PROGRAMMEZ !
Le magazine du développement
PROgrammez !
www.programmez.com
mensuel n°127 - février 2010

Microsoft 2.010
Tout est neuf !

Développement web : ASP.Net, Silverlight et les Tests 2010

Team Foundation Server 2010
La révolution du multitouch avec Surface

©Microsoft

Webmaster
- *Internet Explorer 8 : les bons réflexes de programmation*
- *Eclipse 4.0 : gérer les CSS*

Modélisation
Quels avantages pour le développeur ?

WinDev 15

SGBD
Ingres veut révolutionner les bases de données

Mobile
Créer des animations iPhone

Java
Surveiller la JVM

Flash / Flex
Les bases d'ActionScript 3

Web
PureMVC : développer une application GWT

M 04319 - 127 - 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®

555 NOUVEAUTÉS

WINDEV 15 est l'environnement de développement professionnel le plus efficace.

WINDEV 15 est totalement intégré (IDE, ALM), intégralement en **français** et réputé pour sa **richesse fonctionnelle**, sa **puissance** et sa **facilité** d'utilisation.

WINDEV 15 est livré complet: Maquettage, Schéma de données (UML,...), RAD, Patterns, Lien avec toutes les bases de données: Oracle, SQL Server, AS/400, Informix, DB2, MySQL, PostgreSQL..., Base de données Client/Serveur HyperFileSQL gratuite, Cluster, Générateur d'états PDF, Codes-barres, Accès natif SAP R/3, Lotus Notes, Outlook, Planning, Exigences, Audit, L5G, SNMP, Bluetooth, TAPI, OPC, FTP, HTTP, Socket, Twain, API, DLL, Webservices, XML, Domotique, Liaisons série et USB, Débogage à distance, Profiler, Refactoring, Génération JAVA, Multilangue, Gestionnaire de Versions, Retours utilisateur, Tests automatisés, Installeur 1-clic et push, etc, etc...

Les applications créées fonctionnent sous Windows 7, Vista, XP, 2000, NT, 2003, sous TSE et Citrix, Netbook et sont compatibles INTERNET et mobiles.

WINDEV 15 gère le Cycle complet de développement, pour des équipes de 1 à 100 développeurs. Le **SUPPORT TECHNIQUE** est gratuit*.

VERSION EXPRESS GRATUITE
Téléchargez-la !

VOTRE CODE EST MULTI-PLATEFORMES:
Windows, Internet et Mobile
Java, .Net, PHP, J2EE, XML, Ajax, Smartphone, Linux (SGBD), Client riche...

VOUS AUSSI, DÉVELOPPEZ 10 FOIS PLUS VITE AVEC WINDEV 15 .

DEMANDEZ LE DOSSIER GRATUIT
252 pages + DVD
+ Version Express
+ 112 Témoignages.

PC SOFT 25 ANNÉE Au service des développeurs

www.pcsoft.fr
Tél: **04.67.032.032** info@pcsoft.fr

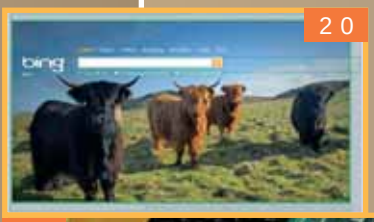
Fournisseur Officiel de la Préparation Olympique

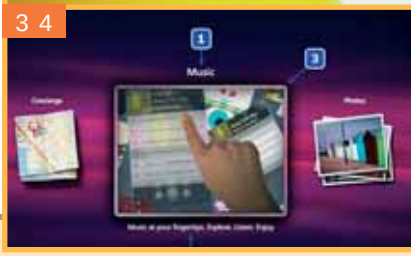
PROgrammez ! le magazine du développement

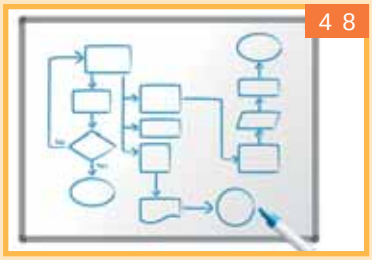
numéro 127 - février 2010

sommaire

vectorwise 13







Istockphoto.com/siumon

FusionCharts | à partir de € 131

InfoSoft Global

Diagrammes interactifs et animés pour les applications Web et les applications de bureau.

- Animez vos applications Web avec les diagrammes Flash animés
- Créez des diagrammes compatibles AJAX pouvant changer côté client sans requêtes serveur
- Exportez les diagrammes en tant qu'images/PDF et les données en CSV pour les rapports
- Créez des jauges, des diagrammes financiers, de Gantt, en entonnoir et plus de 550 mappages
- Utilisé par plus de 15 000 clients et quelques 250 000 utilisateurs dans 110 pays

FarPoint Spread for Windows Forms | à partir de € 659

GrapeCity

Feuille de calcul complète pour applications Windows Forms.

- Contrôle unique, 2 milliards de feuilles, avec chacune 2 milliards de lignes et 2 milliards de colonnes
- Renseignement automatique : anticipation de la frappe dans la cellule
- Nouveau - outil intégré de création de diagrammes avec 85 styles
- Nouveau - préserve les .XLS et restaure les fonctions non supportées

Resco MobileForms Toolkit | à partir de € 495

resco.net

Plus de 20 contrôles et bibliothèques Windows Mobile pour NET Compact Framework.

- Inclut image, list, tree, chart, detailView, grid, zip, notes, calendrier Outlook, CustomKeyboard pour les écrans tactiles et bien plus encore
- Environnement de développement unique et totalement intégré avec Visual Studio
- Inclut des thèmes de composants accessibles directement depuis le concepteur Visual Studio
- Nouveaux contrôles incluant Resco ScrollBar, Resco ProgressBar et Resco MaskedTextBox

Syncfusion Essential Studio Enterprise Edition | à partir de € 855

Syncfusion

Ajoutez des fonctionnalités d'interface utilisateur avancées à vos applications .NET.

- Inclut Syncfusion Essential Tools, Essential Grid, Essential Grouping, Essential XlsIO, Essential Diagram, Essential Chart et Essential Edit
- Fonctionnalités de grille WPF et performances améliorées, et prise en charge des thèmes
- Introduit le 1er générateur de rapports compatible WPF 100 % RDL
- Créez des tableaux de bord Business Intelligence ultraperformants

Siège social en Europe
ComponentSource
30 Greyfriars Road
Reading
Berkshire
RG1 1PE
Royaume-Uni

Siège social aux États-Unis
ComponentSource
650 Claremore Prof Way
Suite 100
Woodstock
GA 30188-5188
États-Unis

Siège social au Japon
ComponentSource
3F Kojimachi Square Bldg
3-3 Kojimachi Chiyoda-ku
Tokyo
Japon
102-0083

Numero vert:
0800 90 92 62
www.componentsource.com

Nous acceptons les bons de commande. Contactez-nous pour demander un compte de crédit.

VISA

éditoll

La modélisation (ne) passera (pas) par moi

À Programmez !, cela fait des années que nous nous penchons sur la modélisation, et sur les tendances MD, UML. Ce mois-ci nous abordons la modélisation et le Model Driven dans les outils et pour le développeur. Un constat s'impose. Le développeur reste souvent peu sensible à ce domaine, même si des événements tels que les MD Days connaissent un beau succès.

Est-ce du désintérêt ? Un manque d'évangélisation ? L'offre est désormais mature et les technologies le sont aussi, bien qu'il existe toujours des points délicats sur des projets très complexes ou dans le round trip. Mais il faut aussi avouer que les éditeurs MDA, il y a quelques années, avaient sans doute trop promis : zéro code à taper, le tout automatique, le modèle-roi... ! Las, il faut avouer que ces promesses ont pu faire douter de la réelle efficacité du MD dans le développement. Alors que la modélisation UML fonctionnait plutôt bien. Aujourd'hui, les éditeurs sont sans doute plus prudents sur les promesses. Oui, cela facilite le travail et permet d'automatiser la création des interfaces et d'une partie du code et de l'application, mais cela ne se fait pas sans efforts, sans changements assez radicaux des méthodes de travail. Il n'y a pas de miracle : il ne suffit pas d'un clic de souris pour générer 100 % de l'application finale.

La peur de la modélisation, du MD vient-elle aussi d'une réticence des développeurs face aux changements et surtout à une éventuelle perte de travail au profit des modèles ? Ici comme sur d'autres technologies, il faut prendre son temps : montrer, expliquer, rassurer. Le développeur aime le changement... mais pas trop non plus. Et en France, il ne faut pas oublier que son métier n'est pas toujours valorisé, soutenu ou tout simplement reconnu à sa juste valeur.

Le modèle permet de se concentrer sur le cœur du code

Nous pensons sincèrement que la modélisation et le modèle sont de précieux alliés. Car le développeur peut se décharger d'une partie du code récurrent et ingrat comme pour la gestion de l'interface ou tout simplement créer le squelette de l'application et son code pour ensuite se concentrer sur le cœur du code, la partie intéressante et « noble ». Là où le développeur apporte son savoir-faire.

Mais on peut aussi se demander s'il ne manquait pas un « booster » au marché. Bien sûr, il existe une profusion d'outils pour le développeur, open source ou payants, mais finalement, pas de véritable 'killer application'. Le prochain Visual Studio 2010, intégrant l'UML, va sans doute sensibiliser le développeur. L'avenir nous le dira. Finalement, c'est comme un légume. Tant qu'on ne l'a pas goûté, on ne peut pas l'aimer ou le détester...

François Tonic

Le magazine du développement
PROgrammez!
www.programmez.com

Rédaction : redaction@programmez.com
Directeur de la Rédaction : Jean Kaminsky.
Rédacteur en Chef : François Tonic - ftonic@programmez.com. **Ont collaboré à ce numéro :** F. Mazué. **Experts :** F. Jehl, V. Bellet, L. Baumann, G. Rouchon, V. Labatut, R. Seltrecht, P-C. Peullemeulle, J-L. Corneliau, P. Desfray, D. Cohen-Zardi, H. Licari, S.Saurel, A. Quinault, O. Martin, A. Zerr, P. Leclercq.
Illustration couverture : ©Microsoft
Publicité : Régie publicitaire, K-Now sarl. Pour la publicité uniquement : Tél. : 01 41 77 16 03 - diff@programmez.com. Editeur : Go-02 sarl, 21 rue de Fécamp 75012 Paris - diff@programmez.com. Dépôt légal : à parution - Commission paritaire : 0712K78366 ISSN : 1627-0908. Imprimeur : ETC - 76198 Yvetot. **Directeur de la publication :** J-C Vaudecrane
Ce numéro comporte sur une partie de tirage
- 1 encart Component source
- 1 encart Solutions-Linux

Abonnement : Programmez 22, rue René Boulanger, 75472 Paris Cedex 10 - Tél. : 01 55 56 70 55 mail : abonnements.programmez@groupe-gli.com
Fax : 01 40 03 97 79 - 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 : 49 € - Etudiant : 39 € - CEE et Suisse : 55,82 € - Algérie, Maroc, Tunisie : 59,89 € - Canada : 68,36 € - Tom : 83,65 € Dom : 66,82 € - Autres pays : nous consulter. **PDF :** 30 € (Monde Entier) souscription exclusivement sur www.programmez.com

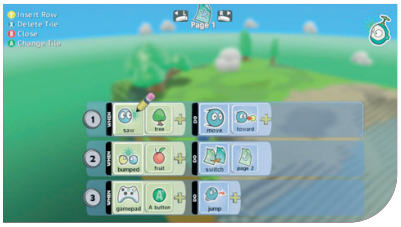
L'INFO PERMANENTE
WWW.PROGRAMMEZ.COM

PROCHAIN NUMÉRO

N°128 Mars 2010, parution 27 février

- ✔ **Dossier** Linux-Open Source
  Spécial Linux Solutions
- ✔ **Tout savoir sur** Silverlight 4.0
- ✔ **Les femmes dans l'IT :**
  Développer se conjugue aussi au féminin !

■ **Samba 4** est en cours de développement. Mi-janvier, une alpha 11 était disponible. Le projet est ambitieux : support de Active Directory, code source largement renouvelé, langage de script embarqué, modèle de process flexible, etc. http://www.samba.org/



■ Microsoft Research dévoile **Kodu Game Lab**, un environnement de développement doté d'un langage simple et accessible à tout le monde avec une grande facilité de création d'interface. Le projet était initialement disponible pour XBox 360. Site : http://research.microsoft.com/en-us/projects/kodu/

■ **NetBeans** ne s'arrête jamais. Après la sortie de la v6.8 en décembre dernier, la 6.9 est en développement, pour une sortie prévue courant mars. Cette version vise à améliorer les performances générales de l'environnement notamment sur la réactivité de l'interface. JavaFX bénéficiera aussi de nouvelles améliorations avec une meilleure stabilité, et de nouveaux composants. Et Spring 3 fera son apparition.

■ **VMware** continue de se renforcer sur le Datacenter et la virtualisation en rachetant Zimbra à Yahoo. Zimbra est connu pour son alternative open source à Exchange. Mais pourquoi racheter du logiciel ? VMware veut renforcer son offre logicielle en ligne (Saas) face à une concurrence féroce. Un nouveau virage stratégique pour VMware ?

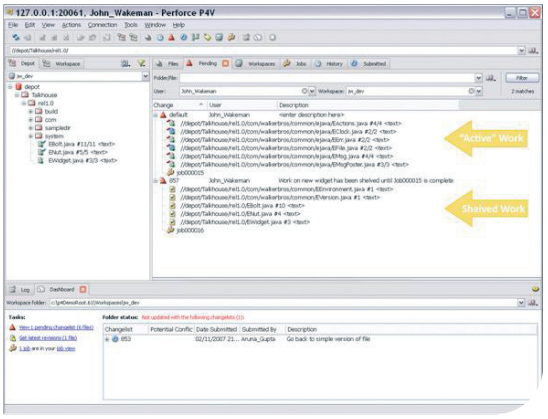
Futur
Firefox se prépare au multicore

L'un des problèmes des logiciels actuels est leur mauvaise adaptation aux processeurs multicœurs. Mozilla a décidé de prendre le problème au sérieux et travaille depuis plusieurs mois à l'adaptation du code et de l'architecture de Firefox au parallélisme et aux processeurs à cœurs multiples. Le projet s'appelle Electrolysis. Il sera intégré dans une future version, et vise trois objectifs : améliorer la réactivité de l'interface, améliorer la stabilité générale et enfin améliorer les performances, notamment sur les machines multicore. Autre objectif espéré : une meilleure sécurité en ayant une exécution de process protégé ou en mode isolé.

Côté process, il s'agit d'avoir une architecture de plugin multi-process. Le plugin est toujours une source d'instabilité. Surtout quand plusieurs plugins fonctionnent sur un même process, les risques sont d'autant plus grands. Il s'agit donc de séparer l'exécution des plugins sur plusieurs process, mais cela nécessite l'emploi d'appels entre les process et les plugins, en s'appuyant sur IPDL (Inter process communication protocol définition langage). C'est un langage spécifique Mozilla en C ++ pour passer des messages entre des process et des threads. Mais cela va nécessiter de longs tests avec les plugins et de savoir si les plugins auront besoin ou non d'adaptation… L'impact du multi-process concerne aussi l'interface utilisateur et les onglets.

Pour en savoir plus :
https://wiki.mozilla.org/Content_Processes

Gestion
Perforce fait évoluer la gestion logicielle



Perforce Software a dévoilé courant janvier la version 2009.2 de sa solution de Gestion de la Configuration Logicielle. Les nouveautés de cette version sont présentées comme permettant d'améliorer la productivité avec l'introduction du Shelving, la réplication des métadonnées en temps réel et de nouvelles fonctionnalités pour travailler en mode déconnecté. Avec la nouvelle fonctionnalité de Shelving les développeurs peuvent mettre les fichiers modifiés en cache sur le serveur Perforce sans avoir à les enregistrer au préalable en tant que changement versionné. Le Shelving améliore le travail en équipe et donne aux développeurs plus de flexibilité dans la gestion de projets multiples. Par exemple le Shelving Perforce permet aux utilisateurs :

• De transmettre les changements à mettre en œuvre à leurs managers dans le cadre de la revue du code ou des processus d'approbation,

• De partager le travail en cours avec d'autres membres de l'équipe ou d'autres stations de travail,

• De faire une modification puis de la tester dans un environnement de build distribué,

• Et de mettre de côté une tâche en cours si une tâche de priorité plus élevée leur est confiée.

Le Shelving est supporté par le serveur Perforce 2009.2, et par les interfaces graphiques et de commandes en ligne. Perforce 2009.2 est d'ores et déjà disponible. Les développeurs intéressés peuvent le télécharger à partir du site Perforce et l'évaluer.

www.perforce.com

agenda \\

FEVRIER

• Du 8 au 10 février, Paris 17, Palais des Congrès, **TechDays 2010.** Le rendez-vous incontournable des développeurs, décideurs et professionnels de l'informatique. Trois jours pour se former sur toutes les nouveautés Microsoft et découvrir les tendances du marché. http://www.microsoft.com/france/mstechDays/

• Le 9 février, Paris, 32 rue Monceau 75508. Sun Microsystems organise son **Roadshow Java 2010** http://fr.sun.com/sunnews/events/2010/jan/java_roadshow/?cid=e10130fr

• 15 - 17 février Paris : **Symfony Live 2010**, conférence technique sur le framework Symfony. Cloud, optimisation, sécurité, Symfony 2… Conférence payante. http://www.symfony-live.com

MARS

Du 16 mars 2010 au 18 mars 2010, Paris Expo, Porte de Versailles. **Solutions Linux/Open Source 2010.** Organisé par Tarsus France, ce salon demeure le carrefour d'échanges incontournables des acteurs du logiciel libre. http://www.solutionslinux.fr

ETRANGER

Du 06 au 7 février 2010, Université Libre de Bruxelles, **FOSDEM** 10e édition. La conférence annuelle libre et non-commerciale pour développeurs de logiciels libres et « open source », organisée par et pour la communauté. www.fosdem.org

egilia®

LEARNING

LE SPÉCIALISTE DE LA
FORMATION CERTIFIANTE
EN INFORMATIQUE
ET MANAGEMENT

Faire de vos succès
notre réussite

www.egilia.com

CONTACTEZ NOS CONSEILLERS FORMATION

N° National 0 800 800 900

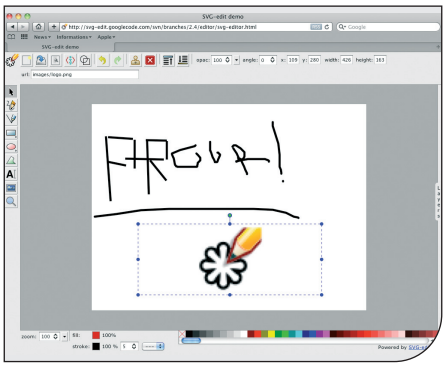
APPEL GRATUIT DEPUIS UN POSTE FIXE

ANVERS . LIEGE . PARIS . LYON . LILLE . AIX-EN-PROVENCE .
STRASBOURG . RENNES . BRUXELLES
TOULOUSE . BORDEAUX . GENEVE . LAUSANNE . ZURICH .

■ **PoshBoard 3.0** en démonstration aux TechDays 2010. Nelite présentera durant l'événement Microsoft la v3 de sa technologie sur écran tactile. Il combine Silverlight et PowerShell. Projet open source, l'objectif est de disposer d'un portail IT orienté administration système (reporting, gestion d'infrastructure). On peut créer rapidement des tableaux de bord d'infrastructure sans besoin de coder, on passe par du scripting. Parmi les nouveautés : le multitouch Silverlight 3, préconfiguration des scripts, différents rendus. Site : www.poshboard.com

■ Avec le Serveur Cloud Dynamique, **1 & 1** lance un modèle de serveur inédit. L'hébergeur est, en effet le premier en France à permettre l'ajustement séparé de chacun des paramètres du serveur. La mémoire vive, le nombre de cœurs et l'espace disque peuvent être sélectionnés individuellement et sont ensuite modifiables à tout moment pour répondre à l'évolution des besoins de l'utilisateur.

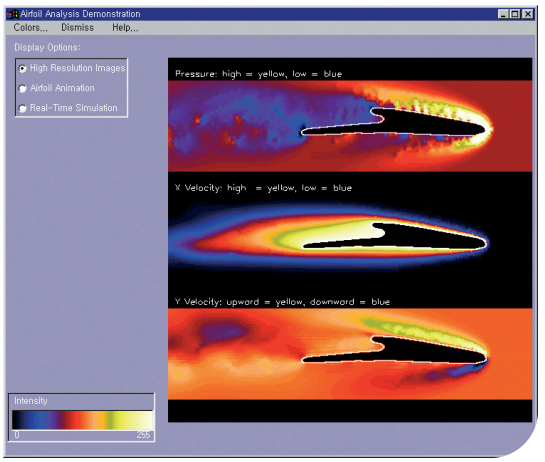
■ Microsoft a annoncé la disponibilité commerciale de la plate-forme Windows Azure depuis début janvier 2010. Cela inclut : **Windows Azure**, SQL Azure, et/ou AppFabric (non finalisé). Sur ce dernier, la facturation débutera en avril. Pour les utilisateurs de la préversion d'Azure, le mois de janvier était gratuit pour une mise à jour.



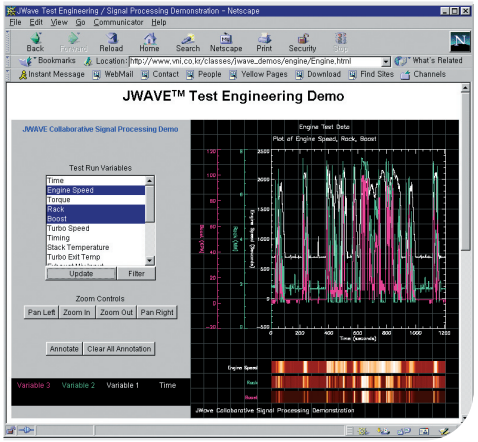
■ **SVG-edit** arrive en version 2.4. Cette version introduit les calques, le zoom, une gestion des images, le multilingue amélioré. De nombreux perfectionnements ont été apportés à l'éditeur depuis la v2.3. SVG-edit est un éditeur SVG pour éditer et créer des images SVG.
Site : http://code.google.com/p/svg-edit/

Rachat
Rogue Wave étoffe son offre

Assez discret depuis quelques mois, Rogue Wave se prépare à une année 2010 pleine d'annonces et de nouveaux produits. Cela passe tout d'abord par le rachat de la société Visual Numerics. Un éditeur spécialisé dans les solutions d'analyses numériques et d'analyses visuelles de données.



Constitués autour du progiciel PV-WAVE optimisé pour le traitement de données, ces outils permettent rapidement d'accéder aux données, de les analyser, de les visualiser et de les interpréter. Ils sont particulièrement performants pour manipuler de gros volumes de données, que ce soit en environnement local ou en environnement distribué. Et comme aucun autre outil sur le marché, PV-WAVE Advantage inclut des fonctionnalités d'analyse sophistiquée basées sur le standard IMSL. Les librairies IMSL sont disponibles pour C/C ++, Java, Fortran et Python, et pour la majorité des environnements informatiques actuels, les Librairies Numériques IMSL constituent depuis de nombreuses années la solution standard pour le calcul mathématique et statistique et pour la représentation de données.



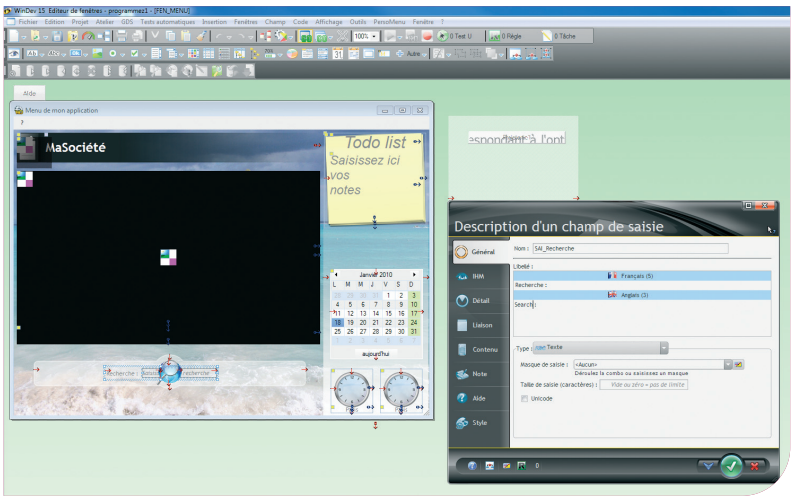
Les algorithmes IMSL, reconnus pour leur robustesse, leur précision et leurs performances, sont directement intégrables dans tous types d'applications : de l'application PC monoposte la plus simple au code de calcul massivement parallèle pour super calculateur (HPC) le plus complexe, en passant par l'application web distribuée.
Les Librairies Numériques IMSL permettent aux développeurs de se reposer sur des technologies à la pointe de l'analyse numérique tout en leur simplifiant le travail. Un environnement de développement, PyIMLS Studio, est aussi disponible pour faciliter le prototypage.

Une édition C#

Ces librairies permettent aussi d'améliorer et de faciliter le parallélisme des applications. Et une version C# était prévue mais là, l'éditeur a rencontré un problème : l'implémentation d'OpenMP sur. Net. Pour avoir les mêmes capacités que sur les autres langages, IMSL s'appuie donc sur les dernières briques parallèles de Visual Studio 2010 et de. Net 4, à savoir : Task Parallel Library, PLinq, et les outils parallèles inclus dans VS 2010 Professional. Visual Numerics a commencé par paralléliser ses bibliothèques C# fin 2008, lorsque les développeurs obtinrent les versions bêta de Microsoft. NET Framework 4 et Visual Studio 2010. Il a alors commencé à investiguer sur la manière d'appliquer les nouvelles fonctionnalités de programmation parallèle fournie par ces versions. « *Nous constatons un grand gain de performances après parallélisation de nos bibliothèques IMSL C# – un effort que le support de la programmation parallèle fourni dans Visual Studio 2010 et. NET Framework a rendu plus facile* », explique Mike Pulverenti, Responsable Technique Programme chez Visual Numerics. Si ces librairies facilitent le développement parallèle et son adoption, le développeur doit tout de même posséder une bonne maîtrise technique.
Au-delà de cette nouvelle stratégie autour du parallélisme, Rogue Wave veut continuer à s'étendre et à compléter sa panoplie technique en rachetant de nouvelles sociétés dans les mois à venir.

WinDev 15 : petit tour du propriétaire

La 15e version de WinDev est disponible depuis fin décembre 2009. Nous allons revenir sur quelques-unes des fonctions phares de l'atelier intégré de développement de PC SOFT ainsi que sur la partie web.



Notre machine de test : MacPro 8 coeurs, 16 Go de Ram, Windows 7 32-bit (partition Bootcamp), deux écrans.

La création des projets WinDev est simplifiée par l'utilisation d'un assistant qui permet de définir les paramètres de base du projet : documents annexes liés, le type de génération (exe, java, .Net, pattern Rad, service, web service, exe. net), on peut aussi définir le déploiement (nb de postes par exemple) et si on privilégie la vitesse à la sécurité… Ce qui est aussi intéressant est de pouvoir créer sa charte de programmation, que l'on respectera durant le projet. Pour chaque projet, on peut aussi choisir son pattern RAD, autrement dit, la possibilité d'avoir telle fonction, de privilégier telle approche projet ou de développement. Une fois l'ensemble des assistants exécutés, l'éditeur génère les squelettes ainsi que les avertissements correspondants comme par exemple dans le cas d'une application multilingue qui ne serait pas encore renseignée sur les traductions.

Un projet très contrôlé

WinDev propose de nombreuses options pour auditer et contrôler à la volée et dynamiquement son projet : fichiers, code, intégrité, occupation mémoire. À cela s'ajoute l'audit statique que le développeur peut demander. Les fonctions de tests et notamment de tests automatiques sont très intéressantes pour avoir une "vue validation" de son projet : les éléments non testés, les scénarios utilisés, etc.

Comme dans les précédentes versions, le développeur accède très rapidement à l'ensemble des propriétés d'un composant graphique. Le but est de limiter le code à écrire et quand on doit le faire, WinDev le limite au strict minimum… Autre élément bien connu de l'environnement, le graphe du projet. Graphiquement, on visualise tout de suite les fichiers, l'enchaînement des différentes parties du projet.

Et pour aller plus loin, en quelques clics on peut créer le modèle UML du projet (Rétro Engineering). Et la v15 introduit aussi une visualisation de l'occupation permettant de connaître le contenu et la taille des fichiers exe, des bibliothèques, etc.

Autre raffinement bienvenu : la possibilité de nettoyer son projet de tout code mort, ou de tout élément non utilisé. Une autre amélioration concerne le tableau de bord qui gagne en visibilité et en ergonomie. La compatibilité avec Windows 7 n'a pas été oubliée. WinDev 15 prend en compte la nouvelle barre des tâches Windows et le « dock ».

WebDev : vers le cloud computing

L'édition web de WinDev partage certaines des nouveautés de WinDev 15. On peut noter une orientation Saas. Il est ainsi rapide de publier en ligne son application et de gérer les utilisateurs, abonnements, accédant aux services en ligne. Cela permet de passer en douceur au cloud computing. Le rôle de l'administrateur est ici très important car c'est lui qui va contrôler les accès, les utilisateurs. Mais on peut aussi autoriser ce processus via la disponibilité d'API. Autre gain intéressant de WebDev, la possibilité de déployer rapidement ses sites chez des hébergeurs (OVH, Free), grâce à des assistants facilitant votre travail. WebDev se complète aussi d'un nouveau module de statistiques : tableaux de bord, réseaux en recherche, fréquentation, parcours des visiteurs, vue géographique… Toutes les fonctions indispensables à un responsable de sites pour les contrôler. Côté création de site, on retrouve les mêmes assistants et réflexes que sous WinDev. Pour ceux qui utilisent un outil de type Dreamweaver, ils trouveront en WebDev un environnement très complet et d'un accès plus aisé. On bénéficie aussi de tous les formats riches actuels (Flash, Silverlight), mais on peut aussi faire des animations d'interface sans avoir recours à Flash.

Pour en savoir plus : www.pcsoft.fr

■ J.V.

Tarification

Jusqu'au 31 mars prochain, un tarif « spécial mise à jour » est disponible. Par exemple, pour une mise à jour v14 vers 15, cela coûtera 499 euros (au lieu de 590). Pour un achat de toute la gamme, la mise à jour reviendra à 1 599 euros (depuis une v14, au lieu de 1 699 euros).

Google dévoile Android 2.1

Google continue à faire évoluer son système mobile : Android. Il arrive désormais en version 2.1. Cette mise à jour, présentée comme mineure, est déployable sur tous les terminaux Android. Cette v2.1 fixe un certain nombre de bugs connus mais surtout modifie quelques API, sans apporter de nouvelles fonctions significatives. Sur la partie API, notons dans la partie Live Wallpapers l'apparition d'un nouveau package : android.service.wallpaper, d'une nouvelle classe wallpaperInfo. Sur la partie téléphonie, une meilleure gestion de l'information du signal réseau (SignalStrenght). La partie WebKit subit aussi quelques ajouts comme sur la géolocalisation ou encore la manipulation sur les données stockées. A noter aussi une mise à jour de l'USB Driver par Windows qui arrive en release 3. Il doit faciliter les tests de ses applications depuis Windows. Ce pilote n'est pas utile en Mac OS X et Linux. En décembre dernier, ce fut au SDK Tools d'arriver en release 4 comprenant des corrections de bugs et quelques nouvelles fonctions telles que la taille sans limite d'une partition système (partie émulateur).

Site : http://developer.android.com/sdk/android-2.1.html

■ Microsoft dévoile **Visual Studio 2010 Team Foundation Server Requirements Management.** Cet outil, disponible sur codeplex, permet de gérer les exigences. Particulièrement utile pour la gestion des projets et la définition des spécifications. Site : http://vstfs2010rm.codeplex.com/wikipage

■ **Google** annonce son intention de migrer vers le système de fichier ext4 en lieu et place de ext2. Les nouvelles contraintes de disponibilité et performances obligent l'éditeur à changer de système de fichier. Cela concerne l'ensemble de l'infrastructure Google.

■ Rififi dans le monde des navigateurs web ? Des organismes français et allemands mettent en garde contre l'usage d'Internet Explorer, tout en demandant à Microsoft de corriger les récentes failles... Du côté français, le **CERTA** a mis en évidence ces failles et concernant IE6, 7 et 8 il met en garde contre : *« Une vulnérabilité due à une référence à un pointeur non valide permettant à une personne malintentionnée d'exécuter du code arbitraire à distance. Des exploitations limitées de cette vulnérabilité ont déjà été constatées ».*

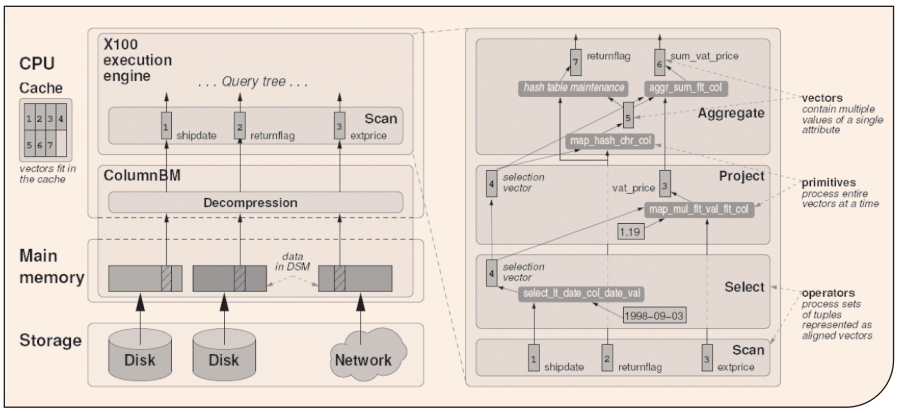
■ **Red Hat** a annoncé la sortie de Cygwin 1.7.1. Notons l'abandon de Windows 95, 98 et Me, le support de Windows 7 et Windows Server 2008. L'outil propose aussi une meilleure conformité à Posix et l'apparition de IPv6.

■ Les **hypers rootkits** sont des rootkits s'attaquant directement aux hyperviseurs. Actuellement aucune offre du marché ne semble viser cette menace. Et sa mesure reste difficile car peu d'études portent sur le sujet. Cependant l'hyper rootkit peut se révéler très dangereux car s'il pénètre dans l'hyperviseur il peut alors se propager à l'ensemble des machines virtuelles qu'il supporte !

Ingres mise sur VectorWise

vectorwise

Le modèle actuel des moteurs de stockage et de base de données est arrivé à un tournant. Limite technique, voire impossibilité de faire évoluer sainement ses moteurs, Ingres veut casser le modèle pour en proposer un nouveau. C'est tout l'enjeu du projet VectorWise.



VectorWise utilise deux fondamentaux : la vectorisation et les colonnes orientées pour le traitement des données. Ces deux techniques doivent permettre un traitement de données plus rapide pour un moindre coût (en ressources, puissance et finances). Cette approche est particulièrement intéressante pour le stockage et l'exécution des requêtes. Pour pouvoir réaliser un tel changement, Ingres a revu en profondeur l'architecture du moteur de stockage en adoptant une nouvelle architecture de traitement des requêtes afin de tirer parti aussi bien des grandes quantités de mémoires vives que des processeurs de dernières générations. Cela signifie en particulier que Ingres veut utiliser le potentiel du multithread, des caches L2 et L3, du SSE, etc. Notons tout de même que le cœur du moteur de ce projet a été développé par le CWI d'Amsterdam, centre de recherche en mathématique et informatique hollandais bien connu.

Mais quelle est l'efficacité de VectorWise ? Ingres annonce une performance en nette hausse grâce à la combinaison du traitement de requête et surtout de l'architecture de stockage dépassant le stockage par colonne actuellement utilisé. Cette nouvelle approche favorise les formats par colonne et par rangée. Tout en incluant une compression automatique et une clusterisation des tables, avec, petit plus, une gestion disque améliorée pour éviter des pertes en Entrée/Sortie pour les accès concurrents des requêtes. Et au final, cette approche doit permettre une bien meilleure utilisation des données et des bases dans un contexte massivement parallèle.

Quelques tests internes permettent de montrer une belle montée en performance entre Ingres 9 et VectorWise (sur du Xeon 5500). Sur la base d'une requête à 4 sommes, un comptage, deux multiplications, deux additions et soustractions sur 6 millions d'éléments, VectorWise exécute les opérations en 0,2 seconde contre 16,5 secondes pour Ingres 9. Seul C ++ fait mieux avec 0,04 seconde.

Une architecture renouvelée

Pour comprendre VectorWise, il faut connaître l'architecture interne et tout particulièrement le moteur d'exécution X100 et le gestionnaire buffer ColumnBM. X100 mappe les ressources matérielles à son moteur d'exécution. L'un des éléments les plus intéressants est l'exécution de la vectorisation. Au lieu d'un simple enregistrement de données, les vecteurs de données sont passés dans le pipeline. Autre fonction intéressante, le traitement dans le cache. Cela signifie que toute l'exécution se réalise dans le cache de la CPU, qui est alors utilisé en tant que mémoire principale comme un buffer pour les E/S et les larges structures de données. Cette technique procure des hautes performances, à condition d'avoir une machine récente et une architecture optimisée et de qualité. Et d'ailleurs le moteur VectorWise est taillé par une exécution en cache.

Les vecteurs sont au cœur du moteur d'exécution. Par vecteur, on entend des requêtes traitées par la transmission de plusieurs « tuples » à un moment donné entre deux opérateurs relationnels. Un tuple est une ligne d'un tableau. Plus généralement, un tuple est une suite de données (une ligne, un enregistrement). Et VectorWise permet de vectoriser les opérateurs relationnels.

Mais au-delà de ces aspects, Vector Wise se préoccupe aussi de la manière de stocker des données sur un disque. C'est pour cela qu'il combine le stockage disque vertical et horizontal. Or, pour optimiser l'exécution des recherches, le travail par colonne (verticalité du stockage, des données) ne suffit pas, notamment dans les indexations de tables ou les techniques de clustering. L'ajout de l'horizontalité limite les E/S et les accès disques. Une autre méthode pour accélérer les traitements est d'utiliser des disques SSD (disque à mémoire flash). Ces disques sont plus performants qu'un disque dur mécanique.

Une approche d'avenir ?

Avec VectorWise, Ingres ouvre une perspective intéressante. Reste à l'implémenter dans les SGBD et à le faire adopter par les utilisateurs, développeurs, DBA. Le marché est-il prêt à une modification profonde des moteurs d'exécution et de stockage ? Pas sûr.

■ François **Tonic**

Optimisez les performances de vos bases de données grâce aux différents outils Altova®

ALTOVA® missionkit®

Altova MissionKit® est une suite intégrée d'outils de bases de données, de XML et d'intégration de donnéesà excellente compatibilité avec toutes les principales bases de données relationnelles.

Nouveautés dans la version 2010:

- Comparaison de schémas de bases de données
- Conversion de structures entre différents types de bases de données
- Nombreuses améliorations dans la transformation et le rendu de données DB
- Et bien plus encore...

Tous les outils de bases de données compris dans le MissionKit prennent en charge les formats suivants :

- Microsoft® SQL Server®
- Oracle®
- IBM DB2®
- Sybase®
- MySQL®
- PostgreSQL
- Microsoft Access™

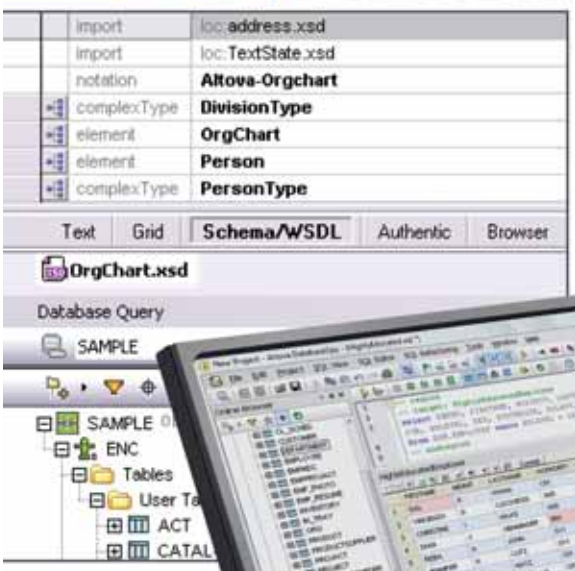
Altova MissionKit comprend plusieurs outils de travail sur bases de données:

DatabaseSpy® – éditeur SQL, outil de requête sur base de données, de conception et de comparaison de bases de donnée

MapForce® – outil graphique de mappage, de transformation et de conversion de données

XMLSpy® – leader parmi les éditeurs XML avec forte intégration de bases de données

StyleVision® – outil de création de feuilles de styles graphiques et de formulaires électroniques pour publier des données DB



Téléchargement gratuit!

Essayez avant d'acheter avec une version d'essai gratuite et entièrement fonctionelle de 30 jours disponible sur **www.altova.com**.

1&1 Serveur Cloud Dynamique

SERVE

Ajustez les performances en fonction de vos besoins

1&1 SERVEUR CLOUD DYNAMIQUE

Le prix du serveur varie en fonction de la configuration choisie, celle-ci étant modifiable à volonté pendant toute la durée de votre contrat.

Constituez votre propre Serveur Cloud Dynamique en optant, par exemple, pour une configuration de base :

- **1 cœur AMD Opteron™ 2352 (jusqu'à 4 cœurs)**
- **1 Go de RAM (jusqu'à 15 Go)**
- **100 Go d'espace disque (jusqu'à 800 Go)**
- **Trafic ILLIMITÉ**
- **OS Linux (Windows optionnel)**
- **1 nom de domaine inclus**
- **Accès administrateur**
- **Parallels® Plesk Panel 9**

Offre de lancement : 3 mois GRATUITS*

Configuration de base : ~~19,99~~ HT/mois ~~23,91 € TTC/mois~~

0 € pendant 3 mois*

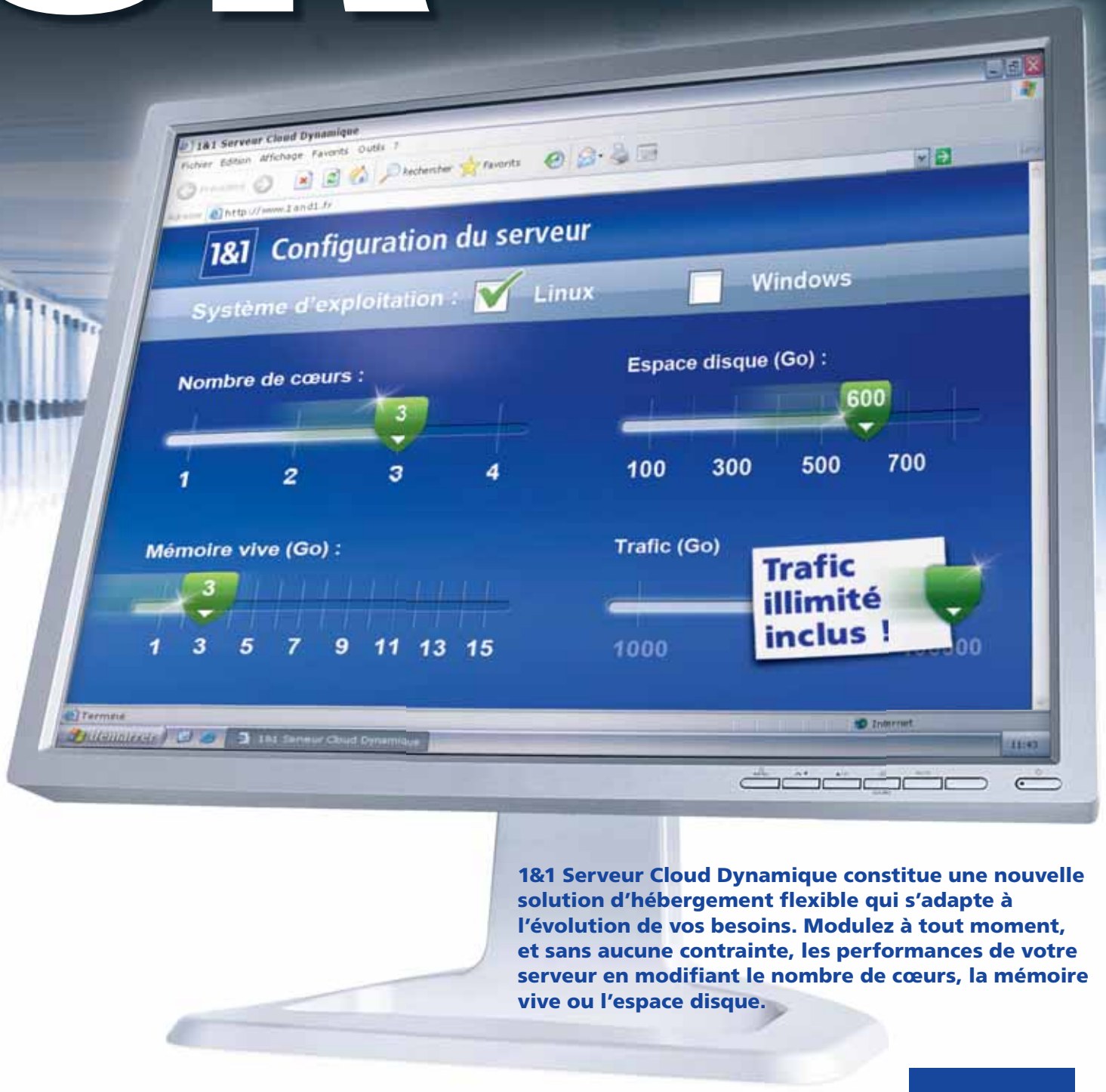
*Les 3 mois gratuits s'appliquent à la configuration de base. Pour une configuration supérieure, le prix appliqué sera égal à la différence entre le prix de la configuration souhaitée et celui de la configuration de base. A l'issue des 3 premiers mois, le prix annoncé sur notre site Internet s'appliquera. Offre soumise à un engagement de 12 mois et à des frais de mise en service de 22,72 € TTC. Conditions détaillées sur www.1and1.fr. Offres sans engagement également disponibles.

MEMBRE DE united internet

0970 808 911 Appel non surtaxé

Innovation 1&1

UR



1&1 Serveur Cloud Dynamique constitue une nouvelle solution d'hébergement flexible qui s'adapte à l'évolution de vos besoins. Modulez à tout moment, et sans aucune contrainte, les performances de votre serveur en modifiant le nombre de cœurs, la mémoire vive ou l'espace disque.

Consultez toutes nos offres serveurs sur notre site Internet !

1&1

www.1and1.fr

E-Commerce et Data Mining

La pratique du Cross-Selling, ventes croisées en français, est devenue de plus en plus fréquente chez les professionnels du e-commerce. Le principe est assez simple : il s'agit d'augmenter le panier moyen de l'acheteur en lui proposant des articles complémentaires. Cette action de conseil permet alors non seulement d'augmenter le chiffre d'affaires généré, mais aussi potentiellement de faire émerger des produits peu visibles et de donner au site un côté exclusif et personnalisé.

La complémentarité des produits proposés peut être évaluée de manière très simple voire simpliste – proposer les accessoires de l'objet choisi, où les œuvres du même auteur pour un article culturel - ou au contraire découverte par le travail complexe de professionnels du secteur, capables de discerner, de par leur analyse précise du marché, des articles potentiellement corrélés. Une autre approche, principalement responsable de l'explosion du nombre de sites équipés d'interfaces de ventes croisés, repose sur l'idée directrice que **les achats déjà effectués permettent de générer les suggestions** pour les nouvelles commandes.

Génération automatique ?

Le terme de génération automatique ramène à celui de Data Mining, terme recouvrant l'ensemble des techniques d'extraction de connaissance depuis les bases de données. Ces techniques ont principalement pour but **d'analyser l'historique** et d'utiliser ce qui a été découvert **pour générer des règles utilisables.** Ces règles peuvent viser à optimiser des processus aussi divers que le ciblage de campagnes de mailing, la détection de transactions frauduleuses, l'analyse de Churn et bien sûr, les ventes croisées.

Dans ce cas précis le principe est de suggérer les articles pouvant intéresser l'acheteur d'un panier donné, au vu des paniers existant dans l'historique de la base du site.

Réalisation de ce type de plate-forme

Ceci peut empiriquement sembler très simple à mettre en œuvre mais vu le nombre de références existant dans un site e-commerce, même de petite taille, analyser mathématiquement tous les articles potentiellement intéressants pour chaque acheteur est un processus très lourd, dont la complexité tend à devenir factorielle.

En plus de cette difficulté technique, la gestion fonctionnelle des règles générées nécessite des outils spécifiques : quand les services marketing veulent pouvoir suivre en temps réel l'évolution des règles, le développeur, lui, voudra simplement pouvoir intégrer les résultats des règles de manière aisée dans son code sans bouleverser ses habitudes. Quant aux services IT, ils doivent pouvoir facilement gérer la plate-forme d'analyse à l'aide d'outils appropriés.

C'est pourquoi pendant longtemps ce type de service a été l'apanage de grands groupes, seuls à même de financer le développement des plates-formes et outils nécessaires.

Cependant, depuis quelques années, des solutions complètes se proposent de répondre à ce genre de problématiques. Celles-ci sont présentées comme capables non seulement de proposer une plate-forme à haute disponibilité pour des calculs intensifs, mais aussi de fournir des interfaces appropriées allant de l'IT aux services opérationnels.

Data Mining en environnement Microsoft

Microsoft a été le premier acteur généraliste à s'inviter sur le segment assez élitiste du Data Mining. Les développements ont commencé chez Microsoft Research en 1999 et ont abouti à une première implémentation au sein de SQL Server 2000. [Fig.1]

Cette sortie a permis à Microsoft de formaliser la norme OLE DB For Data Mining et le langage DMX (Data Mining Extensions), vu comme une extension à SQL pour pouvoir requêter des modèles. De plus l'intégration du Data Mining au sein du serveur Analysis Services - comprenant déjà le moteur OLAP, existant depuis la version 7 de SQL Server - a permis à celui-ci de profiter des protocoles et API déjà développés (ADOMD, XMLA) pour une meilleure intégration avec les applicatifs. Les versions 2005 puis 2008 ont vu la suite se compléter pour arriver au total de neuf algorithmes, utilisés dans un environnement complet, et disposant d'une interface poussée avec d'une part les technologies SQL Server mais aussi avec le poste client au travers, entre autres, d'Excel 2007.

Cross-selling en environnement Microsoft

Un des algorithmes disponibles est particulièrement adapté à ce type de problématique : il s'agit de Microsoft Association Rules. Alors que la plupart des techniques cherchent à établir des corrélations entre les informations disponibles pour prédire une information manquante – cas typique du prospect dont on va chercher à deviner s'il peut devenir client grâce à son âge ou sa ville - Association Rules va s'intéresser à l'association d'objets sans tenir compte de leur nature. La raison est simple : dans ce type de scénario, les informations au sujet de l'acheteur sont généralement assez minces et insuffisantes pour prédire une appétence, quand l'examen des objets achetés ensemble démontre un potentiel prédictif bien plus élevé (1).

Génération de modèle

La création de modèles d'analyse se fait au sein de Business Intelligence Development Studio, installé avec SQL Server 2008, dans le

Fig.1

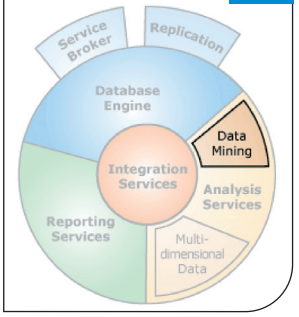
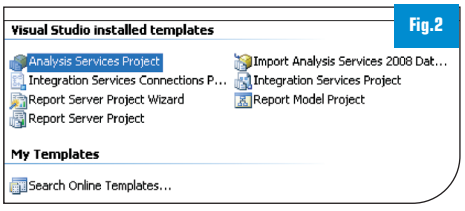


Fig.2



(1) Cette approche est aussi utilisée dans la censure automatique de contenus : pour déterminer si un contenu est impropre, il est plus efficace de connaitre des associations impropres plutôt que de considérer chaque mot unitairement.

cadre d'un projet Analysis Services. [Fig.2]. Au sein d'un projet Analysis Services, bien connu des développeurs OLAP, seuls trois composants doivent être définis pour faire de l'exploration de données :

1. Une **Data Source,** l'adresse physique d'un serveur ou fichier où récupérer les données
2. Une **Data Source View**, dans laquelle seront définis les éléments de données à utiliser
3. Une **Mining Structure** qui définit d'une part le typage des données à utiliser mais aussi l'algorithme - ici Association Rules - et son paramétrage. [Fig.3]

Dans le cas de Microsoft Association Rules, la seule contrainte au niveau données est de disposer des lignes historiques sous la forme d'une table de panier et d'une table d'associations entre panier et objets les constituant. [Fig.4]. La création de la structure d'analyse (Mining Structure) est ensuite assez simple. Dans le langage d'Analysis Services, le panier est le cas (*case*) à analyser dont les objets sont compris dans un champ multi-value vu comme une table imbriquée (*nested*). [Fig.5]. Une fois l'algorithme calculé sur le serveur, il est disponible pour analyse. Il est possible de visualiser les règles qui vont être utilisées pour prédire, classées selon deux critères :

- La **probabilité** qui correspond à la chance qu'une règle a de se vérifier
- L'**importance** qui qualifie l'écart entre la probabilité de la règle et celle issue des objets la composant.

En somme, une bonne suggestion d'achat doit être **probable** (être vraie) et **importante** (ne pas renvoyer les objets achetés quasi systématiquement et donc ne présenter aucun intérêt). Ces deux quantités sont bien sûr analysables dans une fenêtre dédiée. [Fig.6]

Il est en plus possible de parcourir les règles visuellement au sein d'un graphe de dépendances, où les nœuds sont les différents produits et les liens orientés les règles. [Fig.7]

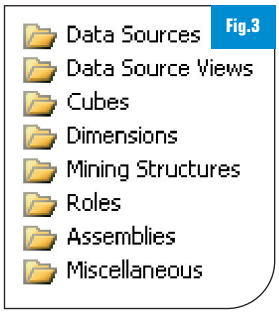


Fig.3

Le requêtage du modèle avec DMX

Pour requêter le modèle on emploie le langage DMX et sa syntaxe proche de SQL. Alors qu'en SQL on effectue des requêtes de sélections sur des tables souvent jointes entre elles, en DMX on effectue ces mêmes requêtes de sélection sur des modèles joints à des données d'entrée. Dans le cas présent, la requête typique à effectuer est une sélection sur le modèle d'association joint au panier courant de l'utilisateur. Le but est de récupérer une liste d'objets probables à afficher dans la zone de suggestion, contenant par exemple jusqu'à 10 objets.

```
//Prédiction
SELECT FLATTENED
PredictAssociation
(
 [Association].[v Assoc Seq Line Items],
 INCLUDE_STATISTICS,
 10
)
//Modèle
FROM [Association]
//Données d'entrée
NATURAL PREDICTION JOIN
(
SELECT
(
SELECT 'Objet 1' AS [Model]
UNION
…
UNION
 SELECT 'Objet N' AS [Model]
 )
 AS [v Assoc Seq Line Items]
)
AS t
```

La syntaxe est très facile à apprendre lorsque l'on possède quelques notions de SQL. Le résultat est renvoyé sous forme tabulaire dont la similarité avec un résultat relationnel est évidente. [Fig.8]. Cette ressemblance est poussée encore plus loin lorsque l'on examine les moyens d'intégrer cette requête et ses résultats dans un frontal .NET.

L'intégration dans un front-office

Lorsque l'on souhaite requêter une base SQL Server en .NET, on utilise SqlClient et un formalisme de code assez commun : utilisation de blocs using, création de SqlCommand au sein de SqlConnection, exécution et récupération via un SqlDataReader ou binding dans un

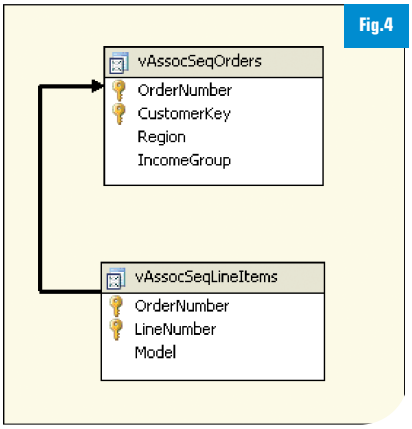


Fig.4

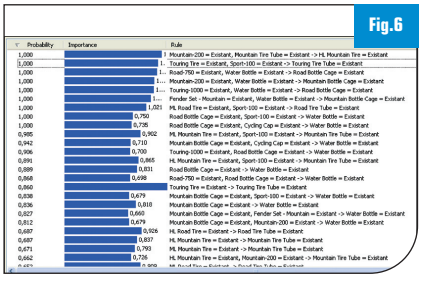


Fig.6

Model	$SUPPORT	$PROBABILITY	$ADJUSTEDPROBABILITY
Mountain Tire Tube	2908	0,671283471837488	0,688964226599737
Road Tire Tube	2216	0,651685393258427	0,71429730291936
Sport-100	6171	0,312096029547553	0,509034880331457
Patch kit	3010	0,295475530932595	0,590913112366106
Water Bottle	4076	0,191766643142 79	0,174500518018287
Mountain-200	2477	0,116537285344625	0,110003087483564
Cycling Cap	2095	0,098565043519172	0,0938632476257742
Fender Set - Mountain	2014	0,0947541754881204	0,0904034918952222
Mountain Bottle Cage	1941	0,0913196894848271	0,0872741246115121
Road Bottle Cage	1702	0,0800752764055516	0,0769531398784552

Fig.8

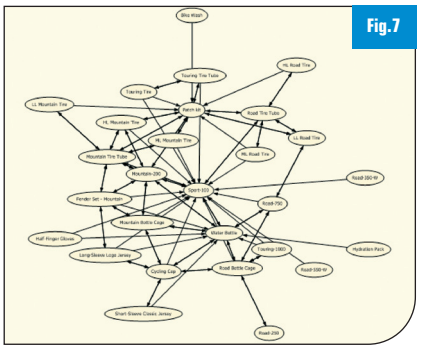


Fig.7

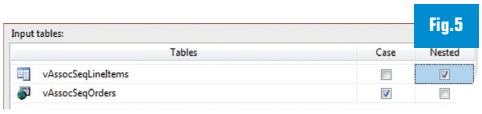


Fig.5

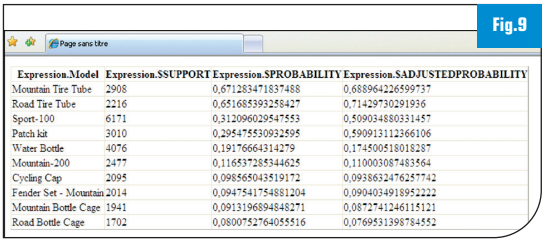
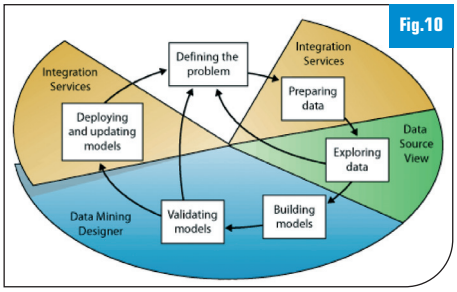
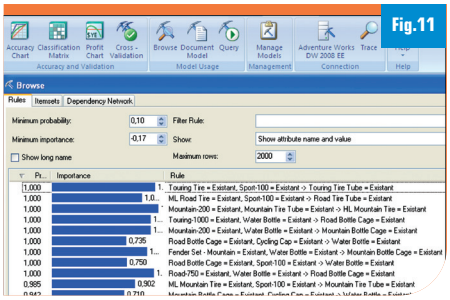
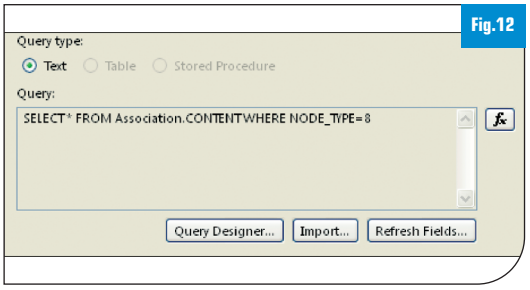


Fig.9







élément adéquat à travers un SqlDataAdapter. Dans le cas de DMX sous ADOMD.NET le principe est le même comme le montre l'exemple de code ci-dessous.

```
try
{
 using (AdomdConnection conn =
       new AdomdConnection("ConnectionString"))
 {
   AdomdCommand cmd =  conn.CreateCommand();
   cmd.CommandText = "requête"
   using (AdomdDataReader dr = conn.ExecuteReader())
   {
     while(dr.Read())
     {
     //Récupération du résultat
     }
     dr.Close();
   }
   conn.Close();
 }
}
catch (AdomdException ex)
{
 //Gestion des erreurs
}
```

Le DataBinding .NET marche exactement de la même manière puisque l'on dispose d'un DataAdapter dédié.

```
AdomdDataAdapter da = new AdomdDataAdapter(cmd);
DataSet ds = new DataSet();
da.Fill(ds);
gridView1.DataSource = ds.Tables[0].DefaultView;
gridView1.DataBind();
```

Les résultats sont donc intégrables en quelques clics dans une page ASP.NET. [Fig.9]. Cela peut sembler trivial, mais dans nombre de solutions concurrentes la simple récupération de résultats nécessite un volume de développement très élevé. Il s'agit sans conteste d'un des grands points forts de l'offre Microsoft.

Les outils de maintenance

Les fonctionnalités citées étant intégrées au sein d'Analysis Services, la stratégie de maintenance ne nécessite aucune formation supplémentaire pour les services d'exploitation. Le protocole de communication (XMLA) et les outils d'administration (SQL Server Management Studio) sont bien sûr strictement inchangés. Quant à l'alimentation des modèles, comme pour une base OLAP, il est recommandé d'utiliser SQL Server Integration Services, l'ETL inclus dans SQL Server. Certes le Data Mining amène des problématiques spécifiques par rapport à l'OLAP. Il est par exemple souvent fait référence à la nécessité d'être en mesure de **suivre l'évolution des performances prédictives** du modèle. Là encore les autres outils de la suite disposent de fonctionnalités prévues pour cela : SSIS permet de requêter et ré-entraîner un modèle de Data Mining, afin de mesurer et d'améliorer sa performance. [Fig.10]

Interaction avec les services fonctionnels

On sait généralement que le Data Mining apporte des capacités prédictives mais on oublie souvent que son usage premier est la compréhension des données. Il est nécessaire d'être en mesure de fournir aux services amenés à piloter l'activité du site les moyens d'évaluer l'impact des solutions mises en place. Le choix de Microsoft en la matière a été de développer un addin Excel donnant à l'utilisateur non technique accès à la plate-forme d'analyse. Dans le cas du modèle de ventes croisées, les visualiseurs abordés plus haut sont donc directement mis à disposition sur le poste client au sein du logiciel le plus utilisé en entreprise. [Fig.11]. Il ne faut bien sûr pas oublier la plate-forme de création et de consultation de rapports de SQL Server, Reporting Services, qui permet aussi de mettre simplement à disposition de chacun les résultats des modèles, via le connecteur spécifique à Analysis Services. [Fig.12]. Une fois un DataSet Analysis Services créé, il est tout à fait possible de publier les métadonnées d'un modèle, par exemple les règles de suggestion. Enfin, ceux qui le souhaitent peuvent utiliser les visualiseurs dans des interfaces personnalisées. Ceux-ci sont disponibles en tant que contrôles Winforms au sein de l'assembly Microsoft.AnalysisServices.Viewer.dll.

Conclusion

Être capable de dépasser le stade de la vente par correspondance, d'être au-delà d'un simple catalogue est devenu un challenge pour tous les sites d'e-commerce. À travers des solutions d'analyse automatique, ils apportent une nouvelle dimension à leur site Web, en offrant des capacités de conseil personnalisé, en mettant en avant des produits rares, en incitant à un parcours ludique d'article en article, le tout dynamiquement. Grâce à la solution de data mining de Microsoft SQL Server 2008, la mise en place d'une telle plate-forme, auparavant crainte tant pour des raisons de tarifs des serveurs que de temps de développement des outils liés, devient accessible au plus grand nombre, sans pour autant sacrifier la performance. Le moteur OLAP d'Analysis Services est devenu en quelques années le plus utilisé dans les entreprises, grâce à une convivialité hors pair alliée à une performance reconnue. Il serait intéressant que le moteur de Data Mining, dont les capacités vont bien au-delà de l'exemple de cet article, suive cette voie et devienne tout aussi incontournable.

■ François **Jehl** - *Consultant/ Formateur Access-IT*

OFFRE D'EMPLOI
MONTPELLIER

DÉVELOPPER EST VOTRE PASSION?

PC SOFT recrute :
- 4 Ingénieurs développement C++
- 2 Ingénieurs développement Java
- 1 Ingénieur développement Android

Environnement de travail optimal: équipe de haut niveau, jeune, à taille humaine, organisé par bureau de 2 ingénieurs, matériel de pointe (double ou triple écran, 2 machines par développeur), dernières versions des logiciels, plus de 100 machines dédiées équipées d'automates de tests,...

JEUNES DIPLÔMÉS BIENVENUS !

D'où viennent les collaboratrices et collaborateurs à PC SOFT ?
Ils et elles sont diplômés de ces écoles (parmi d'autres) : Polytechnique, Supaero, Supelec, ENST, Telecom INT, ENSI, EFREI, ENSEA, E3i, Epita, EPSI, ESSI, Polytech, MASTER...

PC SOFT est éditeur de logiciel: vous travaillerez au siège de la société, à Montpellier. Vous ne serez pas détaché chez un client.

Niveau de rémunération élevé :
Salaire, Primes, Participation aux bénéfices, Intéressement.

Vous êtes **passionné de développement** et accroc aux nouvelles technologies ?

Vous désirez participer à la conception et à la réalisation d'un projet d'envergure mondiale ?

Vous voulez être reconnu pour votre excellence technique ?

Venez rejoindre nos équipes d'ingénieurs !

Vous serez acteur des innovations de WINDEV et WEBDEV, AGL N°1 en France, distribué dans 98 pays.

Vous mettrez en œuvre des **technologies avancées** - programmation parallèle, multi-plateformes, MVC, Design Pattern, Ajax...- dans des domaines innovants - clients riches, Web, Mobilité -.

Vous concevrez et développerez des nouveautés selon les **méthodes agiles** en étant autonome et responsable de vos propres projets, avec le soutien de chefs de projets expérimentés et d'une équipe Qualité.

Vous programmerez en **langages objet** C++, C#, Java, PHP, JavaScript sur des plateformes multiples: Windows, Linux, Mac, Mobile, Android, Web.

Vous renforcerez ainsi votre savoir et vous acquerrez de nouvelles compétences, tant techniques que méthodologiques et ce dans un cadre humain et agréable.

Déposez votre candidature sur www.pcsoft.fr

Postes situés à Montpellier

Fournisseur Officiel de la Préparation Olympique

PC SOFT 25 ANNÉE Au service des développeurs

www.pcsoft.fr
3 rue de Puech Villa, 34197 Montpellier
Tél: **04.67.032.032** **info@pcsoft.fr**

Internet Explorer 8 pour les développeurs web

Internet Explorer 8.0 est rentré progressivement dans notre vie quotidienne depuis mars 2009. Il est proposé en mise à jour via Windows Update pour les PC avec Windows XP SP2 et supérieur. A noter qu'il fera partie intégrante des navigateurs proposés à l'installation de Windows 7 en Europe.

Pour rappel voici les systèmes d'exploitation qui supportent IE8 en architecture 32 et 64 bits :

- Windows XP SP2 et SP3
- Windows Server 2003 SP2
- Windows Vista
- Windows 7
- Windows Server 2008

Dans le cadre du développement Web, nous verrons quels sont les impacts dans notre manière de développer une page Web, comment profiter des nouvelles fonctionnalités offertes par IE8 et gérer les problématiques de compatibilité, notamment lors des migrations d'applications.

LE DÉVELOPPEMENT WEB

Respect des normes

Comme vous le savez, le développement Web nécessite la connaissance de langages, protocoles dans l'objectif d'obtenir les comportements souhaités pour une page Web. Ces protocoles et langages évoluent au même titre que les produits et un des points reproché aux versions précédentes d'Internet Explorer 8.0 était son respect partiel voire insuffisant des normes HTML et CSS.

Une des grandes évolutions est la prise en charge complète de CSS 2.1 qui permet à IE8 de passer avec succès le test ACID 2. Ce test a pour objectif de valider la capacité des navigateurs à respecter les standards du Web et notamment sur l'interprétation du balisage HTML, CSS 2.1 et les images PNG. On peut notamment souligner les avancées de cette version dans ce domaine :

- Prise en charge complète du CSS 2.1 (Cascading Style Sheet)
- Supporte HTML 5.0 notamment pour la navigation AJAX, l'incontournable du Web 2.0
- Supporte CSS.30
- Amélioration du DOM (Document Object Model) avec l'ajout de classes et fonctions pour compléter le portefeuille existant.

Outils intégrés

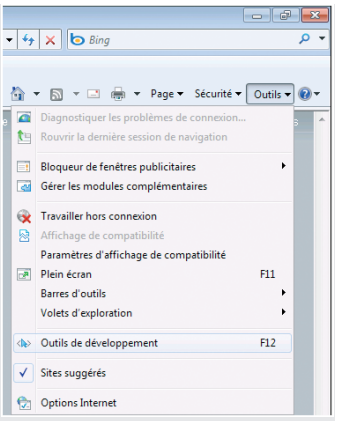
Afin de faciliter le travail du développeur Web, IE8 intègre des outils complémentaires et essentiels pour gérer la complexité du développement Web :

- Manipulation DOM
- Exploration HTML détaillée et intuitive d'une page
- Exploration du CSS (Cascading Style Sheet)
- Debug et manipulation JavaScript

Ces outils sont parfaitement intégrés au produit et il n'est nullement nécessaire d'installer un complément, une mise à jour pour les utiliser. Une fois que vous commencerez à les utiliser, il n'est même plus envisageable de s'en débarrasser tellement ils vous apporteront une flexibilité et un réel appui dans vos développements !

Comment y accéder ?

Méthode 1	Méthode 2
Un simple raccourci avec la touche F12 dans IE8	Depuis le menu, cliquez sur **Outils** puis **Outils de développement**



Depuis cette nouvelle page, vous avez accès à l'ensemble des outils pour les développeurs Web. [Fig.1]

Onglet HTML

L'onglet HTML vous permettra d'identifier le code HTML de la page en cours. Son principal atout n'est pas de nous montrer basiquement le code HTML que l'on pourrait très bien obtenir simplement avec un clic droit Voir source mais bien d'inspecter la page. Par exemple, si vous vous déplacez au sein de la hiérarchie HTML, en cliquant sur un élément, par exemple une table, vous obtiendrez l'ensemble des détails sur cet élément. Ils se trouvent sur la partie droite de la fenêtre de l'outil de développement. Notez bien que les outils ci-dessous ne se concentrent pas seulement sur de la visualisation mais permettent d'interagir avec la page. Si vous décidez de décocher une case d'une propriété CSS, vous verrez le résultat sur la page immédiatement. Que demander de plus pour trouver la propriété CSS à l'origine d'un problème de positionnement ou d'affichage ?

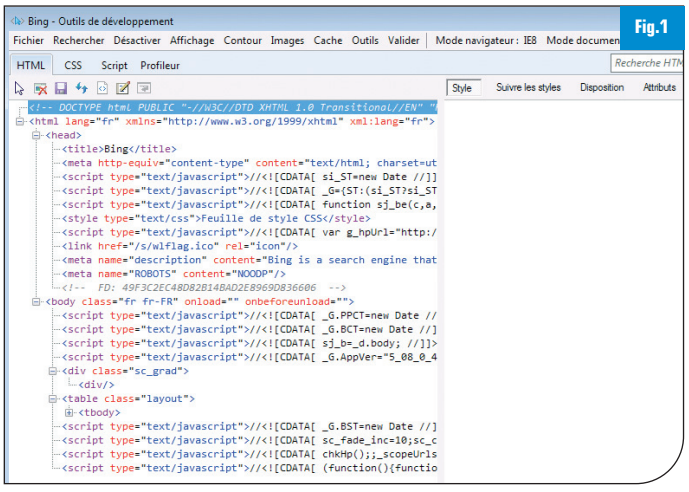


Fig.1

Styles

Styles permet d'identifier les règles qui sont appliquées à l'élément sélectionné dans le code HTML. A noter que l'ordre des règles respecte l'ordre dans lequel elles sont appliquées. Pratique pour déboguer, notamment lors de surchage CSS.

Suivre les styles

Cet outil s'adresse évidemment aux styles CSS à la différence de **Styles**, l'outil classe les règles CSS par propriété. Par exemple si vous recherchez toutes les occurrences sur la hauteur, il sera préférable d'utiliser cet outil.

Disposition

L'outil de disposition se focalise sur le positionnement des éléments. Par exemple, vous ne comprenez pas pourquoi votre padding ne s'applique pas, utilisez l'outil Disposition sur votre élément.

Attributs

Cet outil vous permet de définir des attributs pour l'élément sélectionné. Très pratique lorsque vous voulez simuler un comportement dans votre page. Vous pouvez bien entendu en ajouter, supprimer ou modifier.

L'onglet CSS dresse l'ensemble des propriétés CSS utilisées par la page

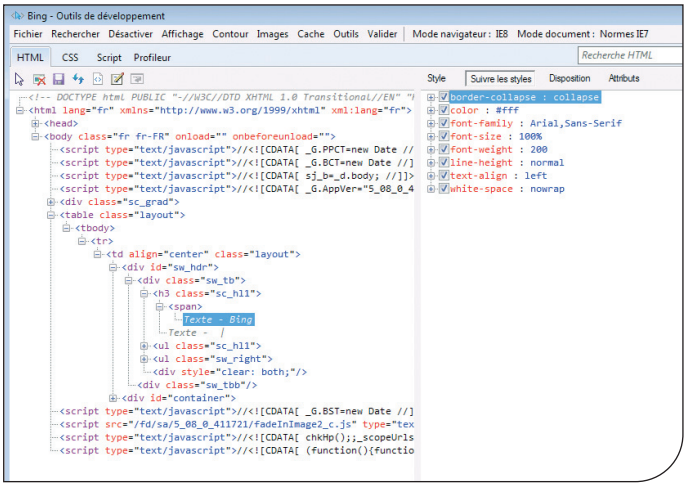
L'onglet Script permet d'inspecter l'ensemble des fonctions qui sont employées dans la page. Une fonction intéressante sera de pouvoir lancer le débogage d'une page au sein du code JavaScript.

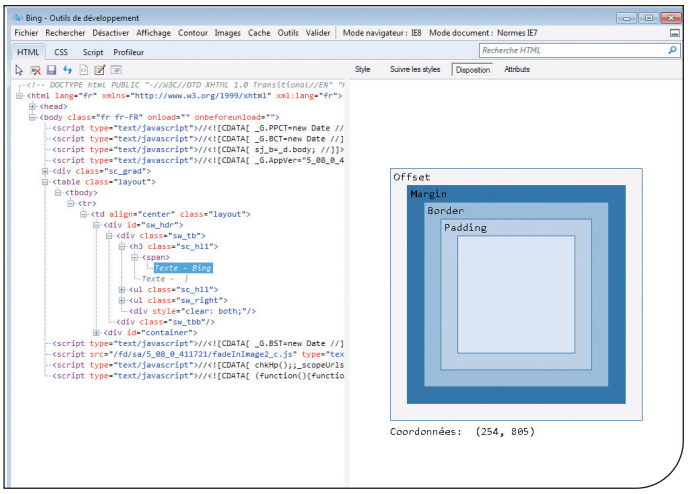
On retrouvera l'ensemble des fonctionnalités propres au débogage, comme l'ordonnancement via des points d'arrêt, la possibilité de contrôler les actions lecture/pause, vérifier les valeurs de variables à un état précis… De la même manière que les outils de l'onglet HTML vus précédemment, vous pouvez agir directement sur les styles de la page en cochant/décochant les cases propres à chaque propriété CSS. [Fig.2]

L'onglet Script

L'outil Script a pour but de vous aider à déboguer le code JavaScript de votre page. En tant que développeur Web, vous savez combien il peut être complexe de déboguer du code JavaScript ! Cet outil vous permet de :

- Définir des points d'arrêt pour contrôler l'exécution de vos scripts
- L'affichage du Console, notamment pour l'affichage d'informations durant l'exécution. A noter que les messages d'erreur sont logués dans le fichier console.log
- L'affichage et la gestion des variables locales pour vérifier si telle ou telle variable a bien telle ou telle valeur à un moment T





- Afficher les fonctions en cours d'appel. Cela permet de voir quelles sont les fonctions et le code associé en cours d'exécution.

L'onglet Profileur

Un outil intéressant pour améliorer votre page Web. Le profileur va vous permettre d'identifier rapidement les fonctions ou bouts de code qui sont les plus consommateurs au sein d'une page. Depuis le profileur vous avez un lien direct vers la fonction et la ligne concernée. Le profileur vous donne la possibilité de gérer plusieurs sessions, notamment avec la notion de rapport. Vos rapports peuvent être également exportés en format CSV. Vos exports peuvent ensuite vous aider à consolider vos résultats et fournir des rapports complets d'analyse de vos sites Web.

Le sélecteur

Le sélecteur est un mode de l'outil de développement qui vous permet de sélectionner n'importe quel élément de manière visuelle directement sur la page Web. Une fois sélectionné, l'outil de développement actualise ses données et vous positionne dans la portion de code HTML concernée.

C'est un mode réellement intéressant pour avoir accès rapidement au code correspondant. Je me rappelle personnellement de mes débuts dans le développement Web il y a une dizaine d'années où l'on devait faire soi-même la corrélation pas à pas entre ce qu'il y avait dans l'aperçu et le code HTML qui peut être parfois réellement complexe. Dans l'illustration [Fig.3], on a utilisé le sélecteur sur la boîte de texte pour saisir nos mots-clés de recherche sous Bing. Automatiquement les définitions HTML et CSS correspondantes sont apparues.

Gestion des liens

Quand on travaille sur le développement d'un site Web et en particulier de portails, un des éléments fondamentaux et utilisé N fois est le lien hypertexte. L'outil de développement permet de remplacer le contenu texte des liens par les URL. Cela peut présenter un vif intérêt lorsque l'on souhaite vérifier les liens sans devoir regarder les sources ou sans passer la souris dessus.

Une autre fonctionnalité fournie qui permet de compléter cette approche est le rapport de liens. [Fig.4]

Une page s'affiche et répertorie l'ensemble des liens de la page en cours. On peut ainsi vérifier très rapidement la cohérence texte affiché/URL ou toute autre correspondance.

Ces fonctionnalités sur les liens sont accessibles depuis le menu **Affichage** :

- **Chemins de lien**
- **Rapport de lien**

Architecture de la page

L'architecture de la page peut être visuellement identifiée par de la coloration du contenu. En effet, on peut par exemple, afficher le contour des tableaux ou des DIV de la page. Ci-dessous on voit une illustration de cette fonctionnalité. [Fig.5]

Les éléments dont le contour est vert correspondent aux différents DIV qui architecturent la page.

En plus de l'architecture visuelle, on a bien entendu accès à la structuration DOM de la page.

Ces différentes options sont disponibles depuis les menus **Affichage** et **Contour** :

- Structuration DOM : **Affichage > Source >** …
- Structuration des éléments : **Contour >**
  - **Tableaux**
  - **DIV**
  - **Images**
  - …

Gestion du cache et des cookies

Le cache est un composant très important des serveurs Web mais également des clients et donc des navigateurs internet. Le cache est de plus en plus utilisé dans les applications Web d'aujourd'hui et permet d'optimiser les performances d'affichage des pages en stockant des pages, des éléments de page localement afin de ne pas devoir les recharger à chaque affichage. Vous vous en doutez, l'inconvénient majeur du cache pour un développeur peut être de maintenir des développements dans un état qui n'est pas l'état réel. En effet, il arrive souvent qu'on ne comprenne pas qu'un développement ne soit pas pris en compte, très souvent le cache se trouve être le « coupable » de ce genre de comportement. L'outil de développement IE8 permet de contrôler le cache et notamment de le vider à la demande. Pour rappel, une combinaison souvent efficace est également d'utiliser Ctrl+F5. L'avantage de l'outil est de pouvoir ajouter une notion de granularité. En effet on peut gérer le cache du navigateur entièrement ou pour un domaine donné selon la page où on se trouve. Nous retrouvons exactement la même notion sur la suppression des Cookies. Ces fonctionnalités sont accessibles depuis le menu **Cache.** Pour conclure, une fonctionnalité intéressante est de pouvoir afficher les cookies qui dépendent de la page en cours. On peut ainsi vérifier la date d'expiration, son nom, sa valeur, le domaine d'application et son chemin via un rapport global (voir ci-dessous). Un rapport s'affichera dans une nouvelle page HTML, ce n'est pas sans rappeler le rapport de liens vu précédemment.

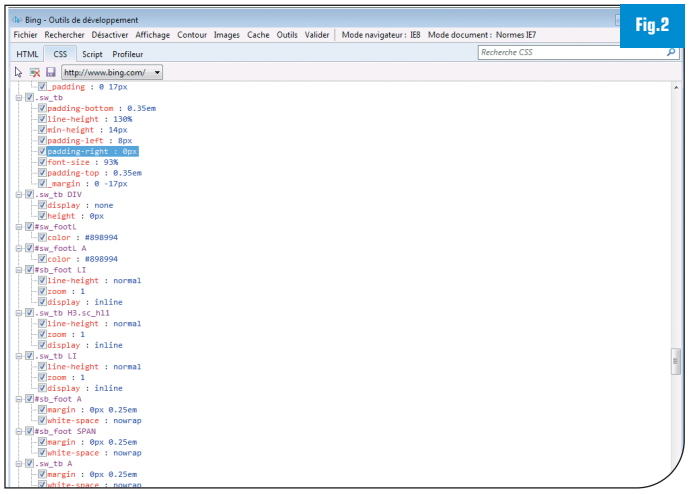


Fig.2

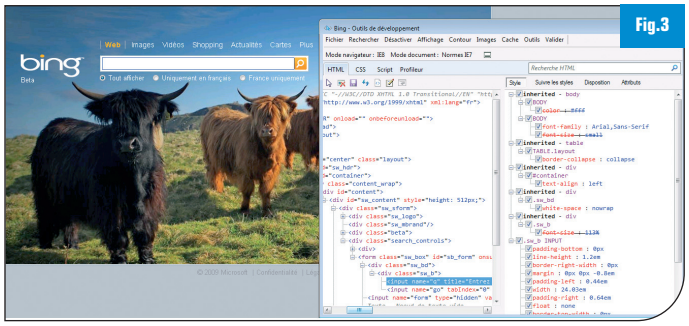


Fig.3

Validation & conformité

Pour conclure sur l'outil de développement, nous allons voir les fonctions de validation fournies. Comme vous le savez, on peut développer une page Web de diverses façons tout en ayant un résultat visuel relativement semblable. Cependant, le HTML permet cette flexibilité mais si on souhaite avoir des pages visibles de façon identique par un plus grand nombre, le principe de base est de se conformer à certaines règles afin d'obtenir des pages conformes aux règles de développement Web, établies notamment par le W3C. Pour faciliter ce travail de mise en conformité, on peut valider sa page, voire son flux de données, via l'outil de développement.

On peut choisir son niveau de granularité sur la validation :

Link Report - http://www.bing.com/ — Fig.4

http://www.bing.com/

1. http://fr.msn.com/
2. http://mail.live.com/
3. http://www.bing.com/#

OnClick - function onclick() { setHp()

4. http://login.live.com/login.srf?wa=wsignin1.0&rpsnv=11&ct=1254664849&rver=5.5.4177.0&wp=MBI&wreply=http:%2F%2Fwww.bing.com%2FPassport252f&lc=1036&id=264960
5. http://www.bing.com/worldwide.aspx?FORM=WHLH
6. http://www.bing.com/settings.aspx?ru=%2f&FORM=SELH
7. http://www.bing.com/images?FORM=Z9LH

OnClick - function onclick() { selectScope(this, 'images');

8. http://www.bing.com/?scope=video&FORM=Z9LH1

OnClick - function onclick() { selectScope(this, 'video');

9. http://ms.ciao.fr/ms/products?v=1

OnClick - function onclick() { selectScope(this, 'commerce');

10. http://www.bing.com/?scope=news&FORM=Z9LH2

OnClick - function onclick() { selectScope(this, 'news');

11. http://www.bing.com/maps/?FORM=Z9LH3

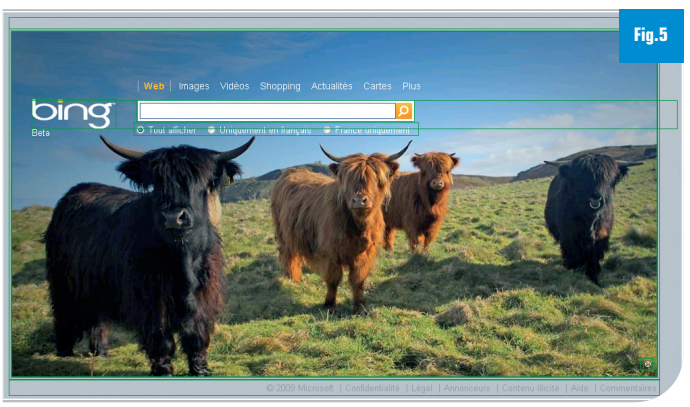


Fig.5

- HTML : respect des normes HTML
- CSS : respect des normes CSS
- Flux : vérifier si la définition du flux est cohérente
- Liens : vérifier si les liens existent bien
- Tous (validations multiples)

Par défaut, nous utilisons les outils de validation du W3C ou FeedValidator mis à la disposition des développeurs.

Modes de compatibilité

Internet Explorer 8.0 intègre nativement des modes de compatibilité avec ses versions précédentes. Dans quel but ? Tout simplement pour permettre de continuer à utiliser une application, un site qui ne respecterait pas les standards et qui s'affichait correctement avant. Ce type de problématique est très important lors d'une migration du poste de travail. Imaginez que des applications Web sont fondamentales au business d'une entreprise avec plusieurs milliers de postes clients…on peut se trouver très rapidement freiné par une migration, quand bien même elle serait envisagée ! Les choses sont réellement facilitées avec la possibilité d'afficher tel site ou telle application avec un rendu spécifique en attendant par exemple son remplacement ou sa migration. Les cas typiques de problèmes rencontrés lors de migration vers IE8 sont clairement autour du développement CSS et JavaScript. Le navigateur étant moins permissif que dans le passé, on peut avoir des comportements inattendus et notamment sur l'affichage du contenu. Ce genre de problèmes couvre tout de même au moins 70 à 80% des problèmes rencontrés à ce jour. Ce ne sont pas le plus souvent des problèmes critiques, mais les modes de compatibilité seront là pour vous aider dans votre démarche.

Quels sont les modes de compatibilité ?

Depuis Internet Explorer 6.0 on trouve deux modes principaux :

- Quirks
- Standard

Le mode Quirks est celui qui permet d'afficher une page avec un rendu quasi équivalent à IE5 et le mode *Standard* dit également *Strict* permet d'afficher le contenu avec le moteur de rendu du navigateur. Sur IE6, le mode standard était le moteur IE6 et à partir d'IE7 le mode standard a été actualisé par l'utilisation du rendu IE7. Quelle différence avec IE8 ? Celui-ci est devenu bien entendu le moteur de rendu standard mais il intègre également le moteur de rendu IE7 ! On peut ainsi émuler des rendus IE7, et pas simplement utiliser un mode de compatibilité. Vous l'aurez compris, on peut donc faire varier le rendu de notre page avec des versions anciennes (IE5) mais également dans un mode IE7 comme IE8 !

Comment spécifier le mode de compatibilité à utiliser ?

La première méthode consiste à indiquer au navigateur que telle page devra utiliser tel mode. Cela est rendu possible par l'utilisation d'une balise *meta* bien particulière.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
    <head>
        <meta http-equiv="X-UA-Compatible" content="IE=7.5">
        <title>Ma Page</title>
    </head>
    <body onload="javascript:alert(document.documentMode);">
        Ma page en IE7 avec mon IE8!
    </body>
</html>
```

La balise *meta* doit être placée dans le *header* de la page et avant les autres éléments, exception faite pour *title* et les autres balises *meta*. La fonction JavaScript document.documentMode permet d'afficher la version du navigateur utilisée.

Internet Explorer 5.0 – Mode de compatibilité	`<meta http-equiv="X-UA-Compatible" content="IE=5">`
Internet Explorer 7.0 – Mode de compatibilité	`<meta http-equiv="X-UA-Compatible" content="IE=7.5">`
Internet Explorer 8.0 – Mode de compatibilité	`<meta http-equiv="X-UA-Compatible" content="IE=100">`
Préciser à IE8 de prendre le mode de compatibilité le plus récent	`<meta http-equiv="X-UA-Compatible" content="IE=Edge">`
Internet Explorer 7.0 - Emulation	`<meta http-equiv="X-UA-Compatible" content="IE=EmulateIE7">`

A noter que si vous définissez plusieurs meta X-UA-Compatible, seule la première balise sera interprétée. Cependant, si vous souhaitez rendre vos sites disponibles avec plusieurs modes de compatibilité, il vous suffira de le spécifier en les séparant par un point-virgule :

```
<meta http-equiv="X-UA-Compatible" content="IE=7.5; IE=8" >
```

La question suivante est bien entendu : comment définir un mode de compatibilité suivi par N sites, N applications. C'est trop compliqué, trop chronophage et trop impactant de rajouter ce genre de code dans chacune des pages ! Et bien on va le définir au niveau du serveur, on peut le faire sur un serveur Apache comme IIS. Je vais prendre ici l'exemple IIS. Le principe et l'implémentation sont relativement simples puisqu'il ne vous faudra définir qu'un *header* particulier. En reprenant l'exemple précédent pour qu'IE8 émule les sites avec IE7 :

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <system.webServer>
    <httpProtocol>
      <customHeaders>
        <clear />
        <add name="X-UA-Compatible" value="IE=EmulateIE7" />
      </customHeaders>
    </httpProtocol>
  </system.webServer>
</configuration>
```

Pour définir une réponse http personnalisée dans IIS, il vous suffit de créer une nouvelle réponse http avec comme nom « X-UA-Compatible » et pour valeur le mode de compatibilité voulu (ex : « IE=EmulateIE7 »).
Que se passe-t-il si je ne précise pas de balise *meta* ?
Internet Explorer 8.0 va alors rechercher le DOCTYPE de la page afin d'identifier si une information sur le mode de rendu a été définie. Si ce n'est pas le cas, la page sera automatiquement en mode quirks (IE5). Si la balise DOCTYPE spécifie un document standard, alors le navigateur interprètera le rendu avec son moteur de rendu.
A noter que quelques exceptions existent sur ce comportement comme la définition de l'**affichage de compatibilité** qui surcharge la définition du DOCTYPE. Enfin, une dernière méthode consiste à modifier une clé de registre pour spécifier un comportement global d'IE8 sur un poste de travail.

HKEY_LOCAL_MACHINE (or HKEY_CURRENT_USER)
SOFTWARE
Microsoft
Internet Explorer
Main FeatureControl
FEATURE_BROWSER_EMULATION
iexplore.exe = (DWORD)

■ Vincent **Bellet**
Consultant, application Platform optimization Microsoft France

Microsoft 2010 : tout est

Février marque la grand-messe de Microsoft en France avec les TechDays, l'événement majeur de l'éditeur en Europe avec plus de 12 000 visiteurs en trois jours. Et cette super-conférence sera le point de départ de nombreuses évolutions pour le développeur cette année. Sharepoint 2010, Office 2010, . Net 4, le développement parallèle, Silverlight 4, Visual Studio 2010, de nouveaux langages, de nouveaux outils, l'arrivée de l'UML, la sortie d'Azure, de nouveaux SDK, du multitouch, etc. Bref, le développeur ne va pas s'ennuyer.

« L'année 2010 sera très riche, ce fut le cas dès fin 2009 avec Windows 7, Exchange 2010, Windows Server 2008 r2. Nous avons aussi beaucoup d'actualité autour du multitouch, sur la sécurité, la programmation parallèle, la compatibilité des applications, XP Mode. .Net 4, Silverlight 4 et Visual Studio 2010 forment un trio d'envergure »

Laurent Ellerbach (directeur marketing de la division plate-forme et écosystème).

Objectif TechDays

Les TechDays 2010 en chiffres représentent : 300 sessions et ateliers, 5 500 m2 d'exposition, des dizaines de communautés, des centaines de speakers. Cette année, 6 thèmes sont à l'honneur : découvertes, architecture et développement, infrastructure, efficacité individuelle et collective, enjeux et enfin le rendez-vous décideurs informatique. Plusieurs focus sont mis en avant : le cloud computing, le jeu, le développement web, Sharepoint, Visual Studio 2010, la virtualisation, la sécurité, Windows 7... Le premier jour sera plutôt orienté IT et développeur, le 2e, tourné vers Office 2010 et enfin le 3e, portera de nouveau sur le développeur, avec une grande session plénière par jour. Office 2010 sera une des vedettes de l'événement avec plusieurs sessions pour les utilisateurs mais aussi l'administrateur et le développeur.

À ne pas manquer :

Le cloud et le développement : session centrée sur le cloud computing et les modèles de développement, lundi 8 février de 13 heures à 14 heures, par François Tonic (rédacteur en chef de Programmez !)

Concours « Je vois des Windows partout ! »

Un concours de développement embarqué autour des futures applications embarquées (utilisant Windows Embedded) va se dérouler durant les TechDays ! Original, le développeur devra utiliser : VS 2010, Windows Embedded Standard 2011 ou Embedded CE 6.0 R3 ! Soyez nombreux à participer !

Site officiel : http://msdn.microsoft.com/fr-fr/windowsembedded/msdn.windows.partout.aspx

neuf pour le développeur !

FOCUS sur WINDOWS AZURE

Le cloud computing fait désormais partie du vocabulaire même si son usage reste très limité. Mais avec la disponibilité de Windows Azure, des outils de développement, des API et SDK, les entreprises, les développeurs peuvent dès à présent réfléchir, tester. Et le développeur achetant Visual Studio 2010 bénéficiera de nombreuses heures gratuites sur Azure ! Et surtout l'interopérabilité d'Azure avec Java et PHP, et même avec Eclipse, peut inciter le développeur non Windows à s'y intéresser. Avec l'approche de plus en plus marquée pour le Software + Services chez Microsoft, l'avenir est clairement vers les services en ligne, les environnements logiciels mixtes (déploiement desktop et en ligne).

Visual Studio 2010 : la tour de contrôle du développeur .Net et Windows

VS 2010 est souvent présentée comme une version majeure de l'IDE, la plus importante depuis la v2003. *« Beaucoup de fonctions proviennent du feedback des développeurs »* affirme Laurent Ellerbach. Cette version apporte de très nombreuses nouveautés comme nous l'avons vu dans le n° 126, notamment sur les tests, la modélisation UML, le cloud, le développement web, avec une vraie intégration de Silverlight et d'Expression. Mais surtout, cette version inaugure une nouvelle génération d'éditeur avec l'apparition d'un éditeur basé sur WPF, la librairie d'interface de .Net. Les développeurs avaient eu des craintes sur la vélocité de l'IDE WPF. Sans être à 100 % rassurés, les derniers tests semblent plutôt bons, même s'il faudra attendre quelques mois avant d'en tirer la moindre conclusion.

Une des grosses nouveautés de VS 2010 et de .Net 4 est la programmation parallèle. Côté librairie, nous disposons de PLinq et de Task Parallel Library. À cela s'ajoutent des librairies natives telles que le Parallel Pattern, Agents. Le but est de simplifier l'accès au parallélisme dans son code en le modifiant au minimum tout en proposant des accès .Net et C ++. Parmi les outils dédiés au parallélisme, notons l'apparition du Parallel Debugger Tool Windows. Par rapport aux premières pré-versions, VS 2010 a parfois modifié son approche et épuré certains mécanismes comme les Tasks au sein du framework .Net 4. Pourtant, le plus dur reste à faire : convaincre le développeur de passer au multicore et maîtriser les bonnes pratiques du parallélisme. Et là le chemin risque d'être long ! Mais au-delà du parallélisme pur et dur, c'est tout l'univers des hautes performances que Microsoft veut adresser : HPC, gros système, industrialisation des tests et du développement (version Ultimate, Team System).

A la recherche de nouveaux langages

2010 sera aussi l'année des nouvelles expériences dans les langages. Microsoft travaille sur de nombreux langages expérimentaux ou plus avancés qui ont vocation, ou non, à être inclus dans la plate-forme. F# est un des exemples les plus connus. Même si aujourd'hui il demeure discret sur .Net, il n'en demeure pas moins un langage d'avenir. Autre nouveauté des laboratoires MS Research : Kodu. Il s'agit d'un langage Visual simple et dédié aux jeux ! Autre langage en pleine élaboration : Vedea. Son principe : être un langage pour la visualisation des données et pour créer des infographies interactives. Il utilise .Net 4 et la DLR, et sa syntaxe est proche de celle de C#.

François Tonic

L'agenda 2010 des principales sorties

- **Visual Studio 2010** : initialement prévu pour le 22 mars, la disponibilité générale de l'outil ne devrait pas intervenir avant courant avril. Pas de date fixée officiellement. Les équipes attendent le retour des testeurs.
- **Net 4.0** : disponibilité avec VS 2010
- **Les éditions Express de Visual Studio** : ces versions gratuites de l'IDE seront disponibles en édition 2010 après la disponibilité de VS 2010, pas de date annoncée.
- **Silverlight 4.0** : sans doute courant de l'été.
- **Silverlight Mobile** : annonce attendue à Mix'2010. Pas de date annoncée
- **Gamme Expression** : le principal changement interviendra après la sortie de .Net 4 avec le support du nouveau .Net (et de Silverlight 4) par Expression Blend. Actuellement en préversion.

Comment génère-t-on l'agenda des TechDays ?

Pour générer et assurer la cohérence de l'agenda bien chargé des TechDays, Pascal Belaud, le grand manitou français de SQL Server, a eu besoin de 24 cœurs, de 16 Go de Ram pour créer et résoudre l'agenda, le tout en 18,04 minutes et en s'appuyant sur la bibliothèque Solver Foundation 1.2.

Au cœur de Visual Studio 2010 : Team Foundation Server

2e partie

Si le socle commun est toujours le Team Foundation Server, ses prérequis ont bien évolué. Concernant les systèmes d'exploitation, Windows 2003 Server est toujours supporté mais nous préférerons plutôt la version 2008. Aux oubliettes SQL Server 2005, maintenant seule la version 2008 est acceptée afin d'avoir des rapports plus beaux et plus de disponibilité (la licence étant comprise dans le produit, pourquoi s'en priver ? !).

Si IIS et le Database Engine de SQL Server sont des prérequis immuables il n'en est plus de même pour Analysis Service, Reporting Service et SharePoint !, qui se trouvent relégués (et ce n'est pas trop tôt) au rang de composantes optionnelles.

INSTALLATION DE TFS

Tous ceux qui ont goûté à l'installation (ou pire encore : la migration) d'un Team Foundation Server peuvent dire fièrement : « Je l'ai fait ! », ou le plus souvent ne rien dire du tout…

S'il est un des domaines où Microsoft devait faire des efforts, c'est bien celui-ci et nous pouvons dire que le cauchemar prend fin avec 2010. L'étape d'installation (ou de migration) a été découpée en deux : l'installation des binaires en premier, puis un assistant disponible à tous moments pour configurer ou migrer. Le challenge n'est plus le même qu'avant, mais on ne va pas s'en plaindre ! **(Fig. 1)**.

Une fois les binaires installés, le Configuration Wizard apparaît afin de choisir le type de configuration souhaitée, de la paramétrer et enfin de l'exécuter. Cet assistant permet d'installer une version Basic, Standard en déploiement sur serveur unique (80 % des cas), Advanced pour une configuration exotique, Application-Tier Only pour configurer un nouvel AT (souvenez-vous, TFS supporte le Native Load Balancing) et enfin Migrer une instance existante (2005 ou 2008). La vérification des prérequis ainsi que la cohérence des paramètres permettent une configuration plus fiable, bien joué Microsoft ! **(Fig. 2)**.

Fig.1

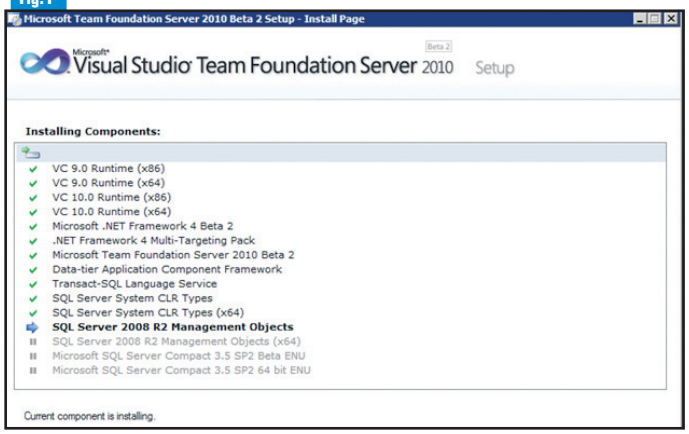
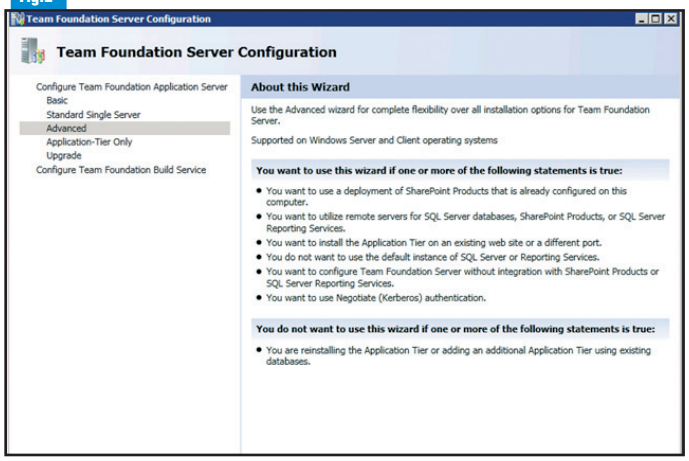


Fig.2



NOUVELLES FONCTIONNALITÉS DES WORK ITEMS

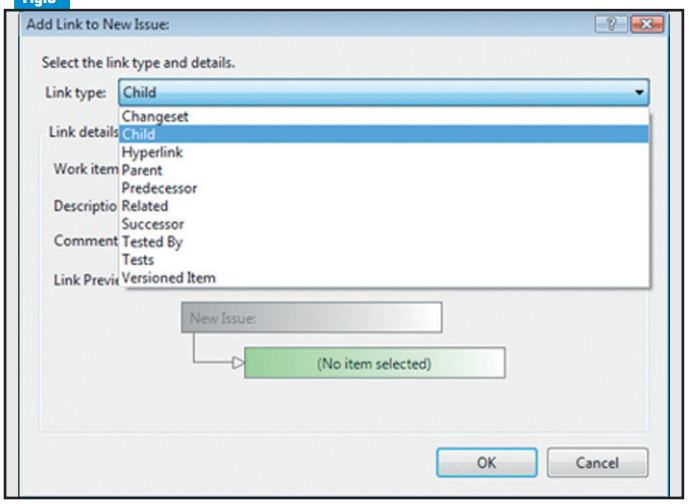
Les Work Items permettent de matérialiser des informations relatives au développement d'un projet. Cette brique déjà phare dans les éditions précédentes est revenue avec de nouvelles fonctionnalités qui étaient très attendues. Tout un article ne suffirait pas à les décrire, nous allons donc citer les plus marquantes.

Les liens typés

Sans conteste la fonctionnalité la plus attendue, les liens typés ont cruellement manqué aux éditions précédentes où on ne pouvait que créer un lien bidirectionnel et non versionné entre Work Items pour faire état d'une relation. Cette esquisse était, dans la vie de tous les jours, insuffisante.

Nous avons maintenant le choix entre plusieurs types de liens prédéfinis (Previous/Next, Parent/Child, Predecessor/Successor, etc.) et la possibilité d'en créer de nouveaux. Les possibilités sont diverses : on peut choisir la direction du lien (avant, arrière ou les deux), un type (défini par un namespace) et un nom. **(Fig. 3)**

Fig.3

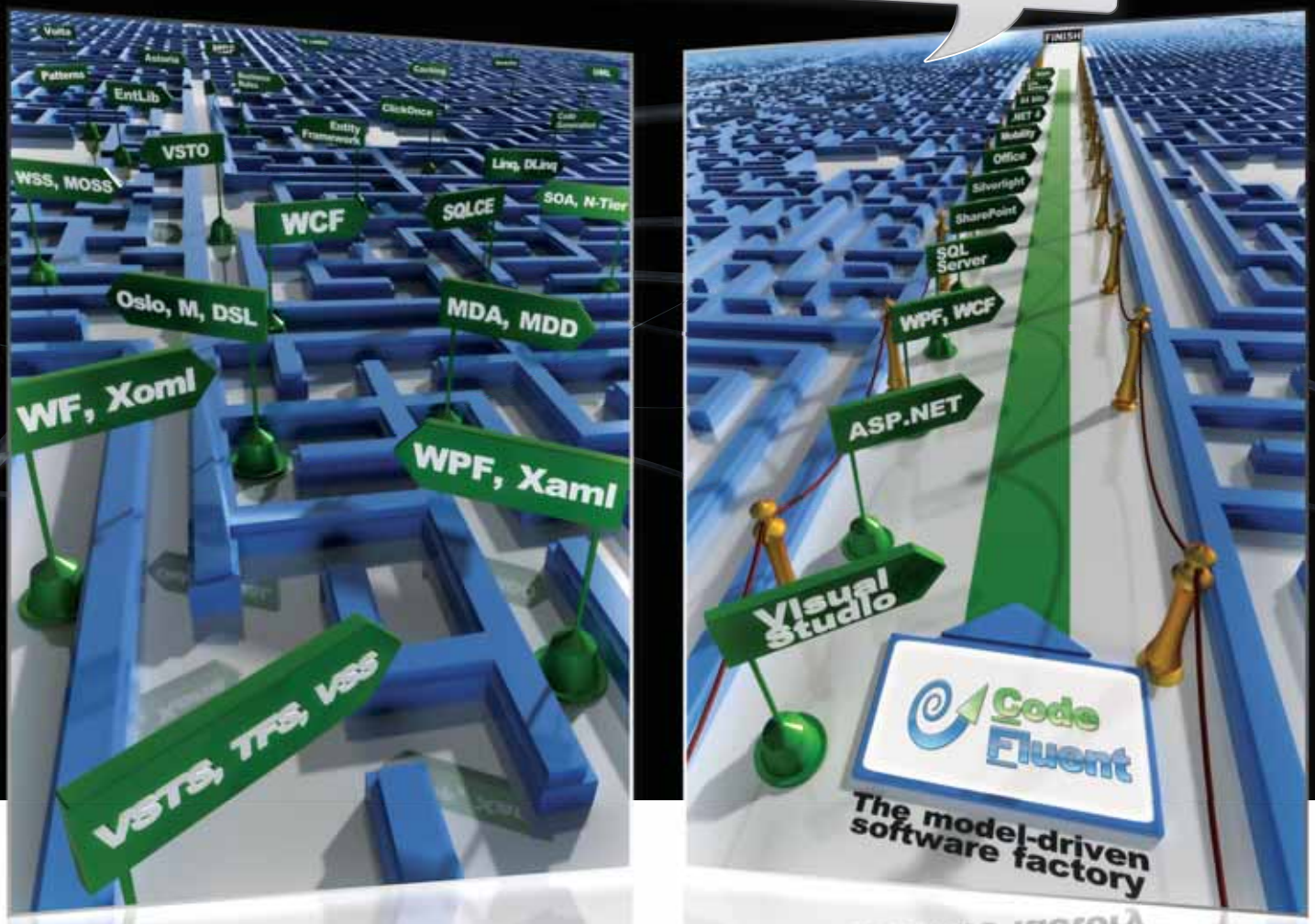


CODEFLUENT ENTITIES

La première fabrique logicielle .NET entièrement pilotée par les modèles

Session le 9 février à 14h30
aux **Microsoft Techdays** 2010

sans **avec**



Le plus court chemin vers les technologies .NET

CodeFluent Entities 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. L'utilisation de CodeFluent Entities vous assure **évolutivité**, **productivité**, **qualité** et **facilité de maintenance**.

SoftFluent
3 rue de la Renaissance - 92160 Antony
01 75 60 04 45 - sales@softfluent.com

idEES
L'innovation commence ici

Mise en version des liens

Les versions précédentes ne permettaient pas cette fonctionnalité, la relation entre les Work Items ne se conjuguait qu'au présent. Dans un monde où la traçabilité est reine, il fallait corriger ce problème. **(Fig. 4)**

Visualisation des Work Item liés

Un nouveau contrôle graphique de la fiche des Work Items permet de visualiser les Work Item liés selon des critères prédéfinis (type du lien, type de Work Item, nombre de niveaux à afficher, etc.). L'affichage lui-même est déterminé par avance parmi les trois types prédéfinis : Flat List, Direct Links et Hierarchy. Cette évolution est importante et a impacté tous les endroits où l'on manipule les work items pour qu'ils supportent un affichage hiérarchique.

TEAM PROJECT COLLECTIONS

Voici une nouveauté d'envergure dans la partie serveur : les Team Project Collections (TPC). Elles définissent une racine commune à un ensemble de Team Projects. Au final, les collections sont considérées comme des silos indépendants, comme si l'on avait plusieurs instances de serveurs différentes. L'analogie avec les collections de sites de SharePoint est volontaire. Un même serveur TFS pourra donc abriter plusieurs collections, et chacune pourra contenir plusieurs Team Projects. D'ailleurs, chaque TPC est mappée par défaut sur une collection de sites SharePoint (avec bien sûr une instance de site par projet). Les numéros de ChangeSets et de Work Items sont uniques dans une même collection. Un utilisateur ne peut donc se connecter qu'à une seule collection à la fois **(Fig. 5)** , une collection est en quelque sort un serveur logique. Les TPC apportent de la souplesse dans la gestion de l'infrastructure, puisqu'un même serveur peut abriter plusieurs collections, on peut fusionner plusieurs serveurs TFS en un seul, ou exporter des collections d'un même serveur vers d'autres serveurs afin de répartir la charge par exemple.

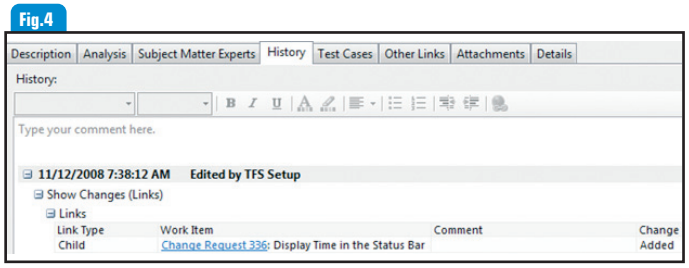
FERMES DE SERVEURS ET MONTÉE EN CHARGE

D'ailleurs, on ne peut plus vraiment dire « le » serveur TFS, on va désormais parler de fermes de serveurs. En effet, une des faiblesses de TFS a été corrigée avec le support du load balancing sur la couche applicative du serveur (AT). Une même TPC peut ainsi être hébergée sur plusieurs serveurs en concurrence. Dernier point, les bases de données ont évolué pour aboutir sur un modèle encore une fois volontairement inspiré de SharePoint. Elles ont été regroupées en une base unique, avec comme principe une base par TPC. Par défaut, on retrouve ainsi :

- Tfs_DefaultCollection : La base de la TPC nommée DefaultCollection
- Tfs_Warehouse : La base du Warehouse de tous les projets
- Tfs_Configuration : La base de configuration de la ferme TFS (par analogie avec la base de configuration de SharePoint)
- Les bases de Reporting Services
- Les bases de SharePoint

Enfin, il est maintenant possible de monter en charge très naturellement du point de vue SQL Server en choisissant des serveurs différents pour héberger les différentes bases des TPC.

Fig.4



CONSOLE D'ADMINISTRATION

Vous l'avez compris, les combinaisons en termes de topologie se multiplient. Afin d'y voir clair dans tout cela et surtout afin de simplifier un nombre important d'opérations qui jusqu'à présent nécessitaient des successions de lignes de commandes, nous sommes désormais dotés d'une vraie console d'administration **(Fig. 6)** pour TFS qui permet très facilement :

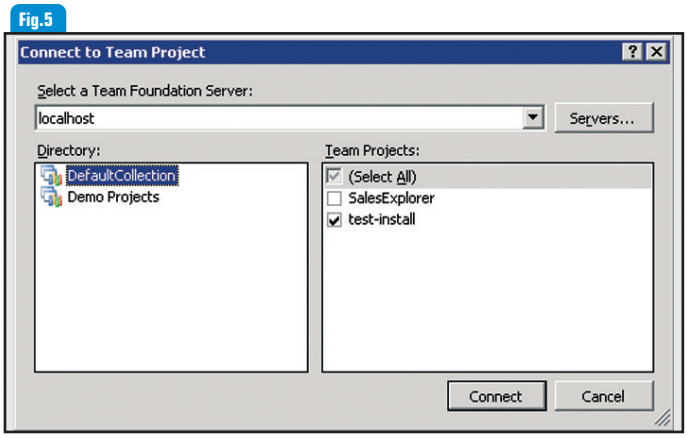
- De changer les paramètres de sécurité, les comptes de services de la couche applicative.
- D'administrer les TPC (création, suppression, sécurité, import/export de TPC).
- De gérer les applications Web SharePoint et leurs comptes de services.
- De gérer les bases du Warehouse et Reporting Services.
- Configurer Lab Management.
- Configurer les contrôleurs et agents de build.
- Accéder directement aux divers logs.

La console affiche bon nombre d'informations comme l'état de tous les services, les URL, les noms des bases, des comptes et des machines impliquées. On a donc une excellente visibilité de l'état et de la topologie d'une ferme TFS, ce qui facilite grandement l'administration de l'infrastructure.

MISE À JOUR DES POWER TOOLS

Concernant les fameux Power Tools, cet ensemble d'outils fournis par Microsoft en avance de phase sur la prochaine version de TFS, on retrouve maintenant une partie des fonctionnalités directement dans le produit de base. Pour faire bref, Visual Studio 2010 implémente nativement le build notifier qui sert pour les Gated-Checkins, et les améliorations sur les labels et sur les branches rendent également obsolètes certains outils en ligne de commande. L'éditeur de Process Template a évolué principalement pour supporter la nouvelle structure hiérarchique des Work Items et pour rajouter le support des types de liens entre ces derniers – un effort important de stabilisation a également été fait sur cet outil. D'autre part, le Best Practice Analyzer a dû être en bonne partie réécrit puisque bon nombre de contraintes ont été levées d'un point de vue architecture (Il n'est pas complètement terminé à l'heure actuelle). Pour le reste des Power Tools, rien de vraiment nouveau, ils ont simplement été mis à jour pour fonctionner avec la dernière version de TFS. Attention, à chaque version ses outils, ne comptez pas éditer un Process Template sur un TFS 2008 avec Visual Studio 2010 et ses Power Tools.

Fig.5



INTÉGRATION AVEC SHAREPOINT

La présence d'un portail SharePoint est devenue optionnelle, mais on va voir que l'interface de navigation a été repensée et la création de nombreuses WebParts dédiées à TFS donne une vraie valeur ajoutée au portail d'un Team Project, qui disons-le, ne présentait jusqu'à présent que peu d'intérêt en dehors des bibliothèques de documents. TFS 2010 sera toujours livré avec Windows SharePoint Services 3.0, mais fonctionnera également avec MOSS 2007 et Windows SharePoint Server 2010. Il y aura même plus de tableaux de bord et plus de rapports sur ces 2 derniers car les équipes Microsoft ont été bien plus productives à produire des rapports en utilisant Excel Services qu'avec Reporting Services. On dispose maintenant d'une panoplie de vraies WebParts SharePoint dédiées à TFS qui permettent :

- L'affichage des derniers builds avec de nombreuses options de filtres, en cliquant sur un build on en voit le détail et on peut l'effacer ou changer son niveau de qualité.
- L'affichage des Work Items résultant d'une requête, on accède à toutes les requêtes existantes, et on peut personnaliser les colonnes. De là, on peut éditer en détail chaque Work Item.
- D'afficher un décompte des Work Items par catégorie.
- De lister les derniers ChangeSets archivés avec des options de filtre (par chemin par exemple).
- De créer des nouveaux Work Items.

Par contre l'affichage de rapports Reporting Services se fait toujours avec la WebPart générique Page Viewer. C'est décidément Excel qui a le vent en poupe pour la partie reporting. Vous pouvez voir sur la **Fig. 7** que le Project Dashboard donne une visibilité immédiate sur le projet. On peut bien sûr le customiser comme bon nous semble. Mieux, chaque utilisateur dispose aussi de son propre Dashboard personnalisable à volonté. Par défaut il nous indiquera nos propres bugs et tâches. On peut maintenant dire que le portail est bien le meilleur endroit pour commencer sa journée : voir les derniers builds, l'état des Work Items, et opérer quelques modifications sur les tâches en cours avant de coder.

Fig.6

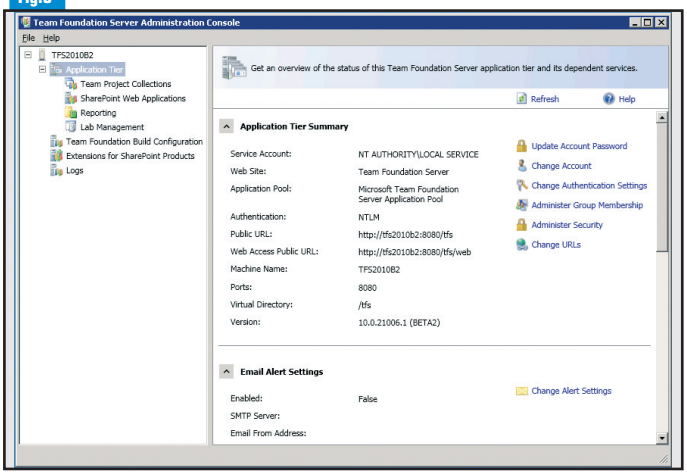
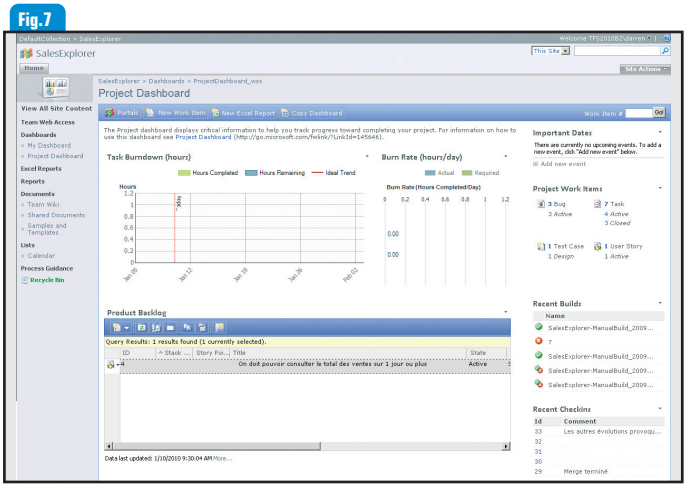


Fig.7



TEAM WEB ACCESS

Team System Web Access devient Team Web Access et son rôle est toujours de fournir un accès client très complet à TFS : gestion des Work Items, Rapports, Documents, Builds, consultation des sources et ChangeSets. La dernière version est complètement intégrée au produit (plus d'installation séparée) mais reste volontairement indépendante du portail SharePoint. Vous remarquerez le lien vers TWA sur le Project DashBoard, il en est de même en sens inverse. Quant au site Work Item Web Access, le site d'accès sans CAL aux Work Items, il est maintenant disponible à la même URL que TWA qui détecte si l'utilisateur appartient au groupe global « Work Item Only View Users ».

LA REFONTE DE TEAMBUILD

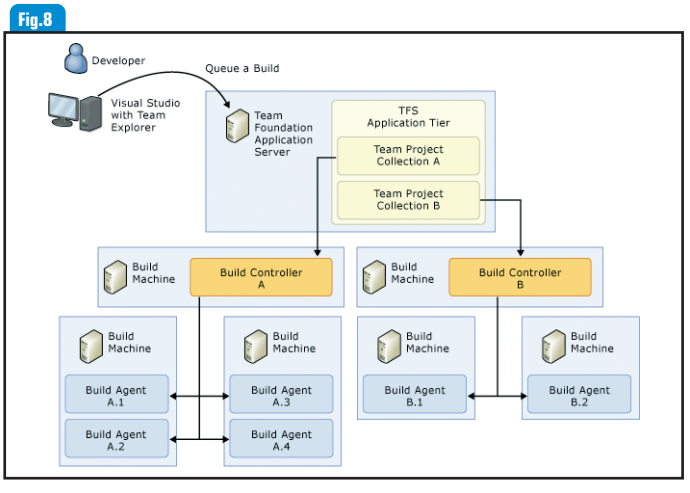
Team Foundation Build est l'outil de TFS permettant de compiler, tester et déployer les applications de manière automatique. La version 2010 de TeamBuild a été entièrement revue : l'architecture des services est maintenant distribuée, le moteur est basé sur Workflow Foundation 4.0 et la configuration des services peut se faire via une vue dédiée dans la nouvelle interface d'administration de TFS 2010. En plus de ces mises à jour, on retrouve une nouvelle version de MSBuild ainsi que des nouveautés au niveau des définitions de build.

Nouvelle architecture

Afin de pouvoir être utilisé aussi bien par les petites entreprises que par les grandes, l'architecture de TeamBuild a évolué vers une architecture distribuée composée de contrôleurs et d'agents.

Les agents sont des services s'occupant des tâches « lourdes » telles que la récupération des sources, la compilation et l'exécution des tests. Ils sont gérés par un contrôleur qui exécutera les tâches nécessitant moins de ressources comme la mise à jour du nom du build, le log et bien entendu la distribution des build vers les agents qu'il gère. Les contrôleurs orchestrent ainsi les builds de la collection de projets d'équipe à laquelle ils sont rattachés. En modifiant le nombre de contrôleurs et d'agents, il est possible de dimensionner

Fig.8



son processus de build à ses besoins. La **Fig. 8** montre un exemple de topologie. Afin de permettre la mise en place d'agents spécialisés, il est possible de leur attribuer des tags. On pourra alors dans la définition d'un build indiquer le ou les tags que devra avoir l'agent qui prendra en charge la compilation et les tests. Un exemple d'utilisation serait la mise en place d'une machine de build spécifique pour les projets Biztalk (une machine sur laquelle Biztalk a été installé par exemple). En ajoutant un tag « Biztalk » aux agents de la machine on pourra ensuite spécifier dans les définitions de build des projets Biztalk qu'il faut utiliser un agent avec ce tag, le contrôleur se chargera de trouver le bon agent.

Nouveau moteur : Workflow Foundation 4.0

L'autre grosse nouveauté de TeamBuild 2010 est l'abandon de MSBuild pour les scripts au profit de Workflow Foundation 4.0 (les projets et solutions restent, eux, en MSBuild). Le processus de build est maintenant défini sous forme de workflow séquentiel avec un ensemble d'activités spécifiquement développées pour le build. On retrouve donc en plus des activités de gestion de flux (If, Switch, ForEach,…) des activités telles que :

- AgentScope : une séquence d'activités s'exécutant sur un agent.
- CreateWorkspace, GetWorkspace, DeleteWorkspace : pour gérer les workspace.
- MSBuild : pour lancer un script MSBuild.
- MSTest : pour lancer les tests.
- WriteBuildMessage, WriteErrorMessage, WriteWarningMessage : pour les logs.

On retrouve aussi des activités spécifiques pour le déploiement sur des environnements de test pour MTLM.

Microsoft fournit par défaut trois modèles de workflow de build :

- DefaultTemplate : le template de base.
- UpgradeTemplate : un template permettant de réutiliser ses anciens scripts TeamBuild écrits en MSBuild.
- LabDefaultTemplate : un template de base pour compiler, déployer et tester son application avec MTLM.

Ces modèles peuvent être utilisés facilement par différents builds car Workflow Foundation apporte des possibilités de paramétrage beaucoup plus riches que MSBuild. Chaque modèle va donc exposer un ensemble de paramètres dont les valeurs par défaut pourront être spécifiées au niveau de la définition de build. Ces valeurs pourront ensuite être modifiées ponctuellement au moment du déclenchement du build. C'est bien plus pratique que les paramètres de type ligne de commande que l'on passait directement à MSBuild !

DÉFINITION DE BUILD

Les définitions de build ont été revues afin de prendre en compte les modèles de type workflow et permettre de paramétrer ceux-ci comme le montre la **Fig. 9**. On retrouve aussi deux nouveaux types de build, le Gated Check-in et le Private Build. Le Gated Check-in est un build qui sera lancé lors de chaque check-in et ne réalisera l'archivage effectif des sources que si le build s'est terminé avec succès. D'un point de vue technique, les modifications du développeur sont mises en étagère et lors de la récupération des sources, ces fichiers seront fusionnés avec les dernières sources. En cas d'échec l'utilisateur pourra, via l'explorateur de build, récupérer son code afin de le corriger. Le Private build utilise le même mécanisme mais c'est l'utilisateur qui le lance et définit l'étagère à utiliser. Cela permet de valider que les modifications apportées seront bien compilées et testées le cas échéant dans l'environnement de TeamBuild.

Enfin, la gestion des données supprimées avec le build a été améliorée afin de permettre de sélectionner plus finement ce qui doit ou ne doit pas être supprimé parmi les éléments suivants :

- Les fichiers générés.
- Les résultats des tests.
- Le label.
- Les symboles.

MSBUILD 4.0

Bien que TeamBuild 2010 utilise Workflow Foundation, des ajouts ont aussi été apportés à MSBuild, celui-ci continuant d'être le langage de script utilisé pour les projets et solutions. On retrouvera dans MSBuild 4.0 :

- Des fonctions sur les propriétés permettant de manipuler des chaînes de caractères, de faire des opérations mathématiques, d'utiliser des expressions régulières,…
- La possibilité de créer des tâches inline dont le code .Net sera contenu directement dans le script.
- L'ajout des attributs *BeforeTarget* et *AfterTarget* à l'élément Target. Ces attributs permettent de spécifier l'ordre d'exécution des Target.
- Le support des projets Visual C ++.

Fig.9

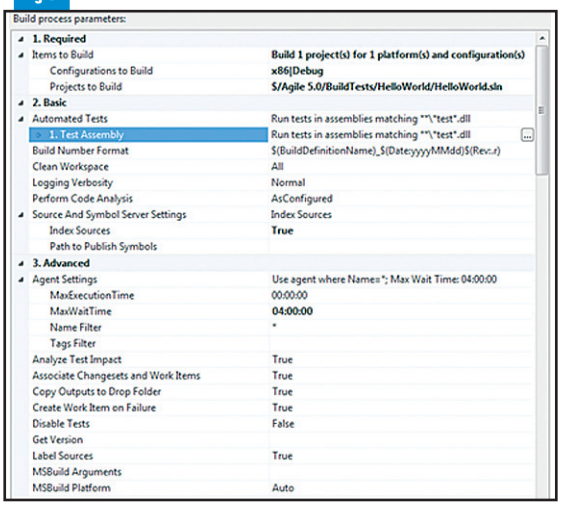
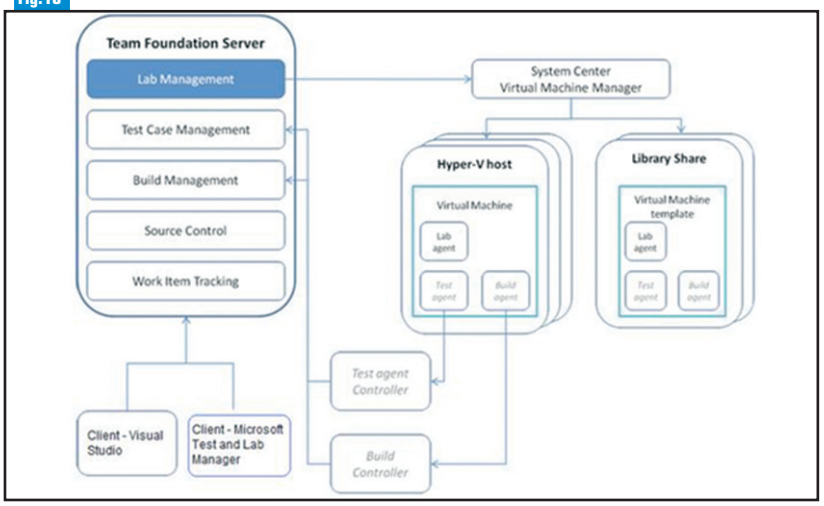


Fig.10



VISUAL STUDIO TEAM LAB MANAGEMENT

Le développement de la plupart des applications nécessite de mettre en œuvre des environnements de test afin de qualifier la build d'une version en cours, un patch ou une localisation spécifique de l'application. Pour répondre à ce besoin, la plupart des équipes emploient des solutions plus ou moins artisanales et trop souvent le temps nécessaire pour la mise en place d'un tel environnement est « trop de temps ». En effet, cette mise en place est une tâche relativement manuelle et la procédure est quasiment identique entre deux environnements, ce qui rend le tout très rébarbatif.

Team Lab Management (Team Lab pour les intimes), nouvel arrivant dans la gamme des composants serveurs de Visual Studio 2010, est la solution proposée par Microsoft. Il s'agit d'une plate-forme qui permet de gérer des environnements virtuels qui seront consommés lors du développement du projet (programmation, campagne de test, mise en pré-production, etc.)

La solution de Microsoft

Encore une fois, Microsoft s'appuie sur ses produits maison pour développer cette nouvelle brique. Team Lab est de loin le composant serveur le plus complexe à mettre en œuvre car il vous faudra respecter un bon nombre de prérequis : l'utilisation d'un Active Directory, l'installation de System Center Virtual Machine Manager (qui lui-même aura besoin d'un SQL Server) et pouvoir adresser un Windows 2008 Server avec le rôle Hyper-V.

Pour ceux qui ont des compétences en infrastructure système et réseau, vous aurez compris que Microsoft s'appuie sur Hyper-V et System Center Virtual Machine Manager (SCVMM) pour la gestion des environnements virtuels. La **Fig. 10** montre un schéma d'architecture de Team Lab. Bien qu'on dise souvent qu'un schéma vaut mieux qu'un long discours, dans le cas présent un discours s'impose quand même. Team Lab servant à gérer des environnements de Test il est normal que celui-ci s'inscrive au-dessus des principales briques de TFS (notamment les Work Items, le Build et Test Case Management). Il va encapsuler SCVMM pour gérer la création/modification/suppression de réseaux et machines virtuelles pour la mise en place d'un environnement Team Lab. En pratique vous n'aurez donc pas à connaître SCVMM pour vous servir de celui-ci. Votre environnement de test aura son propre réseau virtuel privé et pourra être composé de plusieurs machines virtuelles, par exemple un serveur de données (hébergeant SQL Server), un 2e serveur pour la couche applicative web (avec un IIS) et deux autres machines pour simuler un client sous Windows 7 FR et un autre sous Windows 7 US (afin de tester les problématiques liées à la localisation et aux paramètres régionaux). Ces quatre machines virtuelles devront avoir au moins le Lab Agent d'installé afin de pouvoir dialoguer avec Team Lab. Le Test Agent devra être installé si vous voulez pouvoir exécuter des tests manuels et le Build Agent si vous voulez gérer la compilation.

Utilisation de Team Lab

Plusieurs scénarios d'utilisation sont possibles, nous allons parler de deux d'entre eux.

Création d'un template d'environnement

Comme nous le disions précédemment, à chaque fois que l'on veut tester une application, il faut mettre en place un environnement dédié, sa création et mise en œuvre sont une étape manuelle qui est, la plupart du temps, prédéfinie. Team Lab offre la possibilité de créer un template d'environnement afin d'automatiser cette tâche **(Fig. 11)**.

Pour faire court, le template modèle sera nommé et stocké dans SCVMM, pourra cibler n'importe quel serveur Hyper-V ou un en particulier, puis il faudra créer des templates modèles de machines virtuelles correspondant aux machines à instancier lors de la création de l'environnement, leur donner un rôle et des compétences (entre autres).

Un testeur fonctionnel pourra plus tard instancier un environnement pour pouvoir dérouler une campagne de test et ce sans tracas.

Intégration de Team Lab dans la chaîne d'intégration continue

Le summum de l'intégration continue : « End-to-end daily build » permet de compiler, déployer et enfin tester une application. Si la première étape commence à être maîtrisée, les deux autres nécessitent plus de travail (et d'outils !)**(Fig. 12)**.

L'utilisation d'un environnement virtuel est primordiale car l'enjeu dans ce scénario est de pouvoir restaurer à un état propre l'environnement de test avant chaque déploiement de l'application (la restauration de snapshot de machine virtuelle sera utilisée), puis créer un nouveau snapshot après le déploiement de l'application afin de pouvoir exécuter les tests plusieurs fois si nécessaire.

La rapidité d'exécution et la simplicité de Team Lab permettent de mettre en place ce scénario qui aurait auparavant demandé de nombreux développements spécifiques et un travail colossal !

TESTS DE MONTÉE EN CHARGE

Il y a du nouveau en ce qui concerne le modèle de licence des agents de tests de montée en charge : au lieu de payer par agent de test, on pourra désormais déployer autant d'agents que l'on souhaite, mais ceux-ci seront limités à une quantité de 1 000 utilisateurs virtuels. Il faudra alors acheter des « packs » de 1 000 utilisateurs virtuels supplémentaires pour pouvoir faire monter la charge.

Ce nouveau modèle de licence est une bonne nouvelle puisqu'il apporte de la souplesse. La licence porte désormais sur le contrôleur de tests qui pilote les agents afin de générer la charge voulue. Microsoft assure que les prix, déjà moins chers que la concurrence, ne seront pas plus hauts qu'ils ne l'étaient avec TFS 2008 et donc toujours bien inférieurs à la concurrence. D'ailleurs, le modèle de licence

Fig.11

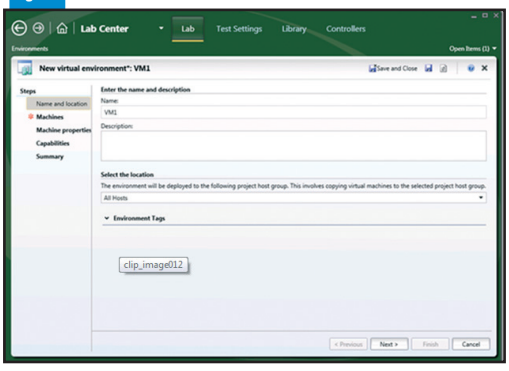
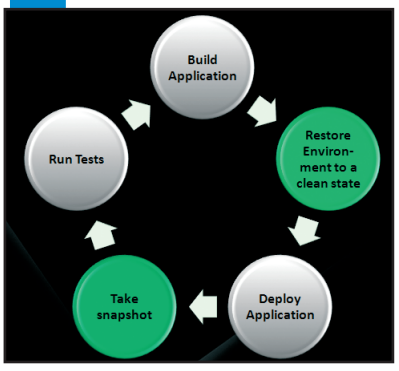


Fig.12



étant désormais aligné sur celui de ses concurrents, il sera possible de comparer directement les différentes offres.

LES MÉTHODOLOGIES DE TFS 2010

Avec Team Foundation Server 2010, Microsoft propose toujours les deux mêmes modèles de processus qu'auparavant : le modèle Agile et le modèle CMMI. Ces modèles ont bien sûr évolué et sont passés en version 5.0. Comme pour les versions précédentes, ces modèles de processus contiennent :

- Un ensemble de définition de type de work items.
- Un guide complet de processus.
- Un template de site Sharepoint.
- Des requêtes de work items.
- Des rapports.

Parmi ces éléments, la première des nouveautés que l'on remarquera est que le guide de processus est maintenant en ligne. Il sera donc possible de profiter automatiquement de toutes les mises à jour et corrections apportées par Microsoft.

Fig.13

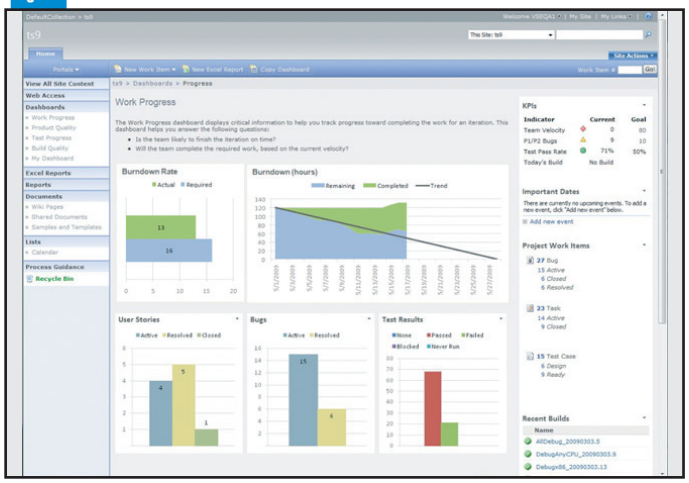


Fig.14

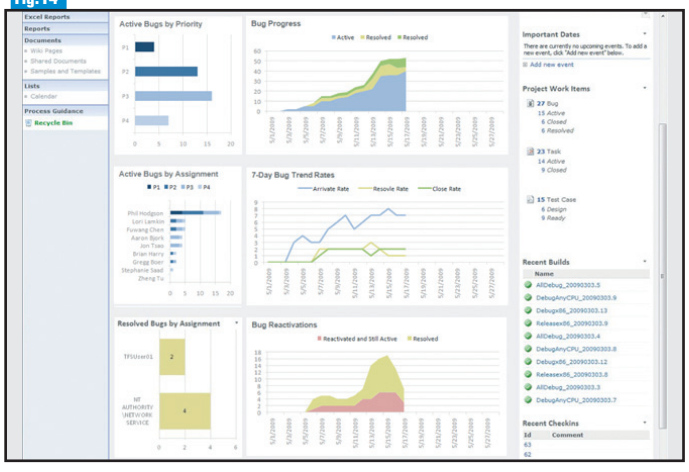
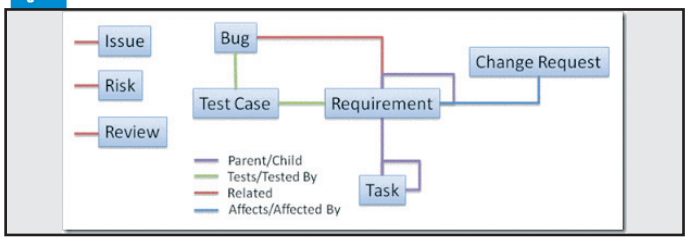


Fig.15



Agile 5.0

Le nouveau modèle de processus Agile 5.0 est clairement orienté vers la méthodologie SCRUM. Le guide de processus explique cette méthodologie ainsi que les différents éléments qui la composent tel que les différents rôles, les types de documents à fournir, les réunions, les sprints… On retrouve bien entendu des work items spécifiques tel que le « User Story » et leurs contenus ont eux aussi été adaptés. Concernant le site SharePoint, et si vous avez MOSS, vous aurez droit à 4 « Dashboards » supplémentaires qui compilent des rapports Excel par thèmes (Work Progress **(Fig. 13)**, Product Quality **(fig. 2)**, Test Progress et Build Quality). Quoi qu'il en soit, vous avez toujours accès aux mêmes rapports individuellement avec les documents Excel fournis dans les bibliothèques de documents.

On retrouve aussi des Workbooks permettant de piloter un projet via Excel. Ces Workbooks sont liés directement à TFS et vont permettre de gérer, par exemple, un sprint via les fonctionnalités suivantes :

- Saisie des tâches (en gérant l'aspect hiérarchique (Iteration Backlog)).
- Paramétrage du sprint (date de début et de fin, area et itération de TFS à utiliser).
- Définition des interruptions comme les congés.
- Gestion de la capacité sous forme de graphique mis à jour automatiquement via les informations présentes dans TFS **(Fig. 14)**.

CMMI 5.0

Contrairement au modèle Agile qui a énormément changé, le modèle de processus CMMI 5.0 n'a pas beaucoup évolué. Pour la plus grande partie il s'agit d'une mise à jour pour intégrer les nouvelles fonctionnalités de Team Foundation Server 2010. On retrouve les évolutions suivantes :

- La conformité avec CMMI 1.2.
- De nouveaux types d'exigences.
- Des rapports améliorés.
- L'intégration avec les liens hiérarchiques entre work items.
- L'intégration avec toutes les fonctionnalités de tests.

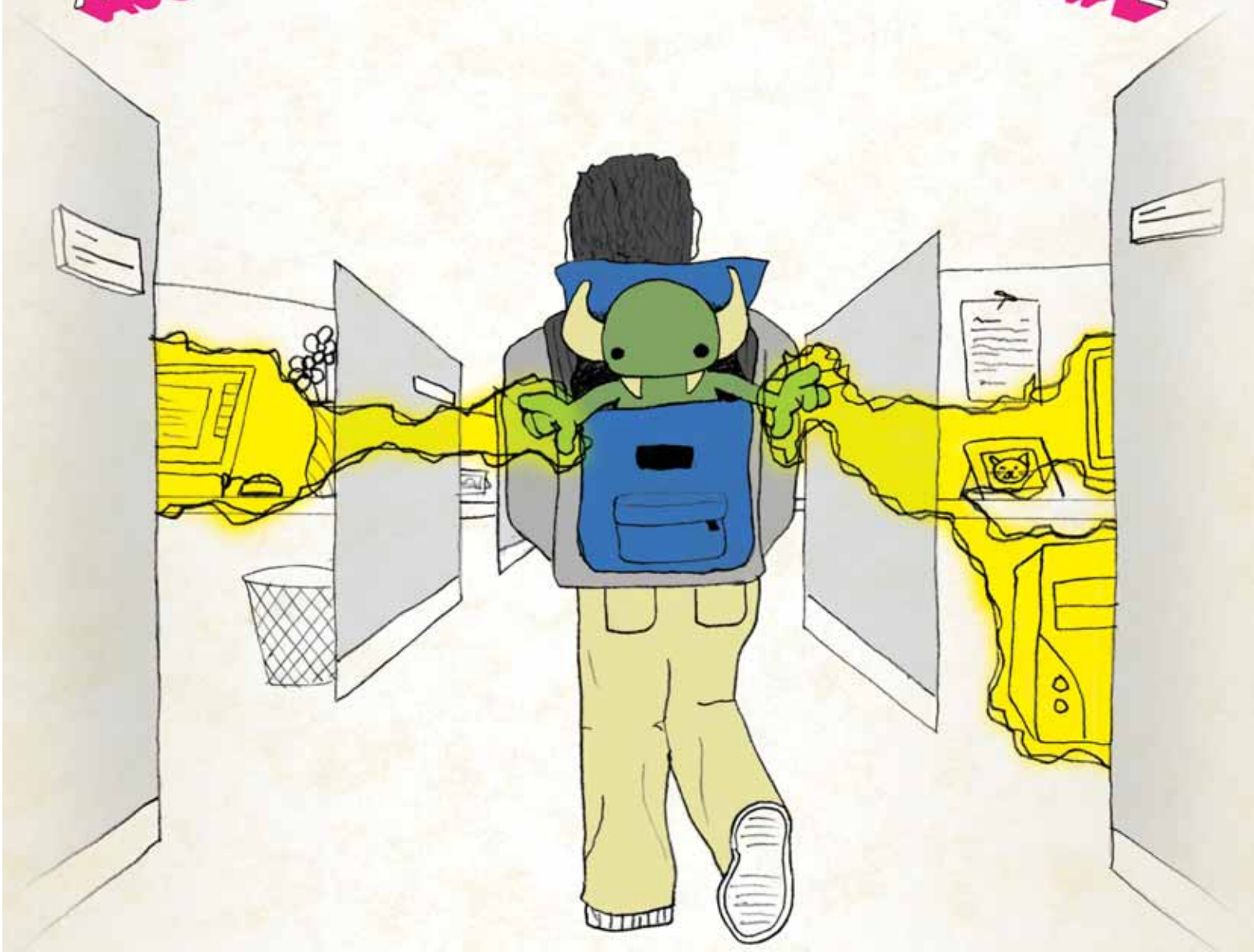
Le site SharePoint a lui aussi été mis à jour comme pour le modèle Agile 5.0 et proposera donc des dashboard ainsi que des web parts spécifiques à TFS. Comme vu précédemment, les work items peuvent maintenant être liés entre eux par des liens typés. Ce nouveau mécanisme permet, quelle que soit la méthodologie, de simplifier la gestion des tâches en les ordonnant et en définissant des dépendances **(Fig. 15)**. Bien entendu il existe d'autres modèles de processus comme par exemple le modèle SCRUM For TeamSystem (http://scrumforteamsystem.com/default.aspx).

■ Loïc Baumann
Architecte Logiciel et expert en ALM chez Winwise ainsi que MVP Team System. Son Blog : http://loicbaumann.org

■ Guillaume Rouchon
Consultant/Formateur .NET et expert ALM chez Winwise. Son Blog : http://blog.qetza.net

■ Vincent Labatut
Consultant/Formateur .NET et expert ALM chez Winwise. Son Blog : http://blogs.codes-sources.com/vlabz

AJOUTEZ DE L'EXTRÊME À VOTRE ÉQUIPE



Insufflez à votre équipe le pouvoir de créer des interfaces utilisateurs extrêmement fonctionnelles, d'une grande facilité d'utilisation et possédant "le facteur WOW!" avec NetAdvantage® dans votre boîte à outils de développement .NET. En disposant des data grids les plus puissantes et les plus rapides sur le marché pour Windows Forms, ASP.NET, Silverlight et WPF, Vous aurez l'impression d'avoir la force de 10 développeurs sur chaque bureau.
Allez sur infragistics.com/killerapps découvrir comment vous et votre équipe pouvez commencer à créer vos propres Killer Apps .

Infragistics®
KILLER APPS. NO EXCUSES.

Infragistics Ventes France N° Vert 800 231 8588
Infragistics Europe Ventes +44 (0) 800 298 9055
Infragistics India +91-80-6785-1111

Programmer la table Surface

Microsoft a débuté ses travaux sur les nouvelles interactions homme-machines en 2001 et plusieurs prototypes furent réalisés, dont une table IKEA en 2003, avant d'obtenir le produit Microsoft Surface actuel. La commercialisation de la table tactile Microsoft Surface a commencé aux États-Unis en 2007 et est arrivée en Europe début 2009.

Surface se présente sous la forme d'une table de 108 cm de largeur, 69 cm de profondeur et 54 cm de hauteur pour un poids de 90 kg. En choisissant un concept de table, Microsoft redéfinit un certain nombre de notions ; ainsi les applications n'ont plus de haut et de bas, ne sont plus définies pour être utilisées par un seul utilisateur devant son écran mais au travers d'une interface à 360° où plusieurs personnes peuvent utiliser les applications, visualiser du contenu, voire collaborer. La reconnaissance des contacts s'effectue à l'aide d'un système de vision basé sur des caméras avec éclairage direct par LED infrarouge. L'image d'une résolution maximale de 1024 par 768 pixels est affichée par un projecteur DLP XGA. La table intègre un ordinateur doté de 2 Go de RAM, un processeur Intel Core 2 Duo cadencé à 2,13 GHz et un disque dur de 250 Go. Le système d'exploitation qui gère la table est Windows Vista Professionnel avec SP1 et notamment .NET framework 3.5 SP1, Microsoft Surface Runtime 1.0 SP1 et un SDK spécifique qui permet de pleinement tirer parti des capacités multi-touch du produit.

LES CONCEPTS

Multi-touch

La table Surface est capable de reconnaître et d'interpréter 52 points de contacts simultanés grâce aux caméras infrarouges. Le produit est donc résolument multi-touch et il est ainsi possible, par exemple, d'agrandir des photos en utilisant les mains, même si plusieurs contacts par main sont établis. Le parcours du globe en utilisant Virtual Earth est un exemple concret d'utilisation des fonctionnalités multi-touch du produit.


Fig.1

Multi-utilisateur

L'écran 30 pouces permet à plusieurs utilisateurs de manipuler simultanément les objets virtuels ou concrets affichés ou disposés sur la table ; les contenus affichés peuvent être partagés ou différents en fonction de la programmation des applications.

Interface à 360 degrés

Le fait que le produit se présente sous la forme d'une table change beaucoup les principes de programmation et explique pourquoi on ne peut pas simplement « porter » une application Microsoft Vista pour la table Microsoft Surface mais qu'il est aussi nécessaire de la « redesigner ». Ainsi, contrairement à une application pour ordinateur classique, il n'y a plus de haut et de bas et la visualisation se fait à l'horizontale ; les utilisateurs doivent avoir accès à l'interface et doivent être en mesure de visualiser les informations où qu'ils soient autour de la table. C'est pourquoi lorsqu'un contact est détecté sur la table, le SDK indique les coordonnées du contact (abscisse X et ordonnée Y) mais aussi l'orientation. L'orientation des applications est importante pour l'expérience utilisateur et Microsoft a notamment transposé ce concept dans le menu de présentation des applications : en fonction du point d'accès (voir le point numéro 5 sur le schéma ci-dessous) choisi par l'utilisateur, le menu et l'application choisis s'orienteront différemment. [Fig.2]

Reconnaissance d'objets

La table, à l'aide de ses caméras infrarouges, est capable de reconnaître des objets soit par des algorithmes spécifiques de reconnaissance d'objets soit à l'aide de raccourcis sous forme de tags 2D de Microsoft [Fig.3 et 4]. Les applications sont ainsi capables de réagir sur détection des objets ou des tags. Ces tags sont des tags visuels et ne contiennent pas de puces RFID afin d'identifier précisément la position de l'objet sur la table.

L'exemple de la figure 5 illustre l'affichage d'un menu sur détection d'une carte de fidélité équipée d'un tag 2D.

LES OUTILS DE DÉVELOPPEMENT NÉCESSAIRES

Visual Studio 2008

Le développement des applications pour la table Microsoft Surface se fait en WPF (ou en XNA) et en XAML ; il est donc nécessaire d'utiliser Visual Studio 2008. La création de l'interface graphique peut se faire en XAML et la gestion des événements en C# ou Visual Basic.

Expression Blend

Expression Blend s'avère fort pratique pour concevoir les éléments

Fig.2

graphiques et les animations qui seront alors exportés en XAML et intégrés au projet Visual Studio.

Le SDK

Le Software Development Kit de la table Surface est en version 1.0 SP1 depuis mai 2009 et s'intègre directement dans Visual Studio 2008. Au lancement de Visual Studio 2008, il suffit de choisir « Nouveau Projet » puis dans le type de projet « Surface » et « v1.0 ». Le développement peut se faire en WPF ou en XNA. Parmi les fichiers automatiquement créés pour le nouveau projet, le fichier XAML décrivant les éléments graphiques de l'application contiendra le code suivant :

```
< s : SurfaceWindow x : Class = "Surface_Magazine_Programmez.
 SurfaceWindow1"
xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns : x = "http://schemas.microsoft.com/winfx/2006/xaml"
xmlns : s = "http://schemas.microsoft.com/surface/2008"
Title = "Surface_Magazine_Programmez"
>
  <s : SurfaceWindow. Resources >
    <ImageBrush x : Key = "WindowBackground" Stretch = "None" Opacity
 = "0.6" ImageSource = "pack ://application :,,,/Resources/
WindowBackground.jpg"/>
  </s : SurfaceWindow. Resources >

  <Grid Background = "{StaticResource WindowBackground}" >

  </Grid >
</s : SurfaceWindow >
```

Afin de tirer pleinement parti des fonctionnalités multi-touch du produit, Microsoft a développé de nouveaux contrôles dédiés et quasiment tous préfixés par Surface : SurfaceButton, SurfaceCheckbox etc… Tous ces contrôles supportent nativement le multi-touch et les contacts multiples avec des remontées d'événements pouvant indiquer combien de contacts simultanés sont sur le contrôle. Le SDK de Surface fournit aussi des outils forts utiles pour simplifier et accélérer le développement des applications pour la nouvelle plateforme Surface. Parmi ces outils, on retrouve notamment des exemples divers et qui illustrent l'utilisation des contrôles spécifiques Surface mais aussi une application de génération des Identity Tag (Fig.6), le simulateur de Surface (très important pour tester les applications sur un PC), un outil de stress d'applications et un programme de gestion de l'économiseur d'écran de la table « Water configuration ».

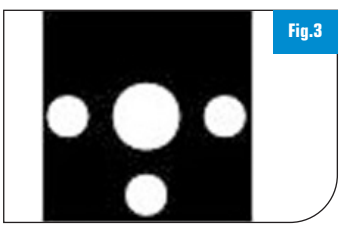


Fig.3

Byte Tag

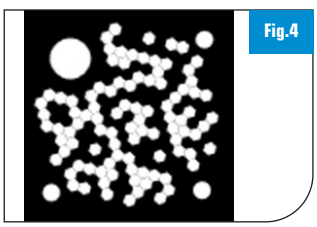


Fig.4

Identity Tag

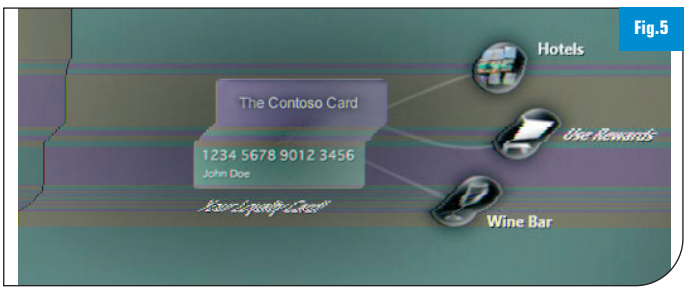


Fig.5

Le simulateur

Le simulateur est un outil indispensable pour développer sur un PC et pour tester ses applications avant installation et tests sur une table Microsoft Surface. Pour l'utiliser vous devez disposer de Microsoft Vista Professionnel (au moins) et d'un écran dont la résolution est au moins égale à 1280 * 960. L'outil remplit fort bien sa tâche et va permettre de simuler tout type de contacts sur la table Surface et même de simuler plusieurs contact simultanés ; ainsi si vous appuyez sur les deux boutons de la souris simultanément, vous créez un contact permanent et vous pouvez alors cliquer à un autre endroit pour générer un autre contact, vous avez alors le loisir de zoomer, d'étendre ou de diminuer le contrôle et son affichage. Par exemple en appuyant sur le bouton n° 2, la souris va simuler un contact de doigt ; en appuyant sur le bouton n° 4, la souris va simuler un tag 2D de type Byte Tag dont la valeur sera celle indiquée par le champ n° 5. Microsoft nous a donc fourni un outil riche et très utile afin de ne pas avoir à tester systématiquement son application sur la table Surface ; néanmoins, il faut impérativement effectuer des tests régulièrement sur Surface car le fait que le produit soit à l'horizontale, contrairement à un écran de PC peut changer la vision des choses surtout si l'application prend en compte l'orientation des contacts. Enfin, la validation de l'application devra obligatoirement se faire sur la table Microsoft Surface. [Fig.7]

LES TAGS 2D POUR SURFACE

Les Byte Tags

Les valeurs de ces tags sont codées sur un octet et il y a donc 256 valeurs possibles (de 0x00 à 0xFF). Les tables Microsoft Surface sont livrées avec 2 jeux de Byte Tags autocollants imprimés (voir Fig.3).

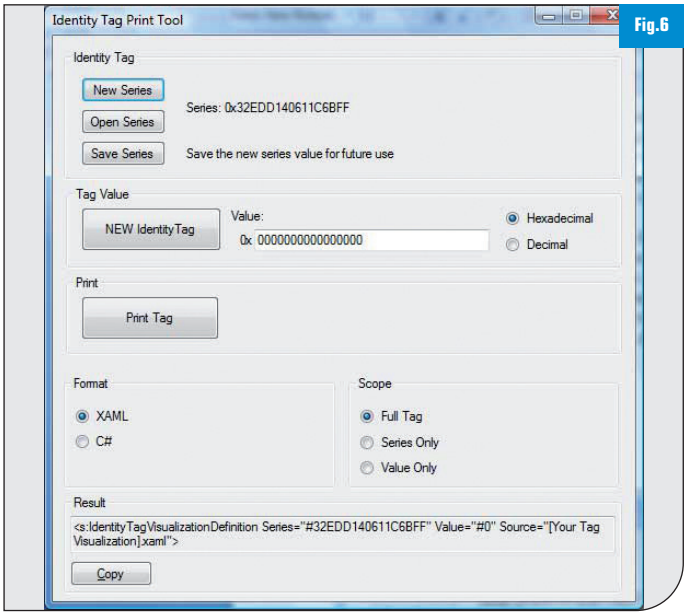


Fig.6

Les Identity Tags

Les Identity Tags sont eux codés sur 128 bits (soit 16 octets) et sont composés de 2 valeurs codées sur 64 bits : la série et la valeur. Pour générer ces tags, vous pouvez utiliser le logiciel de génération d'Identity Tag fourni avec le SDK ou bien utiliser dans votre code la classe IdentityClassGenerator (voir Fig.4).

Lancement d'une application avec un tag 2D

Chaque application Microsoft Surface possède un fichier XML qui doit se trouver dans le répertoire « Programs » de Surface et qui décrit certaines des propriétés de l'application : son nom, ses icônes mais aussi les actions à réaliser sur détection de tags.

Ainsi dans le fichier XML ci-contre, le tag 2D de type Byte Tag de valeur OxCO (voir le contenu de l'élément XML < ByteTag >) va permettre de lancer l'application depuis le menu, l'économiseur d'écran ou toute autre application.

```
< ? xml version = "1.0" encoding = "utf-8" ? >

<!--
This file contains the information needed to install your application with the Surface Shell. Please refer to the documentation for deployment instructions.
-- >
<ss : ApplicationInfo
xmlns : xsi = "http://www.w3.org/2001/XMLSchema-instance"
xmlns : ss = "http://schemas.microsoft.com/Surface/2007/Application Metadata">
  <Application >
    <Title > Surface_Pre_Carre_01 </Title >
    <Description > Surface_Pre_Carre_01 </Description >
    <ExecutableFile > Surface_Pre_Carre_01.exe </ExecutableFile >
    <Arguments > </Arguments >
    <IconImageFile > Resources\icon.png </IconImageFile >
    <Preview >
      <PreviewImageFile > Resources\iconPreview. png </PreviewImageFile >
    </Preview >
    <Tags >
      <!--
If your application uses tagged objects, please uncomment this section to register the tags with the Shell.
You can register ByteTag (s), IdentityTag (s) or both by using the appropriate instructions below.
-- >
      <!--
To register ByteTags :
1. Please uncomment the Byte Tag Element below.
2. Replace "C0" below with the value of your Byte Tag (in hexadecimal format). Repeat this section (this element and its children) for other Byte Tags
3. Please remove the Launch element if you do not want to register the tag with Object Routing.
-- >
      <ByteTag Value = "C0">
        <Actions >
          <Launch/>
        </Actions >
      </ByteTag >
      <!--
To register IdentityTags :
1. Please uncomment the Identity Tag Element below.
2. Replace "0000000000000000" below with the series of your Identity Tag (in hexadecimal format). Repeat this section (this element and its children) for other Identity Tags
3. Please remove the Launch element if you do not want to register the tag with Object Routing.
-- >
      <!--
< IdentityTag Series = "0000000000000000">
        <Actions >
          <Launch/>
        </Actions >
      </IdentityTag >
      -- >
    </Tags >
  </Application >

  <!--
Uncomment this section and comment out the Application element above if you are creating an Attract Mode Application.
```

Fig.7

Fig.8

```
-- >
  <!--
< AttractApplication >
    <ExecutableFile > Surface_Pre_Carre_01.exe </ExecutableFile >
    <Arguments > </Arguments >
  </AttractApplication >
  -- >
</ss : ApplicationInfo >
```

La photo 8 illustre une détection sur tag située sous la voiture miniature ; en appuyant sur « Renault Laguna », on démarrera l'application liée à la Renault Laguna.

EXEMPLE DE DÉVELOPPEMENTS EN WPF POUR SURFACE

En utilisant le contrôle « ScatterView »

Le contrôle « ScatterView » est l'un des nouveaux contrôles créés pour exploiter les capacités multi-touch du SDK ; il est recommandé de l'utiliser lorsque vous souhaitez présenter dans votre application du contenu que l'utilisateur pourra librement agrandir, réduire, déplacer ou tourner.

```
< s : ScatterViewItem Name = "sc_item_port_01" Visibility = "Hidden"
 Orientation = "0" Width = "520" Height = "440" Background =
"AntiqueWhite">
  <Grid >
```

```
   <Viewbox >
     <Grid >
       <Grid. RowDefinitions >
         <RowDefinition Height = "40"/>
         <RowDefinition Height = "400"/>
       </Grid. RowDefinitions >

       < s : SurfaceTextBox IsEnabled = "False" Grid. Row = "0"
Margin = "20,0" FontSize = "16" FontWeight = "Bold" >
                                                Le Port
 </s : SurfaceTextBox >
                          <Image Width = "450" Height = "337"
 Grid. Row = "1" Source = "pack ://application :,,,/Resources
/le_port_01.jpg" Stretch = "Fill"/>
                          s : SurfaceButton Name = "bouton_
fermer_sc_item_port_01"  ContactEnter  =  "bouton_fermer_sc_
item_port_01_ContactEnter"  Margin  =  "-45,-10"  Style  =  "
{StaticResource style_bouton_fermer}"> </s : SurfaceButton >
     </Grid >
   </Viewbox >
 </Grid >
</s : ScatterViewItem >
```

Regardez le résultat sur la table Surface : le contrôle affiche une image, son titre et un bouton permet de fermer le contenu. Ce contenu peut être librement déplacé, retaillé et retourné par l'utilisateur. [Fig.9]

En utilisant le contrôle « TagVisualization »

Le contrôle « TagVisualization » permet d'indiquer dans son application que l'on souhaite gérer des tags. Une partie est décrite dans la fenêtre principale et l'affichage des éléments graphiques est décrit. Code dans le fichier XAML qui gère la détection du tag :

```
< s : TagVisualizer
xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns : x = "http://schemas.microsoft.com/winfx/2006/xaml"
xmlns : s = "http://schemas.microsoft.com/surface/2008"
>
            <s : TagVisualizer. Definitions >
                <s : ByteTagVisualizationDefinition
Value = "0x02"
Source = "TagVisualization_Korrigo. xaml"
MaxCount = "1"
LostTagTimeout = "2000"
OrientationOffsetFromTag = "90"
UsesTagOrientation = "True"
TagRemovedBehavior = "Fade"/>
            </s : TagVisualizer. Definitions >
        </s : TagVisualizer >
```

Code dans le fichier XAML qui affiche les éléments graphiques sur détection du tag (fichier TagVisualization_Korrigi. xaml) :

```
< s : TagVisualization x : Class = "Le_Star_01.TagVisualization
_Korrigo"
xmlns = "http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns : x = "http://schemas.microsoft.com/winfx/2006/xaml"
xmlns : s = "http://schemas.microsoft.com/surface/2008"
Loaded = "TagVisualization_Korrigo_Loaded">
    <Grid >

        <s : ScatterView Name = "scatterView">


            <s : ScatterViewItem AllowDrop = "True" s :
  SurfaceDragDrop. DragLeave = "tag_carte_korrigo_DragLeave"
  s : SurfaceDragDrop. Drop = "tag_carte_korrigo_Drop" s :
   SurfaceDragDrop. DragCompleted = "tag_carte_korrigo_Drag
Completed" s : SurfaceDragDrop. DragEnter = "tag_carte_korrigo
_DragEnter" s : SurfaceDragDrop. DragOver = "tag_carte_korrigo
_DragOver" Width = "225" Height = "150" Name = "tag_carte
_korrigo" CanScale = "False" Background = "{StaticResource back
_carte_korrigo}">

                <s : ElementMenu
ActivationMode = "HostInteraction"
ActivationHost = "{Binding RelativeSource = {RelativeSource Find
Ancestor, AncestorType = {x : Type s : ScatterViewItem}}}"
HorizontalAlignment = "Center"
VerticalAlignment = "Top" Background = "SteelBlue">
                    <!-- Position the ElementMenu partially
 off the top edge. -- >
                    <s : ElementMenuItem Icon = "{Static
Resource logo_services01}" Header = "Services" >
                    </s : ElementMenuItem >
                    <s : ElementMenuItem Icon = "{Static
Resource logo_culture}" Header = "Loisirs">
                    </s : ElementMenuItem >
                </s : ElementMenu >
            </s : ScatterViewItem >


        </s : ScatterView >

    </Grid >
</s : TagVisualization >
```

L'image 10 illustre l'affichage du contenu sur détection de la carte de transport Korrigo sur laquelle se trouve le tag 2D. La carte est détectée et une image entoure la carte et va permettre à l'utilisateur de s'envoyer les nouveaux horaires de bus par Drag & Drop vers sa carte.

CONCLUSION

Le développement des applications est facilité par le SDK et les nouveaux contrôles fournis par Microsoft, mais il reste nécessaire de porter les applications Windows sur Surface pour tirer pleinement parti de l'interface tactile, du multi-touch et de l'interface à 360 degrés.

■ Richard **Seltrecht**
Directeur de Selten, spécialisé dans le développement Surface


Fig.9

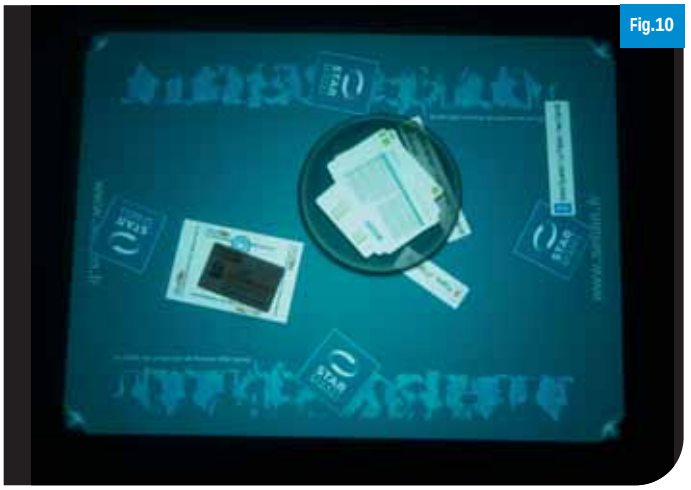


Fig.10

Développement Web : de la conception aux tests

Visual Studio 2010 renforce considérablement ses fonctions de développement web et de tests. Avec la gamme Expression et les technologies web telles que ASP .Net ou Silverlight, le développeur dispose de tous les outils nécessaires au développement web. Dans cet article nous allons utiliser et mettre en œuvre Team Foundation Server. La phase d'installation est disponible sur notre site web : www.programmez.com.

MISE EN ŒUVRE DU PROJET WEB

Création d'une solution Visual Studio

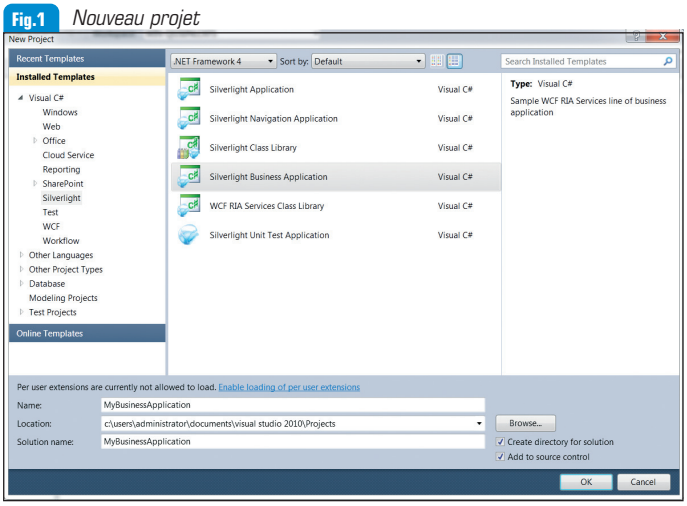
Compte tenu de la complexité du projet, il est souvent nécessaire de découper la solution en de nombreux projets, en fonction de règles de découpage des responsabilités assez classiques :

- Couches d'accès aux données et aux services.
- Services transverses.
- Couche métier.
- Couche présentation.

Dans le cadre de cet article, nous allons utiliser une structuration assez simple, issue d'un des nouveaux modèles de projet : le projet « Silverlight Business Application ». Ce type de projet est apporté par la couche RIA Services. Il peut être vu comme une surcouche d'un projet Silverlight classique fournissant les éléments suivants :

- Une gestion simplifiée de l'exposition des services. Au lieu de devoir modifier manuellement les fichiers de configuration de l'application, l'exposition des classes dérivant de la classe de base **DomainService** se fait automatiquement (en pratique, le type de projet embarque la déclaration d'un module http dont le rôle est, entre autres, de configurer le système à l'exécution).
- Une intégration pour le client Silverlight des informations des MembershipProvider et RoleProvider ASP .Net. Ainsi, l'application Silverlight peut utiliser les informations de login et de profil de l'utilisateur.
- Une génération des proxies WCF à la compilation, avec possibilité de partager du code entre le serveur et le client. A la compilation, le système produit des proxies en se basant sur des informations liées à la réflexion et à certaines conventions de nommage. Ce faisant, et même s'il faut toujours revalider côté serveur ce qui est posté pour des raisons de sécurité, il est alors possible de partager les métadonnées des objets métiers avec le client, ce qui permet de réaliser des tests de validation sans avoir besoin de repasser par le serveur.

Fig.1 *Nouveau projet*



Mais avant de mettre cela en œuvre, il faut d'abord installer quelques éléments :

- Les outils Silverlight 4 pour Visual Studio 2010, sur http://silverlight.net/getstarted/silverlight-4-beta/#tools. Ces outils incluent la couche RIA Services.
- Blend 3.1, sur http://www.microsoft.com/downloads/details.aspx?FamilyID=6806e466-dd25-482b-a9b3-3f93d2599699&displaylang=en. Cette version est compatible avec le nouveau format de fichier de projet de Visual Studio 2010.
- Et pour le principe, le toolkit Silverlight, sur http://silverlight.codeplex.com/Release/ProjectReleases.aspx?ReleaseId=36060. Ce dernier contient de nombreux contrôles complémentaires (graphiques, autocomplétion, etc.).

Une fois ces modules installés, vous pouvez enfin créer votre projet en l'intégrant au gestionnaire de source : **[Fig.1]**

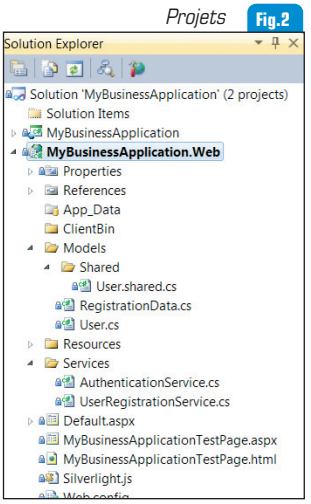
Le résultat est une solution avec deux projets :

- Un projet Web contenant les services.
- Un projet Silverlight contenant l'interface utilisateur. **[Fig.2]**

Si vous regardez les propriétés du projet Silverlight, vous verrez qu'un élément inhabituel est présent sur l'onglet Silverlight : **[Fig.3]**

Le lien « .Net RIA Services » permet en fait d'indiquer quel projet C# doit être utilisé comme source de la génération des proxies. Et si vous regardez les fichiers cachés du projet Silverlight, vous verrez qu'un sous-répertoire « Generated_Code » a été créé : **[Fig.4]**

Fig.2 *Projets*



Ce répertoire contient plusieurs éléments :

- Un fichier MyBusinessApplication. Web.g.cs. Ce fichier définit plusieurs classes :
  - **WebContext**, qui contient le contexte en cours (l'utilisateur, s'il est authentifié).
  - Une classe **XxxContext** par classe publique héritant de

DomainService dans le projet C#. Par défaut, vous trouverez **AuthenticationContext**, permettant de dialoguer avec le service d'authentification, et **RegistrationContext**, pour gérer l'enregistrement d'un nouvel utilisateur.

- Les classes correspondant aux types de données consommées par les précédentes classes.

- Un fichier *. shared.cs par fichier *. shared.cs présent dans le projet C# (par défaut, un seul). Ces fichiers sont en fait recopiés : c'est donc à vous de vous assurer que le fichier compile bien en .Net et en Silverlight. Pour gérer les cas trop complexes, une directive de compilation (SILVERLIGHT) existe afin de permettre une compilation conditionnelle en fonction de l'environnement.

Cette structuration est la plus simple possible. Dans le cadre d'une vraie application, les services seraient certainement définis dans des projets dédiés, tout comme les objets métiers. Un projet Silverlight ne pouvant se lier qu'à un seul projet .Net, il faut alors penser à structurer ses projets en conséquence :

- Par exemple, un projet .Net par ensembles de domaines fonctionnels.
- Pour chaque projet .Net, un projet Silverlight lié mettant en œuvre la partie cliente associée au domaine fonctionnel.
- Et une application Silverlight mettant en œuvre ces divers projets. **[Fig.5]**

Accès au projet depuis Expression Blend

Pour pouvoir travailler sur le projet, les infographistes auront besoin de deux outils :

- Team Explorer.
- Blend 3.1

La gestion des sources est maintenant native dans Blend, il n'y a donc plus de configuration spécifique à faire ou de patch à installer. Et comme précédemment, il faut d'abord récupérer le projet avec Team Explorer sur le poste client. Une fois cela fait, lorsque le projet est ouvert dans Blend, les commandes de gestion de sources sont disponibles : **[Fig.6]**. Les graphistes peuvent alors directement travailler sur la solution et en modifier les écrans, sans passer par des étapes d'échanges de fichier ou de conversion de format. Typiquement, ce travail se fait en parallèle aux développements…

Partage de métadonnées et appels de services

Comme indiqué précédemment, l'un des avantages de ce type de projet est de pouvoir partager aussi des informations de métadonnées. Ces informations peuvent provenir d'un modèle EDMX si la couche de données a été générée avec Linq To SQL ou Entities Framework. Mais dans le cas d'objets .Net « pur » (des POCO, pour Plain Old C# Object), il faut fournir les informations en question. Pour cela, on va utiliser un attribut introduit avec le Framework 3.5 : l'attribut MetadataType.

Cet attribut permet d'associer à une classe existante une classe décrivant des métadonnéees, sans modifier la classe existante, et sans adhérence à une technologie d'interface particulière. Typiquement, ces métadonnées pourraient aussi resservir avec des projets de type ASP .Net MVC ou ASP .Net Dynamic Data.

Concrètement, pour une classe donnée, disons Panier, vous devez créer une deuxième classe, comportant des champs du même nom (mais le type importe peu) :

```
[MetadataType (typeof (PanierMetadata))]
    public class Panier
{
public double Montant { get ; set ; }

public Guid PanierId { get ; set ; }

public string NomUtilisateur { get ; set ; }

internal sealed class PanierMetadata
{
[Required]
            [Range (0, double. MaxValue)]
            public object Montant ;
```

Fig.3 *Option Silverlight*

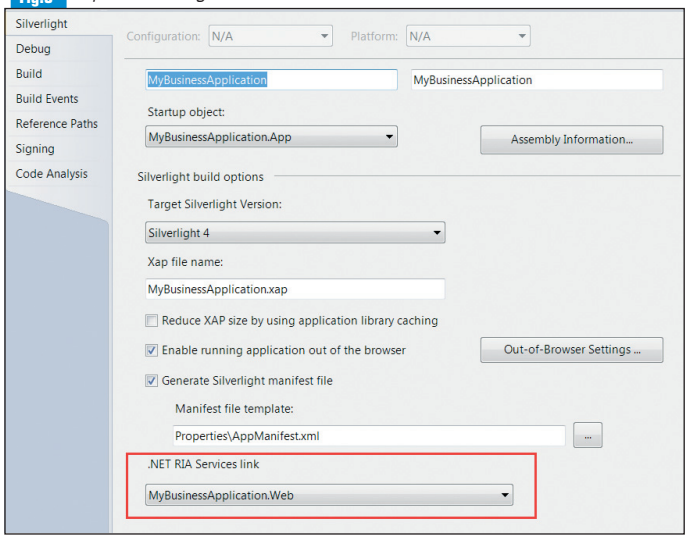


Fig.5 *Intégration de plusieurs projets*

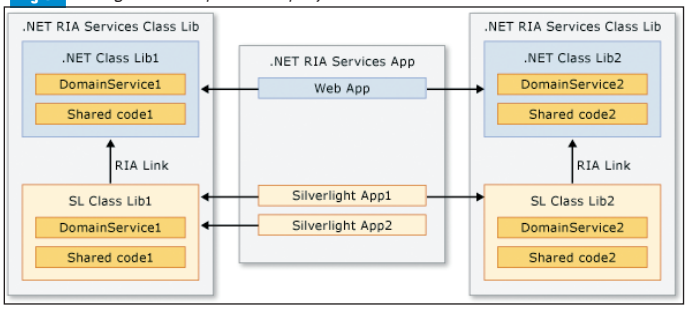


Fig.4 *Code généré*

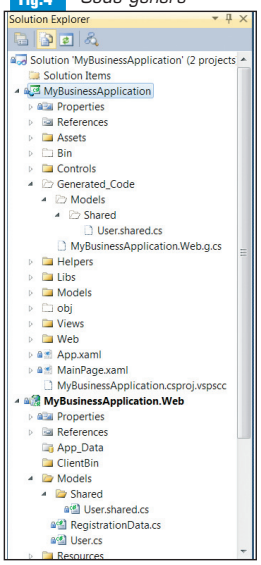
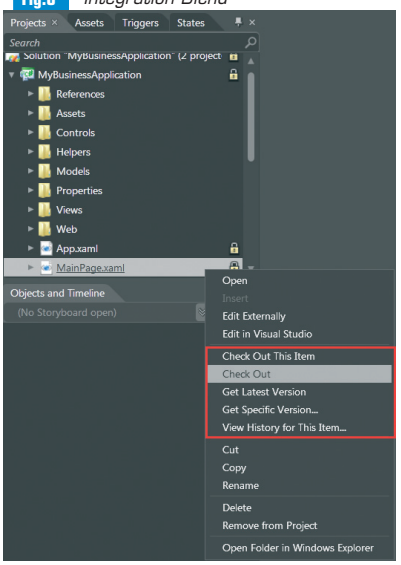


Fig.6 *Intégration Blend*



```
[Required]
            [Key]
            public object PanierId ;

[Required]
            [StringLength (8, MinimumLength = 8)]
            public object NomUtilisateur ;
}
}
```

Ici, la classe fournissant les métadonnées est créée comme imbriquée à la classe définissant l'objet, mais cela n'est pas obligatoire. Par exemple, dans le cas de classes générées, ou pour des raisons de lisibilité, vous pouvez tout à fait définir cette classe et positionner l'attribut dans un fichier séparé (en définissant Panier comme une classe partielle typiquement).

Les principaux éléments à retenir sont :

- L'utilisation de l'attribut **Key.** Cet attribut indique au système quelle est la clé unique identifiant l'objet. La présence d'un élément avec cette clé est obligatoire.
- Des attributs de validation (**Range, Required, StringLength,** etc). D'autres attributs existent (expressions régulières par exemple) et vous pouvez étendre le système en héritant de l'attribut **Validation**.

Une fois votre classe créée, vous devez l'exposer via un domaine de service :

```
[EnableClientAccess]
    public class PanierService : DomainService
{
[Query (IsComposable = false)]
        public Panier GetPanier (Guid id)
{
return new Panier () {PanierId = id, Montant = 0, NomUtilisateur
 = "John Doe"} ;
}
}
```

Le service doit hériter de la classe **DomainService** et avoir l'attribut **EnableAccessClient**. L'attibut **Query** permet ici de préciser que la requête ne peut pas être utilisée comme source d'une autre requête (en clair, vous ne pouvez par associer un filtre supplémentaire lors de la composition de la requête côté client).

Après avoir recompilé la solution, vous constaterez que le fichier MyBusinessApplication. Web.g.cs a changé : il inclut maintenant une classe **PanierServiceContext** (qui permet d'exécuter la méthode GetPanier) et une classe **Panier**. Cette dernière expose les champs de la classe source, les attributs liés aux métadonnées et des méthodes partielles vous permettant de modifier le comportement de la classe sans craindre de perdre votre travail en cas de regénération du code.

```
public sealed partial class Panier : Entity
{
#region Extensibility Method Definitions

/// <summary >
        /// This method is invoked from the constructor once
initialization is complete and
/// can be used for further object setup.
/// </summary >
        partial void OnCreated ();
        partial void OnMontantChanging (double value) ;
partial void OnMontantChanged ();
        partial void OnNomUtilisateurChanging (string value) ;
partial void OnNomUtilisateurChanged ();
        partial void OnPanierIdChanging (Guid value) ;
partial void OnPanierIdChanged ();

        #endregion

[DataMember ()]
        [Required ()]
        public double Montant
{
get
{
return this._montant ;
}
set
{
if ((this._montant ! = value))
                {
                   this. ValidateProperty ("Montant", value) ;
this. OnMontantChanging (value) ;
this. RaiseDataMemberChanging ("Montant") ;
this._montant = value ;
this. RaiseDataMemberChanged ("Montant") ;
this. OnMontantChanged ();
                }
            }
        }
    }
```

Enfin, pour créer, modifier ou supprimer des éléments, vous devez pour chaque type d'éléments devant supporter ces opérations :

- Soit définir les méthodes **InsertXxx, DeleteXxx, UpdateXxx.**
- Soit marquer des méthodes avec les attributs **Insert, Delete, Update.**

Les méthodes sont à définir dans le domaine de service, et les attributs ne sont nécessaires que si la convention de nom n'est pas respectée (ici, dans le cas de l'insertion) :

```
[EnableClientAccess]
    public class PanierService : DomainService
{
[Insert]
        public void CreatePanier (Panier panier)
{
}

[Update]
        public void UpdatePanier (Panier panier)
{
}

[Delete]
        public void DeletePanier (Panier panier)
```

```
{
}
}
```

Une fois ces méthodes définies, lorsque le code client invoquera une sauvegarde du contexte de données, la méthode **SubmitChanges** de la classe de service sera automatiquement invoquée, et cette dernière invoquera alors implicitement toutes les méthodes associées aux types des données modifiées : il n'est pas utile côté client d'appeler explicitement ces méthodes pour chaque donnée gérée… Et comme en Silverlight, tous les appels sont asynchrones, pour récupérer une exception côté client, penser à utiliser la surcharge de la méthode prenant un **callback** en paramètre :

```
protected void Submit ()
    {
        PanierContext ctx = new PanierContext ();
        ctx. SubmitChanges (OnSubmitCompleted, null) ;
}

private void OnSubmitCompleted (SubmitOperation so)
{
if (so. HasError)
{
MessageBox. Show (string. Format ("Submit Failed : {0}", so. Error.
 Message));
            so. MarkErrorAsHandled ();
        }
    }
```

Enfin, pour exposer une méthode qui n'est pas liée à une opération de données, pensez à marquer la méthode avec l'attribut **ServiceOperation,** afin qu'une méthode d'appel côté client soit générée.

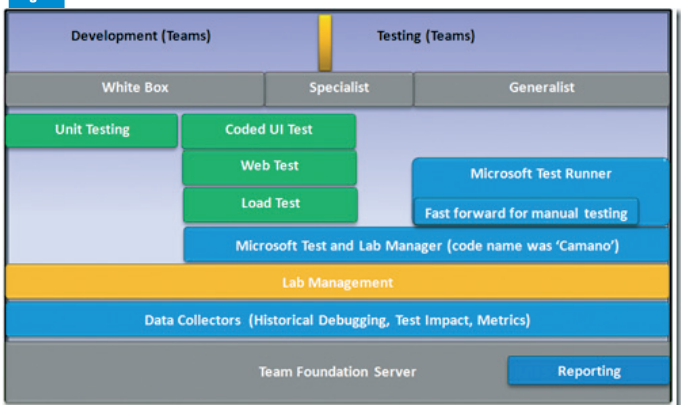
TESTS DE MONTÉE EN CHARGE

Présentation de Visual Studio Team Test 2010

La version 2010 de Visual Team Test a subi une évolution majeure dans la présentation du produit. Trois versions sont maintenant disponibles pour répondre aux différents besoins de tests applicatifs :

- Visual Studio Team Test 2010 Essentials (en bleu sur le schéma ci-dessous) destinés aux testeurs non-développeurs pour création et l'exécution de cas de tests manuels ou automatiques.
- Visual Studio Team Test 2010 (en vert + bleu sur le schéma ci-dessous) destiné aux développeurs complétant les possibilités de Visual Studio Team Test 2008.

Fig.7 *Versions de Visual Studio Team Test 2010*



- Visual Studio Lab Management (en orange sur le schéma) destiné à la mise en place d'environnement virtualisé. **[Fig.7]**

Nous allons ici voir la création de tests web et la mise en place de tests de charge via l'outil Visual Studio Team Test 2010.

Application Web à tester

Dans le cadre de cet article, nous utiliserons une application Web quelconque (celle créée précédemment ou tout autre) : **[Fig.8]**

Création d'un projet de Test

Quel que soit le type de tests à créer, la première étape consiste à créer un projet de type Test. **[Fig.9]**

- Par défaut, le projet de Test créé contient les fichiers suivants :
- UnitTest1.cs : Template de test unitaire.
- Fichier vsmdi : Offre une vue listant les tests présents dans la solution. Cette vue permet par exemple de sélectionner les tests à exécuter.
- Fichiers testsettings : Fichiers de configuration des tests. Le choix de la configuration à utiliser est possible via le menu « Test ». **[Fig.10]**

- Le menu Test se présente de la façon suivante : **[Fig.11]**

Ajout d'un test Web

Dans le projet de test précédemment créé, il est possible d'ajouter un fichier de type « Web Performance Test ». Visual Studio Team Test lance automatiquement un enregistreur de requête http et un navigateur. Il nous est alors possible de naviguer sur notre site de test, toutes les requêtes client/serveur étant enregistrées. **[Fig.12]**

Dans le cadre de cet article, le test consiste à enregistrer un nouvel

Fig.8 *Application Web par défaut*

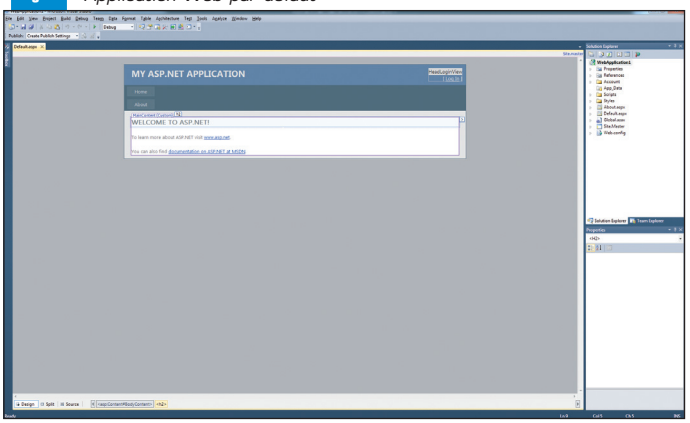
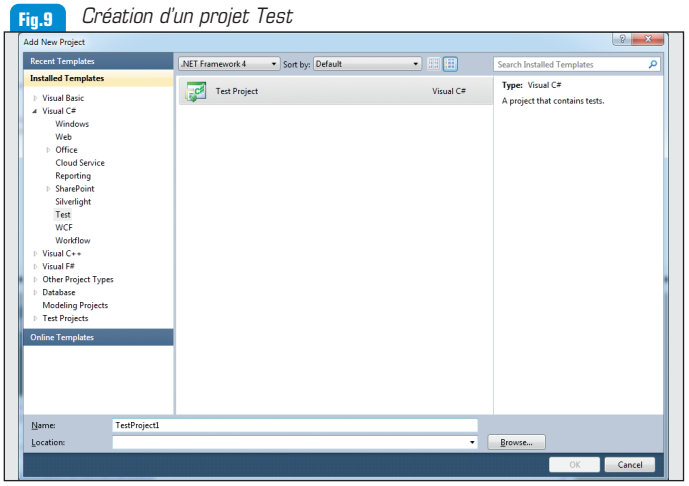


Fig.9 *Création d'un projet Test*



utilisateur. Une fois le scénario enregistré, nous retrouvons l'ensemble des pages sur lesquelles nous avons navigué ainsi que les paramètres envoyés au serveur. [Fig.13]

MODIFICATION DU TEST WEB

Modification des requêtes

Une fois le test web créé, il est alors possible de modifier les propriétés de chaque requête, notamment le temps d'attente à simuler (ThinkTime) lors de la relance du test. Ce temps d'attente est pré rempli avec le temps d'attente lors de l'enregistrement du scénario. Ce temps nous permet de simuler la navigation d'un vrai utilisateur. [Fig.14]

Dans la version de Visual Studio Team Test 2010, une nouveauté consiste à afficher de manière globale certaines propriétés importantes de ces requêtes. (Bouton ). « Reporting Name » est aussi une nouvelle propriété qui permet de modifier le nom affiché lors de l'affichage du résultat d'exécution d'un test. Ce nom sera en effet plus compréhensible que la requête en elle-même. [Fig.15]

Paramètres des requêtes

- Paramètres statiques :

Le bouton  permet de détecter tous les serveurs web dans les requêtes du test. Un paramètre est alors créé pour chaque serveur détecté. Les requêtes du test sont automatiquement mises à jour. Il nous sera alors plus simple de modifier les url dans le cas d'un changement de plate-forme ou de serveurs. Dans notre cas, un seul serveur a été détecté. Web Server [Fig.16]

A noter : *Il est nous est possible de créer notre propre paramètre. La référence à ce paramètre est possible via la commande « {{ParameterName}} ».*
À noter : *Il est possible de créer des règles d'extraction qui permettent d'extraire d'une réponse des informations pour les positionner en tant que paramètre statique.*

- Paramètres dynamiques :

Dans notre scénario, la requête de création d'un utilisateur enregistré contient les informations de l'utilisateur. Si nous relançons notre test, il sera en échec puisque l'utilisateur a déjà été créé. Il nous faut donc pouvoir utiliser une source dynamique de données pour référencer les informations des utilisateurs à créer.

Le bouton  permet de créer une source de données. Dans notre cas, nous utiliserons un fichier csv.

À noter : *Il est possible d'utiliser une base de données SQL ou un fichier XML.* [Fig.17]

Notre source de données est alors disponible dans notre test. Trois modes d'accès aux données sont disponibles (Aléatoire, Séquentiel, Unique). [Fig.18]

Nous pouvons alors modifier la requête pour utiliser cette source de données. [Fig.19]

Validation des requêtes

Pour valider qu'une requête est exécutée correctement, il est possible d'ajouter des règles de validation. [Fig.20]

Les règles principales sont les suivantes :

- Response Time Goal : Pourcentage autorisé pour chaque requête

Fig.10 *Fichiers créés par défaut lors de l'ajout d'un projet de type Test*

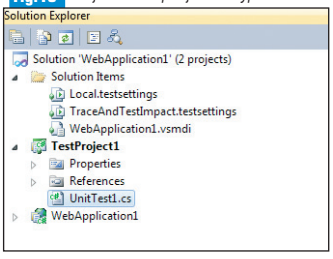


Fig.11 *Menu "Test"*

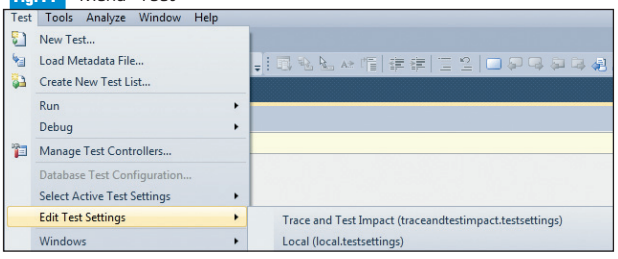


Fig.12 *Enregistrement des requêtes*

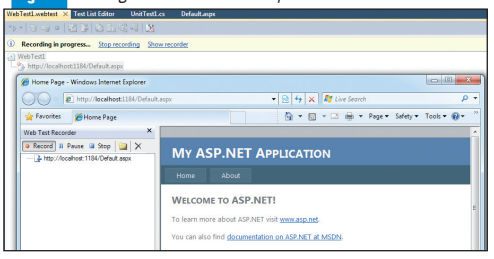


Fig.13 *Web Test : Résultat de l'enregistrement*

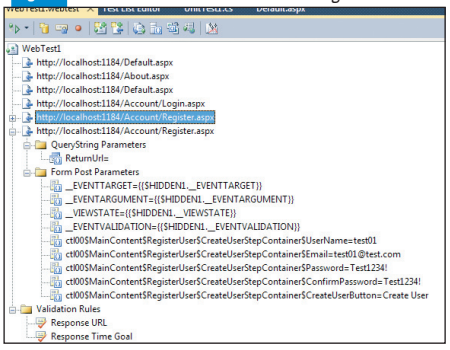


Fig.14 *Propriétés d'une requête*

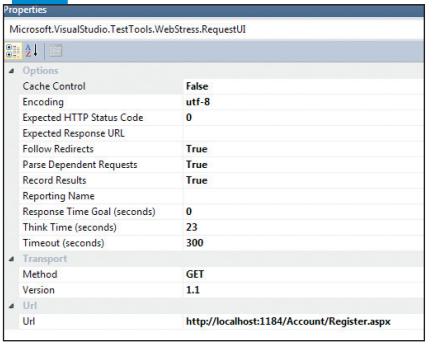


Fig.15 *"Set Request Details"*

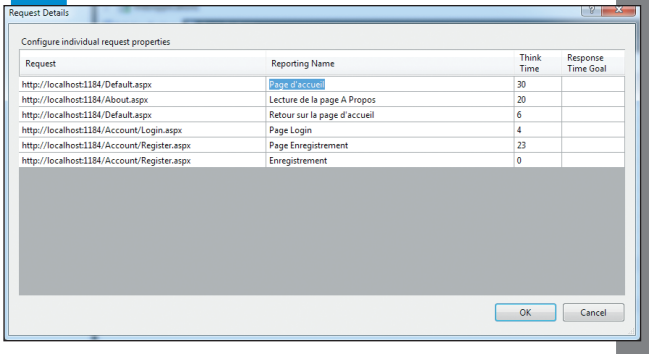


Fig.16

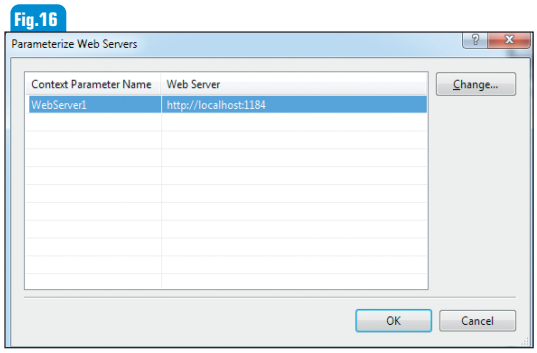


Fig.17 *Création d'une source de données (CSV)*

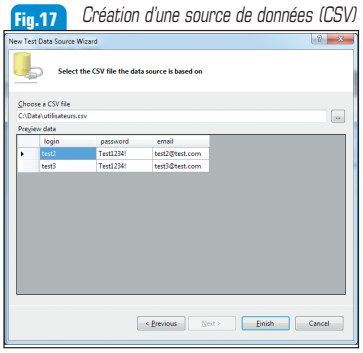


Fig.18 *Source de données*

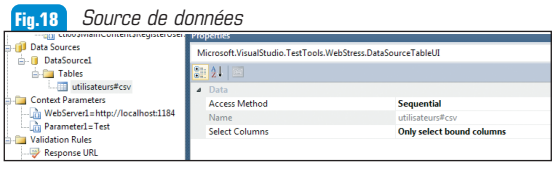
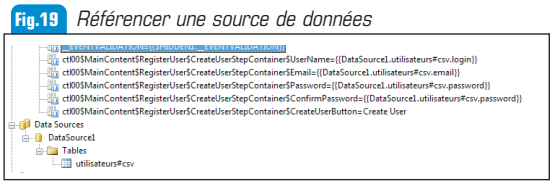


Fig.19 *Référencer une source de données*



par rapport à la valeur de référence « Response Time Goal », propriété propre à chaque requête.
- Response Url : Vérifie que l'url de réponse est identique à celle enregistrée dans le scénario.
- Required Tag/Required Attribute Value : Vérifie la présence d'un tag dans la requête de réponse.
- FindText : Vérifie la présence d'un texte dans l'url de réponse.

Deux nouvelles règles de validation ont été ajoutées dans cette version :
- Selected option : Permet de vérifier que la valeur sélectionnée dans une combobox ou une dropdownlist est correcte
- Tag Inner Text : Permet de vérifier la valeur d'un attribut HTML.

À noter : *Il est possible de créer ses propres règles de validation.*

Dans notre cas, nous utiliserons les règles de validation par défaut « Response Time Goal » et « Response Url ». Il nous faut donc modifier le « Response Time Goal » de chaque requête. **[Fig.21]**

Exécution du test Web

Le bouton  permet d'exécuter le test en mode debug ou non : **[Fig.22]**

L'évolution et le résultat du test sont disponibles via 2 écrans :
- « Test Results » : Résultat d'exécution du test **[Fig.23]**
- « Web test » : Résultat de chaque requête. Le nom de la requête est bien « Reporting Name » comme vu précédemment. **[Fig.24]**
- L'onglet « Web Browser » affiche le résultat HTML de la requête.
- L'onglet « Request » affiche toutes les informations de la requête envoyée (GET/POST, Headers, Cookies, QueryString, Post Parameters).
- L'onglet « Response » affiche l'en-tête et le corps de la réponse.
- L'onglet « Context » affiche tous les paramètres du test comme par exemple la valeur des paramètres dynamiques du test.
- L'onglet « Details » affiche le résultat des règles de validation ainsi que les éventuelles exceptions.

Modification du flux d'exécution du test

L'exécution du test s'effectue de manière séquentielle. Avec la version 2010, il est maintenant possible de conditionner la logique d'exécution en ajoutant des règles de type « Loop » ou « Condition ». Dans notre cas, conditionnons le déroulement du test en vérifiant si la première page a correctement répondu. **[Fig.25]**

À noter : *La condition peut porter sur des critères variés :*

- Présence d'un cookie
- Règle de probabilité : le résultat de la condition sera aléatoire basé sur le pourcentage indiqué.
- Comparaison d'un paramètre statique et d'une valeur.
- Code de retour d'une page
- Valeur d'un cookie
- Résultat d'une requête

Une fois la condition validée, elle est visible dans l'écran du test : **[Fig.26]**

À noter : *La règle de type boucle fonctionne sur le même principe.*

Fig.20 *Validation Rule*

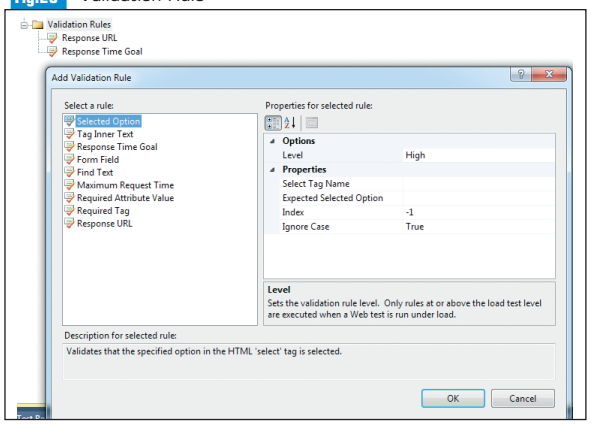


Fig.21 *Détail de la requête*

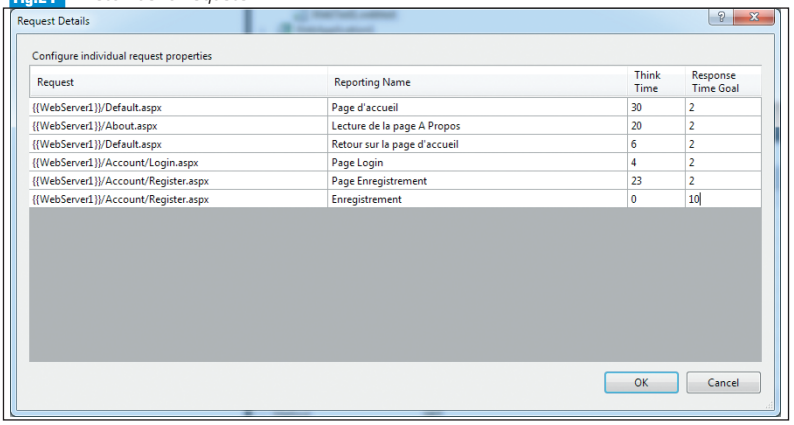


Fig.22 *Lancement du test*

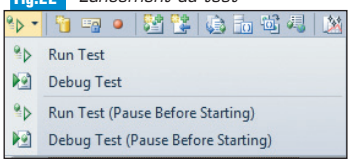


Fig.23 *Résultat*

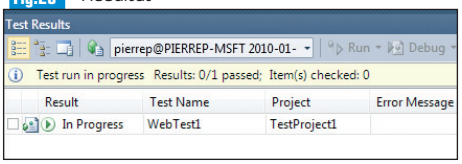


Fig.24 *Résultat des requêtes*

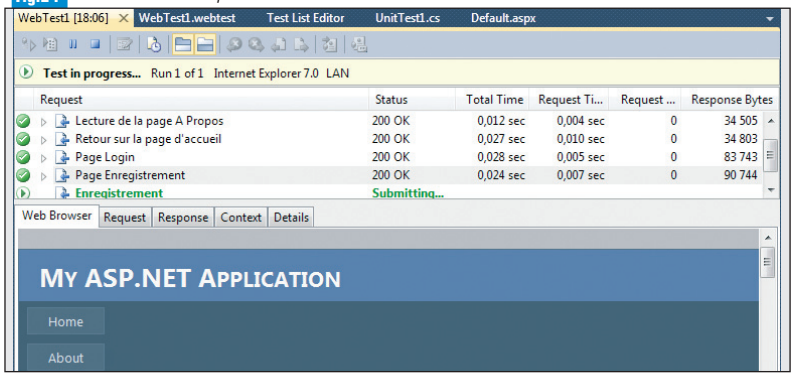
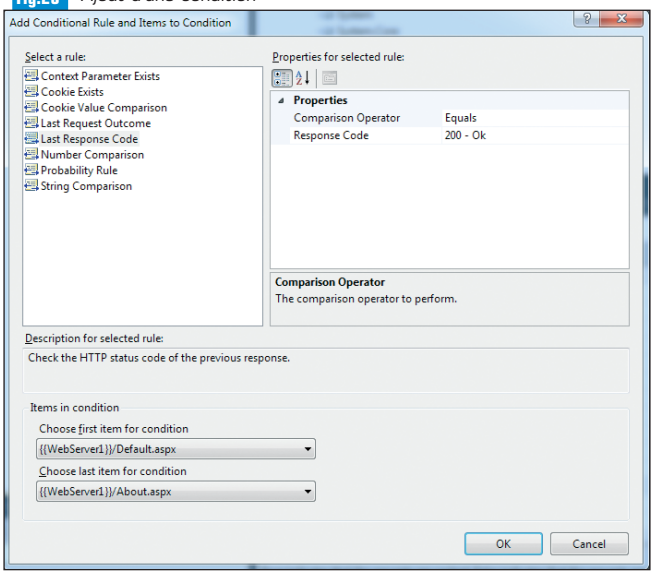


Fig.25 *Ajout d'une condition*



Création de tests de charge

Notre test web permet de tester un scénario d'utilisation de notre application en mode mono utilisateur. L'objectif des tests de charge est donc de tester notre application en mode multi- utilisateurs.

Ajoutons dans notre projet de test, un fichier de type « LoadTest ». **[Fig.27]**

Ce wizard va nous guider dans la configuration de notre test de charge.

Il existe 2 temps d'attente :

- Celui entre 2 tests
- Celui pour chaque requête.

Etape 1 : Nom du test de charge et configuration des temps d'attente :

- Soit utilisation des temps d'attente enregistrés
- Soit pas de temps d'attente

Fig.26 *Test avec Condition*

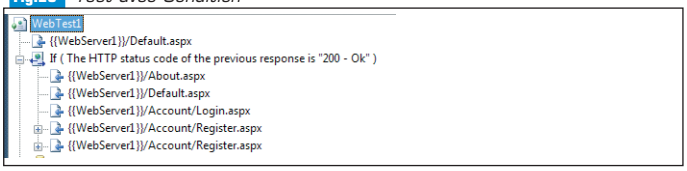


Fig.27 *Création d'un test de charge*

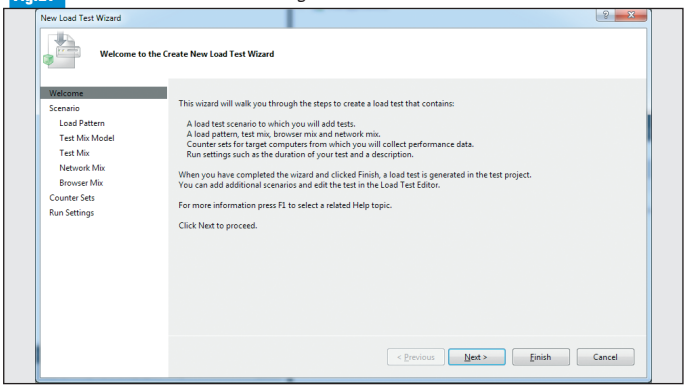


Fig.26 *Temps d'attente*

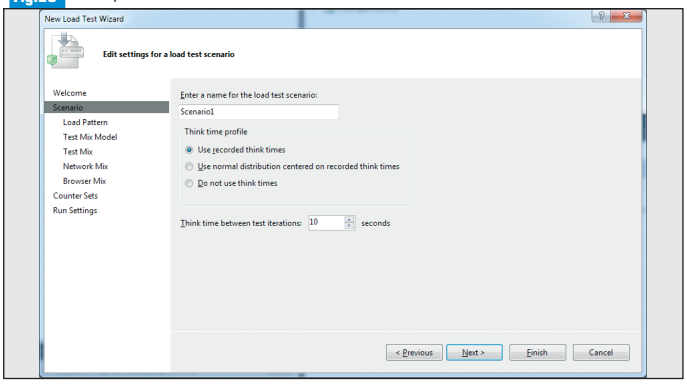
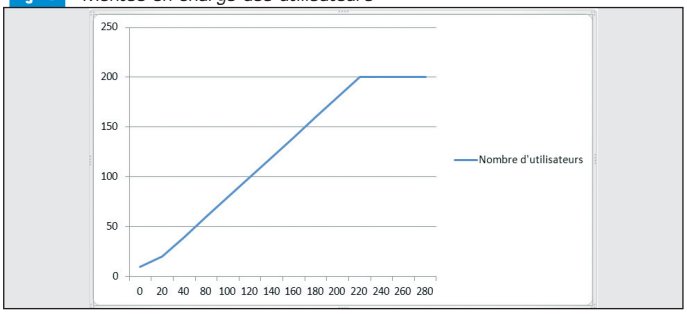


Fig.29 *Montée en charge des utilisateurs*



- Soit une distribution aléatoire autour des temps enregistrés.

Ici nous positionnons le temps d'attente entre chaque requête à 10 secondes et utilisons des temps d'attente enregistrés pour chaque requête. **[Fig.28]**

Etape 2 : Configuration de la montée en charge des utilisateurs :

- Soit utilisation d'un nombre constant d'utilisateurs.
- Soit utilisation d'une montée contrôlée.

Dans notre cas, nous choisissons de contrôler la montée du nombre d'utilisateurs de la manière suivante : **[Fig.29 et 30]**

Etape 3 : Configuration des tests à exécuter. Dans notre cas, notre test de charge est basé sur un test web unique. Cette étape permet de configurer l'ordre des tests à exécuter.

Etape 4 : Choix de tests à réaliser **[Fig.31]**

Etape 5 : Choix des cartes réseau à utiliser. Il est possible de configurer la distribution des cartes réseaux utilisées pour ce test de charge. Cela permet de contrôler l'évolution des flux selon différents canaux. Dans notre cas, nous nous limiterons à la carte Ethernet.

Etape 6 : Choix des navigateurs. Distribution des navigateurs à utiliser pour le test de charge.

Dans notre cas, nous utiliserons Internet Explorer 7 et Internet Explorer 8 (50 % chacun). **[Fig.32]**

Fig.30 *Configuration des utilisateurs*

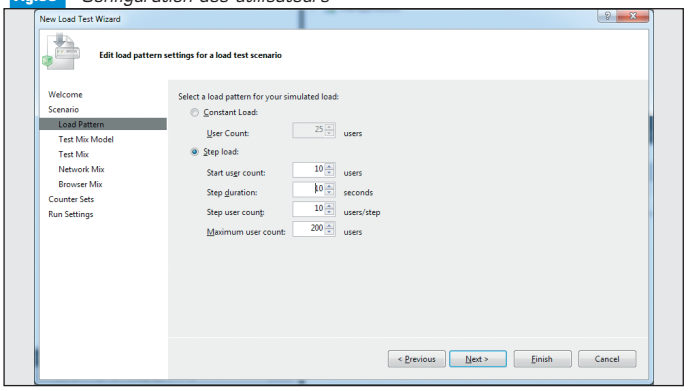


Fig.31 *Choix des tests*

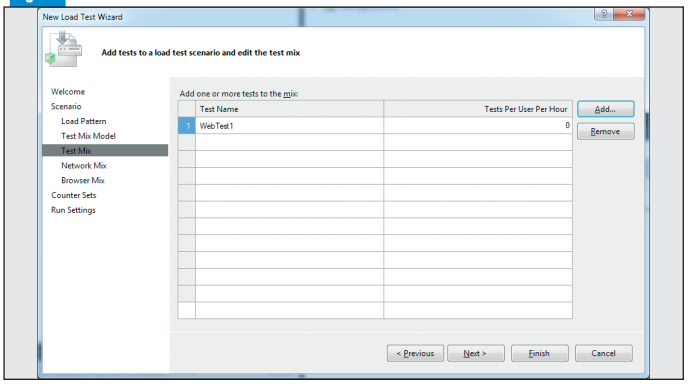
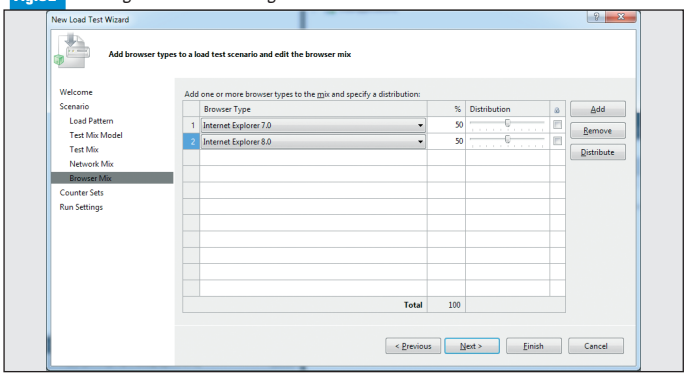


Fig.32 *Configuration des navigateurs*



Etape 7 : Configuration des serveurs à monitorer. Durant le test de charge, il faut récupérer des informations utiles sur chaque machine concernée pour pouvoir évaluer comment notre application réagit. Des profils existent en fonction de la machine ajoutée : dans le cas par exemple, d'un serveur SQL ou d'un serveur web, les compteurs de performance récupérés ne seront pas identiques, (accès au disque ou évolution du process IIS par exemple). **[Fig.33]**
Dans notre cas, nous nous limiterons à notre serveur Web qui est aussi un serveur SQL.

Etape 8 : Configuration de la durée du test.

- Soit basée sur le nombre d'itérations des tests.
- Soit basée sur le temps.

Dans notre cas, nous limiterons notre test de charge à 20 minutes et un temps de démarrage d'une minute pour charger les 10 premiers utilisateurs et ne pas tenir compte des premières valeurs qui comprennent le temps de démarrage de l'application. **[Fig.34]**

À noter *Le « sampling rate » permet de configurer le nombre de secondes entre chaque récupération des compteurs de performance.*

Ainsi configuré, notre test est prêt à être lancé. **[Fig.35]**

Fig.33 *Configuration des compteurs de performance*

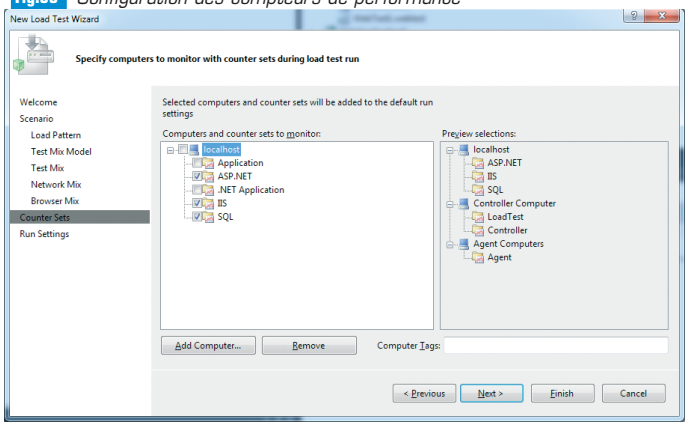


Fig.34 *Durée du test de charge*

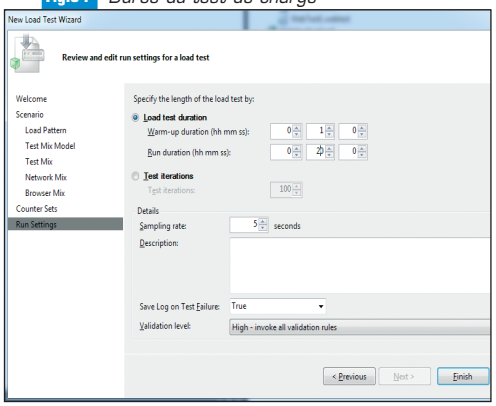


Fig.35 *Test de charge configuré*

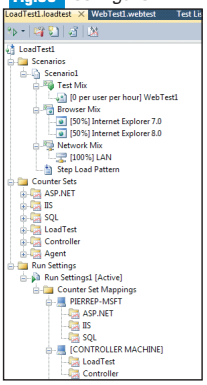
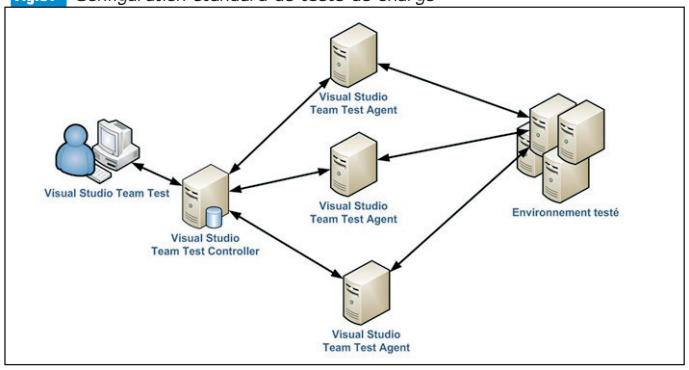


Fig.37 *Configuration standard de tests de charge*



Lancement du test de charge

Une fois lancé, il est possible de suivre l'évolution du test à travers des graphiques représentant les valeurs des compteurs de performance sélectionnés.

Evolution du test [Fig.36]

À noter : *il est possible d'ajouter des graphiques pour contrôler certaines valeurs.*

Configuration du test de charge

Dans notre cas, nous avons utilisé une seule et unique machine pour l'application et le test de charge. Il est cependant possible de différencier chaque rôle sur des machines différentes. **(Fig.37)**

À noter : *Dans Visual studio 2010, le modèle de licence a changé :*

- Visual Studio 2008 : Le contrôleur est gratuit et une licence par processeur et par Agent est nécessaire.
- Visual Studio 2010 : Le coût est basé sur le nombre d'utilisateurs à simuler. Il est donc possible d'installer autant d'agents que vous le souhaitez, l'objectif étant de ne pas dépasser le nombre d'utilisateurs virtuels achetés.

EN CONCLUSION

Ainsi se termine cette présentation des différentes étapes liées à la mise en œuvre d'un nouveau projet Web, avec :

- Mise en place d'un gestionnaire de source.
- Création du projet Silverlight.
- Instrumentation du projet pour la montée en charge.

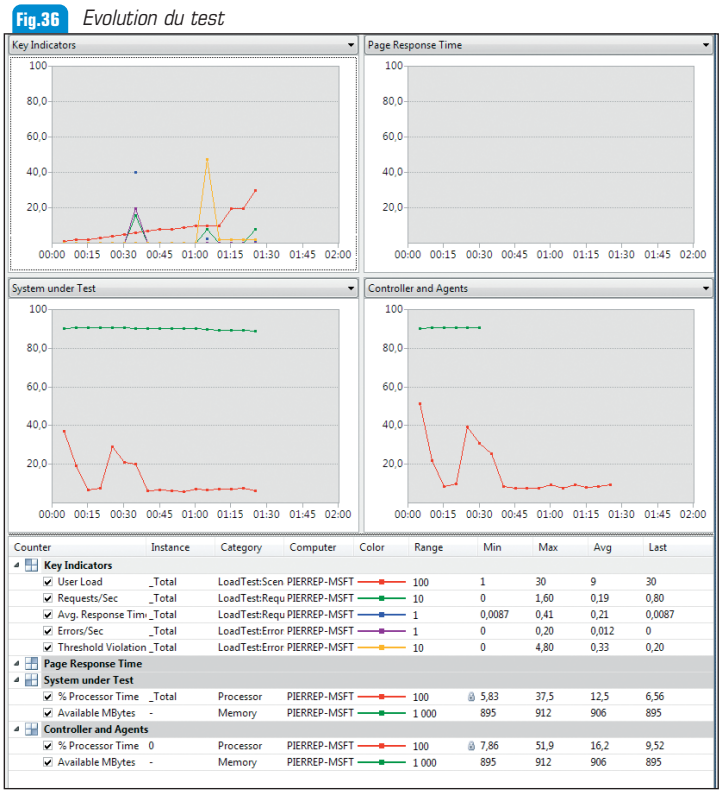
Bien entendu, il y aurait toujours plus à faire :

- Tests unitaires.
- Build quotidienne avec audit du code.
- Génération d'environnements de tests dédiés,
- Personnalisation des process et des work items.

Mais cela, c'est pour une autre fois ☺

■ Pierre-Charles **Peullemeulle**

Fig.36 *Evolution du test*

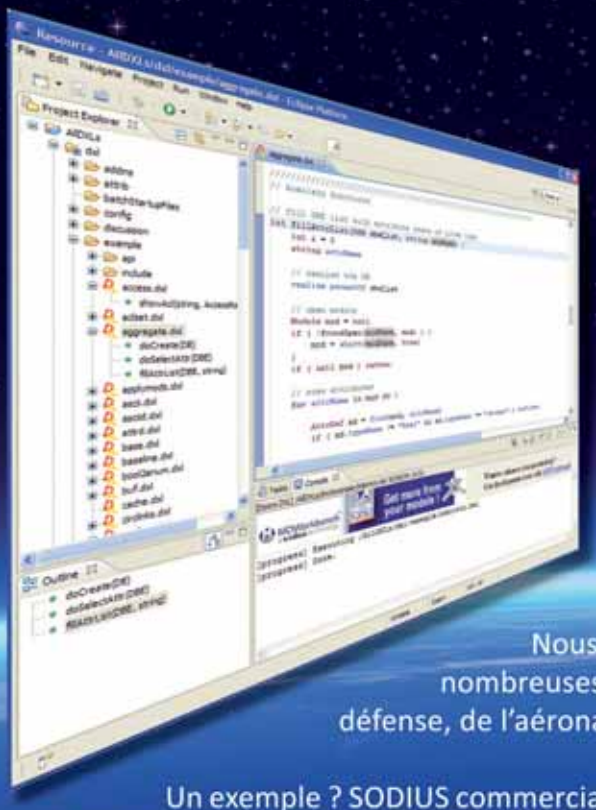


De nouveaux horizons pour vos développements

Il n'y a pas que l'Open Source pour réussir...

Vous avez développé un framework* logiciel performant (technologie ou métier) ?
Ce framework a été éprouvé par des clients professionnels exigeants ?
Vous avez identifié un marché transversal ou de niche ?
Vous souhaitez passer à la vitesse supérieure ?

Conjuguez vos talents avec SODIUS pour assurer l'industrialisation du framework et une mise sur le marché efficace !



SODIUS, Technology Partner de Telelogic et IBM depuis 7 ans, a développé un savoir-faire reconnu dans le domaine du génie logiciel, des ateliers de conception système et de l'interopérabilité sous Eclipse et Jazz.

Maintenant Business Partner de IBM Rational, SODIUS met a votre disposition son expertise pour la diffusion de solutions professionnelles sur des marchés exigeants.

IBM Business Partner

Nous fournissons des produits OEM à de nombreuses entreprises dans les domaines de la défense, de l'aéronautique, de l'automobile, du tertiaire, ...

Un exemple ? SODIUS commercialise MDWorkbench for DOORS et DXL Editor (www.dxleditor.com), des framework pour l'application DOORS de IBM Rational.

Si vous avez un framework validé en main, nous proposons aux meilleurs une alliance pour profiter de notre savoir-faire et de notre réseau de distribution, et ouvrir de nouveaux horizons.

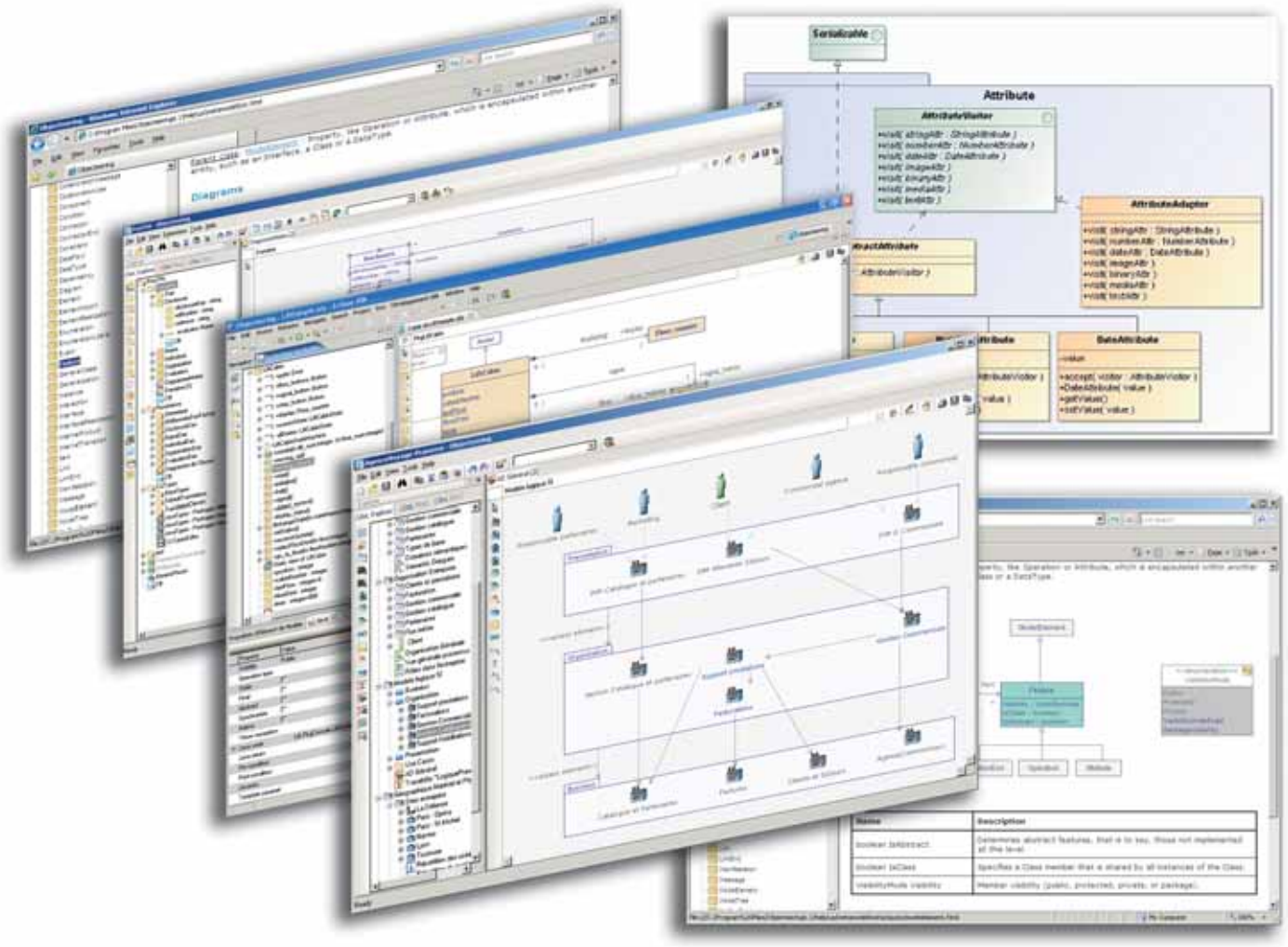
contactez nous !

Gérard Bouvet - Responsable des Partenariats
gbouvet@sodius.com

Sodius SAS
6 rue de la Cornouaille
44300 Nantes
Tél. : 02.28.23.60.60
www.sodius.com

*framework : atelier de développement.

Modélisation, Model Driven : facilitez-vous la vie !



2010, année de la modélisation, de l'UML et du Model Driven ? Ce dernier a fait un chemin remarquable dans les tests. En revanche, le développeur reste encore trop souvent timide sur ces techniques de développement, de conception.

Dans ce dossier, nous allons nous focaliser sur plusieurs points : le Model Driven pour le Développement (MDD), le Model Driven et les développeurs, la modélisation de bout en bout d'un projet, et pour finir, nous reviendrons sur une question, trop souvent oubliée ou éludée : l'interopérabilité des modèles entre les différents outils de modélisation. Or, ce dernier point mérite autant de prudence que le choix d'un outil en lui-même.

Au final, est-ce un bien ou un mal pour le développeur ? Ces outils, ces techniques peuvent lui permettre de se décharger du codage fastidieux ou récurrent comme les requêtes SQL, la couche d'accès aux données. Cela permet d'emblée au développeur de se concentrer sur le code « utile ». Mais attention, la modélisation, le Model Driven, cela ne va pas de soi. Ce n'est pas une formule magique. Il y a toute une phase d'apprentissage, de formation. Le retour sur investissement ne se fait pas d'un clic de souris.

Malgré tout, le développeur aurait tort de ne pas considérer positivement cette aide. Car, avec des projets de plus en plus hétérogènes, complexes, aux contraintes d'intégration fortes, une approche pilotée par des modèles, la modélisation peut grandement faciliter son travail même s'il ne faut pas en attendre de miracles. En revanche, l'arrivée de Microsoft avec le support officiel de UML (même partiel) peut redynamiser un marché qui a parfois du mal à cibler le développeur.

Alors pourquoi ne pas tenter l'aventure UML et Model Driven ?

■ François **Tonic**

Développez agile avec le MD de LEONARDI

L'ingénierie pilotée par les modèles, ou MDD (Model Driven Development) est l'une des avancées les plus marquantes de la dernière décennie en matière de logiciel. L'encapsulation prônée par les technologies objet dans les années 80-90 et les gains en productivité qu'elle permet avaient forcé les développeurs d'applications à penser différemment, la programmation objet prenant le pas sur le procédural. De même, l'arrivée à maturité de l'approche MDD, qui bâtit l'application autour du modèle, se manifeste par la mise sur le marché d'outils visant à augmenter la productivité des équipes projet.

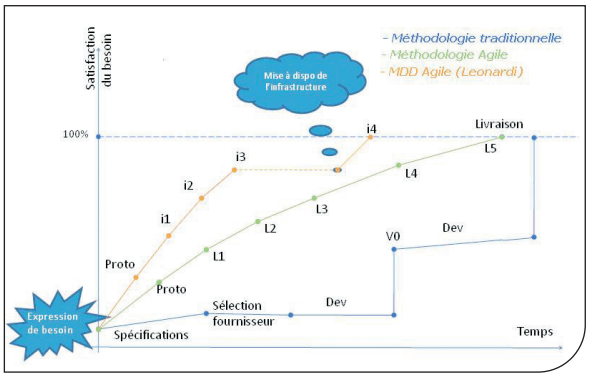
Pour tirer le meilleur profit de cette approche et de l'agilité qu'elle peut conférer, la méthodologie de mise en œuvre sur les projets doit être adaptée. Ici, nous rappelons les principaux bénéfices de l'approche MD et expliquons comment l'organisation des projets est affectée par rapport aux projets traditionnels (Waterfall ou cycle en V).

Bénéfices et impact du MDD sur l'organisation des projets

Lorsque l'on veut développer une application métier intégrant le SI, la logique fonctionnelle et l'IHM, l'expérience prouve que si l'on peut souvent se passer d'étude de faisabilité, de spécifications formelles ou d'une conception détaillée de l'architecture, le modèle de données reste un passage obligé. Forts de ce constat, les éditeurs logiciels se réclamant du MDD proposent des produits pour définir des modèles élaborés depuis lesquels on peut déduire de manière quasi-automatique une application opérationnelle répondant aux besoins exprimés. Comment ? Soit par génération de code, avec des outils tels que oAW, Acceleo d'Obeo, Mia Generation de Mia-Software, MDWorkbench de Sodius, BluAge de Netfective ou Celerio de Jaxio, soit par interprétation directe du modèle, avec Leonardi de W4, CodeFluent de SoftFluent ou OlivaNova de Care Technologies. Ainsi, il devient aisé de produire un logiciel opérationnel validé rapidement par les utilisateurs finaux et les clients. Pour éviter le tout UML, de nouvelles manières de modéliser le domaine, plus conviviales, sont offertes : découverte automatisée des SI, spécialisations GED, utilisation d'ateliers avec fonctionnalités pour le mode collaboratif et la traçabilité (voir Modelio d'Objecteering Software…), qui rendent possible la tâche aux intervenants fonctionnels n'ayant pas forcément une connaissance approfondie des techniques formelles de modélisation.

Cet effort de simplification permet de bien dissocier les rôles sur un projet : les éditeurs prennent à leur charge la partie technologique de l'application alors que l'intégrateur se concentre, avec l'implication forte du client, sur la partie fonctionnelle. Ainsi, la connaissance du métier, des données manipulées et des règles qui les régissent prend une part prépondérante lors du développement : le gros des efforts peut porter sur les tâches où réside réellement le savoir-faire d'une entreprise. Les gains de productivité sont incontestables : les applications sont prêtes plus rapidement, leur qualité et leur robustesse accrues et leur indépendance relative aux technologies sous-jacentes les rendent plus pérennes. De plus, l'un des atouts considérables du MDD est l'allégement significatif de la maintenance. En effet, quand les besoins ou les spécifications changent, il suffit de mettre à jour le modèle et l'application peut évoluer de manière quasi transparente.

Sur ce dernier sujet primordial, la présentation d'Andrew Watson, CTO de l'OMG à MD DAY 2009 est édifiante. Elle indique que le coût de la maintenance dans les applications logicielles croît régulièrement (90 % du coût total en 2000, Erlikh, 2000) cela étant dû au fait que les équipes passent plus de la moitié de leur temps à



comprendre le code. Andrew Watson préconise l'approche MDD comme un remède efficace. Plusieurs études renforcent ce constat dont l'une porte sur un système centralisé de réservations de la société Disney, où 2 équipes travaillaient en parallèle sur le même projet, la première utilisant une approche traditionnelle et la seconde suivant les principes du MDD. Les résultats obtenus par la seconde équipe ont été six fois plus efficaces en tenant compte, a posteriori, de la productivité et de la qualité de l'application.

Le MDD, allié des nouvelles méthodologies agiles

En 2001, le manifeste Agile jetait les bases du développement agile, en énumérant des concepts pragmatiques pour la réussite d'un projet informatique. Dans la lignée de RAD et de DSDM qui préconisaient déjà dans les années 90 une approche itérative, incrémentale et adaptative, ces concepts ont favorisé l'émergence de diverses méthodologies agiles, dont la plus répandue aujourd'hui est Scrum, qui fixe un cadre organisationnel pour les projets et qui peut favorablement s'allier avec des méthodes plus axées

sur la technique, telles que XP, où l'accent est mis sur les bonnes pratiques de construction de l'application : programmation en binôme, appropriation du code par l'équipe, refactorisation et intégration continue etc.

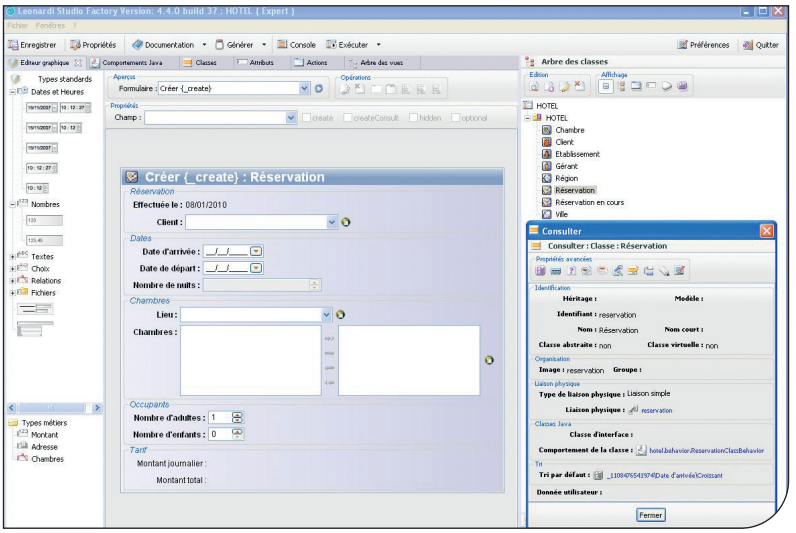
En quoi le MDD est-il agile ? Si l'on se réfère aux principes fondateurs du manifeste agile, il favorise clairement leur mise en œuvre :

- **Les individus (pas les outils) sont au centre du processus de développement.** Le modèle est le moyen de communiquer entre intervenants pluridisciplinaires du projet : ingénieurs, concepteurs, clients et services marketing en viennent à utiliser naturellement cette abstraction du monde métier pour collaborer et obtenir un résultat conforme aux attentes.
- **Le logiciel est opérationnel :** l'industrialisation des applications à partir du modèle favorise un résultat opérationnel de qualité, où les services escomptés sont présents pour accompagner les données modélisées.
- **Le client est impliqué tôt et fortement**. Avec le MDD, les résultats sont montrables rapidement : pour un prototype initial, on peut utiliser un modèle simplifié du domaine que l'ont fait évoluer par itérations.
- **Le processus de développement répond au changement :** en concentrant dans le modèle la richesse du métier, les applications restent flexibles et évolutives. Quand de nouvelles technologies apparaissent, l'évolution des outils et leur compatibilité ascendante permettent aux applications en place d'en bénéficier.

On le voit donc bien : les concepts du MDD sont particulièrement propices à une application des méthodes agiles. Sans appliquer à la lettre ces méthodes, le MDD conduit naturellement à l'agilité. Leonardi en est une illustration.

Leonardi, commercialisé par W4, existe depuis 2001. Sa version V4.4 vient de sortir, intégrant notamment des améliorations notables en version web, une API de composants (flux RSS, SMS, email…) et son orientation SOA avec la possibilité d'exposer le modèle en tant que web services (voir www.lyria.com pour plus d'informations).

Leonardi V4.4



Cette offre s'inspire directement du MDD : le modèle métier décrit la structure des données manipulées et est enrichi par des informations de connexion aux données du SI et par des informations exploitées lors de la production automatisée de l'interface utilisateur. Il n'y a donc pas ici génération de code, mais exécution directe du modèle par un moteur spécialisé qui produit dynamiquement les vues demandées par l'utilisateur.

Contrairement au MDD traditionaliste où il faut souvent créer des modèles élaborés, mis au point par des spécialistes, avant de générer le code applicatif, Leonardi favorise la création de modèles intermédiaires (non finalisés) qui jouent un rôle fondamental pour orienter les efforts de développement en impliquant le client lors d'itérations successives. Avec son atelier de conception Studio, le MDD est à la portée de tous : plus besoin de connaître MOF, XMI, MDA, les Software Factories, DSL ou même UML pour créer un modèle riche et complet.

Les PIM et PSM du MDA (Model Driven Architecture) sont ici confondus en un seul modèle logique, qui devient assez abstrait pour exprimer l'IHM indépendamment de la plate-forme d'exécution cible, mais qui reste assez concret pour générer automatiquement les écrans en fonction de l'action demandée par l'utilisateur, au moment où il la demande sur la plate-forme cible. Avec Leonardi, les spécifications sont réellement exécutables puisque ces dernières résident en grande partie dans le modèle, et que le modèle est exécuté par le moteur. On peut ainsi créer des applications cœur de métier évoluées, sans spécifier de manière formelle des écrans. La non-génération de code accroît encore l'agilité car la gestion de la cohérence entre le modèle et le code est réduite au strict minimum.

Cette approche n'adresse pas la problématique d'écrans sur mesure, travaillés au pixel près pour des besoins pointus et spécialisés. C'est une approche plus industrielle (de type « prêt à porter ») du développement de l'IHM qui est visée ici. Elle accroît considérablement la productivité et industrialise véritablement la production des applications cœur de métier, tout en les rendant pérennes et évolutives et en facilitant leur maintenance.

Conclusion

Le MDD montre son pragmatisme sur le terrain et son utilisation se généralise. Cette tendance est d'ailleurs confirmée par une étude récente du Gartner (décembre 2009) qui prédit que la composition, définie ici comme « des assemblages orchestrés de données, de processus et de services », remplacera graduellement les applications en tant qu'implémentation première des systèmes logiciels métier. L'étude indique aussi que le MD devient l'une des approches principales de création et de maintenance des systèmes.

■ Jean-Loup **Comeliau**

Quelle interopérabilité entre les ateliers de modélisation ?

L'interopérabilité est depuis l'origine un sujet sensible pour les ateliers de modélisation. L'enjeu pour les utilisateurs est de ne pas être verrouillé dans un atelier, en ayant l'ensemble de ses travaux et modèles stockés dans un format propriétaire spécifique.

Le deuxième enjeu est de pouvoir partager ses modèles avec d'autres équipes ou organisations utilisant des ateliers différents. Un dernier enjeu important enfin, est de faire coopérer des ateliers de nature différente rendant des services différents, comme par exemple modélisation, simulation, exécution, etc. Il faut alors être capable d'échanger des modèles entre ces ateliers, avec une garantie d'absence de perte sur le contenu.

L'échange de modèles UML : le bénéfice du standard

UML étant un standard, on pourrait naïvement en déduire que tous les ateliers le supportant sont capables d'échanger leurs modèles. Las, la réalité est beaucoup moins rose, et il est loin d'être garanti que deux ateliers UML échangent correctement leurs modèles.

Très tôt après la standardisation des premières versions UML, l'OMG (organisme standardisant UML) s'est soucié de l'interopérabilité des modèles, et a produit un standard d'échange bâti sur XML : XMI.

Différentes versions XMI 1.x ont donc été émises par l'OMG pour assurer l'interopérabilité. Avec le recul, on peut qualifier d'échec ces premiers standards : Pour que deux ateliers échangent via XMI, il faut qu'ils s'appuient sur la même version d'XMI et la même version d'UML. Par ailleurs, de nombreuses ambiguïtés dans le format d'échange ont conduit les outilleurs à avoir des interprétations différentes. Échanger des modèles en format XMI 1.x était ainsi une pénible course d'obstacles pour l'utilisateur, avec un résultat incertain. Cet échec a conduit l'OMG à réaliser un nouveau standard d'échange : XMI 2.1, plus compact, plus lisible et mieux formalisé pour éviter les ambiguïtés. XMI 2.1 s'applique à UML version 2.x. XMI 2.1 a indubitablement amélioré les choses. Cependant, des différences d'interprétation existent toujours, faisant courir des risques sur le succès d'opérations d'échange de modèles. Pour complexifier davantage la situation, la plate-forme open source Eclipse EMF (Eclipse Metamodel Factory) délivre un support d'UML 2 en format XMI... qui n'est pas tout à fait conforme au standard OMG XMI 2.1, EMF ayant divergé du standard OMG MOF pivot pour les métamodèles. Un certain nombre d'ateliers ont été construits sur cette infrastructure open source EMF.

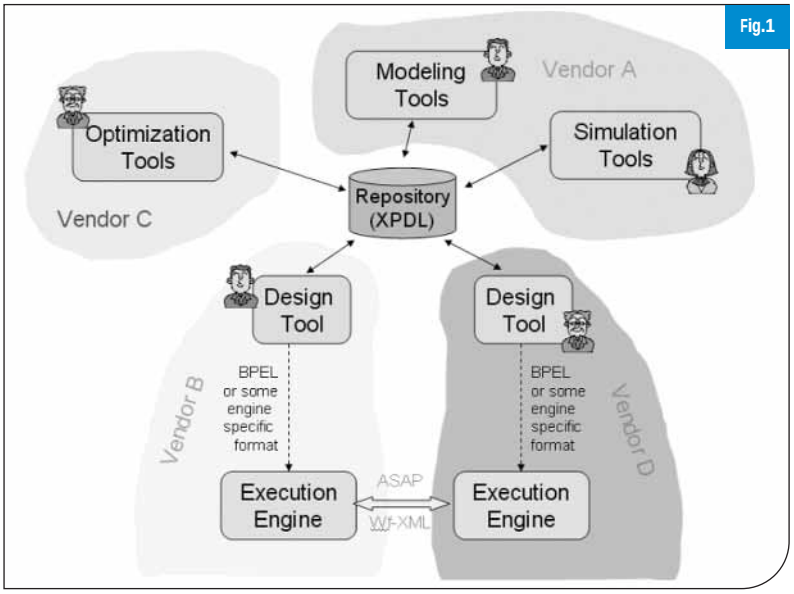
Depuis l'été 2009, l'OMG a lancé une initiative appelée « Model Interchange Working Group » (MIWG) destinée à mettre en place des tests d'interopérabilité standard, pour certifier les ateliers interopérables. Ce travail s'effectue en coopération avec des outilleurs - Artisan Studio, IBM Rhapsody, IBM RSx, SOFTEAM Modelio, No Magic MagicDraw, Sparx Systems Enterprise Architect – (voir l'annonce presse OMG http://www.omg.org/news/releases/pr2010/01-04-10.htm) qui valident l'interopérabilité par des échanges « pair à pair » entre ateliers, et par des import/export avec des documents XMI de référence.

A ce jour, 9 jeux de tests ont été mis en œuvre avec succès par ces outilleurs, et des démonstrations ont été effectuées.

Il faut ajouter que cet effort a été fait sous la pression du gouvernement américain, qui dans ses appels d'offres exigera cette interopérabilité prouvée. Le NIST (US National Institute of Standards and Technology) fournit un outil de validation XMI pour mieux certifier la validité du processus de certification.

Pour avoir une bonne garantie d'interopérabilité entre ateliers UML, il est donc recommandé de choisir un atelier parmi ceux qui ont participé à ce groupe de travail : ils ont validé l'interopérabilité effective des modèles. Cette interopérabilité couvre naturel-

Fig.1



XPDL assure l'interopérabilité entre la grande variété d'ateliers existant dans le monde BPM

lement les extensions UML qui peuvent être standard (comme par exemple SysML ou SoaML) ou propriétaires, grâce au mécanisme des profiles UML.

Certains ateliers, comme par exemple Modelio (www.modelio.fr), permettent un export / import XMI au format standard OMG, ou au format UML2 EMF, offrant ainsi une interopérabilité dans les deux mondes.

Il reste un aspect non couvert par l'interopérabilité des ateliers : les diagrammes. En effet, XMI permet d'échanger les modèles, mais n'échange pas les diagrammes. Concrètement, après un échange, vous verrez vos modèles sous l'explorateur de votre atelier, mais les diagrammes sont à recréer manuellement, ou semi-automatiquement via des aides de l'atelier. Tous les travaux de présentation faits sur le modèle antérieur sont perdus.

L'OMG travaille sur ce problème, en mettant en place un standard dédié à l'échange de diagrammes. La réponse au cahier des charges (Diagram Definition RFP) est attendue pour fin mai 2010. Serons-nous au bout de nos peines ? Pas encore ! Ce standard précise comment les diagrammes doivent être définis pour être interopérables, de manière générique (niveau MOF) c'est-à-dire pour tout modèle comme par exemple, UML, BPMN, ou d'autres modèles. Pour que ceci fonctionne, il faut que les standards désignés spécifient précisément les diagrammes et représentations graphiques en suivant ce nouveau formalisme. Donc, il ne faut pas attendre de standard finalisé pour UML avant… 2012, en estimant au mieux des implémentations dans les outils pour fin 2012. En résumé, nous avons une solution d'échange des modèles UML qui fournit un niveau de service validé et intéressant. Pour une capacité complète d'échange, il nous faut encore un peu de patience.

Les ateliers EA / BPM, la voie BPMN

Les ateliers de modélisation de processus métier, et les ateliers de modélisation d'architecture d'entreprise ont été historiquement basés sur des modèles propriétaires. Il est donc très difficile de parler d'interopérabilité en ce qui les concerne. Bien entendu, des ponts « pair à pair » ont été bâtis, notamment vis-à-vis des plus grands acteurs du marché, pour fournir des solutions ad hoc. Une difficulté est que la plupart du temps, ces ateliers sont extensibles, et que les utilisateurs peuvent adapter les modèles et notations. Dans ce cas, inutile d'espérer une solution clé en main d'interopérabilité. Il faut développer des ponts spécifiques selon les besoins. Heureusement, la quasi-totalité des ateliers offre un import / export XML, ce qui facilite techniquement le travail.

BPMN (Business Process Modeling Notation) est un standard OMG très populaire pour la modélisation des processus métier. Ce standard doit mécaniquement améliorer l'interopérabilité. Cependant, la version actuelle (BPMN 1.1) est simplement un standard de notation, sans métamodèle. Ceci ouvre la porte à des interprétations divergentes des outilleurs, qui chacun ont leur propre métamodèle, et des notations adaptées.

Un standard indépendant de BPMN a néanmoins été défini pour assurer l'interopérabilité des modélisations de processus métier. Il couvre à la fois des modèles propriétaires de processus et BPMN1.1. Il s'agit de XPDL (XML Process Definition Language – standard WfMC), qui bénéficie de plus de 70 implémentations dans le monde. L'enjeu est important, du fait de la grande variété d'outils pouvant interopérer dans le contexte des processus métier : Outils de modélisation, outils de conception dédiés à des moteurs Workflow, outils d'optimisation, de simulation, etc. [Fig.1]

Dans sa version 2.1, XPDL offre une couverture spécifique de BPMN1.x, et est de facto le format d'interopérabilité utilisé. XPDL permet également d'échanger les diagrammes BPMN, avec cependant deux problèmes :

1 - il couvre mal les disparités entre outils ;
2 - il attache le diagramme à la sémantique, ce qui rend très délicat les modèles ayant plusieurs diagrammes représentant le même élément. XPDL est la solution opérationnelle. Elle permet par ailleurs d'attacher des compléments XSLT, pour attacher au format standard des adaptations liées à telle ou telle solution propriétaire.

L'OMG vient de définir le standard BPMN2.0, pas encore réellement implémenté, qui étend les capacités de BPMN, définit enfin un métamodèle pour BPMN, et fournit un format d'interopérabilité basé sur XMI. XPDL va évoluer de son côté pour couvrir BPMN 2.0. Nous nous retrouvons donc à terme avec deux standards d'interopérabilité ! Ceci se résoudra probablement par le support des deux standards par les outilleurs, ou par des convertisseurs d'un standard à l'autre, ou par le non-support d'un des deux standards. XPDL, qui reste historiquement le standard adopté par les outilleurs, sera probablement la solution préférée d'interopérabilité.

La voie vers une solution complète est ouverte

Qu'en est-il de la coopération BPMN / UML ? C'est un cas important, car BPMN ne supporte pas la modélisation des données, et n'est donc pas capable de relier les objets manipulés dans un processus, pas plus qu'il ne modélise les acteurs ou entités responsables des tâches modélisées. Certains ateliers comme Modelio (www.modelio.fr) fournissent un support intégré BPMN / UML, mais sont obligés, du fait d'une absence de standard sur ce point, d'apporter leur solution. BPMN et UML peuvent être exportés vers d'autres environnements, mais pas leurs liens d'intégration. L'interopérabilité a beaucoup progressé pour les modèles standardisés. Les solutions actuelles ont atteint une bonne maturité et rendent de grands services aux utilisateurs. Il reste les problèmes liés aux diagrammes et à l'intégration des standards, notamment BPMN et UML.

Je fais le pari que ces problèmes seront derrière nous dans une dizaine d'années.

■ Philippe **Desfray** – *SOFTEAM* – *philippe.desfray@softeam.fr*

MODEL2CODE

Agile Development in action!

CERTAINS AGL VOUS PROMETTENT DE DÉVELOPPER 10 FOIS PLUS VITE?

ET SI VOUS VOUS LAISSIEZ... DIRIGER PAR LES MODÈLES ?

NE DÉVELOPPEZ PLUS, MODÉLISEZ !

Passez à la vitesse du Model Driven!

Maquettez vos IHMs en HTML et modélisez simplement vos processus métier sous forme de diagrammes UML, nos générateurs les transforment instantanément en application Spring, Java EE ou Flex, prête à être déployée !

Nouveau ! Plugin CRUD Booster

Créez en quelques minutes vos prototypes et accélérez considérablement vos projets applicatifs. Générez une application CRUD fonctionnelle (y compris les IHMs) à partir d'un simple diagramme de classe !

VERSIONS D'ÉVALUATION GRATUITES & TUTORIAUX :

WWW.MODEL2CODE.COM

Model2Code propose des ateliers agiles et collaboratifs de génération d'applications web et riches, combinant les meilleures technologies Model Driven du marché : l'outil de modélisation UML Magicdraw® associé au moteur de génération par transformation de modèles BLU AGE®.

magicdraw™ BLU AGE AGILE MODEL TRANSFORMATION

BLU AGE Software
Immeuble le Gabriel Voisin - 79, rue Jean-Jacques Rousseau 92158 Suresnes cedex FRANCE
Tel +33 1 56 05 60 91 Fax +33 1 56 05 88 01

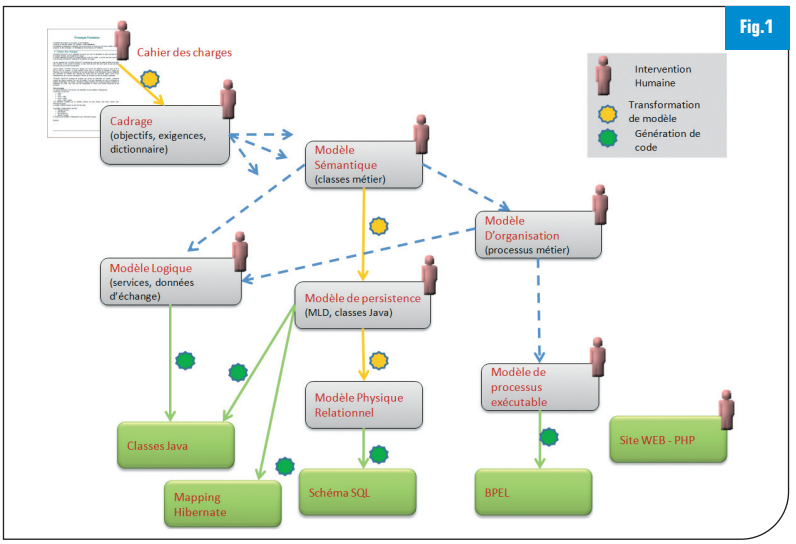
www.model2code.com

Modélisation de bout en bout d'un projet : de la vision de l'entreprise à l'implémentation dans son SI

Les avancées des techniques de modélisation, MDA (Model Driven Architecture) et les outils de modélisation permettent de supporter l'ensemble de la démarche, depuis la définition des orientations de l'entreprise, en passant par la formalisation du métier, par celle de l'architecture technique, puis par le développement, pour aboutir à la réalisation guidée par le modèle de tout ou partie d'un SI.

Pour assurer une continuité de la démarche sur toute la portée d'une entreprise, faire en sorte que les différents intervenants communiquent et soient en phase, et garantir que le résultat final corresponde aux attentes, comment faut-il procéder ? Il est nécessaire de s'appuyer sur une démarche générale d'entreprise, par exemple la méthode ouverte Praxeme (www.praxeme.org) ou d'autres pratiques s'appuyant sur le cadre méthodologique TOGAF. Il est également recommandé de mettre en œuvre un atelier de modélisation unique couvrant toute la portée : besoin, objectifs, terminologie, règles métier, architecture d'entreprise, logiciel. Dans cet article, nous allons nous intéresser à un cas d'étude illustrant la modélisation de bout en bout, en s'appuyant sur un cahier des charges publié par un organisme indépendant - le Ceisar (www.ceisar.org) dédié à l'évangélisation de l'architecture d'entreprise -. Ce cas d'étude s'appuie sur l'atelier de modélisation Modelio (www.modelio.fr), du fait qu'il supporte toute la portée de modélisation, qu'il intègre la technologie MDA et fournit un ensemble de générateurs permettant d'automatiser le codage et la documentation.

Le cas d'étude porte sur le support de réservations de voyages via internet pour une agence de voyage factice appelée « DiscountVoyages ». Un white paper détaillé, le cahier des charges Ceisar, une base de modélisation complète Modelio, ainsi que l'application résultante peuvent être téléchargés sur (http://www.modelio-soft.fr/tutorials-fr/model-realise-soa-application-fr.html).



Différents types de modèles, liaisons, interventions automatiques ou manuelles

La solution technologique s'appuie sur une architecture SOA, une application Java, l'utilisation de Hibernate et MySQL, la mise en place de services WEB, la mise en œuvre de BPEL, et l'utilisation du serveur d'application open source Glassfish (NetBeans).

La séquence des modèles aboutissant à la solution

La Figure 1 représente schématiquement les différents modèles manipulés, et les types d'intervention et liaisons existant entre ceux-ci. Le Cadrage et le modèle Sémantique permettent de guider totalement le reste de la modélisation. Plus on affine le modèle et on s'approche de la solution, et plus l'automatisation devient conséquente. Ainsi, des parties importantes du modèle logique (données d'échange) et du modèle de persistance sont automatiquement déduites. Le modèle physique relationnel est intégralement généré par transformation de modèle. De même, les classes Java pour les services, la manipulation des données d'échange, et l'accès à la base de données sont intégralement déduits par génération.

Le cadrage : Les modèles de l'entreprise et du système d'information doivent « coller » au cahier des charges. Une première phase dite de « cadrage » va permettre à partir de ce document initial d'identifier les objectifs de l'entreprise, les termes du métier révélant les notions principales de celui-ci, les règles métier devant être suivies par l'entreprise et son SI, ainsi que les exigences de l'entreprise sur son SI. Le modèle sera ainsi construit sur ces éléments préliminaires de cadrage, en se référant à eux grâce à des liens de « traçabilité » supportés par Modelio. On commence tout simplement par éditer le cahier des charges sous Word, en utilisant des commandes (macros) spécifiques

composants « processus » et marron pour les composants « entité »). Ils sont assemblés via les services requis ou les services fournis. Leur déduction est guidée par la démarche : les classes sémantiques saillantes, les plus importantes qui souvent fédèrent des classes satellites, sont traduites par des composants de service « Entité » dont l'objet est de gérer les accès et les modifications des données du référentiel métier. Il s'agit des composants Client, Voyage et Commande, en marron. Les processus métier devant être supportés par le SI seront traduits en composants processus (ici « Réservation Voyage »), et s'appuieront sur les composants entité pour réaliser leur traitement.

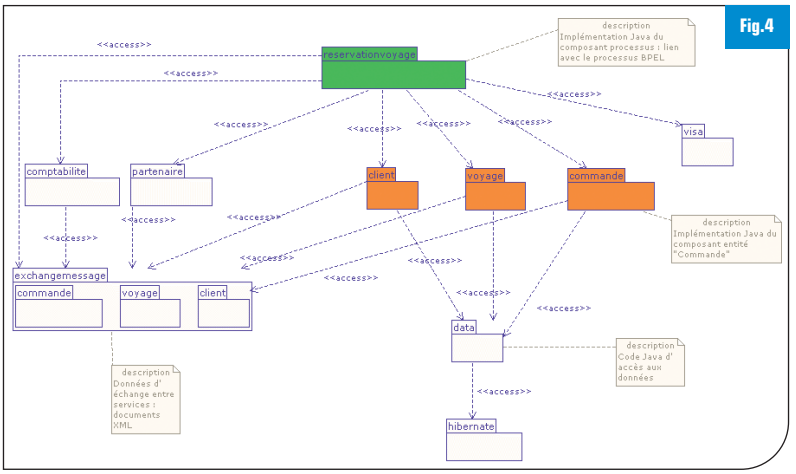
L'architecture logique est modélisée de manière détaillée, notamment pour définir précisément les services et les données d'échange. La signature des services est fournie complètement. Les données d'échange sont déduites du modèle sémantique, qui est retravaillé pour fournir les données attendues par les services, mises à plat lors d'échanges typiquement de documents XML.

Production de la base de données : Le modèle logique des données a été obtenu automatiquement sous Modelio en transformant le modèle sémantique. Il est retravaillé manuellement dans un but technique, pour respecter les formes normales, pour définir des identifiants et des clés primaires sur les éléments et pour optimiser la gestion du modèle en base.

Ce modèle est ensuite automatiquement transformé en modèle physique des données faisant apparaître des mécanismes relationnels, comme typiquement des clés étrangères.

Ce modèle peut être retouché pour des raisons physiques, d'optimisations spécifiques d'une base. Modelio maintient la cohérence entre le modèle logique et le modèle physique lors de modifications. Puis le code SQL est automatiquement généré. Le choix de la plate-forme Hibernate, permet à Modelio de générer automatiquement les classes Java d'accès aux données.

Nature de l'élément	Nombre dans l'application	Nombre généré depuis le modèle	% automatisation
Classes Java	40	38	95 %
Lignes de code Java	4035	2987	74 %
Lignes SQL	148	148	100 %
Lignes BPEL	251	251	100 %



Modèle logiciel de l'implémentation Java – Diagramme de Packages UML

Architecture logicielle – traduction du modèle logique : L'architecture logicielle reprend le modèle logique, en réalisant un modèle dédié à la cible technique. Ici, le choix des techniques XML, WSDL, BPEL (Glassfish), SQL (MySQL), Hibernate et Java, va conditionner la forme du modèle.

La figure 4 montre la structure de l'implémentation Java. On y retrouve l'implémentation des services pour les composants de service, avec les couches processus et entité.

Les données d'échange sont manipulées par des classes Java et sérialisées en XML. La couche « Data » gère l'accès à la base de données relationnelle.

Le code Java d'accès aux données est intégralement déduit du modèle logique de données. Il produit des classes Java et un « mapping Hibernate » d'accès à la base. Le code Java de manipulation des données d'échange (XML) est intégralement déduit du modèle des données d'échange réalisé avec l'architecture logique. Il permet de sérialiser/désérialiser les données d'échange, et de les manipuler depuis le code Java. En ce qui concerne l'implémentation des services WEB, le squelette du code, avec la signature des opérations de service est déduit du modèle logique des services. Il faut ajouter des compléments de programmation, afin de décrire quels traitements réalisent ces services. Dans ce cas d'étude, il s'agit essentiellement d'accéder aux données en base. Avec Modelio, il est possible d'éditer directement sous l'atelier le code Java, ou d'utiliser un environnement comme Eclipse, le code étant toujours maintenu de manière synchrone avec le modèle.

Le composant « RéservationVoyage » est un composant processus développé avec le langage BPEL. Il a été modélisé en BPMN sous Modelio, pour le générer en BPEL.

La partie IHM WEB a été réalisée manuellement en PHP, et a utilisé les services WEB.

Bilan

Ce cas d'étude complet montre comment modéliser de bout en bout, depuis la définition des objectifs d'une entreprise, en passant par l'architecture d'entreprise, les besoins, l'architecture SOA, le logiciel pour aboutir au code résultant. Le tableau ci-contre résume la volumétrie de code.

Ce cas d'étude peut être consulté en détail sur (http://www.modeliosoft.fr/tutorials-fr/model-realise-soa-application-fr.html).

Des exemples de transformations de modèle et de capacités MDA sont visibles sur la vidéo http://www.modeliosoft.fr/modules-fr/modelio-mda-designer-fr.html.

Nous remercions le Ceisar pour nous avoir autorisés à publier leur cahier des charges.

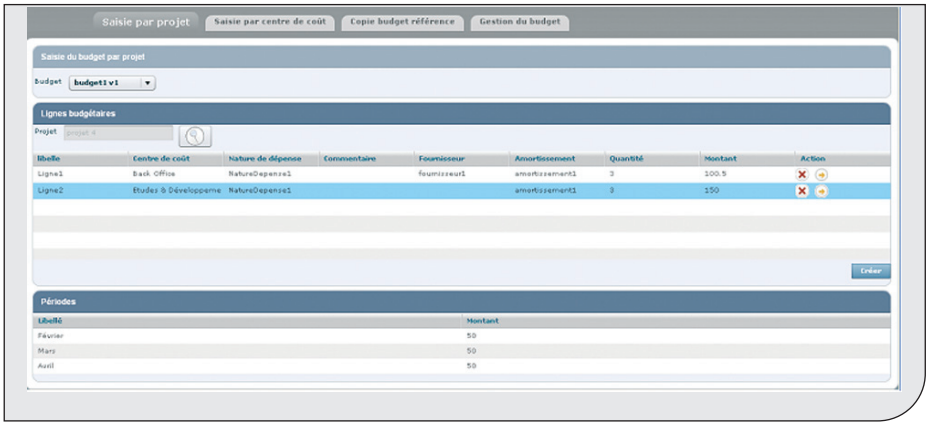
■ Philippe **Desfray**
Directeur R & D SOFTEAM
philippe.desfray@softeam.fr

Retour terrain
Manpower France : cap sur le Model Driven

Aujourd'hui, à l'informatique de Manpower France, **Hélène Monin** (Directeur des Etudes et Développements à la Direction de l'Innovation Opérationnelle et des Systèmes d'Information) s'appuie sur une équipe IT de compétences mixtes JEE & Mainframe.

« Nous avons récemment adopté une démarche Model Driven et UML. Cette démarche était essentielle pour nous. Nous n'avons pas de véritable maîtrise d'ouvrage et le métier et l'informatique avaient du mal à se comprendre », précise Hélène Monin.

« Nous avons réfléchi sur la manière de mieux nous comprendre pour être plus efficaces dans l'évolution du SI. Le Model Driven et la modélisation nous ont paru être un bon moyen pour construire les applications sans parler d'outils aux utilisateurs mais d'abord adéquation de la solution à leurs besoins fonctionnels », poursuit Hélène Monin. En parallèle de la refonte du socle technique JEE, une réflexion globale s'est engagée après la réalisation réussie d'un premier projet en production depuis fin 2008, conçu au forfait, avec la démarche MDA et l'UML par Netfective. Les deux approches pour Hélène Monin vont ensemble. Un programme de formation a ainsi été construit pour les équipes car *« évidemment, il ne suffit pas de quelques clics pour générer le code »*. Aujourd'hui, un premier projet pilote MDA et UML est en cours depuis novembre. Dans quelques semaines un nouveau projet métier stratégique démarrera. *« Dès à présent, nous avons prévu de partager cette expérience avec nos collègues informaticiens américains »,* conclut Hélène Monin.



2010: l'Odyssée du Model Driven

La précédente décennie aura vu la modélisation et le Model Driven passer des mains des universitaires et des chercheurs dans celles de l'industrie logicielle. La prochaine verra certainement l'abandon des techniques et outils classiques de développement au profit des méthodes agiles et d'un outillage complètement dirigé par les modèles, replaçant enfin l'utilisateur et les besoins métiers au cœur des projets applicatifs. La virtualisation de serveurs, grâce au principe d'abstraction des ressources matérielles, a permis de simplifier considérablement l'exploitation et la tolérance aux pannes des Datacenters. Le Model Driven apporte cette même notion d'abstraction entre la conception des applications et leur réalisation technique par transformation successive de modèles. Au même titre qu'une machine virtuelle est entièrement portable d'un serveur physique à un autre, il est désormais possible de transformer automatiquement et intégralement une même représentation graphique (diagrammes) des besoins métiers en une application Java EE ou. Net, implémentant les frameworks de son choix : Spring, Hibernate, Struts, Seam, etc. Les directions informatiques et les développeurs peuvent capitaliser sur le modèle, et plus sur des architectures ou compétences le plus souvent éphémères…

Projections ou réalités ?

Jusqu'en 2007/2008, l'outillage de développement dirigé par les modèles disponibles sur le marché n'était pas suffisamment industrialisé pour permettre aux utilisateurs et aux développeurs un usage en toute autonomie : les projets applicatifs étaient donc le plus souvent conduits et réalisés par les concepteurs de ces outils — éditeurs ou SSII -, seuls capables de prendre en main ce qui ressemblait davantage à une "boîte à outils" pour experts. Les acteurs du Model Driven ont depuis redoublé d'efforts pour rendre accessible la technologie, tant d'un point de vue simplicité d'usage que tarifaire, transformant ce marché de services en véritable marché de logiciels. Pour sa part, Blu Age Software, en partenariat avec l'américain No Magic (éditeur du modeleur UML MagicDraw), a récemment contribué avec succès à cette évolution du marché, en lançant une gamme verticale de générateurs d'applications (M2Spring, M2Flex et M2Code for Java), pouvant être librement évaluée et prise en main avec un minimum d'investissement, notamment grâce à la diffusion de tutoriaux en ligne et à l'intégration de nombreux assistants (wizards) au sein des produits. Grâce au module CRUD Booster, un débutant en UML peut par exemple désormais générer une application CRUD complète d'une trentaine d'écrans (y compris les IHMs) en moins de deux minutes, et ce à partir d'un simple diagramme de classe. Cette maturité de l'outillage Model Driven permet d'ores et déjà sa diffusion et son utilisation directement sur le poste de travail des utilisateurs finaux. Preuve très concrète de la maturité du marché pour Blu Age : sur le dernier trimestre 2009, nous avons commercialisé nos premières licences sur le sol américain (pourtant réputé hostile au MDA…) via notre partenaire No Magic, et démarré un projet conséquent de modernisation d'application Legacy, réalisé conjointement avec HP Services et outillé par notre technologie MDD. Autre preuve de l'essor de ce marché : des géants américains de l'industrie du logiciel (tel qu'Adobe, avec le prochain Flash Builder 4) introduisent désormais des fonctionnalités Model Driven au sein de leurs gammes. L'IDE est (bientôt) mort, vive le développement dirigé par les modèles !

■ Harold **Licari** –
Directeur Channel et Marketing – Blu Age Software
www.model2code.com *et* www.bluage.com

Le « Model-Driven » et le développeur

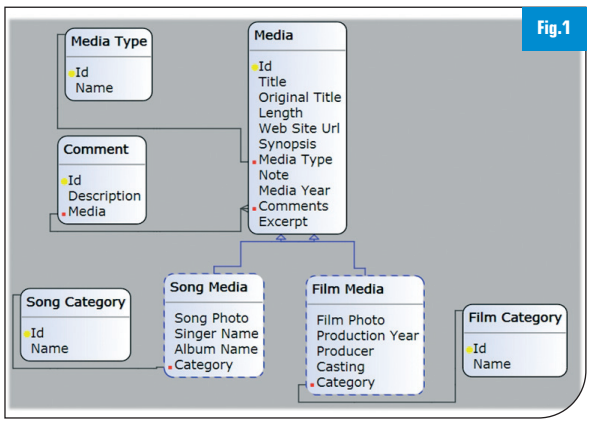
Impossible d'évoluer aujourd'hui dans le métier du développement et d'échapper à la vague du « Model-Driven » qui se profile, et deviendra probablement la norme des projets de développement au cours de la décennie 2010. Le succès grandissant de l'événement MD Day (http://www.mdday.fr), qui réunit les 10 acteurs majeurs du secteur témoigne de cette tendance.

Qu'est-ce que le Model-Driven ?

La raison fondamentale de ces nouvelles approches est que l'évolution technologique ne cesse de s'accélérer, et tout le monde s'accorde aujourd'hui sur le fait que cette évolution ne peut que se poursuivre. Dans ce contexte, les entreprises se doivent de trouver des solutions leur permettant de pérenniser leurs investissements en modélisant les concepts de leur métier de la façon la plus indépendante possible des technologies. Prenons l'exemple d'une application de gestion de médiathèque, vous permettant de gérer vos films et musiques. L'approche pilotée par les modèles propose de décrire les éléments fonctionnels selon une représentation « objet » des concepts, sans se lier à une quelconque technologie.

Dans notre cas, nous aurons une entité Media, de laquelle hériteront les entités FilmMedia et SongMedia. Ces entités ont des relations vers les entités FilmCategory et SongCategory. Notre modèle est enrichi d'une notion de Type et de possibles Commentaires, ce qui donne le schéma ci-dessous : [Fig.1]

La réalisation de l'application selon une architecture ou une autre s'obtient ensuite en choisissant la cible d'architecture que l'on souhaite : web, client-riche ou intelligent (synchronisé avec des services), Office, SharePoint ou Silverlight.



Qu'est-ce que cela change pour le développeur ?

Concrètement, cela fait évoluer le métier du développeur vers une plus grande importance de la part de conception et vers un niveau d'abstraction plus élevé. Au lieu de devoir coder de nombreux éléments à la main, comme les requêtes SQL Server ou les méthodes de chargement de collections d'éléments, le développeur écrira une méthode CFQL comme par exemple :

```
load (string categoryName) Where
Category. Name = @categoryName
```

Concrètement, la procédure stockée qui chargera les données voulues sera générée de manière optimisée pour SQL Server ou Oracle suivant le producteur de base de données retenu (ou éventuellement les deux dans le cas d'un produit d'éditeur). Ces procédures stockées incluront les éventuelles jointures induites par le besoin. De même, la méthode d'exécution de cet accès sera automatiquement générée dans la couche métier, et potentiellement exposée en service Web si le producteur WCF est configuré. D'autres règles peuvent également être décrites au niveau du modèle avec l'avantage de pouvoir être vérifiées à différents niveaux selon les architectures cible. Par exemple, le bon formatage d'un e-mail suit une expression régulière relativement complexe qui nécessite un certain code de validation. Dans un outil Model-Driven tel que CodeFluent, il suffit de positionner une règle de type EmailValidate sur la propriété voulue et cette règle sera vérifiée potentiellement dans plusieurs couches de l'application. Cela permet par exemple de la vérifier au plus près sur un client-riche, en évitant un aller-retour serveur, tout en faisant aussi cette vérification sur le serveur afin de garantir l'intégrité quel que soit le client qui utilise le service.

Pour autant, le développeur continuera de devoir utiliser un environnement de développement tel que Visual Studio pour implémenter certaines fonctionnalités avancées qui ne sont pas directement générables automatiquement : envoi d'un e-mail, gestion d'un workflow, comportements personnalisés sur des événements particuliers.

Le Model-Driven est donc avant tout un moyen pour le développeur d'être plus productif et de concentrer son effort vers plus de valeur ajoutée, sans risque de se perdre dans des méandres techniques difficiles à maîtriser.

Le Model-Driven et les méthodes agiles

Nous avons vu plus haut le rôle prépondérant du modèle pour fabriquer une partie significative de l'application. Lorsqu'une application évolue, que ce soit au cours de sa mise au point où les spécifications sont rarement 100 % exhaustives, ou tout simplement parce que la vie de l'application pousse à étendre son périmètre fonctionnel, on est fréquemment amené à modifier ou ajouter certaines propriétés.

Cela est d'autant plus vrai lorsque l'on pratique les méthodes agiles, ou les spécifications sont faites par itérations successives et où le changement est prévu par conception.

Le Model-Driven prend alors tout son sens car il est aisé de faire une modification et de la répercuter dans l'ensemble des composants puisque cela se fait par simple regénération du modèle. Concrètement, dans le cas de notre application, nous avons

Modelio pour annoter dans le texte ce qui correspond à des *objectifs, termes du dictionnaire, règles métier* ou *exigences.* Par cette technique, le cahier des charges est analysé, et chaque fragment textuel intéressant est annoté en fonction de sa nature, puis, ces éléments sont importés sous Modelio, où un travail complémentaire est engagé pour compléter les éléments : ajouter des éléments absents, compléter les descriptions, etc. Modelio fournit des éditeurs tabulaires, des explorateurs et des éditeurs graphiques dédiés pour éditer, organiser et modéliser ces éléments.

Le modèle Sémantique : À partir des notions issues du cahier des charges, que l'on retrouve dans les termes du dictionnaire et des règles métier, le modèle sémantique est créé sous Modelio. On détermine à quel terme correspond une classe sémantique, quelles règles métier s'appliquent à telle classe (ou propriété de classe). Le modèle sémantique (voir www.praxeme.org) est un modèle de classe UML simplifié, qui est focalisé sur la modélisation des notions métier, indépendamment de l'organisation de l'entreprise, et de toute considération technique sur l'implémentation possible. Le modèle ci-dessous exprime que la classe Voyage dérive du terme Voyage (dictionnaire), et que sur cette classe s'appliquent plusieurs règles métier (ex : localisation voyage). Modelio intègre dans un même référentiel ces différents éléments que l'on fait ici apparaître sur un même diagramme. [Fig.2]

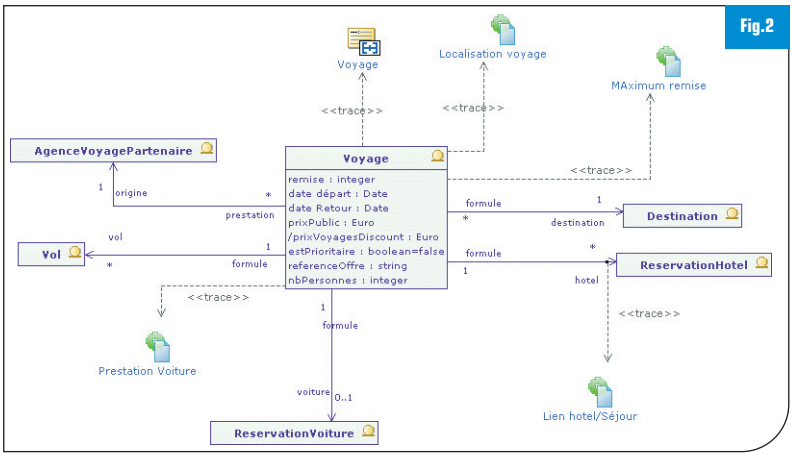


Fig.2

Vue détaillée de la classe Voyage

Chaque classe métier possède des états de gestion significatifs que l'on modélise par un diagramme d'état. Les transitions explicitent les liaisons possibles entre états.

Le modèle organisationnel de l'entreprise : On décrit l'organisation de l'entreprise, en représentant les rôles et leurs liens, les unités d'organisation, les flux échangés entre ces éléments. On trouve pour chaque rôle ses responsabilités, objectifs assignés, liens hiérarchiques, liens de communication.

La modélisation de leurs participations dans les processus métier complétera cette description.

Un modèle général des processus permet de positionner les processus métier, en représentant leurs événements et flux métier majeurs en entrée/sortie, et en indiquant leurs initiateurs et intervenants (rôles ou unités d'organisation). L'existence du modèle d'organisation permet d'identifier immédiatement les rôles et les « unités d'organisation » impliqués.

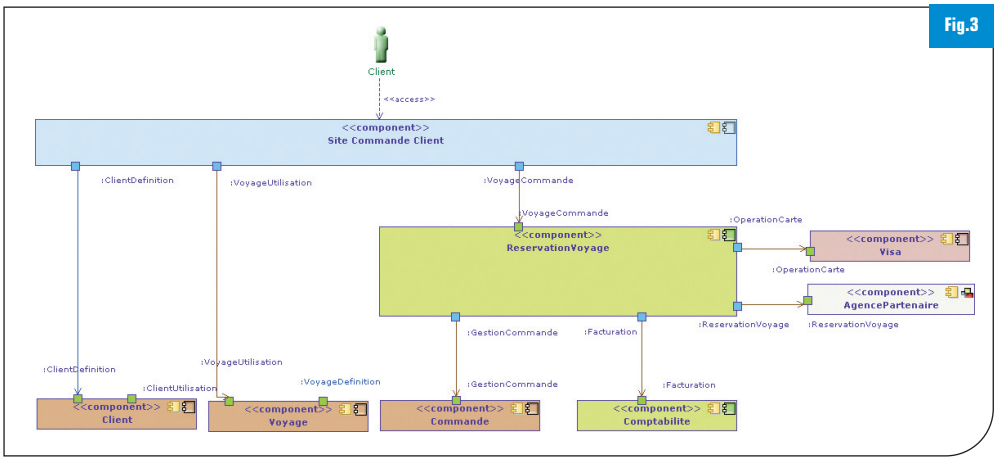


Fig.3

Architecture logique – partie liée au site de réservation de voyages

Les flux métier identifiés sur les diagrammes d'organisation sont également des éléments intéressants pouvant être repris. Les objectifs assignés aux processus sont détaillés pour en déduire les indicateurs de performance clés (KPI) correspondant pour les processus retenus.

Le modèle détaillé de chaque processus s'appuie sur le standard BPMN. Les « Lanes » (lignes) BPMN permettent de représenter la responsabilité des tâches. L'intégration UML/BPMN de Modelio permet d'associer ces « Lanes » aux rôles dans l'entreprise, ou aux unités d'organisation, qui sont les responsables habituels des tâches des processus. Les tâches à automatiser sont marquées, afin de préparer l'étude de l'architecture nécessaire à leur support.

Analyse des exigences : Les exigences sont recueillies à partir du cahier des charges du Ceisar, et modélisées sous Modelio. Elles sont ensuite modélisées plus en détail par des Use Case UML, ou directement tracées sur les parties de modèle qui les implémentent.

Architecture logique : L'architecture logique va définir la cible en organisant les grands modules logiciels du futur SI. Une approche s'appuyant sur une démarche SOA nous permet de définir une architecture évolutive, décomposée en composants autonomes et interchangeables, connectés entre eux via des « services ». [Fig.3]

Les composants de services sont répartis en différentes couches (ici, les couleurs : Bleu pour les composants « présentation », vert pour les

prévu une notion de commentaires sur les différents éléments de la Médiathèque. Au-delà de la zone textuelle prévue, nous pouvons vouloir ajouter une évaluation chiffrée. En ajoutant simplement une propriété « Note » à l'entité « Commentaire », et en re-générant l'application, cette nouvelle fonctionnalité sera gérée intégralement. Si l'interface utilisateur est réalisée spécifiquement et non générée, il sera au maximum juste nécessaire d'ajouter ce champ et de le lier. En l'absence d'un outil piloté par les modèles, il faudrait modifier la base de données, potentiellement en perdant des données de test ou en jouant des scripts (alors que CodeFluent gère le moteur de différences), retoucher à la main différentes procédures stockées manipulant les commentaires, intervenir au niveau de la couche métier, des couches de service et enfin de l'interface utilisateur. Cet exemple simple mais fréquent illustre bien l'apport des outils pilotés par les modèles pour suivre une démarche agile.

Le Model-Driven et l'évolution technologique

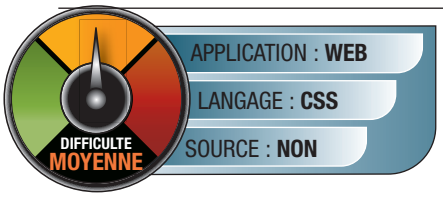
L'évolution technologique ne cesse de s'accélérer, avec aujourd'hui des architectures potentielles multiples, et souvent des besoins hybrides suivant les différentes parties de l'application. Il est fréquent de devoir disposer d'une application purement Web pour toucher une large population d'utilisateurs, mais de souhaiter une application plus interactive et client-riche pour certains utilisateurs avancés, tout en rendant certaines fonctions accessibles sur des terminaux mobiles. L'avènement d'environnements avancés tels qu'Office 2010, SharePoint, Silverlight et Azure (Cloud Computing) — pour ne parler que du monde Microsoft mais l'équivalent est vrai aussi sur la plateforme Java — ne va pas simplifier les choix d'implémentation. Heureusement, le Model-Driven permet de relativiser l'importance de ces choix et de garder également une forte flexibilité dans le temps. En modifiant les producteurs configurés, le développeur va pouvoir rapidement générer les composants de son application vers telle ou telle architecture, et ne recoder que les parties très spécifiques, souvent pour affiner l'interface utilisateur. Les gains de cette approche étant particulièrement significatifs sur le long terme pour la maintenance et l'évolution, le développeur indépendant qui réalise des applications pour ses clients a, lui aussi, tout intérêt à s'équiper de tels outils.

Il pourra ainsi leur proposer des solutions innovantes sans coût excessif et suivre l'innovation technologique sans avoir besoin d'une équipe d'architectes à demeure.

■ Daniel **Cohen-Zardi**
Président Softfluent

Eclipse 4.0 et le moteur CSS

Aujourd'hui le développeur souhaite de plus en plus personnaliser son application de manière à coller au plus près des exigences des utilisateurs. Ce paramétrage passe par la modification de la police, des couleurs des composants graphiques, … La spécification standard CSS (Cascading Style Sheets) utilisée dans le domaine des clients légers est une solution simple et flexible pour la personnalisation des applications Web. Toutefois, la personnalisation de l'apparence des applications de type client lourd (qui nécessitent une installation sur le poste client) est plus difficile puisque les technologies utilisées dans ce domaine sont différentes et ne permettent pas d'utiliser CSS.



La future version d'Eclipse e4 annoncée pour l'année 2010 est très ambitieuse. Diverses fonctionnalités sont prévues (voir article Dans les coulisses du projet Eclipse e4 du n° 119 de Programmez!), dont la prise en compte de CSS. En effet, il sera possible de modifier l'apparence du noyau graphique d'Eclipse en utilisant des feuilles de style CSS. Par ailleurs, cela s'appliquera également aux applications construites via la plate-forme Eclipse (Eclipse RCP).

Le code origine du moteur CSS e4 vient du projet open source TK-UI http://tk-ui.sourceforge.net/fr/user-guide/cssengine.html qui propose un moteur CSS générique. Il est capable de modifier l'apparence graphique via CSS de n'importe quel type d'objet Java comme des composants graphiques Swing ou SWT (la boîte à outils graphiques utilisée dans la plate-forme Eclipse).

Le projet TK-UI a été accepté par la fondation Eclipse et les sources du moteur CSS de TK-UI sont aujourd'hui hébergées dans le repository CVS de e4. Ce dernier contient plusieurs projets concernant CSS dont :

- **org.eclipse.e4.ui.css.core** : moteur CSS générique capable de charger des feuilles de styles CSS et de calculer le style CSS à appliquer pour un objet Java donné. L'application des styles s'effectue ensuite dans les implémentations du moteur CSS.
- **org.eclipse.e4.ui.css.swt** : implémentation du moteur CSS permettant d'appliquer les styles CSS calculés pour des composants graphiques standard SWT.
- **org.eclipse.e4.ui.css.nebula** : implémentation du moteur CSS permettant d'appliquer les styles CSS calculés pour des composants graphiques Nebula (basé sur SWT). Ce projet est un exemple d'extension du moteur CSS qui permet de gérer des composants graphiques autres que le standard SWT.

EXEMPLE CSS - CONTACT E4

Dans le repository CVS de e4 vous pourrez trouver plusieurs démonstrations qui utilisent le moteur CSS. L'une d'entre elles créée par Kai Tödter, propose une petite application qui gère une liste de contacts. Le menu Theme permet de basculer d'un thème à un autre, autrement dit d'appliquer une feuille de style CSS différente.

Voici une copie d'écran de la démonstration avec le thème *Dark Theme* : [Fig.1] Vous pouvez remarquer que l'onglet « Contacts List » qui est sélectionné présente une fonte rouge et on peut voir un léger dégradé de couleur du gris vers le noir. Ceci se concrétise par la déclaration du style CSS :

```
CTabFolder.active:selected {
  background-color: #777777 #373737 #101010 50% 50%;
  color: #ff5500;
  font: bold;
}
```

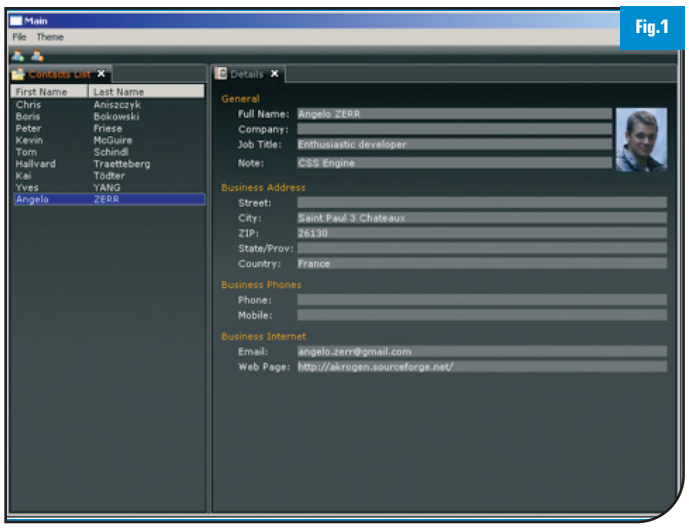


Fig.1

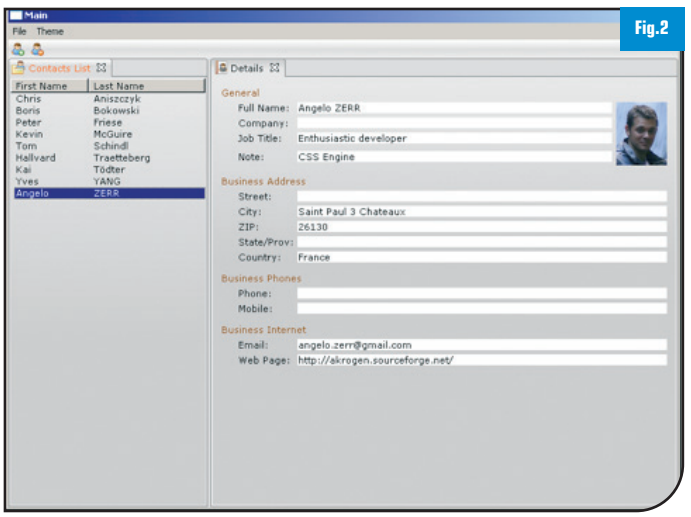


Fig.2

CTabFolder est le composant graphique SWT qui gère les onglets. Cette déclaration de style CSS utilise la pseudo classe **:selected**, ce qui signifie que ce style n'est applicable à un *CTabFolder* que lorsqu'un onglet est sélectionné. Et voici une copie d'écran avec le thème *Bright Theme* : [Fig.2]. Vous pouvez remarquer que l'onglet « Contacts List » qui est sélectionné présente une fonte rouge et il y a cette fois un léger dégradé de couleur du blanc vers le gris. Ceci se concrétise par la déclaration du style CSS :

```
CTabFolder.active:selected {
  background-color: #ffffff #cccccc #dddddd 50% 50%;
  color: #ff5500;
  font: bold;
}
```

Cette démonstration met en évidence la possibilité de styler une application à l'aide de feuilles de styles CSS et de pouvoir appliquer « au runtime » des feuilles CSS.

EXEMPLE CSS - SWT

Maintenant, intéressons-nous à la relation code Java et CSS. Voici un exemple basique de code SWT qui permet d'appliquer la feuille de style CSS avec les selectors de type SWT (autrement dit les noms des classes des composants graphiques SWT sont ici utilisés en tant que selector CSS). Le code mis en gras est celui du moteur CSS. Voici la feuille de style CSS que l'on souhaite appliquer à une interface SWT :

```
Label {
  color:red;
}

Text {
  background-color:green;
}
```

Voici le code Java SWT :

```
Display display = Display.getDefault();
// Create SWT CSS Engine
CSSEngine engine = new CSSSWTEngineImpl(display);
// Parse style sheet
engine.parseStyleSheet(new StringReader(
       "Label {color:red;} Text {background-color:green;}"));

/*--- Start  UI SWT ---*/
Shell shell = new Shell(display, SWT.SHELL_TRIM);
FillLayout layout = new FillLayout();
shell.setLayout(layout);

Composite panel1 = new Composite(shell, SWT.NONE);
panel1.setLayout(new GridLayout(2, true));

// Label
Label label1 = new Label(panel1, SWT.NONE);
```

```
label1.setText("Label 0");

// Text
Text text1 = new Text(panel1, SWT.NONE);
text1.setText("bla bla bla...");

/*---  End UI SWT ---*/

// Apply Styles
engine.applyStyles(shell, true);

shell.pack();
shell.open();

while (!shell.isDisposed()) {
        if (!display.readAndDispatch())
                display.sleep();
}

display.dispose();
```

Fig.3

Après exécution de ce code, la fenêtre SWT s'affiche en appliquant la feuille de style CSS. [Fig.3]. Si vous connaissez SWT, vous pouvez remarquer que le code du moteur CSS n'est pas intrusif. Il peut être utilisé dans des applications SWT existantes.

EXEMPLE CSS – XWT

XWT (Eclipse XML Windowing Toolkit) est une des solutions déclaratives d'interfaces utilisateurs (Declarative UI) proposée dans e4 par la société Française Soyatec (http://www.soyatec.com). Il permet de séparer la définition IHM avec la technologie d'affichage. Le moteur CSS a été intégré de manière basique à XWT.
Si on reprend l'exemple SWT décrit ci-dessus, il se déclare dans un fichier *.xwt comme ceci :

```
<Composite xmlns="http://www.eclipse.org/xwt/presentation">
<Composite.Resources>
<CSSStyle x:Key="style" url="/test/style.css"/>
</Composite.Resources>
<Composite.layout>
<GridLayout numColumns="2" />
</Composite.layout>
<Text text="bla bla bla "/>
</Composite>
```

L'élément XML CSSStyle permet d'indiquer la feuille de style CSS test / style.css à utiliser :

```
Label {
color:red;
}
Text {
background-color:green;
}
```

Ici l'exemple avec XWT se veut très simple, mais XWT est beaucoup plus puissant que cela. En effet, il est par exemple possible de déclarer le binding avec des objets métiers. L'exemple Contact e4 géré exclusivement avec XWT est une bonne démonstration de ce que l'on peut faire avec XWT.

EXEMPLE CSS - ADEO (GROUPE LEROY MERLIN)...

Groupe ADEO est une entreprise entièrement tournée vers l'amélioration de l'habitat et du cadre de vie qui poursuit son développement à travers le monde : Brésil, Chine, Espagne, France, Grèce, Italie, Pologne, Portugal, Russie, Ukraine et Turquie et au travers de ses différentes enseignes : Leroy Merlin, Weldom, Bricocenter, Aki, Bricoman ou encore Zodio et Kbane. L'utilisation d'Eclipse RCP s'inscrit dans un projet de portail d'applications magasin.
Les défis auxquels cette solution doit répondre sont :

1. Bénéficier d'une authentification unique.
2. Etre extrêmement réactive : certaines applications sont utilisées conjointement avec le client.
3. Avoir une ergonomie poussée : les erreurs de saisie coûtent cher, le temps d'apprentissage de l'outil doit être minimisé.
4. Être Modulaire : chaque Enseigne peut choisir ses applications "à la carte" en fonction de ses besoins spécifiques.
5. Mutualiser la partie commune du métier tout en laissant la possibilité d'adaptation aux enseignes. Notamment concernant le "look and feel" des applications.

La solution Eclipse RCP est apparue comme la meilleure réponse possible à toutes ces exigences. Elle présentait néanmoins l'inconvénient de ne fournir en standard aucune solution simple permettant d'adapter l'IHM en fonction de l'enseigne.
Afin de répondre à cette problématique, Groupe ADEO a fait le choix du moteur CSS de TK-UI. Sa conception permet une intégration facile au modeleur graphique SWT Designer (http://www.instantiations.com/windowbuilder/swtdesigner/) par le biais d'une extension de la Factory FormToolkit. Un graphiste a fourni les maquettes et la charte graphique des applications, ensuite déclinées par le développeur grâce au modeleur. Afin de répondre à des besoins non couverts par le moteur CSS, il a été facilement étendu tout en gardant une compatibilité maximale avec le moteur d'origine : la migration du moteur CSS TK-UI vers celui de e4 n'a pris que quelques minutes !

CSS E4 – TRAVAIL EN COURS

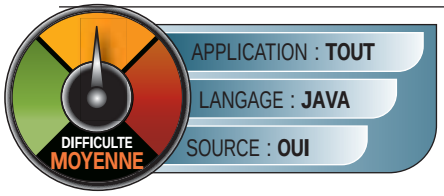
Voici des liens utiles concernant le moteur CSS e4 :

- http://wiki.eclipse.org/E4/ProjectSetFiles : lien à partir duquel vous pouvez trouver les fichiers *psf pour télécharger le moteur CSS et ses exemples.
- http://dev.eclipse.org/blogs/kevinmcguire/ : blog de Kevin Mc Guire qui est l'un des responsables de la partie Declarative UI / CSS dans e4.
- http://www.toedter.com/blog/ blog de Kai Tödter, créateur de la démonstration Contact e4.
- http://wiki.eclipse.org/E4/UI/Running_the_photo_demo et http://wiki.eclipse.org/E4/UI/Running_CSS_demos liens sur les démonstration CSS.
- http://www.w3.org/TR/CSS2/ spécification CSS2.
- http://wiki.eclipse.org/E4/XWT/Running_the_demos#e4_contact_demo démo en XWT avec integration CSS.

■ Angelo **Zerr** & Pascal **Leclercq**
projet open source TK-UI

Scala : le Java nouveau est arrivé ! 2e partie

Dans la première partie de cet article consacré à la présentation du langage Scala, nous avions mis en avant les caractéristiques de son paradigme objet. Dans le second volet, nous allons passer en revue ses aspects fonctionnels avant de terminer par quelques mots sur son intégration avec le langage Java.



UN LANGAGE FONCTIONNEL

En plus d'être un langage objet pur, Scala est aussi un langage fonctionnel. Ainsi en Scala, tout est considéré comme étant une valeur que ce soit les variables ou les fonctions, et les valeurs sont des instances de classes. Les principes d'immuabilité et de transparence référentielle apportés par Scala prennent toute leur importance dans sa partie fonctionnelle. En effet, ils garantissent que l'appel à une fonction peut être remplacé par son résultat dans un fragment de code sans que cela change l'exécution de ce dernier.

Comme tout bon langage fonctionnel qui se respecte, Scala autorise les fonctions d'ordre supérieur. En clair, cela signifie qu'il est possible de définir des fonctions en prenant d'autres en paramètre et dont le résultat peut être également une fonction. Le fait que les fonctions soient considérées comme des valeurs prend ainsi tout son sens. Cette particularité issue du monde fonctionnel est un atout très puissant dans la trousse à outils du développeur Scala.

Autre atout dans cette trousse, les fonctions supportent la **curryfication.** Pour les profanes des langages fonctionnels, malheureusement encore trop nombreux, la curryfication permet de créer des fonctions pures en transformant les fonctions à plusieurs arguments en fonction à un seul argument qui en retourne une autre traitant le reste des arguments. En Scala, l'emploi de la curryfication permet de définir des fonctions partielles de manière élégante, sans avoir recours à des sucres syntaxiques comme l'emploi du caractère magique _ pour les arguments d'une fonction. Ces fonctions partielles, qui sont donc définies seulement sur une partie de leur domaine de valeur en entrée, s'avèrent très utiles comme le démontre brièvement l'exemple suivant :

```
object FunTest extends Application {
// Fonction permettant de filtrer les éléments d'une liste d'entiers
// à l'aide de la fonction p
def filter(xs: List[Int], p: Int => Boolean): List[Int] =
  if (xs.isEmpty) xs
  else if (p(xs.head)) xs.head :: filter(xs.tail, p)
  else filter(xs.tail, p)

  // fonction modN classique
  //  def modN(n:Int, x:Int) = ((x%n) == 0
  // fonction modN curryfiée
  // def modN(n: Int) = (x: Int) => ((x%n) == 0)
  // fonction modN curryfiée en version abrégée
  def modN(n: Int)(x: Int) = ((x % n) == 0)

  // définition d'une liste d'entiers
  val nums = List(1, 2, 3, 4, 5, 6, 7, 8)

  // définition partielle de modN et application sur la fonction filter
  println(filter(nums, modN(2)))
  println(filter(nums, modN(3)))

  // forme curryfiée d'une fonction de multiplication
  def mul(x: Int)(y: Int) : Int = x * y
  // application sur une liste à l'aide de l'itérateur map
  val l = List(1,5,3)
  println(l map mul(2)) // equivaut à l.map(mul(2))

  // Renvoie une fonction inc => fonction d'ordre supérieur
  def inc(inc: Int): (Int => Int) = (x: Int) => x + inc

  // fonctions partielles
  val inc10 = inc(10)
  val inc15 = inc(15)

  // utilisation
  println(inc10(12))
  println(inc15(12))
}
```

Comme on peut le constater sur cet exemple, la forme curryfiée d'une fonction peut être abrégée pour soulager son écriture. Le résultat de l'exécution de ce code est présenté à la figure 1. Le lecteur attentif aura remarqué que l'objet Scala FunTest ne possède pas de méthode main mais son exécution se réalise correctement. Cela vient du fait qu'il étend le trait Application qui permet l'exécution de l'ensemble du code placé au sein du corps de notre objet Scala.

DES CLOSURES PLEINEMENT FONCTIONNELLES

En outre, Scala permet l'utilisation de fonctions anonymes. Ceci ouvrant la voie aux closures qui sont partie intégrante de Scala en bon langage fonctionnel qu'il est. Pour les développeurs Java, le concept de *closure* est sûrement le concept fonctionnel qui lui est le plus familier. En effet, avec Java 7 un long débat s'était créé quant à

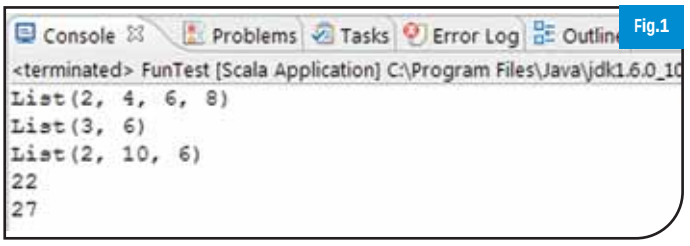


Fig.1

l'implémentation de closures au sein du langage, ce qui a permis d'ouvrir de nouvelles perspectives à bon nombre de développeurs. Malheureusement, cela n'a pas abouti pour cette mouture de Java ! Les développeurs Java vont continuer à devoir utiliser des classes anonymes pour simuler de manière moins puissante les possibilités offertes par les closures.
Heureusement Scala est là, et va vous permettre de les découvrir et d'en profiter pleinement dès aujourd'hui. Les closures sont intimement liées aux fonctions anonymes qui en fait sont elles-mêmes des closures. Souvent appelées expressions lambda dans le monde fonctionnel, les closures sont des fonctions un peu particulières de par leur capacité à disposer du contexte d'un objet ou d'une autre fonction. Elles peuvent être nommées ou bien anonymes auquel cas on les invoque à l'aide de leur méthode apply, comme le détaille l'exemple suivant :

```
object ClosureTest extends Application {
  // définition d'une fonction prenant en entrée une closure
  def oncePerSecond(callback: () => Unit) {
    while (true){
      callback()
      Thread.sleep(1000)
    }
  }

  // fonction affichant un simple message est de type () => Unit
  def simpleMsg() { println("SimpleMessage") }

  // définition d'une closure nommée affichant l'argument passé
 en entrée
  val printArg = (x: Int) => println("Print arg = " + x)
  // invocation de la closure avec la méthode apply
  printArg.apply(12)
  // définition et utilisation d'une closure non nommée
  println(((x:Int, y:Int) => x + y).apply(12,15))

  // appel de la fonction oncePerSecond avec des closures nommées
 ou anonymes
  oncePerSecond(simpleMsg)
  oncePerSecond(() => println("Time flies in anonymous function"))
}
```

Cet exemple basique d'utilisation, faute de place, ne met pas en valeur la puissance des closures, pourtant très réelle, bien que déroutante pour certains au départ. Cependant, une fois le temps d'adaptation passé, on devient vite accro et on se demande encore comment on a pu s'en passer jusque-là !

UN PATTERN MATCHING ÉVOLUÉ

Souvent comparé à un switch-case évolué, le pattern matching est un concept fonctionnel qui permet à Scala de filtrer facilement des variables en alliant discrimination et décomposition. Ainsi, la variable que l'on cherche à matcher va être décomposée suivant les motifs appliqués dans les différentes clauses du match. En sus, le filtrage peut être fait suivant le type de la variable et donc des éléments la composant également. L'ordre des clauses du match a son importance puisque les tests de filtrage s'arrêtent dès qu'un motif *matche* la variable que l'on cherche à filtrer. Une fois, le motif matché, le bloc de code qui lui est associé est exécuté. Enfin, il est bon de savoir que le compilateur Scala lève une erreur lorsque le pattern matching n'est pas exhaustif. Voici un petit exemple mettant en lumière ces quelques points :

```
// classe couleur RGB
class Color(val red:Int, val green:Int, val blue:Int)

// classes couleurs spécialisées
// le mot-clé case permet d'utiliser ces classes dans un match
case class Red(r:Int) extends Color(r, 0, 0)
case class Green(g:Int) extends Color(0, g, 0)
case class Blue(b:Int) extends Color(0, 0, b)
object ColorMatchTest extends Application {
  // méthode utilisant le pattern matching
  def printColor(c:Color) = c match {
    case Red(v) => println("Red: " + v)
    case Green(v) => println("Green: " + v)
    case Blue(v) => println("Blue: " + v)
    case col:Color => {
      print("R: " + col.red + ", ")
      print("G: " + col.green + ", ")
      println("B: " + col.blue)
    }
    case null => println("Invalid color")
  }

  printColor(Red(100))
  printColor(Blue(220))
  printColor(new Color(100,200,50))
  printColor(null)
}
```

La classe Color définit une couleur de type RGB. Ses classes filles devant être utilisées pour faire du pattern matching, on les préfixe du mot-clé case. Le pattern matching d'une variable s'utilise via le mot-clé match. Il est réalisé dans la fonction printColor au sein de laquelle on met en avant la décomposition de la variable matchée et sa discrimination à l'aide du test de typage du motif. Enfin, il est important de préciser que le type de donnée doit être identique pour chacune des clauses d'un même match. Ceci étant bien le cas ici puisqu'on renvoie un type Unit. Le résultat de l'exécution de ce code est présenté à la figure 2.
L'emploi du pattern matching s'avère particulièrement efficace pour le traitement de flux XML par exemple ou bien pour la gestion des exceptions. Dans ce dernier cas, il permet d'utiliser un seul bloc *catch* pour récupérer plusieurs exceptions renvoyées dans le bloc *try* correspondant. Les exceptions étant ensuite filtrées par pattern matching et le traitement adéquat pouvant être appliqué à chacune d'entre elles.

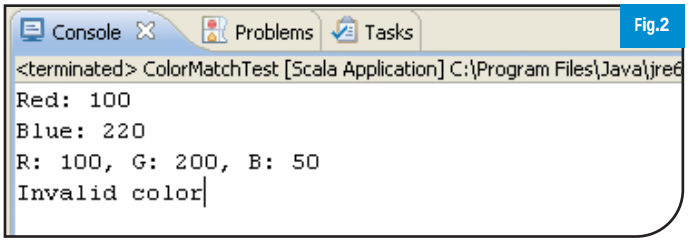


Fig.2

UTILISATION SIMPLIFIÉE DES COLLECTIONS

Le dernier point majeur issu du monde fonctionnel concerne les collections et les itérateurs proposés par le langage Scala. Cet apport permet de simplifier grandement l'usage des collections et les opérations qu'on peut leur appliquer. Un itérateur prend en entrée une closure et l'applique à l'ensemble des éléments d'une collection. Ceci permettant par exemple d'appliquer et de raccourcir grandement l'écriture de traitements complexes sur ces collections. Il va sans dire qu'il est tout à fait possible de définir ses propres itérateurs qui viennent s'ajouter aux itérateurs classiques que sont le *foreach*, le *fiter*, le *map* et le reduceLeft notamment que l'on peut voir dans l'exemple suivant :

```
object ScalaCollections extends Application {
  val list = List(50,10,-15,0,12,-1,-80,82,125,-120)

  // indice du premier 0
  println(list findIndexOf((x) => x == 0))
  // perl touch
  println(list findIndexOf(_ == 0))

  // la liste contient -15 ?
  println(list exists((x) => x == -15))
  // perl touch
  println(list exists(_ == -15))

  // ajoute 5 à chaque élément de la liste
  println(list map (x => x + 5))
  // perl touch
  println(list map (_ + 5))

  // filtre les éléments négatifs
  println(list filter (x => x >= 0))
  // perl touch
  println(list filter (_ >= 0))

  // affichage de tous les éléments
  list foreach (x => println(x))
  // perl touch
  list foreach (println)

  // somme des éléments de la liste
  println(list reduceLeft((x,y) => x + y)) // utilisation de
l'inférence de type ici
  // perl touch
  println(list reduceLeft(_ + _))

  // composition de ces fonctions
  println(list.filter(_ < 0).map(_* 2).reduceLeft(_ + _))
}
```

Ainsi, pour calculer la somme des éléments d'une liste on a utilisé l'instruction suivante : *list.reduceLeft((x,y) => x + y)* et pour filtrer les éléments d'une liste : *list.filter(x => x >= 0)*. L'écriture des closures appliquées à ces itérateurs peut être encore plus raccourcie via quelques sucres syntaxiques que Scala a empruntés à Perl notamment. Mais ce raccourcissement extrême entraînera forcément des problèmes de lisibilité du code.

PROGRAMMATION CONCURRENTE

Avec la multiplication des machines multicœurs, la programmation concurrente est devenue un enjeu réel pour le développement d'applications. Désormais, les évolutions en termes de performance passeront par une meilleure utilisation de ces cœurs. Force est de constater que le langage Java, apparu bien avant ces machines multicœurs, n'est pas adapté à ce modèle de programmation. Bien que les apports de l'API Concurrent depuis Java 5 soient réels et aient permis d'améliorer ce problème, il n'en reste pas moins que la mise en œuvre de la concurrence en Java reste compliquée. Elle ne permet pas de bénéficier pleinement des potentialités des machines actuelles et à venir.

Là encore, Scala apporte une solution élégante et puissante à ce problème. Il s'agit de l'API **Actors** qui repose sur le modèle éponyme, issu du langage Erlang dont les qualités en matière de programmation concurrente ne sont plus à démontrer. Le modèle des acteurs repose sur 2 principes fondamentaux :

- Un acteur est une entité logicielle indépendante
- Les acteurs communiquent en échangeant des messages asynchrones ou synchrones mais ne partagent pas d'information d'état entre eux.

Considérons maintenant le petit exemple suivant :

```
// Test d'un simple acteur
object ActorTest {
  def main(args : Array[String])  = {
    // creation d'un acteur
    val msgActor =
      actor {
                  receive {
                    case msg => println(msg)
                  }
      }
      // envoi d'un message via l'utilisation de !
      msgActor ! "Hello Programmez !"
  }
}
```

Ici, on crée un acteur qui va exécuter le bloc receive qu'on lui passe en entrée. Dans ce bloc, on peut reconnaître l'utilisation du pattern matching pour traiter le message retenu. Enfin, on envoie un message à notre acteur via l'utilisation du !. En outre, la communication entre acteurs à l'aide de messages est bidirectionnelle. Ainsi, un acteur recevant un message d'un autre acteur peut lui répondre à l'aide de la méthode reply. Ceci nous permettant de compléter légèrement l'exemple précédent pour obtenir cela :

```
import concurrent.ops._
import scala.actors._, Actor._

object ActorTest extends Application {
  // consommateur de messages
  val msgActor = actor {
    var done = false
    while(!done){
      receive {
        case msg => println("received : " + msg)
        done = (msg == "DONE")
```

```
        reply("RECEIVED")
      }
    }
  }

  // producteur de messages
  // création d'un nouveau thread via spawn
  spawn {
    val msgs = Array("Hello Programmez !",
      "Magazine de tous", "les langages", "DONE")
    msgs foreach((msg) => msgActor !? msg)
  }
}
```

L'acteur msgActor est instancié avec une boucle qui se met en attente de réception de messages tant que le message DONE n'est pas reçu. Pour alimenter ce consommateur de messages, on crée un autre thread via l'utilisation du bloc *spawn* (issu d'Erlang également). Ce bloc va permettre de créer et de démarrer un nouveau thread qui va exécuter le code placé en son sein. Ainsi, on va se contenter de produire des messages que l'on envoie à l'acteur msgActor. Ici, l'envoi de messages est réalisé via le symbole **!?** ce qui a pour effet de mettre le producteur en attente d'une réponse du consommateur.

Comme on vient rapidement de le constater, la programmation concurrente par acteurs est quelque peu différente de sa version objets. Dans ce modèle, on raisonne en termes de messages. Ceci a pour effet premier de rendre le code concurrent plus simple à comprendre mais cela permet également de réduire les risques d'utilisation de données partagées par différents threads. De fait, les problèmes de deadlocks et autres cauchemars de la programmation concurrente seront désormais de l'histoire ancienne en Scala. Et c'est tant mieux car bien souvent, ils ne sont visibles qu'en production et donnent des sueurs blanches aux développeurs Java.

INTÉGRATION PARFAITE AVEC JAVA

Bénéficiant d'un écosystème à la fois riche, éprouvé et performant, la plate-forme Java est la plate-forme de référence en entreprise. En outre, ses performances sont en constante amélioration. Ainsi, le choix d'en faire la plate-forme cible du langage Scala était une évidence pour Martin Odersky. Le compilateur qu'il a mis au point pour Scala a bénéficié de son expérience en la matière et se révèle très performant. Compilées directement en byte-code Java, les applications Scala ont un temps d'exécution au minimum équivalent à celui des applications Java pures. L'ensemble des API et Frameworks existants sont immédiatement utilisables en Scala, ce qui est considérable, compte tenu du travail effectué autour de Java depuis sa création. L'intégration de code Scala au sein de programmes Java rentre quant à elle dans le cadre des efforts faits par Sun pour améliorer l'*interfaction* de son langage avec les langages dits "dynamiques". Ainsi, l'invocation du langage Scala en Java respecte les fondements apportés par la JSR-223 (cf. Programmez ! n°115 et l'article "Invocation de langages dynamiques en Java") et sera grandement améliorée avec la JSR-292 implémentée dans Java 7. Cette dernière va apporter à Java un support natif des langages dynamiques tout en optimisant au maximum les appels faits à ces fragments de code avant de rendre leur utilisation quasi-équivalente en termes de performance au code Java.

CONCLUSION

À la fois orienté objet et fonctionnel, le langage Scala tire parti de sa richesse dans cette mixité inédite pour un langage à typage statique. Facilement extensible de par la puissance de ses concepts, il offre de nouvelles perspectives de par sa scalabilité. Ainsi, il est simple de l'utiliser pour définir de nouveaux paradigmes de programmation comme cela a été démontré avec l'implémentation du modèle des acteurs pour la programmation concurrente. Adoubé par le créateur de Groovy en personne, il se présente comme un langage à surveiller de près dans le futur. Il pourrait bien être de l'aveu d'un grand nombre de spécialistes de la plate-forme Java un remplaçant idéal au langage Java (figure 3) ou tout du moins un compagnon de route de plus en plus présent dans les années à venir.

Aussi Séduisant soit-il sur le papier, il commence tout juste à pointer le bout de son nez dans l'industrie. Ainsi, il semble partir avec une longueur de retard sur le langage Groovy, qui bien que moins séduisant en théorie, bénéficie grandement du support de SpringSource et est plus répandu en entreprise. Mais Scala semble bien parti pour combler son retard. Le remplaçant du langage Java, si remplaçant il doit y avoir, sera sûrement parmi ces 2 langages. Affaire à suivre dans les années qui viennent donc…

■ Sylvain **Saurel** – *Ingénieur d'Études Java / JEE ACP* – www.acp-qualife.fr - *sylvain.saurel@gmail.com*

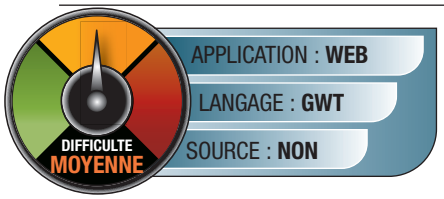
Fig.3

Scala ⊕	Scala ⊖
+ modèle objet pur	- membres statiques
+ surcharge d'opérateurs	- types primitifs
+ closures	- traitement spécial des interfaces
+ mixin-composition	- wildcards
+ données membres abstraites	- break, continue
+ pattern matching	
+ API Actors	

Figure 3 : Comparaison des ajouts/retraits de Scala par rapport à Java

Développer une application GWT avec le framework PureMVC

Depuis l'apparition de GWT, Google n'a cessé d'améliorer les performances et les fonctionnalités de son framework RIA au fil de ses versions successives. Cependant, il n'y a toujours pas de composants livrés en standard implémentant le célèbre pattern Modèle-Vue-Contrôleur. Ce pattern est particulièrement utile à mettre en œuvre lorsque la taille d'une application graphique augmente. Je vous propose aujourd'hui de combler cette lacune en utilisant le portage Java du framework open source PureMVC.



LE PATTERN MODÈLE-VUE-CONTRÔLEUR

La façon de programmer une Interface Homme Machine n'est soumise à aucune règle. Chacun peut développer ses IHM comme bon lui semble, organiser son code en plusieurs fichiers ou bien tout coder dans un seul. Cependant, si l'on souhaite structurer ses développements en séparant les portions de code gérant les données, la présentation et les traitements, on se tournera généralement vers une architecture de type MVC. Cette méthode a été inventée en 1979, dans les laboratoires de recherche Xerox PARC par Trygve Reenskaug.

Le fait de structurer son application en fonction des responsabilités des composants, tout en tenant compte du sens de couplage employé pour les faire collaborer, est capital pour construire et maintenir une application dans le temps.

Explication du trigramme MVC : [Fig.1]

Le Modèle assure la gestion des données qui seront utilisées par l'application. La Vue assure l'affichage des composants graphiques et gère l'interaction de ceux-ci entre l'utilisateur et l'application.

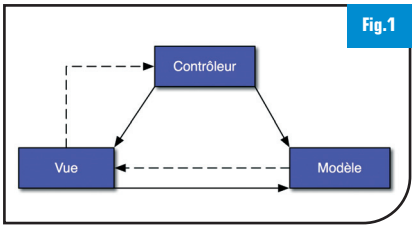


Fig.1

Le Contrôleur assure la synchronisation entre la Vue et le Modèle. Il gère tous les événements provenant de la Vue et déclenche les actions à réaliser. Intéressons-nous maintenant à l'implémentation du modèle MVC dans le framework PureMVC - http://www.puremvc.org

LE FRAMEWORK PUREMVC

PureMVC est un framework pour créer des applications basées sur le célèbre « design pattern » Modèle, Vue et Contrôleur décrit précédemment. Ce framework open source à été initialement développé par Cliff Hall <cliff@puremvc.org> en Action Script 3 pour être utilisé avec Adobe Flex, Flash et AIR. Il a ensuite été porté sur les principaux langages actuellement disponibles sur le marché. Nous allons nous intéresser dans cet article au portage Java 1.5 de PureMVC Multicore. En effet, il existe deux versions de ce framework : la version standard et la version multicore. La version standard fournit une méthode simple de séparation des couches suivant le modèle MVC, adaptée à un usage de type « PageFlow ». La version MultiCore permet en plus de l'usage «PageFlow », de construire une application avec plusieurs modèles MVC fonctionnant en parallèle et communiquant ensemble dans la même JVM. Ces différents modèles sont structurés en modules indépendants «Core independant ».

Un core est un ensemble de quatre acteurs, on retrouve les trois acteurs de la version standard « Modèle », « Vue », « Contrôleur » et un acteur supplémentaire la « Façade ». Nous allons donc avoir plusieurs «Core» en mémoire « les Multitons» plutôt que le « singleton » utilisé dans la version standard. Ainsi chaque « Core» sera référencé par une clef «Multiton» unique. La version Multicore a été développée pour répondre aux besoins des RIA (rich internet Applications) devant charger et libérer des fonctionnalités découpées en module lors du runtime. Par exemple, une application de type messagerie sur PDA gérant une liste de contacts, une liste de tâches et envoyant des mails devra charger et libérer ses modules selon des fonctionnalités demandées par l'utilisateur.

LES PATTERNS : PROXY, MEDIATOR, COMMAND ET FAÇADE

Dans le framework, les concepts MVC s'appellent Proxy (M), Mediator (V) et Command (C) [Fig.2]. L'application communique avec la triade de MVC par l'intermédiaire d'un point d'entrée : La Façade.

Le Proxy relaie les appels vers le modèle de données. Il gère les appels à des services distants pour enregistrer ou lire des données.

Le Mediator crée les composants graphiques, ajoute les événements sur les composants, envoie des notifications vers les autres acteurs du système et/ou reçoit des notifications en provenance du système. La Command est le composant qui orchestre l'ensemble des communications entre tous les composants du système. Il permet d'interagir avec le Proxy, d'envoyer des notifications, lancer d'autres commandes, d'initialiser le système au démarrage et à l'arrêt de l'application. [Fig.2]

MISE EN OEUVRE PUREMVC DANS UN PROJET GWT

Les instructions suivantes nécessitent l'installation de l'IDE Eclipse, du plugin « Subversive» pour la connexion à Subversion et d'une connexion internet. Vous pouvez télécharger Eclipse sur http://www.eclipse.org/downloads/ et le plugin Subversion sur http://www.eclipse.org/subversive/.

Une fois Eclipse installé, téléchargez et installez le « Google Plugin»

sur : http://code.google.com/intl/fr/eclipse/. Le «Google Plugin» vous permet de créer facilement des projets GWT et de les exécuter. Nous allons ensuite importer un projet d'exemple et utiliser le « Google Plugin» pour le compiler et le lancer d'une façon plus conviviale. Sous Eclipse, importez le projet d'exemple se trouvant sur internet : «File > Import > SVN/Project from SVN» et cliquez sur «Next». Cliquez sur «Create a new repository location», puis cliquez sur le bouton «Next». Dans le champ Url, mettre «http://svn.puremvc.org/Demo_Java_MultiCore_GWT_EmployeeAdmin», puis cliquez sur «Next». Dans le champ url, ajoutez «trunk» à la fin de l'url de façon à avoir : http://svn.puremvc.org/Demo_Java_MultiCore_GWT_EmployeeAdmin/trunk, puis cliquez sur «Finish». Cliquez sur «Check out as project configured using the Next Project Wizard». Sélectionnez «Java Project» puis cliquez sur le bouton «Next». Dans le champ «Project Name», mettez un nom de projet valide (ex: ExempleEmployeeAdmin) et cliquez sur «Finish».

Dans la vue «Navigator» ou «Package», clic droit sur le projet [Fig.3] et sélectionnez «Google > Web Toolkit Settings». Cochez «Use Google Web toolkit» puis validez avec le bouton «OK». Rajoutez le fichier «PureMVC_Java_MultiCore-1.0.6.jar» : clic droit sur le projet «Properties > Java Build Path > Add JARs», sélectionnez votre fichier dans le répertoire «war/WEB-INF/lib».

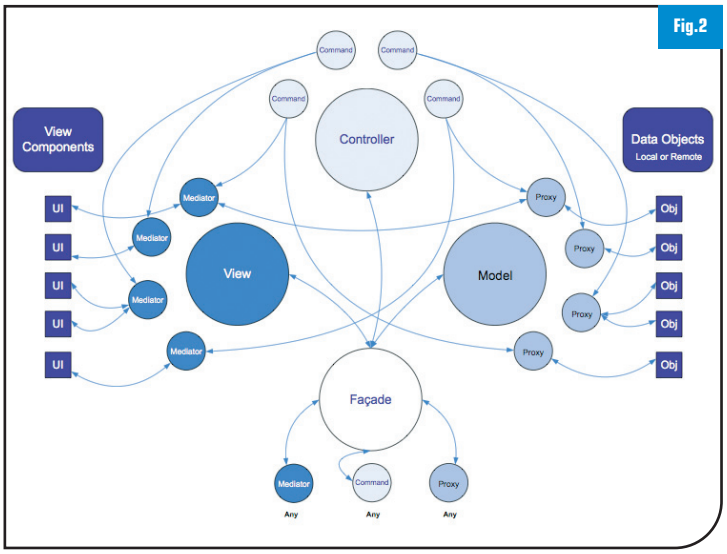
Voilà, votre projet est configuré et prêt à être exécuté avec le «Google Plugin». Pour son exécution, faites un clic droit sur le projet, puis «Run As / Web Application». Si vous avez correctement effectué les manipulations précédentes, l'application se lancera en mode Hosted Browser [Fig.3, 4, 5].

UTILISATION DE PUREMVC AVEC MAVEN

Pour les utilisateurs de Maven, le fichier jar est disponible sur le repository central. Pour l'utiliser avec Maven, il suffit d'insérer dans son fichier «pom.xml» les éléments suivants :

```
<project>
....
<plugin>
<groupId>org.codehaus.mojo</groupId>
<artifactId>gwt-maven-plugin</artifactId>
<version>1.1</version>
<configuration>
<module>org.puremvc.Demo_Java_MultiCore_GWT_Employee  Admin
</module>
<runTarget>EmployeeAdmin.html</runTarget>
</configuration>
<executions>
<execution><goals><goal>compile</goal></goals></execution>
</executions>
</plugin>
....
<dependency>
<groupId>org.puremvc</groupId>
<artifactId>PureMVC_Java_MultiCore</artifactId>
<scope>provided</scope>
<version>1.0.6</version>
</dependency>
....
</project>
```

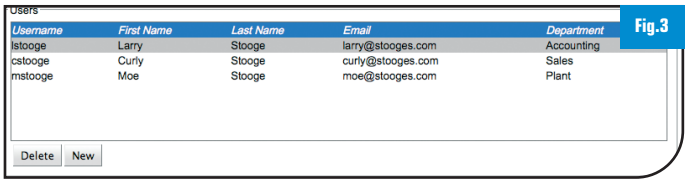
Fig.2



APPLICATION DE DÉMONSTRATION : EMPLOYEE ADMIN

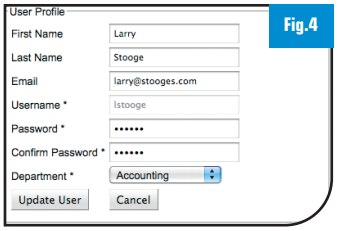
Revenons à notre application d'exemple. Vous avez téléchargé une application illustrant l'utilisation de PureMVC avec GWT, dans le cadre d'une application de gestion d'utilisateurs. L'application est divisée en trois parties :

Fig.3



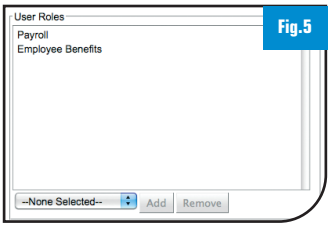
Création et/ou suppression des utilisateurs

Fig.4



Gestion du profil de l'utilisateur

Fig.5



Gestion des rôles de l'utilisateur

L'application déployée en local est accessible via l'URL:
http://localhost:8080/EmployeeAdmin.html
ou déjà déployée sur internet à l'adresse :
http://employeeadm.appspot.com/

INITIALISATION DE L'APPLICATION

Une application GWT pour se lancer à besoin de connaître la classe qui sera appelée en premier. Il s'agit ici de la classe «org.puremvc.java.multicore.demos.gwt.EmployeeAdmin». Cette information est présente dans le fichier «Demo_Java_MultiCore_GWT_EmployeeAdmin.gwt.xml». Vous trouverez également dans ce fichier, la déclaration permettant d'utiliser la librairie de PureMVC.

```
<module rename-to='employeeadmin'>
    ......
  <!-- Inherit PureMVC -->
 <inherits name="org.puremvc.PureMVC_Java_MultiCore" />
```

```
  <!-- Specify the app entry point class.  -->
  <entry-point class="org.puremvc.java.multicore.demos.gwt.EmployeeAdmin" />
</module>
```

Regardons maintenant le point d'entrée :

```
public class EmployeeAdmin implements EntryPoint {
    public void onModuleLoad() {
        ApplicationFacade applicationFacade =
                            ApplicationFacade.getInstance();
        applicationFacade.startup(RootPanel.get());
    }
}
```

Nous récupérons une instance unique de la Façade. Ceci est très important pour bien comprendre le fonctionnement du framework ou chaque «Core» n'est instancié qu'une seule fois en mémoire.
Avec PureMVC une Façade est un singleton identifié par une constante «NAME» déclaré dans la classe «ApplicationFacade».
Vous trouverez également la déclaration des constantes de notifications «STARTUP», «USER_SELECTED» etc.
Lors de la création du singleton de «ApplicationFacade», le constructeur transmet à la classe mère «Facade» son identifiant «NAME».
Par polymorphisme, la classe appelle ensuite les méthodes initializeModel(), initializeController(), initializeView() de la classe fille «ApplicationFacade» si celles-ci ont été surchargées.
Ici, seule la méthode initializeController() a été surchargée pour l'enregistrement des contrôleurs :

```
@Override
protected final void initializeController() {
    super.initializeController();
    registerCommand(STARTUP, new StartupCommand());
    ....
 }
```

Les médiators et proxies seront initialisés dans la méthode startup() plutôt que dans les méthodes initializeModel(), initializeView(). Ce choix a été défini dans la version AS3 de EmployeeAdmin afin de gérer l'affichage des médiators et l'initialisation des proxies à la demande. Dans cet exemple, nous avons une page n'affichant qu'une seule Façade. Mais dans le cas où l'application serait plus complexe, avec plusieurs Façades d'affichées simultanément, alors la méthode startup() prendrait tout son sens. Elle permettrait en fonction de la navigation de n'afficher que certaines parties de l'IHM. Pour en masquer certaines, il suffirait de rajouter une méthode stop() qui dé-enregistrerait les mediators et proxies de la façade.
La méthode startup() envoie une notification STARTUP :

```
public final void startup() {
     sendNotification(STARTUP);
}
```

La notification STARTUP est associée à la classe StartupCommand

```
registerCommand(STARTUP, new StartupCommand());
```

La méthode execute() de la StartupCommand est appelée :

```
public class StartupCommand extends SimpleCommand {
  @Override
  public final void execute(final INotification notification) {
    getFacade().registerProxy(new UserProxy());
    getFacade().registerMediator(new UserListMediator());
    ....
  }
}
```

TRAITEMENTS EFFECTUÉS LORS D'UNE ACTION UTILISATEUR

Lorsque vous cliquez dans l'ihm sur la ligne de l'utilisateur «Istooge», la méthode onSelect() de la classe «UserListMediator» est appelée. onSelect() enverra une notification USER_SELECTED avec l'objet UserVO identifiant l'utilisateur, aux acteurs du système :

```
sendNotification(ApplicationFacade.USER_SELECTED,
                                userProxy.users().get(selectedRow));
```

Les mediators «UserFormMediator» et «RolePanelMediator» seront rafraîchis car ils ont déclaré dans leur méthode listNotificationInterests() leur intérêt pour cette notification :

```
@Override
public final String[] listNotificationInterests() {
  return new String[] {
    ....
    ApplicationFacade.USER_SELECTED
  };
}
```

Les mediators seront notifiés de l'arrivé d'une notification dans leur méthode handleNotification(….).

```
@Override
public final void handleNotification(final INotification notif) {
  ....
if (notif.getName().equals(ApplicationFacade.USER_SELECTED)) {
    user = (UserVO) notification.getBody();
    ....
    }
  super.handleNotification(notification);
}
```

Ils pourront ensuite rafraîchir leur ihm grâce aux propriétés de l'objet UserVO.

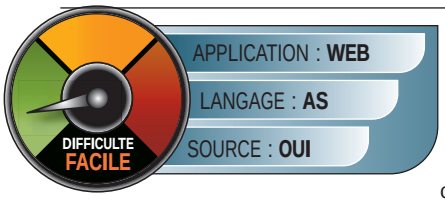
CONCLUSION

Vous venez d'avoir un aperçu de la puissance de PureMVC. L'intérêt principal est d'utiliser un framework pérenne et très bien documenté. Utiliser PureMVC MultiCore dans vos développements en GWT vous permettra de les structurer, vous évitera d'avoir du code spaghetti et rendra l'application plus facilement maintenable. Il vous permettra également de la transposer avec un minimum d'effort dans un environnement autre, comme SWT par exemple, en ne réécrivant que le code des mediators. GWT 2.0 est disponible avec des nouveautés comme le «Code Splitting» et le «Declarative User Interface». J'ai testé ces fonctionnalités dans la version GWT2.0 M1 qui s'intègre parfaitement avec l'architecture de PureMVC. Pour plus d'information, je vous invite à parcourir la documentation sur le site de PureMVC.

■ Anthony **Quinault**
Architecte JEE freelance. Pour du conseil et de la prestation : aquinault@cadesoft.com. Pour toutes questions relatives à PureMVC : anthony.quinault@puremvc.org
Membre du Poitou-Charentes JUG, http://www.poitoucharentesjug.org

Introduction à Flex et ActionScript 3

ActionScript 3 et son framework Flex 3 permettent de développer des applications s'exécutant sur le lecteur Flash. S'il n'est pas immédiat de démarrer avec eux, ils se révèlent passionnants à utiliser. Faisons les premiers pas dans l'univers de la programmation Flex/ Flash.



Des applets Java ou des applications Flash ? C'est un débat souvent lancé. Les deux permettent de faire des choses semblables et chacun a ses propres atouts. Sans entrer dans le débat, nous nous intéressons aujourd'hui au travail avec ActionScript, le langage pour programmer le lecteur Flash. Nous allons travailler dans le contexte de Flex 3. Au moment de la rédaction de cet article, Flex 4 n'était pas loin de sortir. Il existait déjà en bêta et bien entendu il apportera son lot d'améliorations. Nous restons cependant, pour l'instant avec Flex 3, déjà extrêmement riche. Les grandes lignes données dans cet article resteront bien sûr valables avec Flex 4.

1 LES PRÉ REQUIS

ActionScript 3 est un langage orienté objet qui présente beaucoup de similitudes avec Java ou C#. Il n'est pas question d'aborder la programmation objet dans cet article, nous supposons donc que le lecteur a des notions sur un de ces langages, ou à défaut, sur un autre langage orienté objet. Flex 3, quant à lui, est basé sur XML, et nous supposons que le lecteur est familiarisé avec lui.

2 OUTILS ET PRISE DE CONTACT

Programmer en ActionScript est très amusant, dès lors que l'on a vaincu une vraie difficulté: prendre contact avec tout cet univers, ce que nous faisons maintenant. Pour programmer avec ActionScript seul, on utilise normalement un outil commercial proposé par Adobe. Adobe a, petit à petit, ouvert ActionScript et propose maintenant un SDK Open Source du nom de Flex 3. Qu'est-ce que Flex 3 ? Au centre, nous trouvons le langage ActionScript 3. Immédiatement au-dessus, viennent les librairies Flex qui constituent un framework pour coder avec ActionScript. Ensuite les outils du SDK sont principalement un compilateur de fichier MXML.

MXML est une sorte de langage déclaratif, basé sur XML, et servant à décrire une application Flex. Nous avons aussi un compilateur de composants, permettant de générer des éléments réutilisables. La SDK Flex vient sans documentation. La première chose à faire, après le téléchargement du SDK Flex 3 est de vous rendre à http://www.adobe.com/support/documentation/en/flex/ et de télécharger la " Complete Flex 3 documentation". Celle-ci contient la documentation, à la manière javadoc, de toutes les classes du framework. Ensuite les documents PDF devguide_flex3.pdf et progAS_flex3.pdf retiendront plus particulièrement l'attention pour commencer. Enfin pour que les outils du JDK fonctionnent, vous devez avoir Java installé sur votre machine. Des IDE pour développer en Flex commencent à exister, des plugins Eclipse notamment. Nous manipulerons pourtant avec le SDK et un éditeur de texte, car c'est, de l'humble avis de votre serviteur, la meilleure façon de démarrer.

Une fois les bases acquises, le lecteur se tournera avec profit vers un IDE, bien entendu.

3 HELLOWORLD

Sacrifions, une fois de plus, à la tradition. Nous voulons que notre application (HelloWorld sur notre site) soit constituée d'un panneau embarquant un bouton. Un clic sur le bouton affichera une alerte avec le traditionnel message. Avec un outil commercial Adobe, tout pourrait être écrit en ActionScript pur. Avec Flex, nous devons avoir le fichier MXML descriptif de l'application. Au gré de l'humeur du développeur, ce fichier peut être réduit au minimum, ou au contraire, faire la quasi-totalité du travail, comme dans l'exemple ci-dessous, Helloworld.mxml

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

<mx:Application layout="absolute"
xmlns:mx="http://www.adobe.com/2006/mxml">

<mx:Script source="HelloActionScript.as"/>

<mx:Panel layout="absolute" title="HelloWorld!!"
 id="panneau" width="270" x="10" y="10" height="150">
<mx:Button click="afficherMessage()" label="Hello !!" id="bouton_
action" width="120" x="60" y="10" height="20" horizontalCenter
="0" verticalCenter="0"/>
</mx:Panel>

</mx:Application>
```

L'espace de noms mx correspond au paquetage racine des librairies Flex. Dans ces librairies nous trouvons tout naturellement une classe application, qui, comme le nom le suggère, gère l'application dans son ensemble. Le point important pour bien comprendre Flex est le suivant: Quand on travaille avec une balise comme mx:Application, cela équivaut à instancier la classe mx.Application comme ceci :

```
import mx.Application;

new Application();
```

C'est le compilateur MXML qui va se charger de générer le code correspondant. Laissons un instant de côté la ligne mx:Script. En dessous nous déclarons quels seront les contrôles constituant notre application. Même principe que pour la classe application. Le compilateur MXML se chargera de générer le code correspondant.

Imbriquer les balises revient à imbriquer les composants les uns dans les autres. Il y a correspondance directe entre les attributs de balises XML et les propriétés des classes. Une propriété est définie lorsque la classe est dotée de deux accesseurs *get* et *set* (ou seulement *get* pour une propriété en lecture seule) pour ladite propriété, de façon semblable à ce que l'on a en C#.
Si nous regardons attentivement ce code, nous verrons une exception à ce qui vient d'être dit. "click" pour la classe mx.Button, n'est pas une propriété. C'est un nom d'événement de souris. Et la valeur qui lui est affectée est la méthode qui traitera l'événement.
En ActionScript comme en Java on s'enregistre auprès d'une source d'événement (Design Pattern Observer). Ici la méthode est *afficherMessage*. Bien remarquer la présence des parenthèses, sans elles le code compilera, mais l'exécution ne donnera pas satisfaction. Ou se situe le code de cette méthode ? Dans un fichier ActionScript, cette fois. C'est ce qu'indique la balise mx:Script que nous avions laissée de côté un peu plus haut. Cette balise relie notre application au fichier HelloActionScript.as dont voici le contenu :

```
import mx.controls.Alert;

public function afficherMessage():void
{
 Alert.show("Hello World!");
}
```

Très simple, mais surprenant de la part d'un langage orienté objet. Où sont les méthodes, où sont les classes ? Les méthodes s'appellent *function,* c'est ainsi :-) Où donc est la classe ? Nous sommes dans la portée de la classe mx.Application.
Ainsi que nous l'avions déjà évoqué plus haut. Lorsque l'on compile le fichier MXML, le fichier ActionScript lié est compilé également, un amalgame des deux est constitué et nous avons notre application Flash. Nous compilons ainsi :

```
mxmlc HelloWorld.mxlm

Intégrer l'application Flash dans une page HTML est simple:

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0//EN" "http://www.
w3.org/TR/1998/REC-html40-19980424/strict.dtd">

<html>
<head>
<title>Hello World</title>
<meta http-equiv="Content-Type" content="text/html; charset=
UTF-8">
</head>

<body>
<p>
<object type="application/x-shockwave-flash" data="HelloWorld
.swf" width="300" height="300">
<param name="movie" value="HelloWorld.swf">
</object>
</p>
</body>
</html>
```

4 S'AFFRANCHIR DE MXML

Les fichiers MXML c'est très intéressant, mais, surtout si l'on est habitué à un langage comme Java, ou si l'on trouve XML lourd à manipuler, on peut préférer coder au maximum avec ActionScript. L'exemple HelloActionScript que vous trouverez sur notre site est structuré ainsi. Voici le fichier MXML, réduit au minimum :

```
<?xml version="1.0" encoding="UTF-8"?>
<mx:Application layout="absolute" xmlns:mx="http://www.adobe.
com/2006/mxml"
applicationComplete="afficherMessage()">

<mx:Script source="HelloActionScript.as"/>

</mx:Application>
```

Nous avons donc une instance d'Application vide, liée à un script. Nous nous appuyons sur le fait que le runtime émet un événement applicationComplete lorsque notre application est chargée et initialisée. Cet événement est en quelque sorte le point d'entrée de notre code ActionScript, que voici :

```
import mx.controls.Alert;

public function afficherMessage():void
{
 Alert.show("Hello World!");
}
```

Ainsi notre application Flash affiche l'alerte dès qu'elle est chargée. Ceci est très bien, mais du coup, pour l'instant nous n'avons plus de bouton sur lequel cliquer. Nous allons résoudre ce point un peu plus loin.

5 EMBARQUER ACTIONSCRIPT DANS MXML

On peut trouver contraignant, surtout dans les cas simples, de travailler en parallèle avec deux fichiers, MXML et ActionScript respectivement. Flex offre dans ce cas la possibilité d'embarquer le code ActionScript dans le code MXML.
Voici ce que devient notre application minimaliste précédente, une fois remaniée :

```
<?xml version="1.0" encoding="UTF-8"?>
<mx:Application layout="absolute" xmlns:mx="http://www.adobe.
com/2006/mxml"
applicationComplete="afficherMessage()">

<mx:Script>
<![CDATA[
import mx.controls.Alert;

public function afficherMessage():void
{
 Alert.show("Hello World!");
}
]]>
```

```
</mx:Script>

</mx:Application>
```

6 AJOUTER DES CONTRÔLES À LA VOLÉE

Notre exemple précédent étant par trop vide, nous allons ajouter des composants à la volée, ce qui nous fera aussi faire plus ample connaissance avec le langage.
Vous trouverez, bien entendu cet exemple, 'AjoutControles', sur notre site. Le code MXML reste basique :

```
<?xml version="1.0" encoding="UTF-8"?>
<mx:Application layout="absolute" xmlns:mx="http://www.adobe.
com/2006/mxml"
applicationComplete="ajouterControles()">

<mx:Script source="AjoutControles.as"/>

</mx:Application>
```

Mais cette fois, le code ActionScript est des plus intéressants :

```
import flash.events.MouseEvent;
import mx.controls.Button;
import mx.controls.Label;

var monlabel:Label;

public function ajouterControles():void
{
 var bouton:Button = new Button();
 bouton.label = "Clic !";
 bouton.x = 130;
 bouton.y = 250;
 this.addChild(bouton);

 monlabel = new Label();
 monlabel.x = 10;
 monlabel.y = 10;
 monlabel.htmlText = "<b>Programmez!</b>"
 this.addChild(monlabel);

 bouton.addEventListener(MouseEvent.CLICK, monEcouteur);
}

function monEcouteur(evt:MouseEvent) : void
{
 monlabel.htmlText = "<b>Abonnez-vous ! :-)</b>"
}
```

Rappelons-nous que nous sommes dans la portée d'une classe de type application, le compilateur l'ayant généré pour nous lors de la compilation du fichier MXML. Tout se passe comme si notre code était entouré de :

```
public class MonApplication extends Application
{
 // notre code
}
```

Ainsi *monlabel,* bien que déclaré 'var', n'est pas une variable globale, en dépit des apparences, mais bien un membre de la classe dérivée de mx.Application. On remarque une particularité de ActionScript: le type d'une variable, ou le type retourné ou reçu par une méthode suit le nom de l'entité, à la manière du langage Pascal. Pour le reste, notre code ressemble étrangement à du Java. Avec cette différence que les imports de paquetage nécessaires à la compilation d'une classe se font ***à l'intérieur*** de celle-ci, chose qui n'existe pas en Java. Pour ce qui est de la gestion des événements, tout fonctionne à la Java, sur la base du Design Pattern Observateur. (Cf. Programmez! n° 93). Notre classe enregistre, auprès du bouton, via la méthode *addEventListener* de celui-ci, la méthode *monEcouteur* qui fera office de gestionnaire d'événement. Cette méthode sera automatiquement invoquée lorsque l'utilisateur cliquera sur le bouton. Que faisons-nous dans le corps de cette méthode ? Nous modifions le texte affiché par le contrôle de type mx.Label. Nous rencontrons pour l'occasion une différence notable entre ActionScript et Java. En Java nous aurions quelque chose comme :

```
monlabel.setText("Abonnez-vous");
```

Ici nous avons directement

```
monlabel.htmlText = "<b>Programmez!</b>"
```

htmlText est une propriété comme en C#. La syntaxe est différente mais le principe est le même. Nous manipulerons des propriétés dans notre dernier exemple. Le dernier point à commenter est celui du positionnement des contrôles. Comme en Java, il existe des gestionnaires de mises en forme, ou layout. Nous avons utilisé le layout "absolute", qui comme le nom l'indique permet de positionner les contrôles par des coordonnées. L'utilisation de ce layout est annoncée dans le fichier MXML :

```
<mx:Application layout="absolute" etc.
```

Était-il possible de définir le layout dans le code ActionScript plutôt que dans le fichier MXML ? Oui. Layout est une propriété de la classe mx.Application. Nous savons que dans notre fichier ActionScript nous sommes sans la portée d'une classe dérivée, donc tout simplement nous pouvions écrire :

```
public function ajouterControles():void
{
 this.layout = "absolute";
 var bouton:Button = new Button();
 bouton.label = "Clic !";
 // etc, etc.
}
```

7 CRÉER DES COMPOSANTS RÉUTILISABLES

Un des points forts de Java est la réutilisation des composants. Nous avons aussi cette possibilité avec ActionScript. Pour notre dernier exemple (Composant sur notre site) nous voulons créer un panneau présentant les mêmes fonctionnalités que dans notre exemple précédent. Il s'agit donc de mettre un label et un bouton dans un

container de type mx.Panel. Nous voulons que notre composant puisse être configuré de l'extérieur. Concrètement, pour cet exemple, nous voulons que le texte affiché par le mx.Label puisse être défini depuis un fichier MXML utilisant le composant. Nous devons faire en sorte que ce texte soit vu comme une propriété. Voici le code ActionScript :

```
package fred.composants
{
  import flash.events.MouseEvent;
  import mx.controls.Button;
  import mx.controls.Label;
  import mx.containers.Panel;

  public class MonPanel extends Panel
  {
    private var monlabel:Label = new Label();
    private var bouton:Button = new Button();

    public function MonPanel()
    {
      super();

      layout = "absolute";
      bouton.label = "Clic !";
      bouton.x = 110;
      bouton.y = 200;
      this.addChild(bouton);

      monlabel = new Label();
      monlabel.x = 10;
      monlabel.y = 10;
      // initialisation déplacée dans le fichier mxml
      // qui utilise le composant
      //monlabel.htmlText = "<b>Programmez!</b>"
      this.addChild(monlabel);

      bouton.addEventListener(MouseEvent.CLICK, monEcouteur);
    }

    private function monEcouteur(evt:MouseEvent) : void
    {
      monlabel.htmlText = "<b>Abonnez vous ! :-)</b>"
    }

    public function get texte():String
    {
      return monlabel.htmlText;
    }

    public function set texte(t:String):void
    {
      monlabel.htmlText = "<b>" +  t + "</b>";
    }
  }
}
```

Nous voyons que, pour créer un composant réutilisable, la première chose à faire est de placer celui-ci dans un paquetage. Nous sommes dans le royaume de la programmation objet. Il est donc naturel de déclarer une classe dérivant de mx.Panel. Cette fois la classe sera bien toute entière dans notre fichier ActionScript :-) Classiquement pour des langages objet nous avons un constructeur. Bien remarquer l'appel, requis, du constructeur de la classe de base avec super(). La fin du code définit la propriété texte simplement en déclarant deux accesseurs *get* et *set*. Nous devons maintenant compiler ce code. Pour cela, le fichier source doit être déposé dans une arborescence de répertoire conforme au paquetage. Le compilateur dédié à la création de composants s'appelle *compc.* Voici l'incantation pour produire un fichier MonPanel.swc contenant le composant.

```
compc -o MonPanel.swc -source-path ./ -include-classes fred.
composants.MonPanel
```

Il ne nous reste plus qu'à utiliser le composant. Voici le fichier MXML qui réalise cela:

```
<?xml version="1.0" encoding="UTF-8"?>
<mx:Application layout="absolute" xmlns:mx="http://www.adobe.
com/2006/mxml"
xmlns:demoComp="fred.composants.*">

<demoComp:MonPanel id="ComposantMonPanel"
x="0"
y="0"
width="300"
height="300"
texte="Programmez!"
>
</demoComp:MonPanel>

</mx:Application>
```

Nous devons tout d'abord déclarer un espace de noms qui permettra au compilateur de trouver la (ou les) classe du composant. Cet espace de noms est ensuite utilisé comme nous avons utilisé l'espace de nom mx précédemment. On remarque l'initialisation de notre propriété texte, et du même coup, une petite subtilité. Nous ne pouvons pas donner :

```
texte="<b>Programmez!</b>"
```

Le compilateur mxmlc ne l'accepterait pas. C'est pourquoi les balises HTML sont ajoutées dans le getter de la propriété texte.

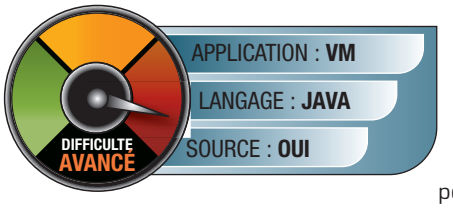
8 CONCLUSION PROVISOIRE

Cet article vous a donné l'essentiel de ce qui faut connaître pour se lancer dans la programmation d'applications Flash qui décoiffent. Mais le sujet est très loin d'être épuisé. Dans un article à venir, nous aborderons des notions plus avancées. Ainsi une application Flash n'est pas confinée dans un bac à sable. Elle peut communiquer avec l'extérieur, le navigateur pour commencer, bien évidemment, ou avec un Web service.

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

Observer le fonctionnement de la JVM Java à travers JVM TI

Comment les choses se passent-elles au sein d'une machine virtuelle. Que ce soit pour résoudre un problème de performance ou pour satisfaire la curiosité, l'interface native JVM TI permet de répondre à cette question.



Une machine virtuelle Java est un environnement logiciel complexe, mais ouvert sur l'extérieur. Elle propose notamment deux interfaces pour travailler avec du code natif. La première interface, JNI, permet d'invoquer, depuis un programme Java, des librairies écrites en C ou C++ pour répondre à un besoin de performance précis. L'autre est JMV Tool Interface ou JVM TI. On utilisera cette interface pour observer le comportement interne d'une JVM au travail ou bien pour déterminer la cause d'un problème de chute de performance ou de deadlock entre deux threads. Nous allons nous divertir un peu avec JVM TI. JVM TI n'est pas le seul moyen de monitorer une JVM. Il existe aussi JMX, pour Java Management Extensions, qui permet de faire ce travail en Java. JMX est à la base de JConsole, un utilitaire inclus dans les JDK depuis Java 5.0. JMX est un sujet à part entière, passionnant lui aussi, que nous aborderons sans doute dans un article à venir. Mais pour aujourd'hui nous dégainons notre bon vieux compilateur C++

1 JVM TI ET LE CODE NATIF

Travailler avec JVM TI est passionnant, ou très utile pour localiser un point de contention entre threads, mais pas nécessairement immédiat pour le codeur Java. Pour lui C/C++ ne sont pas nécessairement des langages qui vont de soi. En outre, si JVM TI est entièrement documentée, on ne peut pas dire que cette documentation soit d'une clarté limpide. Le but de cet article est de montrer les bases et principes de la programmation JVM TI, bases qui permettront de mieux comprendre et exploiter les exemples assez complexes qui viennent dans la JDK. Enfin, pour une raison que l'on s'explique mal, les exemples de la JDK et le peu de documentation sont écrits en C. Alors que la JVM elle-même est écrite en C++, comme chacun sait. Nous allons donc employer ce langage qui allège l'écriture et la rend beaucoup plus agréable.

2 HELLO WORLD

Commençons par l'exemple le plus simple et le plus traditionnel qui soit, afin de déjouer les premières difficultés. Pour travailler avec JVM TI, on n'écrit pas une application autonome en C ou C++ mais une librairie partagée, autrement dit un fichier .dll sous Windows et un fichier .so sous Linux ou Solaris. Une telle librairie s'appelle un agent, dans le jargon JVM TI. Les exemples de cet article, disponibles sur notre site, ont été écrits sous Windows avec Visual Studio 2008. Utiliser un autre compilateur ne pose en principe aucun problème. On configurera l'environnement de développement, C ou C++, afin que le compilateur puisse accéder à deux fichiers en-têtes de la JDK. [Fig.1] Le premier, jvmti.h, se trouve sous le répertoire include de celle-ci, le second, jni_md.h, étant spécifique à la plate-forme hôte, se trouve dans un sous répertoire à include. Par exemple le répertoire linux ou win32. Pourquoi un agent doit-il être une librairie partagée ? Parce que tout son code est constitué de fonctions de rappel, ou callback, déclarés et installés lors du chargement de l'agent par la JVM lors de son démarrage. Un agent n'est donc pas une application autonome. Une seule fonction de rappel doit toujours exister. Elle se nomme Agent_OnLoad, et c'est dans son corps que seront faites toutes les initialisations. Voici un agent minimaliste :

```
#include "jvmti.h"
#include <iostream>

using namespace std;

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM *vm, char *options, void *reserved) {
    jvmtiEnv               *jvmti;
    jint                    res;

    // Obtenir l'environnement JVM TI environment (jvmti).
    res = vm->GetEnv(reinterpret_cast<void **>(&jvmti), JVMTI
_VERSION_1);
    if(res!=JNI_OK)
   cerr << "Impossible d'obtenir un environnement JVM TI" << endl;
  else
    cout << "Hello JVM TI" << endl;

  return res;
}
```

À la différence de tous les autres callback qui reçoivent directement un environnement JVM TI, et sans que l'on sache très bien pourquoi, la fonction de rappel Agent_OnLoad reçoit un pointeur sur un objet 'machine virtuelle' dont la méthode GetEnv nous permet d'obtenir un pointeur sur un environnement JVM TI, ce que nous faisons

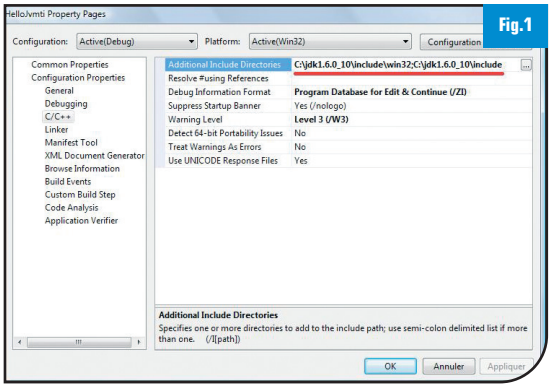


Fig.1

Nos projets sous Visual Studio sont configurés pour trouver les en-têtes de la JDK.

dès la première ligne de code. C'est ici que l'on apprécie C++. En C la ligne de code serait

```
res = (*vm)->GetEnv(vm, (void **)&jvmti, JVMTI_VERSION_1);
```

ce qui est assez laid. Si l'environnement JVM TI est obtenu, nous affichons fièrement Hello World. Une fois ce code compilé, nous avons donc une librairie. Comment l'utiliser ? La première chose à faire est de la faire pointer par le PATH du système si vous travaillez sous Windows, ou par la variable d'environnement LD_LIBRARY _PATH, si vous travaillez sur un système à la Unix. Ensuite on passe le nom de la librairie, sans l'extension du nom de fichier, à la JVM. La compilation de notre exemple, disponible sur notre site, produit la librairie HelloJvmti.dll. En supposant que l'on veuille instrumenter la JVM alors qu'elle exécute la démo Java2Demo de la JDK, on donnera la commande :

```
java -agentlib:HelloJvmti -jar Java2demo.jar
```

3 CAPACITÉS D'UN AGENT ET ÉVÉNEMENTS

Notre premier agent ne fait rien de très excitant. JVM TI fonctionne sur la base d'une politique minimaliste afin de nuire au minimum à la performance de l'application instrumentée. Tant que nous ne lui indiquons rien, elle se limite à invoquer la fonction OnLoad, ainsi que la fonction symétrique OnUnload. Hormis pour ces deux fonctions, JVM TI fonctionne de manière asynchrone, en déclenchant des événements qui doivent être traités dans des fonctions de rappel, ou callback. La liste des nombreux événements susceptibles d'être émis est évidemment donnée dans la documentation. Pour indiquer que notre agent veut travailler avec un événement précis, nous devons procéder en trois étapes.

- D'abord nous devons ajouter cet événement aux capacités de JVM TI, ce qui se fait en renseignant la structure de données prévue à cet effet.
- Ensuite nous devons indiquer au système notre fonction de rappel pour cet événement, en renseignant une autre structure de données.
- Enfin nous devons activer le mécanisme pour cet événement en invoquant une fonction de l'environnement JVM TI. Sans cette dernière étape, qu'il est aisé d'oublier, rien ne se passe.

Voici donc un autre exemple (JvmtiEvent sur notre site) qui suit ces règles mais qui ne fait toujours pas grand-chose. Il permet seulement de visualiser les phases d'initialisation de la machine virtuelle.

```
#include "jvmti.h"
#include <iostream>
#include <cstdlib>

using namespace std;


void JNICALL cbVMDeath(jvmtiEnv *jvmti_env, JNIEnv* jni_env)
{
 cout << "Evenement VMDeath" << endl;
}

void JNICALL cbVMStart(jvmtiEnv *jvmti, JNIEnv *env)
{
 cout << "Evenement VMStart" << endl;
}
```

```
void JNICALL cbVMInit(jvmtiEnv *jvmti_env, JNIEnv* jni_env, jthread
thread)
{
 // Pas très correct, mais pour l'instant ça fonctionne
 // cf exemple suivant JvmtiThreads
 cout << "Evenement VMInit" << endl;
}


JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM *vm, char *options, void *reserved) {
    jvmtiEnv               *jvmti;
    jint                    res;
 jvmtiCapabilities capabilities;
 jvmtiEventCallbacks callbacks;
 jvmtiError error;


    // Obtenir l'environnement JVM TI environment (jvmti).
    res = vm->GetEnv((void **)&jvmti, JVMTI_VERSION_1);
    if(res!=JNI_OK)
 {
   cerr << "Impossible d'obtenir un environnement JVM TI" << endl;
   return res;
 }
 else
   cout << "Hello JVM TI" << endl;

 // Définir les capacités de l'agent
 memset(&capabilities, 0, sizeof(jvmtiCapabilities));
 // rien de particulier pour l'instant ici
 error = jvmti->AddCapabilities(&capabilities);


 // Définitions des fonctione de rappel de l'agent
 memset(&callbacks, 0, sizeof(callbacks));
 callbacks.VMStart = &cbVMStart; // JVMTI_EVENT_VM_START;
    callbacks.VMInit = &cbVMInit;  // JVMTI_EVENT_VM_INIT
    callbacks.VMDeath = &cbVMDeath; // JVMTI_EVENT_VM_DEATH
 error = jvmti->SetEventCallbacks(&callbacks, sizeof(callbacks));
 if (error != JVMTI_ERROR_NONE )
   cerr << "Impossible de definir les fonctions de rappel de l'agent"
<< endl;

 // Activation des événements, un par un
 error = jvmti->SetEventNotificationMode(JVMTI_ENABLE,
   JVMTI_EVENT_VM_START, NULL);
 if (error != JVMTI_ERROR_NONE )
   cerr << "Impossible d'activer JVMTI_EVENT_VM_START" << endl;

 error = jvmti->SetEventNotificationMode(JVMTI_ENABLE,
   JVMTI_EVENT_VM_INIT, NULL);
 if (error != JVMTI_ERROR_NONE )
   cerr << "Impossible d'activer JVMTI_EVENT_VM_INIT" << endl;

 error = jvmti->SetEventNotificationMode(JVMTI_ENABLE,
   JVMTI_EVENT_VM_DEATH, NULL);
 if (error != JVMTI_ERROR_NONE )
   cerr << "Impossible d'activer JVMTI_EVENT_VM_DEATH" << endl;
```

```
  cout << "Chargement de l'agent termine" << endl;
  return JNI_OK;
}

JNIEXPORT void JNICALL
Agent_OnUnload(JavaVM *vm)
{
  cout << "Dechargement de l'agent" << endl;
}
```

Ce programme fonctionne, mais il n'est par encore écrit tout à fait dans les règles de l'art. Nous reprendrons ce point dans notre dernier exemple. En attendant, nous allons encore enrichir notre code.

4 CHARGEMENT DE CLASSE, ALLOCATION MÉMOIRE ET PILE D'APPELS

JVM TI pouvant déclencher des événements pour tout, nous pouvons nous amuser à regarder vivre la JVM lorsqu'elle exécute une application importante. Pour notre exemple, nous avons pris la démonstration Java2Demo.jar de la JDK. Voici le code (JvmtiLoadAlloc sur notre site) d'un agent qui intercepte un événement chaque fois qu'une classe est chargée depuis un fichier et qu'une allocation mémoire pour les données d'une classe est effectuée. Le premier événement, identifié par JVMTI_EVENT_CLASS_FILE_LOAD _HOOK, nous permet d'afficher le nom du fichier depuis lequel la classe est chargée, ce que nous faisons. Mais son utilisation la plus raffinée est la redéfinition à la volée du byte-code de la classe chargée. Par ce moyen, il est possible de modifier des méthodes, par exemple à des fins de profilage, sans devoir toucher au code de l'application Java. Cette approche, appelée byte-code instrumentation (BCI), est assez complexe et mériterait un article à elle seule. Si vous êtes intéressés, n'omettez pas, pour vos essais, de vous appuyer sur l'exemple *hprof* qui vient avec votre JDK, ainsi que sur la librairie que vous trouverez dans l'exemple java_crw_demo. Ces deux ressources vous éviteront l'écriture de beaucoup de code et quelques migraines. Le second événement que nous traitons est JVMTI_EVENT_VM_OBJECT_ALLOC, déclenché lorsque la JVM alloue de la mémoire pour les données d'une classe. Si la taille de cette mémoire allouée est inférieure à 50 octets, nous ne faisons rien. Si elle est supérieure, nous imprimons cette quantité, et si cette quantité est supérieure à 150 octets, nous imprimons le nom des 5 premières méthodes présentes sur le cadre de pile au moment de l'allocation. Voici un extrait de notre code :

```
static void JNICALL cbClassFileLoadHook(jvmtiEnv *jvmti, JNIEnv* env,
            jclass class_being_redefined, jobject loader,
            const char* name, jobject protection_domain,
            jint class_data_len, const unsigned char* class_data,
            jint* new_class_data_len,
            unsigned char** new_class_data)
{
  cout << name << endl;
}

static void JNICALL cbObjectFree(jvmtiEnv *jvmti, jlong tag)
{
  // Jamais appelé si les objets libérés
  // ne sont pas tagués dans bVMObjectAlloc
}
```

```
static void JNICALL
cbVMObjectAlloc(jvmtiEnv *jvmti, JNIEnv *env, jthread thread,
    jobject object, jclass object_klass, jlong size)
{
  char *className;
  char *declaringClassName;
  jclass declaringClass;
  char *methodName;

  if(size > 50)
  {
    jvmti->GetClassSignature(object_klass, &className, NULL);
    cout << "Classe allouee: " << className << endl;
    if (className != NULL)
      cout << "Classe alloue: "
        << className << "Taille: " << size << endl;
  }

  if(size >= 150)
  {
    // Afficher la trace d'appel
    jvmtiFrameInfo frames[5];
    jint nb;
    jvmtiError error;

    error = jvmti->GetStackTrace(NULL, 0, 5, frames, &nb);
    if (error == JVMTI_ERROR_NONE && nb >= 1)
    {
      for (int i = 0; i < nb; i++)
      {
        error = jvmti->GetMethodName(frames[i].method,
          &methodName, NULL, NULL);
        if (error == JVMTI_ERROR_NONE)
        {
          jvmti->GetMethodDeclaringClass(
            frames[i].method, &declaringClass);
          jvmti->GetClassSignature(
            declaringClass,
            &declaringClassName, NULL);
          cout << "Methode: " << methodName
          << " de la classe: " << declaringClassName
            << endl;
        }
      }
    }
  }
}
```

Une fois l'application Java2Demo chargée, il est amusant de regarder "vivre" la JVM, de constater à quelle fréquence des petites zones mémoires sont allouées et libérées, ou encore de constater que les classes ne sont chargées qu'au moment effectif de leur première utilisation.

5 TRAVAILLER AVEC LES THREADS

Abordons notre dernier exemple qui veut montrer les démarrages et arrêts de threads. Examiner les threads est un des intérêts majeurs de JVM TI car cela permet de localiser facilement les points de contention d'une application complexe. Pour nous qui prenons

DÉVELOPPEZ VOTRE SAVOIR-FAIRE

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 !

*11 numéros par an : 49 €**

Economisez 16,45 €*

**Tarif France métropolitaine*

Choisissez votre formule

- **Abonnement 1 an au magazine : 49 €** (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*
- **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 59 €** *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*

+ Abonnement INTÉGRAL

ACCÈS ILLIMITÉ aux ARCHIVES du MAGAZINE pour 0,84€ par mois !

Cette option est réservée aux abonnés pour 1 an au magazine, quel que soit le type d'abonnement (Standard, Numérique, Etudiant). Le prix de leur abonnement normal est majoré de 10 € (prix 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 vous abonner en ligne et trouver tous les tarifs*** www.programmez.com

PROGRAMMEZ

☐ **Abonnement 1 an au magazine : 49 €** (au lieu de 65,45 € tarif au numéro) *Tarif France métropolitaine*

☐ **Abonnement Intégral : 1 an au magazine + archives sur Internet et PDF : 59 €** *Tarif France métropolitaine*

☐ **Abonnement Etudiant : 1 an au magazine : 39 €** (joindre copie carte étudiant) *Offre 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

Offre limitée, valable jusqu'au 28 février 2010

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 127

contact avec JVM TI, cela va aussi nous forcer à bien prendre conscience de la nature fondamentalement asynchrone de JVM TI. Être notifié du démarrage ou de l'arrêt d'un thread s'obtient de la même façon que les autres notifications vues précédemment. Nous écrivons donc un premier code (JvmtiThreadsBad sur notre site) et nous le lançons sur Java2Demo, application multithreadée s'il en est. Et nous obtenons un résultat assez surprenant. D'abord la sortie émise par notre agent est chaotique. Ensuite les résultats ne semblent pas refléter fidèlement la réalité, enfin, et c'est le plus étonnant, nous avons la surprise d'être notifiés du démarrage de threads avant d'être notifiés de l'événement JVMTI_EVENT_VM_INIT, ce qui signifie que les threads démarrent avant que la machine virtuelle Java ne soit initialisée…

6 SYNCHRONISER LES FONCTIONS DE RAPPEL

JVM TI ne donne aucune garantie qu'un événement ne sera notifié que lorsque la fonction de rappel de l'événement notifié avant lui aura rendu la main. C'est à nous de nous charger de cela, ce qui pour parler Java, revient à faire de nos callbacks des fonctions synchronisées. On emploie pour cela des monitors, le mécanisme interne de la JVM pour les méthodes de classes synchronisées. Pour faire simple, un monitor est un sémaphore dans le monde de Java. Nous pouvons donc écrire un code qui donnera cette fois toute satisfaction. Dans le callback Agent_OnLoad, dont on est sûr qu'il rendra la main avant toute notification d'événement, nous créons un monitor, rangé dans une variable globale. Ensuite, le code de tout callback est mis en sandwich entre l'acquisition du monitor et sa libération. Voici un extrait de JvmtiThreads, notre dernier exemple que vous trouverez complet sur notre site.

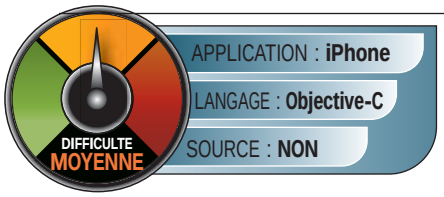
```
jrawMonitorID mon_monitor;

void JNICALL cbVMInit(jvmtiEnv *jvmti_env, JNIEnv* jni_env, jthread
thread)
{
 jvmtiError error;

 error = jvmti_env->RawMonitorEnter(mon_monitor);
 cout << "Evenement VMInit" << endl;
 error = jvmti_env->RawMonitorExit(mon_monitor);
}

void JNICALL cbThreadStart(jvmtiEnv *jvmti_env,
       JNIEnv* jni_env, jthread thread)
{
 jvmtiError error;
 jint nb;
 jthread* ptr;
 jvmtiThreadInfo info_ptr;


 error = jvmti_env->RawMonitorEnter(mon_monitor);
 cout << "Demarrage Thread";
 // Combien de Thread sont démarrés ?
 error = jvmti_env->GetAllThreads(&nb, &ptr);
 if(nb >= 1)
   cout << " " << nb << " threads vivants" << endl;
 else
   cout << endl;
 jvmti_env->Deallocate(reinterpret_cast<unsigned char*>(ptr));

 // Obtenir des infos sur le thread démarré
 if(nb >= 1)
 {
   error = jvmti_env->GetThreadInfo(thread, &info_ptr);
   if(error == JVMTI_ERROR_NONE)
     cout << "Nom du thread: "
       << info_ptr.name << ", priorite: "
     << info_ptr.priority << endl;
     jvmti_env->Deallocate(
       reinterpret_cast<unsigned char*>(info_ptr.name));
 }
 error = jvmti_env->RawMonitorExit(mon_monitor);
}

void JNICALL cbThreadEnd(jvmtiEnv *jvmti_env,
       JNIEnv* jni_env, jthread thread)
{
 jvmtiError error;
 jint nb;
 jthread* ptr;

 error = jvmti_env->RawMonitorEnter(mon_monitor);

 cout << "Arret Thread";
 error = jvmti_env->GetAllThreads(&nb, &ptr);
 cout << " " << nb << " threads vivants" << endl;
 jvmti_env->Deallocate(reinterpret_cast<unsigned char*>(ptr));

 error = jvmti_env->RawMonitorExit(mon_monitor);
}

JNIEXPORT jint JNICALL
Agent_OnLoad(JavaVM *vm, char *options, void *reserved) {
    jvmtiEnv               *jvmti;
    jint                    res;
 jvmtiCapabilities capabilities;
 jvmtiEventCallbacks callbacks;
 jvmtiError error;

 // comme dans les exemples précédents, puis:
 error = jvmti->CreateRawMonitor("mon monitor", &mon_monitor);
 if (error != JVMTI_ERROR_NONE )
   cerr << "Impossible de creer le monitor" << endl;
}
```

On remarquera encore, dans ce code, l'appel à la fonction JVM TI Deallocate, pour libérer de la mémoire allouée pour nous par le système. On ne lira jamais trop attentivement la documentation à ce sujet… Enfin pour bien travailler avec JVM TI, le code de nos callback doit être réentrant. Cela signifie principalement que l'on doit utiliser des variables sur la pile, ou sinon que les accès à des variables globales doivent être effectués dans les sandwiches d'acquisition et de libération d'un monitor.

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

Animer votre iPhone !

Dans notre précédent article qui décrivait la plate-forme de développement iPhone (Programmez ! 121), nous avions survolé rapidement une des fonctionnalités du SDK iPhone : les animations. Je vous propose d'en faire cette fois un tour plus approfondi, cet article s'adresse donc à un public de développeurs avertis. Loin d'être un simple gadget les animations apportent un réel plus pour l'utilisateur, mais attention, les pièges sont nombreux et nous verrons comment s'en prémunir.



1 QU'EST-CE QUE CORE ANIMATION ?

Sur iPhone, les animations font intégralement partie de l'expérience utilisateur, elles sont d'ailleurs très clairement mentionnées dans la référence Apple pour le design d'applications, à savoir le très complet *« iPhone Human Interface Guidelines »* (ce n'est pas pour rien qu'Apple est reconnu pour la qualité de ses interfaces graphiques). Avant tout, il s'agit d'un moyen pour communiquer des informations à l'utilisateur, en termes d'usabilité on parle de « feedback ». Par défaut, dans un *NavigationController* l'écran s'anime de la gauche vers la droite pour montrer que l'on a navigué vers le détail. On peut l'observer par exemple dans les préférences. Pour autant, ces effets graphiques doivent rester subtils et ne pas entraver le déroulement de l'application, il est en effet facile de succomber à l'effet « Star Wars » vu les nombreuses possibilités de cette bibliothèque. Pour résumer simplement, Core Animation est une bibliothèque permettant d'effectuer des animations très simplement sans que le développeur ait à se préoccuper des détails techniques. Ainsi la gestion des threads ou bien la fréquence de rafraîchissement en fonction des ressources disponibles sont prises en charge de façon transparente.

C'est une technologie relativement récente, qui a fait son apparition avec Mac OS 10.5 et qui a été écrite à l'origine pour l'iPhone OS. Si l'on regarde d'un peu plus près, on s'aperçoit d'ailleurs que le niveau d'intégration est très fort entre UIKit et Core Animation. *UIView,* la classe mère pour tous les contrôles graphiques contient une propriété *layer* de type *CALayer* et n'est donc qu'une surcouche à Core Animation. UIKit rajoute la gestion des interactions utilisateur, tandis que la composition des différentes couches qui composent une interface graphique et la possibilité de les animer les unes par rapport aux autres sont déléguées à Core Animation.

2 TRAVAUX PRATIQUES : UN EFFET GÉNIAL

Core Animation permet de réaliser des animations complexes en seulement quelques lignes de code, pourtant l'API est très riche et la présenter sans exemple peut vite se révéler indigeste. Je vous propose donc de la mettre en œuvre et d'introduire les concepts au fur et à mesure. Les exemples *FrontRow* ou *CoverFlow* étant déjà des classiques, intéressons-nous à la nouvelle animation sur les listes Genius qui a été introduite avec iTunes 9 et l'OS 3.1.2 : une mosaïque présente 4 illustrations d'albums qui se retournent successivement pendant que la pochette de l'album en cours de lecture s'agrandit pour occuper tout l'écran. L'animation ne dure même pas une seconde en tout mais l'effet est assez sympathique. [Fig.1]

Fig.1

Bootstrap du projet

Pour commencer, nous allons créer un nouveau projet « Genius » à partir du template *« View based application »* et modifier le ViewController pour créer les éléments graphiques que nous allons animer. Le but n'est pas de faire l'architecture projet la plus parfaite mais de faire court et se concentrer sur Core Animation. Créez le projet, ajoutez quelques images dans les ressources (Img1.png, Img2.png etc.), puis modifiez la classe *GeniusViewController* comme suit :

GeniusViewController.h

```
#import <UIKit/UIKit.h>

@interface GeniusViewController : UIViewController {
  UIButton *buttonView;
  UIImageView *albumView;
}

@property (nonatomic, retain) UIButton *buttonView;
@property (nonatomic, retain) UIImageView *albumView;

@end
```

L'animation consiste à faire pivoter les différentes pochettes et en même temps faire apparaître au premier plan l'image principale. Nous avons donc défini cette image ainsi qu'un bouton qui va contenir la mosaïque de pochettes et qui répondra à l'événement *UIControlEventTouchUpInside* pour lancer l'animation.

GeniusViewController.m

```
#import "GeniusViewController.h"
#import <QuartzCore/QuartzCore.h>
```

```
@implementation GeniusViewController

@synthesize buttonView;
@synthesize albumView;

- (void)dealloc {
 [buttonView release], buttonView = nil;
 [albumView release], albumView = nil;
    [super dealloc];
}

- (void)viewDidLoad {

 // Create the main button
 UIButton *button = [[UIButton alloc] initWithFrame:CGRectMake
(10, 80, 300, 300)];
 [button addTarget:self action:@selector(triggerGeniusAnimation)
 forControlEvents:UIControlEventTouchUpInside];
 [self.view addSubview:button];
 self.buttonView = button;
 [button release];

 // Add the mosaic in the button
 CGFloat width = CGRectGetWidth(button.frame) / 2;
 for (int i = 1; i <= 4; i++) {
   CGRect imageFrame = CGRectMake(width * (i % 2), i < 2 ? 0 : width,
 width, width);
   UIImageView *imageView = [[UIImageView alloc] initWithFrame:
imageFrame];
   [imageView setImage:[UIImage imageNamed:[NSString stringWith
Format:@"Img%d.png", i]]];
   [buttonView addSubview:imageView];
   [imageView release];
 }

 // Create the album view
 albumView = [[UIImageView alloc] initWithFrame:CGRectZero];
 [albumView setImage:[UIImage imageNamed:@"Imgfull.png"]];
 albumView.contentMode = UIViewContentModeScaleAspectFit;
 albumView.layer.position = CGPointMake(CGRectGetMidX(self.view
.bounds), CGRectGetMidY(self.view.bounds));
 [self.view addSubview:albumView];
}

- (void)triggerGeniusAnimation {
}
@end
```

Fig.2

Voici ce qui devrait être affiché si vous exécutez le projet, pour le moment il ne se passe pas grand-chose mais nous avons tous les éléments en place pour la suite. [Fig.2]

Un peu de code, beaucoup de théorie

Implémentons maintenant la méthode *triggerGeniusAnimation* comme suit pour créer la première animation, celle de l'image principale :

```
// Make the main illustration grow
(1) CABasicAnimation *anim = [CABasicAnimation animationWith
KeyPath:@»bounds»];
(2) anim.timingFunction = [CAMediaTimingFunction functionWith
Name:kCAMediaTimingFunctionLinear];
(3) anim.duration = 0.5;
(4) anim.toValue = [NSValue valueWithCGRect:CGRectMake(0, 0, 320, 460)];
(5) anim.fillMode = kCAFillModeBoth;
(6) anim.removedOnCompletion = NO;
(7) [albumView.layer addAnimation:anim forKey:nil];
```

Intéressons-nous au code précédent pour comprendre ce qui se passe exactement. Nous avons mentionné plus haut la propriété *layer* d'une vue, la classe *CALayer* est une des classes fondamentales de Core Animation. C'est ce qui définit une zone dessinée à l'écran ; elle possède en outre, de nombreuses propriétés comme la position, la transparence, une bordure ou plus important : une liste de sous-layers. L'interface graphique est ainsi composée d'un arbre de composants et chaque layer définit son système de coordonnées par rapport à son parent. Il existe plusieurs sous-classes spécialisées en fonction de leur contenu, par exemple *CAEAGLLayer* permet de rendre de l'OpenGL.

Ici nous avons défini une animation qui modifie la taille du *layer* de l'image principale pour lui faire occuper tout l'écran. La plupart des propriétés d'un *layer* sont susceptibles d'être animées, par exemple l'opacité ou la position, mais il en existe plus d'une vingtaine. Il est également possible d'hériter d'une *CALayer* et de lui rajouter des propriétés animables.

Pour définir l'animation, nous avons créé une instance de la classe *CABasicAnimation* qui permet de créer une animation où les étapes intermédiaires sont interpolées automatiquement. Il suffit donc de spécifier l'état de début et de fin et de laisser à Core Animation le soin de tout calculer à notre place. C'est exactement ce qui s'est passé ici (code ci-dessus) :

- (1) L'animation porte sur la propriété bounds que nous avons spécifiée à la création.
- La valeur de départ est la taille actuelle du layer.
- La valeur de fin d'animation est spécifiée via la propriété *toValue* (4), pour les connaisseurs on notera que Core Animation est basé sur la programmation KVC (*Key Value Coding*).

Deux propriétés sont un peu plus obscures et nécessitent quelques explications, à savoir *fillMode* et *removedOnCompletion* (5, 6). Dans Core Animation il faut distinguer les valeurs des propriétés d'un *layer* des valeurs qui sont affichées à l'écran, l'animation modifie l'apparence mais sans modifier réellement ces valeurs. Il est d'ailleurs possible de connaître l'état d'un layer via deux méthodes :

- -(id)modelLayer : renvoie une instance de *layer* représentant l'état de son modèle
- -(id)presentationLayer : renvoie une approximation du *layer* tel qu'il est affiché

Une fois l'animation terminée le *layer* retourne à son état d'origine, ce qui peut surprendre. Comme nous voulons conserver l'état de fin nous avons donc défini que l'animation continue à être appliquée (*removedOnCompletion*) avec la valeur finale de l'animation (*fillMode*).

Animations en 3D

Implémentons le reste de l'animation : les quatre illustrations doivent se retourner les unes après les autres. Pour cela, nous allons utiliser la propriété *transform* d'un *CALayer* qui permet d'appliquer

une transformation de type rotation, translation ou échelle et ce sur les 3 axes X, Y, Z. Bien évidemment cette propriété est susceptible d'être animée. Rajoutez le code ci-dessous à la fin de la méthode *triggerGeniusAnimation* :

```
// Flip albums
CGFloat timeOffset = 0;
for (CALayer *sublayer in buttonView.layer.sublayers) {

  CABasicAnimation *animation = [CABasicAnimation animationWith
KeyPath:@»transform»];
  CATransform3D transform = CATransform3DMakeRotation(M_PI/2, 0,
 1, 0);
  animation.timingFunction = [CAMediaTimingFunction functionWith
Name:kCAMediaTimingFunctionEaseIn];
  animation.toValue = [NSValue valueWithCATransform3D:transform];
  animation.duration = 0.2;
  animation.beginTime = CACurrentMediaTime() + timeOffset;
  animation.fillMode = kCAFillModeBoth;
  animation.removedOnCompletion = NO;
  [sublayer addAnimation:animation forKey:nil];

  timeOffset += 0.1;
}
```

Il suffit d'itérer sur les sous-layers du bouton et d'appliquer une rotation de PI/2 sur l'axe Y, ce qui va donner la sensation que la pochette tourne jusqu'à être complètement masquée.
Nous jouons également sur un délai d'exécution pour donner un effet « domino ».

Pour aller un peu plus loin

N'ayant pas la place de traiter complètement un si vaste sujet, voici deux pistes intéressantes à creuser si vous voulez en savoir plus sur Core Animation :

- Dans les deux exemples précédents nous avons toujours créé et ajouté un objet de type *CAAnimation*, ce mode déclaratif est appelé **animations explicites**. Core Animation permet un deuxième modèle de programmation appelé **animations implicites** où l'on associe une animation qui sera lancée pour toute modification d'une propriété (voir les méthodes *defaultActionForKey* et *actionForKey*).
- L'OS 3 a apporté à *CALayer* de nouvelles sous-classes : *CAGradientLayer*, *CAShapeLayer* qui sont deux layers spécialisées dans le dessin de gradient et de path.

3 NE PAS GRILLER SON BUDGET SUR UNE ANIMATION

Les animations arrivent généralement tard sur un projet iPhone, il est courant de se concentrer d'abord sur le design et l'enchaînement des écrans. Ne les négligez pas pour autant : c'est ce qui fait toute la richesse d'une application iPhone. Il vaut donc mieux anticiper le coût de développement et de débogage.
Règle d'or : **toujours tester sur le device** (si possible les 3 générations) et le plus **tôt possible**. Le simulateur ne suffit absolument pas pour valider une animation et peut masquer un problème de performance ou pire un comportement totalement différent. Rien de pire que de se rendre compte après à la fin de la journée que l'animation ne passe pas du tout sur un iPhone.

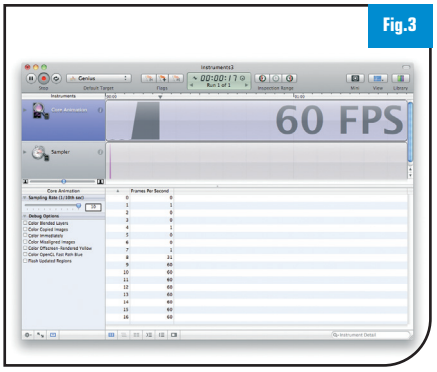


Fig.3


Fig.4

Comment régler un problème de performance ? Tout d'abord le rendre visible, le mesurer et enfin le diagnostiquer. Les outils du SDK iPhone permettent heureusement tout cela.
Dans Xcode exécutez le projet que nous avons réalisé plus tôt avec les outils de diagnostic de performance activés en passant par l'option *Run → Run with Performance Tool → Core Animation* ce qui va nous permettre de mesurer précisément le nombre de FPS (images par seconde) que nous obtenons sur le device. Pour ce genre de test il est intéressant de répéter l'animation pour obtenir une mesure moyenne, on peut utiliser la propriété repeatCount d'une animation pour cela. [Fig.3]
Sur l'iPhone 3GS de test, l'animation tourne déjà au maximum de FPS, c'est normal, les transformations sont accélérées au niveau hardware et il y a peu d'éléments présents à l'écran. Même si en apparence tout est parfait, il se trouve que l'on peut quand même optimiser l'affichage. En cochant l'option *« Color Blended Layers »* l'iPhone se met à afficher des zones rouges et vertes : [Fig.4]
Les zones rouges correspondent aux layers avec de la transparence et qui nécessitent un calcul plus complexe pour le rendu final. C'est évidemment à éviter au maximum pour avoir les meilleures performances possibles. Dans notre exemple, nous n'en avons pas besoin, il suffit donc de supprimer la composante alpha des images pour que tout devienne vert. Une deuxième option d'Instruments permet de diagnostiquer un problème de performances. *« Flash Updated Regions »* montre en temps réel quelles sont les zones qui sont redessinées à l'écran en les affichant en jaune. Il est alors possible de détecter que certaines portions sont redessinées alors qu'elles ne devraient pas l'être. En résumé voici les trois conseils que nous avons présentés, ce n'est évidemment pas exhaustif mais devrait vous éviter quelques soucis :

- Tester au plus tôt sur le device
- Privilégier les vues opaques
- Minimiser les zones de redessin

Core Animation est une bibliothèque très complète et qui à permis à l'iPhone de se démarquer de la concurrence en proposant une expérience utilisateur riche et inédite. J'espère vous avoir fourni les notions nécessaires pour que vos applications en tirent également parti.

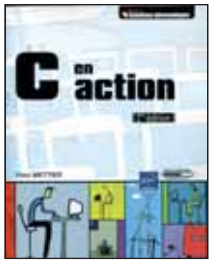
Ressources

- http://developer.apple.com/mac/library/documentation/Cocoa/Conceptual/CoreAnimation_guide/Introduction/Introduction.html
- http://developer.apple.com/iphone/library/documentation/userexperience/conceptual/mobilehig/Introduction/Introduction.html

■ Olivier **Martin** *est expert senior mobilité au sein d'OCTO Technology, cabinet d'architectes en systèmes d'information* (http://blog.octo.com, http://universite-du-si.com)

LANGAGE

C en action 2e édition



*Difficulté : ****
Editeur : Eni éditions
Auteur : Yves Mettier
Prix : 39 €

Vous ne connaissez pas le langage C et vous rêvez de programmer avec ? Ce livre, 650 pages tout de même, est fait pour vous ! Quel que soit votre niveau, C en action pose les bases du langage et permet d'apprendre les fondamentaux : les outils de compilation, les fonctions et bibliothèques, la gestion des erreurs, le réseau, l'exécution parallèle, etc. Bref tout y passe. Très didactique, l'auteur pose le problème, en discute puis donne la solution, avec le code. Pratique ! D'autant plus que le code source est disponible en ligne !

AGILITÉ

L'art du Lean Software Development

*Difficulté : ****
Editeur : Dunod
Auteur : Curt Hibbs
Prix : 24,90 €

Le Lean Software Development hérite de la méthode de production et d'amélioration continue Lean initiée par Toyota. Méthode de agile par excellence, le Lean est un cycle perpétuel d'amélioration. Cet ouvrage n'est pas un cours magistral sur la méthode mais plutôt une introduction pour bien comprendre les bases de cette méthode, sa philosophie.

LANGAGE

Oracle APEX

*Difficulté : ****
Editeur : Eni éditions
Auteur : Ahcène Bourouis
Prix : 54 €

Connaissez-vous Oracle Application Express ? Non ? Il est temps de découvrir cet environnement de développement, gratuit, d'Oracle. L'ouvrage se découpe en trois grandes parties : initiation, niveau avancé et administration. Car on va de la conception au déploiement de son application. L'objectif est de développer des applications web pour l'environnement Oracle. Tout d'abord, on démarre par l'outil dans la gamme d'Oracle, son utilité, l'environnement de travail, puis on attaque avec un hello world. Rapidement, l'auteur dévoile les notions de pages, de composants, les ajouts de validation ou l'utilisation des thèmes et des modèles dans son projet. Les parties administration et déploiement permettent aux développeurs de mieux connaître ces mécanismes internes que l'on ne maîtrise pas toujours...

LIVRES DU MOIS

Programmation GWT 2

*Difficulté : *** - Editeur : Eyrolles*
Auteur : Sami Jaber - Prix : 35 €

Librairie bien connue des développeurs Javascript et Java, Google Web Toolkit est désormais en version 2.0. Elle améliore bon nombre de fonctions et introduit aussi de multiples nouveautés. L'auteur, bien connu de la communauté GWT, propose ici de brosser un tableau très complet de GWT 2 : environnement de développement, CSS, intégration de code javascript, services distants, design pattern, intégration JEE, création de ses composants, les contrôles, etc. Clair et largement illustré d'exemples, ce livre est incontournable pour bien démarrer ou approfondir GWT et son incroyable richesse !

Spring par l'exemple

*Difficulté : *** - Editeur : Pearson*
Auteur : Gary Mak - Prix : 42 €

Spring est un puissant framework de développement et applicatif JEE. Il se veut accessible et simple d'adoption. Il possède de puissantes fonctions : modèle MVC, persistance, tests, debug, déploiement, etc. Aujourd'hui assez largement utilisé, Spring doit cependant être maîtrisé si on veut l'exploiter au mieux. Ici, l'auteur explore le framework avec de nombreux exemples très concrets en expliquant chaque contexte, avec à chaque fois la solution, les conseils. Une bonne initiation.

LANGAGE

L'art du développement Android

*Difficulté : ***
Editeur : Pearson
Auteur : Mark Murphy
Prix : 28,90 €

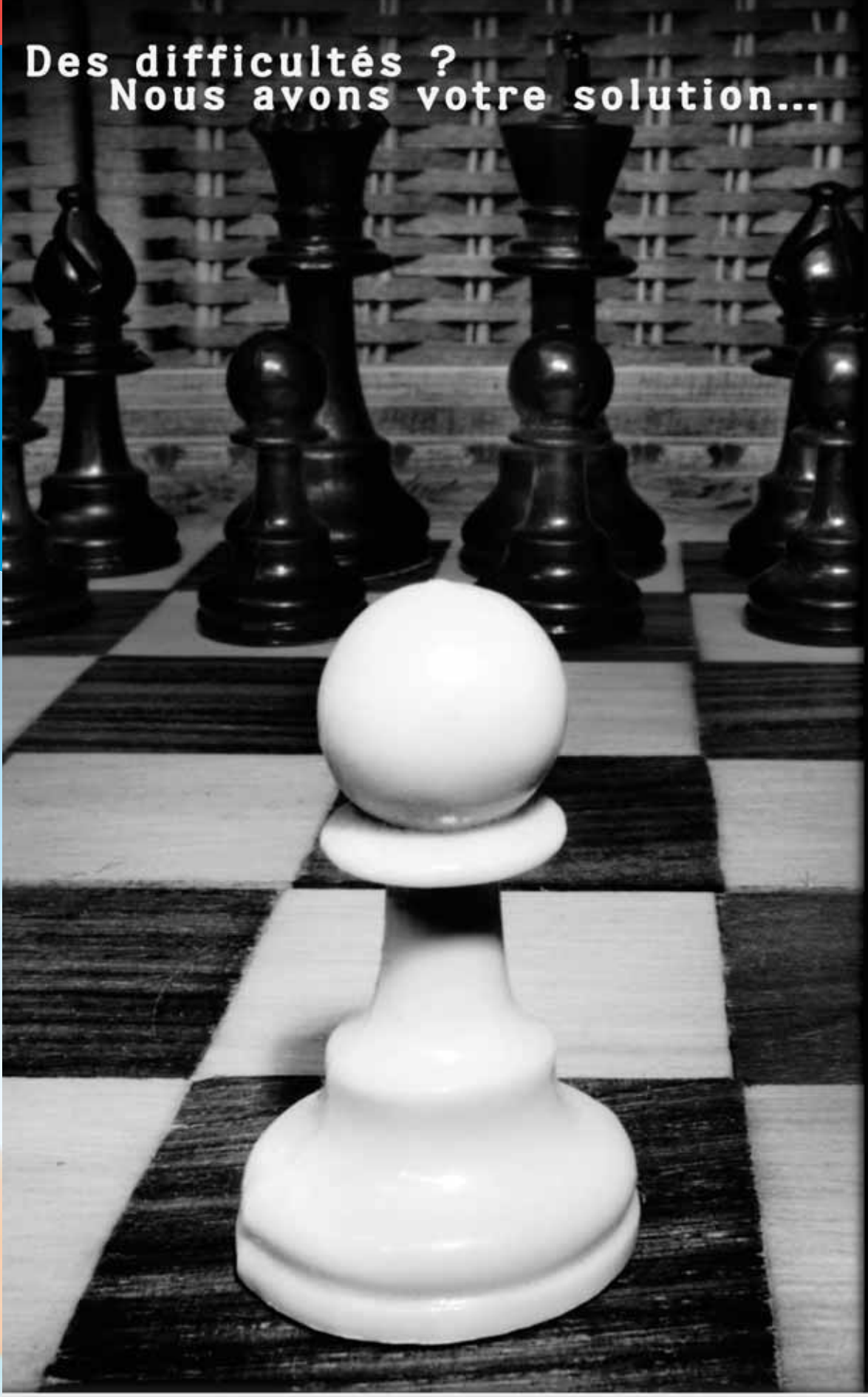
Impossible de passer à côté du phénomène Android ! Développeur et membre de la communauté, Mark Murphy connaît très bien le système mobile et son modèle de développement. D'une manière claire, sans assommer le lecteur de nombreuses lignes de code, l'auteur explique pas à pas la programmation Android : interface, librairies, GPS, services web, sécurité, etc. Les cas concrets de projets permettent d'y voir plus clair et d'éviter les grosses erreurs... Bref une bonne approche !

WEB

Réussir son site web avec XHTML et CSS 3e édition

*Difficulté : ***
Editeur : Eyrolles
Auteur : Matthieu Nebra
Prix : 25 €

Non, HTML n'est pas mort, au contraire ! Il est toujours nécessaire de bien maîtriser HTML, XHTML, les CSS pour concevoir son site web. Mais il n'est pas toujours simple de bien manipuler ces éléments. L'auteur propose une nouvelle édition de son "Réussir". Et le but est que tout le monde puisse comprendre, créer rapidement son premier site avec les outils, les éléments et les techniques fondamentales du Web et notamment les fameuses CSS. Abondamment illustré, le livre procure aussi de nombreux conseils. Bref, voici un ouvrage à mettre entre toutes les mains ! Bon point : les notions d'accessibilité sont abordées.



Virtualisation - Identité & Sécurité - Messagerie - Poste de Travail - Management des Infrastructures - Collaboration

nelite

NELITE FRANCE

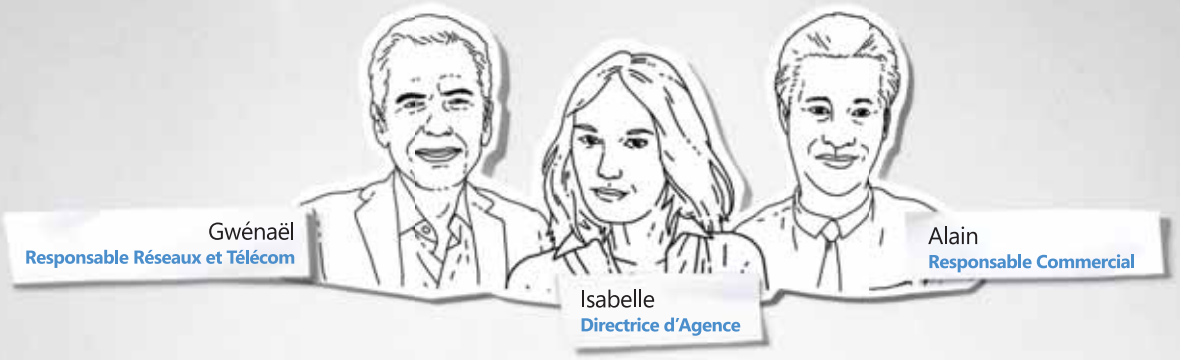
FORMAONE

NELITE NORTH AFRICA

Votre Expert de l'intégration des Solutions Microsoft & de Virtualisation

Web. www.nelite.com | Tel. 0 820 866 899 | Email. contact@nelite.com

Microsoft



Communiquer et partager les idées en toute sécurité

Parfois, l'information que l'on recherche est sur un disque dur, parfois elle se trouve dans la tête de quelqu'un. En déployant Microsoft Exchange Server, avec ses services de Communications Unifiées, SharePoint, et ForeFront, vous ferez des économies en permettant à vos collaborateurs d'échanger des idées où qu'ils soient et d'accéder aux informations dont ils ont besoin en toute sécurité.

Pour une communication et une collaboration plus efficace, plus sûre et plus facile à gérer, rendez-vous sur : **www.nouvelle-efficacite.fr**

Parce que c'est l'affaire de tous