pour réaliser une acquisition, nous devons cette fois travailler dans une fenêtre au sens "Windows" du terme. Ce n'est pas que sous Windows une console ne soit pas avant tout une fenêtre et qu'il ne soit pas possible de *bidouiller.* Mais pour un bon fonctionnement garanti, il est nécessaire d'ouvrir une fenêtre de façon standard et qui sera la fenêtre principale, ou de niveau supérieur, de l'application. Ensuite, au moyen des API on crée une seconde fenêtre qui devra être fille de la précédente. C'est alors dans cette seconde fenêtre, et après connexion avec le driver de la webcam qu'apparaîtra le flux de données (les images). Et c'est à cette fenêtre que l'on postera des messages, par exemple pour définir les paramètres de capture. Il est encore important d'être bien conscient du fait que certains messages reçus par la fenêtre principale (WM_PALETTECHANGED par exemple) ne sont pas automatiquement transmis vers la fenêtre fille qui gère la capture. Le programmeur devra donc se charger de la propagation si besoin est. Pour illustrer le maniement de base de notre API, nous fournissons un exemple qui a été écrit en API Win 32 pure et dure. Par souci de clarté, votre serviteur s'est appliqué à concentrer tout le code "important" au début du source. Celui-ci a été écrit avec Visual Studio 2005 et essayé sous Visual Studio 2008, le tout sous Windows Vista. La compilation avec tout autre compilateur C++ propre sur lui ne doit pas poser de difficulté. La webcam utilisée est une LifeCam VX-3000 Microsoft. Le code est normalement totalement indépendant du périphérique et doit fonctionner avec n'importe quelle webcam, ou même n'importe quel périphérique de capture vidéo.

2 Démarrage de notre application de démonstration

Lorsque notre application démarre, elle ouvre, comme nous l'avons dit, une fenêtre de niveau supérieur. Au moment où cette fenêtre est sur le point d'apparaître sur l'écran, le système émet un message WM_SHOWWINDOW. Dans la procédure de fenêtre, nous utilisons ce message pour appeler notre fonction DoPreview qui initialise en cinq étapes tout ce qui est en rapport avec notre webcam. De ces 5 étapes, seules les deux premières sont absolument requises.

```
void DoPreview(HWND hwnd)
{
    LRESULT result;

    result = CreateCaptureWindow(hwnd);
    if(!result)
    {
        ::MessageBox(hwnd,
        L"Impossible de créer la fenêtre de capture",
            L"Programmez!", MB_OK);
        return;
    }

    result = MakeConnexion();
    if(!result)
    {
        ::MessageBox(hwnd, L"Impossible d'établir la connexion",
            L"Programmez!", MB_OK);
        return;
    }

    result = CreatePreview();
    if(!result)
    {
        ::MessageBox(hwnd,
            L"Impossible d'établir la prévisualisation",
            L"Programmez!", MB_OK);
        return;
    }

    GetInfos();
    InstallCallback(); // Attention à WM_DESTROY
}
```

L'avant-dernière étape, GetInfos() ne sert réellement dans cet exemple qu'à titre d'information. Son rôle est seulement d'écrire des données dans des fichiers texte qui seront créés dans le répertoire du projet si vous travaillez avec Visual Studio ou dans le répertoire de l'exécutable de l'application si vous travaillez avec un autre outil. Dans le premier fichier (driver_name.txt) vous trouverez les nom et numéro de version du pilote de votre webcam avec laquelle nous travaillons. Ces données en Unicode sont obtenues ainsi :

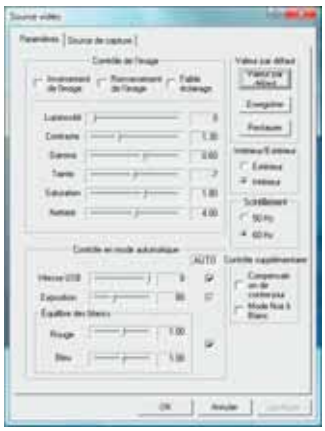
```
#include <fstream>
using namespace std;

void GetDriverName()
{
    WCHAR buffer[128];
    wofstream ofs("driver_name.txt", ios::out|ios::trunc);
    capDriverGetName(capture_window, buffer,
          sizeof(buffer)/sizeof(WCHAR));
    ofs << buffer << endl;
    capDriverGetVersion(capture_window, buffer, sizeof(buffer)/sizeof(WCHAR));
    ofs << buffer << endl;
}
```

Le second fichier (driver_caps.txt) contient les valeurs des membres de la structure CAPDRIVERCAPS, ce qui nous informe sur les capacités du driver de notre webcam, et notamment sur les boîtes de dialogue que nous avons la possibilité d'ouvrir. Enfin le troisième fichier, captureparms-default.txt contient les valeurs des membres de la structure CAPTUREPARMS, ce qui nous renseigne sur les paramètres de capture par défaut. Nous renvoyons le lecteur sur le site www.programmez.com pour le code générant les deux derniers fichiers. Les informations des deux premiers fichiers sont bien sûr des constantes. En revanche, il est possible d'agir sur le contenu de CAPTUREPARMS, soit via des boîtes de dialogue soit via des API pour configurer à notre convenance les paramètres de capture.

3 Les boîtes de dialogues de configuration

Elles sont au nombre théorique de 4 maximum :

Nom du dialogue	Rôle
VideoSource	Affiche les paramètres affectant la capture, tels que contraste ou saturation.
VideoFormat	Affiche les paramètres de format d'image: taille, profondeur, compression.
VideoDisplay	Affiche les paramètres relatifs à l'affichage sur l'écran de la capture, sans que cela n'affecte la capture elle-même.
VideoCompression	Affiche les paramètres de compression post-capture.



La boîte de dialogue VideoSource du driver de notre LifeCam VX-3000.

La boîte de dialogue VideoFormat du driver de notre LifeCam VX-3000.

Des quatre dialogues possibles, le driver de notre LifeCam VX-3000 n'en supporte que deux, illustrés ci-contre.

Quatre boîtes de dialogue ?... Si vous avez eu la curiosité d'examiner la structure CAPDRIVERCAPS, ou bien le contenu du fichier driver_caps.txt, vous avez nécessairement remarqué que la structure ne comporte que 3 membres, pour les trois premiers dialogues seulement du tableau ci-dessus. Le quatrième aurait il été oublié par les concepteurs de l'API ? Toujours est-il qu'il n'existe, sauf erreur de votre serviteur, aucun moyen de tester à l'avance si la boîte de dialogue de compression est supportée :(

4 Ouvrir une fenêtre de capture

Le terme ne doit pas tromper, une fenêtre de capture ne capture rien par elle-même, pas même les instantanés (photos). En revanche c'est le pivot sur lequel tout ce qui concerne les captures s'articule. Pour le programmeur c'est aussi et avant tout une fenêtre enfant qui s'ouvre, avec une fonction assez semblable à l'API bien connue CreateWindowEx :

```
HWND capture_window; // globale dans la démo

BOOL CreateCaptureWindow(HWND hwnd)
{
   capture_window = ::capCreateCaptureWindow (L"Ma Capture Window",
            WS_CHILD | WS_VISIBLE , 0, 0, 640, 480,
                  hwnd, 0);
   return capture_window != 0 ? TRUE : FALSE;
}
```

Nous ouvrons ici une fenêtre de taille 640*480, ce qui correspond à la plus grande résolution dont notre webcam est capable. Il est toutefois possible d'ouvrir une fenêtre plus ou moins grande, ainsi que nous le verrons un petit peu plus loin.

5 Se connecter à un pilote de webcam

Ceci se fait par l'intermédiaire de notre fenêtre. On envoie un message WM_CAP_DRIVER_CONNECT avec l'API dédiée à tout envoi synchrone de messages de fenêtres sous Windows: SendMessage. Voici le code :

```
const int id_webcam = 0; // globale dans la démo

BOOL MakeConnexion()
{
   return static_cast<BOOL>(::SendMessage(capture_window,
        WM_CAP_DRIVER_CONNECT,
        static_cast<WPARAM>(id_webcam),
        0L));
}
```

La variable id_webcam est un identificateur pour la webcam avec laquelle on travaille par l'intermédiaire de son driver. Windows peut gérer jusqu'à 10 périphériques de capture vidéo, numérotés de 0 à 9. Notre machine n'en ayant qu'un, son identificateur (0) est vite trouvé. La question est: comment procéder lorsqu'il y en a plusieurs ? La MSDN donne un exemple pour énumérer les périphériques, ou plutôt leurs drivers, que voici en substance :

```
#include <iostream>
using namespace std;

#include <windows.h>
#include <vfw.h>
#pragma comment(lib, "vfw32.lib")


int main(int argc, char** argv[])
{
   TCHAR name[80];
   TCHAR version[80];

   for (int i = 0; i < 10; i++)
```

```
    {
        if (capGetDriverDescription(
            i,
            name,
            sizeof(name),
            version,
        sizeof(version)))
        {
        wcout << "Peripherique de capture n " << i << endl;
            wcout << "Nom: " << name << endl;
            wcout << "Version: " << version << endl;
            wcout << endl;
        }
    }
    return 0;
}
```

Seulement les informations ne sont rien d'autre que celles obtenues dans le fichier driver_name.txt généré par notre programme de démonstration, à savoir :

```
Microsoft WDM Image Capture (Win32)
Version:  6.0.6000.16386
```

En aucun cas nous n'avons le "friendly name" qui est pour notre webcam

```
Microsoft LifeCam VX-3000
```

Les informations obtenues par le procédé indiqué par la MSDN ne sont finalement pas très utiles car elles ne permettent pas de soumettre un choix digne de ce nom à l'utilisateur humain, celui-ci connaissant a priori le nom de ses périphériques, mais pas celui des drivers. Comment obtenir ce "friendly name" ? Selon la documentation MSDN ces informations sont stockées dans la base de registre, ou même, sur de vieux Windows, dans le fichier system.ini. Même en supposant que nous travaillions avec un Windows récent, l'exploration de la base de registre prend pour cette question une allure de cauchemar. De l'humble avis de votre serviteur, la meilleure démarche est d'énumérer les périphériques avec les interfaces COM de DirectShow telles que l'interface ICreateDevEnum. Ce travail sort du cadre du présent article et est laissé à la sagacité du lecteur.

6 Obtenir une prévisualisation

Au stade où nous en sommes, notre webcam (ou autre périphérique de capture) doit donner signe de vie, sans doute en allumant sa led. Toutefois, dans notre fenêtre de capture, nous voyons, soit rien, soit une image fixe. Nous voudrions voir une image animée, c'est-à-dire le reflet de ce que pourrait capturer l'appareil. Ce reflet s'appelle prévisualisation. On l'installe théoriquement en envoyant des messages à la fenêtre de capture, de la même façon que nous l'avons fait plus haut. Toutefois, cette approche de travail est tellement fastidieuse que les concepteurs de l'API ont écrit un bon nombre de macros pour alléger le code. Voici le code qui démarre la prévisualisation:

```
BOOL CreatePreview()
{
    if(!capPreviewRate(capture_window, 66))
        return FALSE;
    // Eventuellement adapter la fenêtre
    // à la preview avec capPreviewScale
    capPreviewScale(capture_window, TRUE);
    return capPreview(capture_window, TRUE);
}
```

Prévisualisation de la capture dans une fenêtre.

Nous utilisons trois macros, capPreviewRate, capPreviewScale et capPreview. Oui, ce sont bien des macros en dépit de la convention qui voudrait que leur nom soit en majuscules. La première macro définit le nombre d'affichages par seconde. La seconde macro définit si oui ou non les images capturées doivent être adaptées à la taille de la fenêtre de capture lors de la prévisualisation. Si oui et que la taille de la fenêtre ne correspond pas au format défini pour la capture, les images sont étirées. Si non, alors soit les images ne rempliront pas la fenêtre de capture dans sa totalité, soit elles seront coupées. Enfin c*apPreview* démarre ou arrête la prévisualisation, selon le booléen qu'elle reçoit en argument.

7 Capturer une Video pendant 5 secondes, mauvaise approche

La marche à suivre pour capturer une vidéo est toute simple. On définit les paramètres de capture, on définit un fichier temporaire, éventuellement de taille pré-allouée, puis la capture terminée, on demande l'écriture des données du fichier temporaire vers un fichier cible de son choix. Cette approche fonctionne si on démarre et arrête la capture soi-même, par exemple, en cliquant sur un bouton de la fenêtre principale. En revanche, si on demande une capture sur une plage de temps bien définie et que l'on attende que cette plage de temps soit écoulée pour sauvegarder les données, l'expérience montre qu'il est à peu près impossible d'obtenir un résultat correct. Voici un exemple de code qui démarre une capture pour une durée de 5 secondes. Le code est correct, sauf la partie mise en commentaires à la fin, et qui correspond à la mauvaise approche que nous venons de décrire.

```
#define CAPTURE_FILE L"G:\\temp\\capture.avi"
#define TARGET_CAPTURE L"G:\\temp\\ma_capture.avi"

void DoCapture(HWND hwnd)
{
    if(capture_window == 0)
        return;
```

```
    ::DeleteFile(CAPTURE_FILE);
    CAPTUREPARMS cp;
    BOOL result
    capCaptureGetSetup(capture_window,
          &cp, sizeof(cp));
    // demander une capture limitée dans le temps
    cp.fLimitEnabled = TRUE ;
    // demander une capture pendant 5 secondes
    // compte tenu d'un "délai" d'initialisation
    cp.wTimeLimit = 5 + 2;
    // Laisser les autres applications tourner
    cp.fYield  = TRUE;
    cp.fCaptureAudio = FALSE; // <- INDISPENSABLE
    cp.AVStreamMaster = AVSTREAMMASTER_NONE;
    if(!capCaptureSetSetup(capture_window, &cp, sizeof(cp)))
    {
          ::MessageBox(hwnd,
                L"Impossible de définir les paramètres de capture",
                L"Programmez!", MB_OK);
                return;
    }
    // Pré-allocation d'un fichier de 40 Mo
    if(!capFileAlloc(capture_window, 1024*1024*40))
    {
          ::MessageBox(hwnd,
          L"Impossible de prédéfinir la taille du tampon de capture",
                L"Programmez!", MB_OK);
                return;
    }

    if(!capCaptureSequence(capture_window))
    {
          ::MessageBox(hwnd,
                L"Impossible de faire la capture",
                L"Programmez!", MB_OK);
    }
    // MAUVAISE approche
    //::Sleep(10000);
    //capFileSaveAs(capture_window, TARGET_CAPTURE);
    //::MessageBox(hwnd,
    //      L"Fini",
    //      L"Programmez!", MB_OK);
}
```

En attendant de résoudre totalement le problème, ce code apporte déjà son lot d'enseignements. Nous définissons ainsi le temps de capture dans la structure CAPTUREPARMS :

```
cp.wTimeLimit = 5 + 2;
```

Nous ajoutons une valeur empirico-pifométrique de deux secondes au temps de capture. En effet, l'expérience montre qu'il faut "un certain temps" d'initialisation pour que le driver lance effectivement la capture. Malheureusement, il ne semble pas possible de mieux s'informer sur ce "certain temps". Enfin, toujours dans le remplissage de CAPTUREPARMS il faut bien remarquer ceci :

```
cp.fCaptureAudio = FALSE;
```

En effet, une webcam peut faire aussi des captures audio, ce dont ne nous ne nous préoccupons pas aujourd'hui. Toutefois les deux captures, audio et vidéo, ne sont pas découplées par défaut, ce qui est logique puisqu'elles sont toutes deux destinées au même fichier .avi. Cependant sans une initialisation de la capture audio en bonne et due forme, la capture vidéo ne fonctionnerait pas du tout. Nous devons donc inhiber totalement la capture audio si celle-ci n'est pas explicitement initialisée.

8 Capturer une Vidéo pendant 5 secondes, bonne approche

Globalement le code donné plus haut était valable. La seule chose à ajouter est l'installation d'une fonction de rappel qui sera invoquée par le driver à plusieurs reprises pendant les phases de l'opération de capture. Dans notre exemple, cette installation est faite dans la fonction DoPreview et la désinstallation est faite dans l'événement WM_DESTROY de la fenêtre principale. Grâce à notre fonction de rappel, nous pouvons être avertis quand la capture démarre réellement et quand elle se termine. A ce moment nous pouvons écrire les données dans le fichier cible.

```
// La bonne approche pour une capture pendant un temps limité
// est l'utilisation d'une fonction de rappel comme celle ci-dessous
BOOL CALLBACK status_callback(HWND cw, int nID, LPCSTR lpsz )
{
   if(nID == IDS_CAP_BEGIN)
         Beep( 750, 300 );
   if(nID == IDS_CAP_END)
   {
         capFileSaveAs(capture_window, TARGET_CAPTURE);
         Beep( 750, 300 );
   }
   return FALSE;
}

void InstallCallback()
{
   capSetCallbackOnStatus(capture_window, status_callback);
}
```

9 Capturer un instantané (Photo)

Rien de plus simple. On veillera simplement à employer la macro capGrabFrameNoStop plutôt que capGrabFrame si l'on ne souhaite pas que la prévisualisation soit arrêtée par l'opération.

```
void DoPhoto(HWND hwnd)
{
   if(!capGrabFrameNoStop(capture_window))
   {
         ::MessageBox(hwnd,
               L"Impossible de faire la photo",
               L"Programmez!", MB_OK);
         return;
   }
   capFileSaveDIB(capture_window, TARGET_PHOTO);
   Beep(750, 300);
}
```

Frédéric Mazué
fmazue@programmez.com

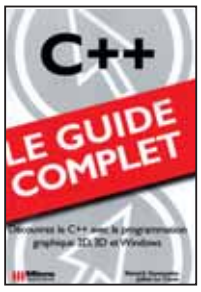
Tutoriels sur
www.programmez.com/tutoriel.php

Programmation concurrente et temps réel avec Java

- ***Difficulté : ********
- ***Editeur : Presses polytechniques et universitaires romandes***
- ***Auteur : Luigi Zaffalon***
- ***Prix : env. 64 €***

Tout le monde ou presque connaît Java. Mais c'est moins vrai dans le monde de l'embarqué, dans l'enfoui et surtout dans le monde temps réel, car Java a souffert d'une mauvaise image. Cet ouvrage, un des rares en français, se plonge dans le sujet. On débute par une longue introduction: programmation concurrente, temps réel. Car pour bien coder dans ces domaines, il faut maîtriser et parfaitement connaître les notions, le formalisme, les processus, etc. Après cela, on attaque le vif du sujet avec les threads en Java, les verrous et sémaphores, l'exclusion mutuelle, etc. Puis, le temps réel débute (ou plutôt on étend notre travail) par la notion de temps : horloge et minuterie. L'auteur parle naturellement d'un élément vital : l'ordonnancement et les threads temps réel, comment gérer la mémoire (un gros problème dans le temps réel et la programmation concurrente). On regrettera (un peu) l'absence d'un chapitre sur la sécurité. Vous l'aurez compris, cet ouvrage est à réserver à des développeurs Java voulant réellement aller très loin dans le langage. Et pour ceux qui veulent en savoir plus sur la concurrence et le temps réel, c'est le moment ! Un must incontournable.

C++ (le guide complet)



- ***Difficulté : ******
- ***Editeur : Micro Application***
- ***Auteur : collectif***
- ***Prix : 20 €***

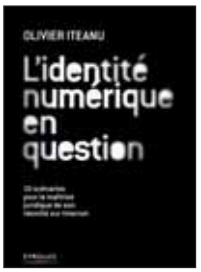
C++ vous tente ? Micro Application vous propose un guide complet sur ce langage orienté objet par excellence. Une bonne connaissance de la programmation permettra de mieux assimiler le C++ même s'il demeure un langage difficile sans une grande rigueur. L'environnement utilisé est Visual C++ 2008 Express (édition gratuite). Les auteurs y abordent les classes, les listes, les templates, la 2D et 3D, la plate-forme .Net et C++, la compilation. À cela s'ajoutent des trucs et astuces pour déboguer, gérer les exceptions, configurer le compilateur, etc. Se voulant ludique, l'ouvrage mise sur la 3D.

Sharepoint 2007

- ***Difficulté : *****
- ***Editeur : Eni***
- ***Auteur : Sandrine Schmitt***
- ***Prix : 39 €***

Vous ne connaissez pas SharePoint 2007 ou vous souhaitez démarrer en douceur sur cette plate-forme de Microsoft surpuissante ? Ce livre se propose de vous guider dans la création d'un site collaboratif. On y parle architecture, les briques pour bâtir le site, la sécurité et les droits d'accès et enfin le déploiement du site collaboratif. Pour aider le lecteur, l'ouvrage est très visuel, pour faciliter la compréhension des fonctions de Sharepoint. Une bonne introduction.

L'identité numérique en question



- ***Difficulté : *****
- ***Editeur : Eyrolles***
- ***Auteur : Olivier Iteanu***
- ***Prix : 19 €***

Qu'est-ce l'identité numérique ? Comment se protéger et protéger ses données personnelles ? Quel est le statut juridique du mot de passe ? L'approche juridique de l'auteur (avocat) est particulièrement intéressante dans la gestion des identités et pour dire qui est détenteur de l'identité. Voilà une heureuse initiative pour tordre le cou à quelques rumeurs et vérités qui n'en sont pas ! À lire sans délai.

Algèbre relationnelle

- ***Difficulté : ******
- ***Editeur : Eni***
- ***Auteur : Michèle Clouse***
- ***Prix : 27,14 €***

Vous vous demandez qu'est-ce que l'algèbre relationnelle ? Il s'agit de savoir comment on construit une base de données relationnelles à partir des besoins des utilisateurs, la définition des données, des tables, etc. L'algèbre relationnelle permet de voir les relations au sens mathématique du terme. Cela vous semble abstrait ? Pourtant cette approche peut permettre de réaliser des bases de données plus performantes.

Livre du mois

L'art du beau code

- ***Difficulté : ******
- ***Editeur : O'reilly***
- ***Auteur : collectif***
- ***Prix : env. 45 €***

Un autre " must have " que tout développeur devrait lire et relire tous les mois ! Plus que jamais, le code ne doit pas s'improviser. Les auteurs du livre abordent un sujet parfois tabou : comment structurer et avoir un beau code ? Les développeurs, plus de 30, dévoilent ici les bonnes pratiques, les usages, ce qu'il faut éviter, mais aussi les entorses à faire pour contourner un problème. Ce n'est pas un cours mais une succession de thématiques, de problématiques : les expressions régulières, les outils de type Subversion, les frameworks de tests, les tests, la génération de code, etc. Chaque chapitre aborde un thème et le développe, le dissèque, fournit des exemples, et surtout une explication approfondie, héritée d'une longue expérience de codage ! C'est beau et utile !

Calculateurs, calculs, calculabilité

- ***Difficulté : *****
- ***Auteur : Collectif***
- ***Editeur : Dunod***
- ***Prix : 25 €***

Que viennent faire les calculs et les calculateurs dans la programmation et les applications ? Cet ouvrage s'apparente à des travaux dirigés de niveau Licence – Master. Vous saurez comment bien utiliser les calculs, les mettre en œuvre et surtout comment les structurer. Une bonne piqûre de rappel !

SQL pour Oracle 3e édition

- ***Difficulté : ******
- ***Editeur : Eyrolles***
- ***Auteur : Christian Soutou***
- ***Prix : 29 €***

Bible incontournable de la programmation SQL dans l'environnement Oracle, on découvre le modèle PL /SQL et les fonctions avancées de ce dérivé de SQL propre à Oracle. Didactique, l'ouvrage permet un apprentissage en douceur. Il se destine à tous les développeurs C, C++, Java, PHP et XML. Couvre la toute dernière version : Oracle 11g.

Groupe LINAGORA, réussissons ensemble les grands projets du libre

LINAGORA RECRUTE
Rejoignez-nous sur job.linagora.com

OBM

(Open Business Management)

LA MEILLEURE SOLUTION DE MESSAGERIE ET DE TRAVAIL COLLABORATIF LIBRE

Agendas partagés, Messagerie, CRM, Annuaire LDAP,
synchronisation Outlook, Thunderbird et Smartphones/PDA

Venez contribuer sur

WWW.OBM.ORG

NOS PROCHAINES "MATINÉES POUR COMPRENDRE ..." SUR PARIS, TOULOUSE ET LYON :

**Libérez-vous des contraintes Open Source :
Choisissez votre propre cycle de vie logiciel**

- Le 15 mai sur Paris
- Le 22 mai sur Toulouse
- Le 29 mai sur Lyon

**Gestion de parc :
La puissance des outils libres pour la gestion industrielle de votre parc informatique**

- Le 12 juin sur Paris
- Le 19 juin sur Toulouse
- Le 26 juin sur Lyon

Séminaires gratuits - Plus d'informations sur
www.linagora.com

LINAGORA GROUPE
The obm company

© 2008 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.