

PROgrammez !

www.programmez.com

mensuel n°120 - juin 2009

Multi-plate-forme Votre code fonctionne-t-il sur tous les systèmes ?

*Qt, compilation croisée,
les outils, le problème des
navigateurs web.*

Web 2.0

Spécial SILVERLIGHT

- *Bien débuter avec Silverlight*
- *Tout savoir sur Expression 3.0*
- *Silverlight sur votre Desktop*
- *Silverlight pour Eclipse*

Le développeur devient AGILE !

- *Adopter une méthode agile.*
- *Scrum + eXtreme Programming : le cocktail parfait*
- *Les experts dévoilent leurs secrets agiles*

Programmation MULTICOEUR

*Devenir un développeur parallèle
Les enjeux des processeurs multicore*

Langage
F# : le nouveau visage de .Net

Cloud Computing
Déployer Java sur Amazon EC2

Méthode
Créer vos DSL sous Eclipse

Concurrence
Maîtriser le Concurrency and Coordination Runtime

SGBD
LINQ et MySQL : c'est possible !



Développez 10 fois plus vite

Logiciel professionnel. Documentation contractuelle. Support technique gratuit. 15 requêtes sur la version en cours de commercialisation.
*: WINDEV a été élu « Langage le plus productif du marché » par les lecteurs de la revue « Programmez! », octobre 2008



WINDEV®

RÉUSSISSEZ TOUS VOS PROJETS
AVEC L'OUTIL DE DÉVELOPPEMENT
ÉLU **LE PLUS PRODUCTIF***



VERSION EXPRESS GRATUITE
Téléchargez-la !

WINDEV 14 est l'environnement de développement totalement intégré (IDE, ALM), intégralement en français, réputé pour sa **puissance** et sa **facilité** d'utilisation.

WINDEV 14 est livré complet: éditeur d'analyses (UML,...), **RAD**, patterns, **lien avec toutes les bases de données (ODBC, OLE DB)**, Oracle, SQL Server, AS/400, Informix, DB2..., lien natif MySQL, PostgreSQL, **base de données Client/Serveur HyperFileSQL gratuite incluse**, Générateur d'états PDF, Codesbarres, **Accès natif SAP R/3**, **Lotus Notes**, Gestion de planning, Gestion des Exigences,

L5G, SNMP, **Bluetooth**, TAPI, OPC, FTP, HTTP, Socket, Twain, API, DLL, **domotique**, liaisons série et USB, **débugage à distance**, profiler, refactoring, **génère d'applications JAVA**, multilangue automatique, **Gestionnaire de versions**, Installateur 1-clic, etc...

Les applications créées fonctionnent avec toutes les versions de Windows: 98, 2000, NT, 2003, XP, Vista, sous TSE et Citrix, sur eeePC...

WINDEV 14 gère le **cycle complet de développement**, pour des équipes de 1 à 100 développeurs. Le support technique est **gratuit***

Vous aussi, développez 10 fois plus vite... avec WINDEV 14.



VOTRE CODE EST MULTI-PLATEFORMES:

Windows, .Net, Java, PHP, J2EE, XML, Internet, Ajax, Pocket PC, SmartPhone, Client riche ...

Logiciel professionnel
DEMANDEZ LE DOSSIER GRATUIT
252 pages + DVD + Version Express incluse + 112 Témoignages.
Tél: **04.67.032.032** ou **01.48.01.48.88**
info@pcsoft.fr



Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr



Sommaire

\\ actus
 L'actualité en bref8
 Agenda8



12

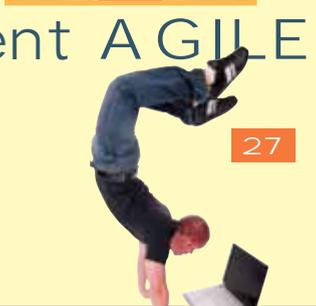
\\ événements
 Piloter l'ordinateur par la pensée12

\\ gros plan Multi-plate-forme
 Multi-plate-forme : où, quand, comment, pourquoi ?15
 Développer multi-plate-forme en.net avec Delphi PRISM16
 La compilation croisée entre Linux et Windows19
 Omake, le système de build multi-plate-forme qu'il vous faut22
 Le multi-navigateur : le guide de survie du développeur web24



15

\\ dossier : Le développeur devient A GILE
 Les méthodes agiles pour mieux développer27
 eXtreme Programming : les fondamentaux pour le développeur agile32
 La journée d'un développeur agile34
 Standardiser la résolution de problèmes dans une équipe agile36
 L'agilité, une révolution culturelle37
 Vendre Scrum à une équipe qui pratique V38
 Adopter le mode agile en début d'un projet ou en cours de route41



27

\\ développement web
Bien débuter en Silverlight44
 Microsoft Expression Studio 346
 Silverlight 3 envahit aussi le bureau50



46

\\ technique
 Cloud Computing : déploiement d'applications Java56
 Pourquoi passer à la **programmation parallèle**58
 Intel Parallel Studio débarque60



60

\\ code
 Comprendre les Design Pattern Fabrique62
 CCR, une API de programmation concurrente pour .net66
 Introduction à la construction d'un DSL sous Eclipse70
 Eclipse Tools for Silverlight : Silverlight sur Eclipse, c'est possible !73
 Utilisation de Linq avec différentes bases de données75
 F# : l'avenir de .net et du développement Windows ? w(1re partie)78



70

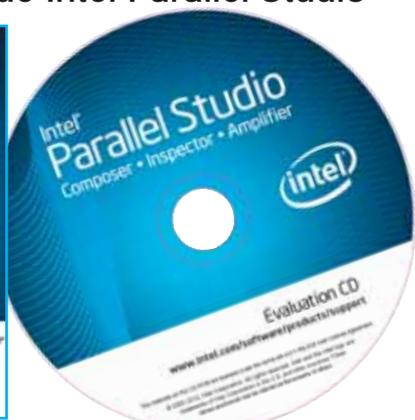
\\ temps libre
 En direct des labos (2e partie)81
 Les livres du mois82

Le CD du mois : Découvrez la version finale de Intel Parallel Studio

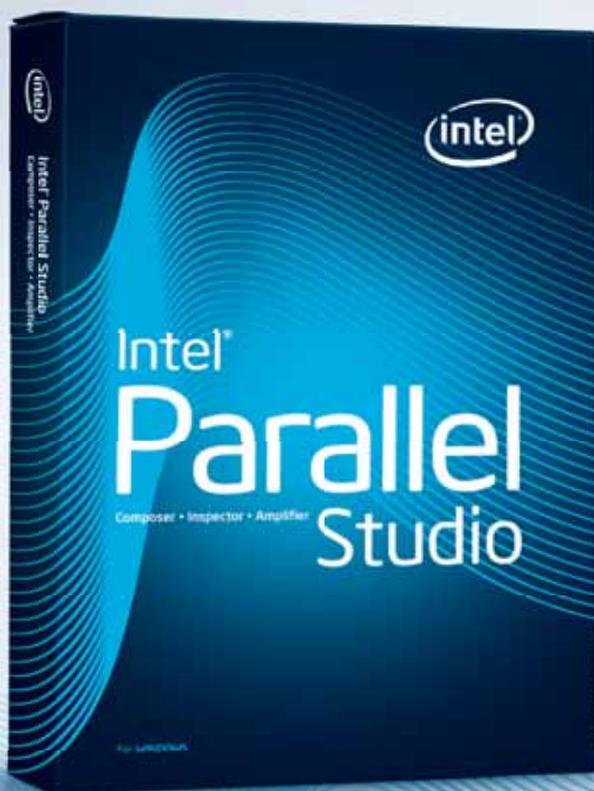
Intel® Parallel Studio
 Create optimized serial and parallel applications with the ultimate all-in-one parallelism toolkit.
 • Intel® Parallel Composer
 • Intel® Parallel Inspector
 • Intel® Parallel Amplifier
 System Requirements
www.intel.com/software/products/parallelstudio/requirements
 Support
 Intel® Parallel Studio products feature access to community forums and a Knowledge Base for all your technical support needs, including technical notes, application notes, and documentation, and all product updates.
 For more information, go to www.intel.com/software/products/parallelstudio/support

NOUVEAU Evaluation CD for Microsoft Visual Studio

Démarrez le développement parallèle avec la toute dernière boîte à outils Intel®



CD ROM



L'INNOVATION EN PARALLELISME.

FAITES EVOLUER VOTRE CODE.

Réalisez des applications parallèles robustes, de l'analyse à la compilation, du débogage à l'optimisation. Intel® Parallel Studio est la panoplie d'outils ultime pour les applications Windows*. Les outils sophistiqués qui simplifient le développement multicore pour tous les développeurs Microsoft Visual Studio* C/C++. Tout ce dont vous avez besoin pour passer du code sériel au parallèle et du parallèle à la perfection.

Disponible maintenant: www.intel.com/software/parallelstudio





Le développeur serait-il un chat ?

3 %. C'est le pourcentage de développeurs ayant sur ces derniers mois introduit une fonction parallèle dans un code source. Une petite poignée! Voilà un chiffre bien intrigant, non ? Et pourtant, depuis plus de 18 mois, Programmez ! s'investit autour de la programmation multicore / parallèle pour vous sensibiliser sur un sujet hautement stratégique et d'une complexité non négligeable.

Aujourd'hui, l'écrasante majorité des PC vendus possède au moins 2 cœurs. Et cela monte très vite à 8, 16 cœurs. D'ici à 2010, les fondeurs estiment à 100 % les nouvelles machines qui seront multicœur, que se soit dans le desktop, le serveur, le portable, et même les téléphones portables ! L'argument anti-parallèle est cependant non négligeable : la plus part des PC actuellement en activité ne sont pas multicœurs. Mais d'ici 5 ans, le parc explosera. Et cinq ans ce n'est pas grand chose...

C'est donc aujourd'hui qu'il faut comprendre la programmation parallèle. Car cette programmation est loin d'être " friendly " comme vous le verrez rapidement dans notre grand dossier à paraître sur plusieurs numéros. On change de philosophie de code. Il faut penser multi dimension du code en quelque sorte : exécuter plusieurs tâches parallèlement. Avec toute la complexité de gestion, de concurrence, de mémoire, etc. Les risques de bugs se multiplient et, en programmation multicœur, ces erreurs ne pardonnent pas. Cependant, n'allons pas croire qu'il faille paralléliser son code à outrance. Au contraire, un code trop parallèle peut s'avérer plus lent qu'un code " normal " ! Il faut donc définir les portions du code à paralléliser. La priorité sera donnée aux fonctions très exigeantes en puissance de calcul comme les traitements Audio / Vidéo, les requêtes de données, la compression, la cryptographie, etc.

Demain, les outils, les langages vont offrir de puissantes fonctions parallèles directement dans la machine virtuelle (ex. : Java et .Net) et des bibliothèques vont aider le gentil développeur à paralléliser en quelques minutes un code. Mais, n'attendez pas non plus la solution miracle ! Le parallélisme nécessite une solide compétence technique et une rigueur de codage extrême. Cependant, le développeur parallèle est réellement l'avenir et là, il apporte une valeur, une compétence.

Autre sujet chaud du moment, lui aussi récurrent me direz-vous, l'agilité. Non pas que le développeur doive devenir aussi souple physiquement qu'un acrobate... mais il doit intégrer le merveilleux monde des méthodes agiles. Depuis quelques mois, le couple Scrum / eXtreme Programming connaît un succès fulgurant dans les équipes de développement. Et pour cause, Scrum offre un cadre organisationnel, et XP redonne joie et plaisir aux développeurs. Et oui, fini le temps du programmeur à la chaîne, codant sans réellement réfléchir... ! Le développeur revient au centre du projet, avec de nouvelles responsabilités. Où, quand, comment ? C'est ce que l'on verra dans notre grand dossier du mois.

Si nous avons prédit la survie, la mort puis la résurrection de notre ami le développeur, aujourd'hui, nous lui prédisons sa réincarnation. Comme le chat, le développeur n'aurait-il que neuf vies ? Rendez-vous au numéro 150 de Programmez ! pour le savoir. Le suspense sera insoutenable...

■ François Tonic
Rédacteur en chef

Rédaction : redaction@programmez.com
 Directeur de la Rédaction : Jean Kaminsky
 Rédacteur en Chef :
 François Tonic - ftonic@programmez.com
 Ont collaboré : F. Mazué, J.B. Boisseau, G Vidal.
 Experts : B. Boucard, P. Guermontprez, O. Lance,
 A. Dupuis, J. Mezhrhid, F. Queudret, L. Laslaz,
 O. Juncu, H. Harrath, A. Peter, S. Boigelot,
 L. Bar, J. Bernard, D. Vojtisek, T. Jaskula,
 N. Biedermann, C. Durand, R. Pierquin, P. Blayo.
 Publicité : Régie publicitaire, K-Now sarl
 Pour la publicité uniquement :
 Tél. : 01 41 77 16 03 - diff@programmez.com
 Editeur : Go-02 sarl, 6 rue Bezout
 75014 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 1 CD-Rom

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 exclusive-
 ment sur www.programmez.com

L'INFO
 PERMANENTE
 WWW.PROGRAMMEZ.COM



PROCHAIN
 NUMÉRO

N°121
 juillet-août 2009,
 parution 30 juin

✓ Développement
**Attention le futur
 est déjà là !**

Préparez-vous cet été :

- Windows 7, Android, iPhone, Surface, Java 7
- OpenVibe : l'informatique pilotée par la pensée

✓ Enquête carrière :

- Etes-vous satisfait de votre job ?

Votre potentiel, notre passion.™

Microsoft®





LES BONNES EQUIPES MARQUENT.
MAIS LES MEILLEURES GAGNENT
LA PARTIE EN DEPLOYANT PLUS VITE.


Microsoft®
Visual Studio®
Team System

RELEVEZ TOUS LES DÉFIS. TOUS ENSEMBLE.

Tout le monde veut un code de qualité, mais personne n'a suffisamment de temps pour le produire. Alors n'attendez plus et jouez collectif : pensez Visual Studio® Team System 2008. Téléchargez une version d'essai sur releveztouslesdefis.com

■ **Embarcadero** sort une nouvelle offre dédiée aux données et Java afin de détecter et de corriger les problèmes de performances, DB Optimizer 1.5 et Performance Center. Ces outils permettent de profiler et de régler finement les codes Java et SQL depuis la couche applicative Java jusqu'à la base de données, facilitant ainsi le développement de codes et la détection des problèmes avant qu'une requête applicative ou de base de données ne soit envoyée à la production.

■ **JetBrains a dévoilé un IDE Ruby** : RubyMine 1.0. L'un des avantages clés de RubyMine réside dans la compréhension poussée des spécificités du langage de Ruby, sa nature dynamique, et les conventions de code. Il a connu un beau succès depuis sa disponibilité. RubyMine apporte une gamme de refactorisation efficace compatible avec Rails, simplifiant considérablement par là-même le processus de modification des codes.

■ **Quest** annonce l'intégration de vWorkplace, gestionnaire de poste de travail virtualisé, aux environnements Microsoft. Cela signifie que cette solution est désormais compatible avec System Center Virtual Machine Manager et App-V (virtualisation d'application).

■ **Perforce** a dévoilé un nouveau composant : Perforce Server Log Analyser. Il permet aux administrateurs de télécharger leurs fichiers de log pour une analyse instantanée. Ils disposent pour cela d'un accès sécurisé à leurs fichiers de log et à leurs résultats d'analyses, qui peuvent être stockés jusqu'à dix jours. Disponible gratuitement sur la Base de Connaissance du site Perforce : <https://kb.perforce.com/psla>.

■ En attendant la disponibilité de **Flex 4** et des nouveaux outils, dont Catalyst, Adobe annonce une vingtaine de sociétés qui supportent Strobe. Strobe est un framework pour aider les éditeurs et constructeurs à avoir un standard pour les media player. Les fonctions sont très intéressantes comme le streaming dynamique, la fonction pause dans du direct live, une gestion fine de la qualité de service, etc. Strobe devrait être disponible dans quelques mois. www.adobe.com/go/strobe

Marché Rachat de Sun, Borland, et d'une partie de Compuware : le nouveau visage du développement

Depuis un mois, les rachats se succèdent ! Est-ce la crise qui accélère les rapprochements et la nécessité de croissance externe des éditeurs ? Quoi qu'il en soit en quelques semaines, deux grandes figures du développement ont été absorbées : Sun et Borland. Pour Sun, on attendait depuis des mois le rachat. Si IBM tenait la corde, c'est finalement Oracle qui rafle une fois de plus la mise pour plus de 7 milliards de dollars. Oracle avait déjà une forte implication dans Java, cela ne fait que renforcer cette tendance. Mais au-delà, Oracle complète ses offres logicielles notamment avec Glassfish et les outils de sécurité de Sun ou encore sur le matériel notamment les lignes de serveurs et de stockage, tout en mettant la main sur un système d'exploitation fiable et robuste : Solaris. Et surtout, MySQL tombe sous la coupe d'Oracle. Ce rachat pose aussi des questions de pérennité de certaines offres, notamment dans le logiciel car elles font doublon avec les gammes actuelles d'Oracle comme pour le serveur d'application, la base de données. De quelle manière, MySQL sera-t-il positionné par rapport à Oracle Database ? L'autre question concerne l'implication des développeurs Sun dans les projets open source. L'effort sera-t-il poursuivi, par exemple dans OpenOffice ou encore Netbeans. Mais pour Oracle, il s'agit aussi de mieux se positionner sur le cloud computing et la virtualisation, deux marchés qui étaient très mal adressés par l'éditeur. L'autre surprise du mois est le rachat pur et simple de Borland par Micro Focus. Borland avait déjà cherché à se recentrer sur le cycle de vie, la modélisation et la gouvernan-



ce, en se délestant totalement des outils de développements, de CodeGear à Embarcadero. La concurrence accrue des outils de cycle de vie y est sans doute pour quelque chose, ainsi que la profusion d'outils de modélisation, marché sur lequel, Borland Together avait du mal à rester un acteur significatif. Cette opération se monte à environ 75 millions de dollars. Reste à savoir si l'ensemble des gammes actuelles seront poursuivies. Micro Focus a aussi mis la main sur une partie des outils de Compuware : gamme test et de qualité. Compuware cherche là aussi à recentrer son activité sur les performances et optimisation sur mainframe et la gestion du portefeuille applicatif. Ces rachats modifient le marché du développement et il faut encore s'attendre à d'importants rachats dans les mois à venir, en particulier, côté Microsoft...

agenda \

JUIN

Le 04 juin, Hôtel Castille, Paris 1er, **invitation au lancement d'Intel Parallel Studio**, la suite complète pour le développement parallèle. Programme détaillé sur : <http://www.sosdevelopers.com/in tel/form.html>

Le 09 juin, Lille, Eura Technologies, **WYGDAY 2009**, pour comprendre comment les

technologies vont supporter votre futur (Green IT, Cloud Platform, Microsoft Surface, Live Mesh, Virtual Earth/Google Maps, .NET 4 ou encore Azure). <http://wygday.wygwam.com/>

Le tour découverte Flex 3 et Air à travers la France. Ateliers gratuits de découverte des outils Adobe. http://www.baao.com/Formations/LE_TOUR_DE_FLEX_ET_AI R_.html

Du 17 au 18 juin 2009, Paris Expo - Porte de Versailles **SALON ONLINE 2009**, l'événement des solutions web et mobile <http://www.online-expo.fr/>

JUILLET

Du 06 au 07 juillet 2009, Grand amphithéâtre de l'EPITA, les **PHP Days "Industrialisez votre PHP!"** vous permettront d'acquies les compétences nécessaires à une utilisation professionnelle de PHP. Organisé par Alterway et Anaska. www.phpdays.com

Perforce, le système de Gestion de Configuration Logicielle rapide



Le support technique Perforce Des traitements rapides, des réponses précises

Nos équipes supports sont toujours disponibles afin de partager leurs expertises en personne - pas de réponses automatiques, de répondeurs ou de centres d'appel. Nos ingénieurs supports hautement qualifiés se font une fierté de traiter rapidement vos appels et de vous apporter des réponses précises.

Réaliser vos projets dans les délais nécessite des équipes supports prêtes à vous aider quand vous en avez besoin. Vous pouvez compter sur le système de GCL rapide Perforce et son support technique réputé, pour vous donner un atout gagnant.

PERFORCE
SOFTWARE

Téléchargez sans conditions une copie gratuite de Perforce sur www.perforce.com. Un service d'assistance technique gratuit est offert pendant toute la période d'évaluation.

■ **Smartesting**, spécialisé dans les outils de tests, annonce Test Designer 3.4. L'outil permet de générer automatiquement des tests fonctionnels à partir d'un module. Cette version simplifie la création des modèles de tests avec l'ajout de nouveaux assistants, l'éditeur est maintenant intégré à Eclipse. D'autre part il sait s'intégrer aux grosses solutions de tests du marché comme HP Quality Center.

■ **Oracle** lance une nouvelle version de Beehive. Celle-ci renforce le système de build unifié pour l'entreprise avec de nouveaux outils de collaboration (mail, calendrier, espace de travail en équipe, etc.). Plus que jamais l'outil cherche à aider l'entreprise à mieux collaborer et à améliorer la partie web pour le travail en équipe.

■ **Ubuntu** lance un concurrent frontal à Microsoft Live Mesh : Ubuntu One. Il s'agit d'un service stockage et de synchronisation des données sur le cloud, comme Mesh.



Une fois inscrit, il faut installer un client desktop sur son Ubuntu. On dispose d'un "desktop" virtuel comme avec Mesh mais en moins ergonomique. Dommage que le client desktop ne soit disponible que sous Ubuntu. La version de base (2 Go de stockage) est gratuite. La version 10 GB est payante (10 dollars par mois).

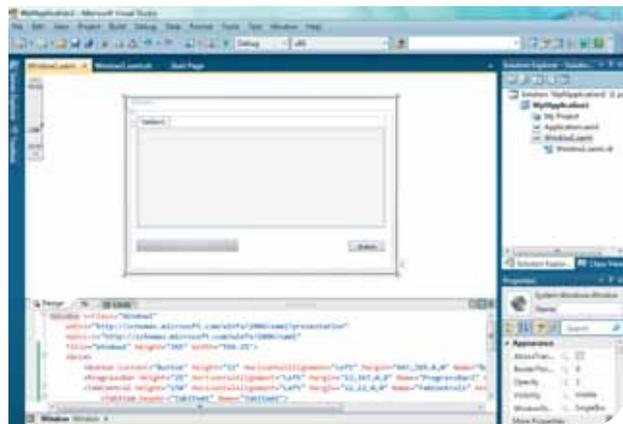
■ **Eclipse 3.5** sera disponible fin juin. Cette version apporte pas mal de changements : support de Solaris 86x pour SWP, nouveau module d'installation de logiciels, changement d'éditeurs et de pages dans les éditeurs par des raccourcis claviers, nouvelles optimisations entre Eclipse et Cocoa (MacOS X), comparateur d'API (pratique en cas de changement). Site : www.eclipse.org

■ **SpringSource** a rendu disponible SpringSource tc Server, une version entreprise de Tomcat. Elle se destine aux Datacenters, aux environnements virtuels et cloud computing. Il est prévu de livrer ce serveur sous forme d'appliance virtuelle optimisée pour VMware. Site : www.springsource.com

IDE

Visual Studio 2010 : enfin disponible en bêta !

C'est la grande nouveauté du mois de mai. Après une pré-version lourde et d'une lenteur souvent horripilante, Microsoft livre enfin aux développeurs une bêta du prochain Visual Studio qui annonce de nombreuses nouveautés.



La version bêta, disponible à l'écriture de l'article, pèse 3,6 Go en environnement .Net (4,6 Go avec C++). Bonne nouvelle dès l'installation : la présence de F# (article dans ce numéro), de Dotfuscator, des outils de tests unitaires, du .Net 4, du framework Sync.

100% WPF

Ce qui frappe d'emblée, c'est une interface assez largement modifiée. Elle se veut plus claire, moins complexe, dotée d'éditeurs plus ergonomiques. On constate aussi la présence d'une nouvelle fonction zoom surpuissante, preuve que Visual Studio s'appuie sur le framework Windows Presentation Foundation (WPF). L'un des focus de cette version concerne le cycle de vie des applications que Microsoft veut démocratiser auprès des développeurs. Team System 2010 se dote en effet

de nouvelles fonctions (fonction découverte des codes existants, nouveaux diagrammes de modélisation, outils améliorés pour la documentation...).

Visual Studio 2010 supporte l'ensemble des dernières évolutions de .Net et tout particulièrement .Net 4.0 que nous détaillerons dans les prochains numéros. Retenons ici le support natif et en managé (.Net) de Windows 7, d'une mise à niveau des outils de développement pour Office, du support de Sharepoint, de Silverlight (uniquement 2.0 dans la bêta testée). Autre grosse nouveauté, la présence du cloud computing avec une intégration transparente des kits de développement Azure. Jusqu'à présent, il était parfois délicat de les installer avec Visual Studio 2008. Autre surprise bienvenue, l'arrivée du support de DB2 et d'Oracle dans les bases de données, en plus du classique SQL Server ! Autre grosse nouveauté : le support par défaut du développement

parallèle pour les processeurs multicoeurs (en C++ et .net). Concernant la modélisation, outre la version Architect de Team System, le projet Oslo sera aussi sans doute présent. Oslo est la pierre angulaire de l'approche transversale et verticale de la modélisation chez Microsoft.

Une version prometteuse

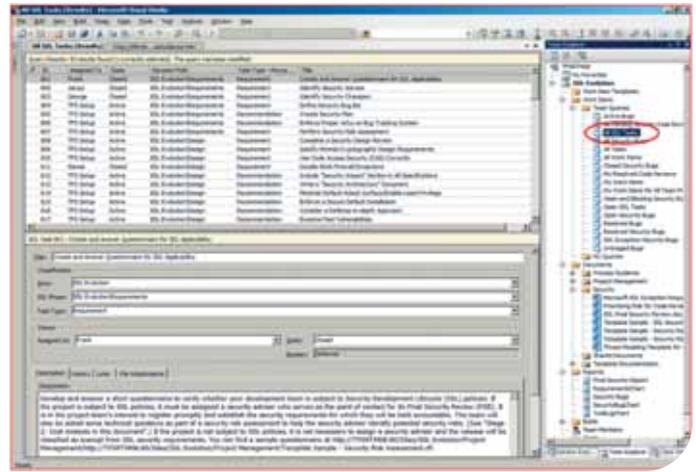
A l'heure actuelle, il n'y a pas de date officielle de sortie. On évoque souvent une disponibilité fin 2009, début 2010, sans plus de précision. Il est probable que cette sortie se déroule plusieurs semaines après Windows 7. Nos premiers tests montrent qu'il faut tout de même une machine puissante avec une grosse réserve de mémoire vive. Les lenteurs constatées ici et là (notamment sur la toolbox) apparaissent normales pour une bêta. Attendons les évolutions de performances dans les prochaines versions pour mieux juger.

Sécurité

Microsoft relance SDL

En 2002, après une série retentissante de vulnérabilités détectées dans les logiciels Microsoft, Bill Gates siffle la fin de la récréation. Les failles d'Internet Explorer et celles du système d'exploitation inquiètent les entreprises utilisatrices, et certaines commencent à tester sérieusement les offres alternatives. Bien sûr, un simulateur de vol dissimulé dans Excel n'est pas la fin du monde.

Sous le nom de Security Development Lifecycle (SDL), soit Cycle de développement sécurisé, l'initiative Microsoft a pour objectifs de mettre la sécurité au cœur des développements de tous les produits, et de partager les résultats de cette politique avec ses partenaires. Il ne faut pas croire que le premier objectif soit si simple : en effet, pour des responsables de lignes de produits, la sécurité dans un premier temps coûte cher. Il n'est donc pas surprenant de devoir attendre 2004 pour voir la démarche SDL généralisée à l'ensemble de Microsoft, et 2005 pour être rendue disponible au reste de l'écosystème. Où en somme-nous aujourd'hui, 7 ans après le lancement de ce processus ? Si vous vous rappelez le dossier dans Programmez ! concernant les 25 erreurs de programmation les plus courantes, il semblerait que nous soyons encore à l'âge de pierre en ce domaine ! D'après les statistiques citées par Microsoft, 70% des éditeurs de logiciels font leurs tests de sécurité... après la mise en vente de leur logiciel. On se heurte en fait à deux types de mentalité : le sempiternel " la sécurité coûte " et également " le système d'exploitation devrait être là pour nous protéger ". Faux, et faux.



A la mi-mai, Microsoft a lancé un " nouveau SDK " comprenant :

- Le guide SDL version 4.1.
- Une intégration d'outils SDL dans Team System, l'environnement de développement intégré de Microsoft. (via un template)
- Un partenariat stratégique : SDL Pro Network avec SANS (25 erreurs de programmation) et SAIC (une entreprise de cyber-sécurité travaillant pour le gouvernement US).

Elle répond donc aux trois objectifs : éduquer (les équipes), automatiser (les processus) et assister (les projets). Les contraintes de sécurité s'intègrent donc désormais de façon transparente aux développements, et permettent enfin le suivi en temps réel des failles de sécurité, tout en donnant une vue synthétique au moment de la revue finale du projet. Un exemple à suivre pour les autres éditeurs ?

■ Gilbert Vidal

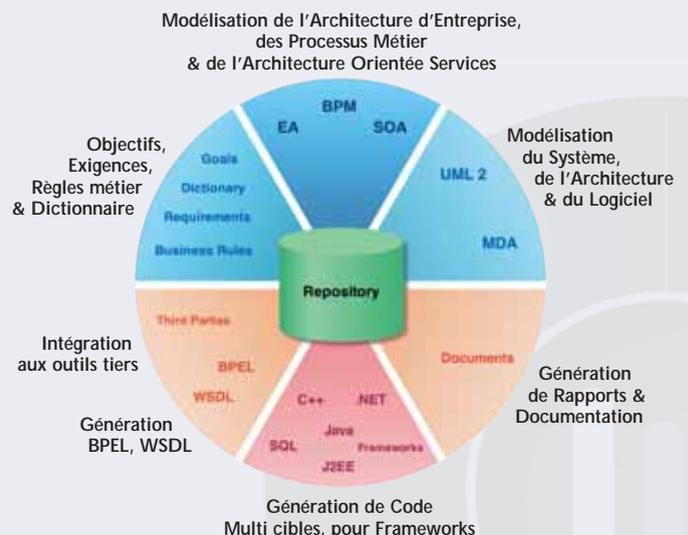
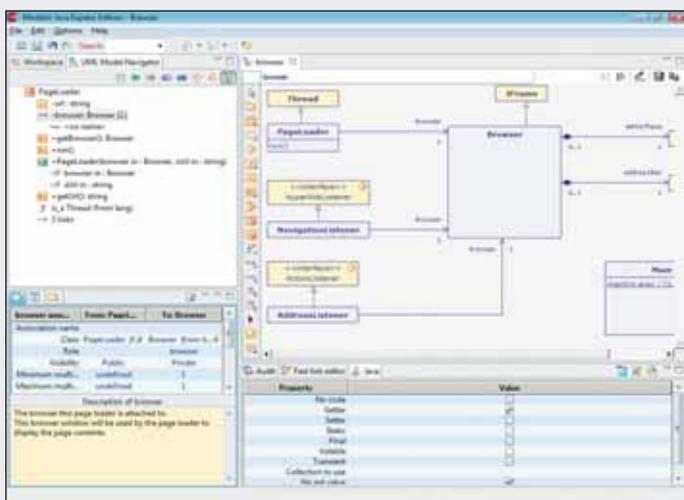


Modelio : Nouvel outil de modélisation

Modéliser sur toute la portée de l'entreprise pour un alignement SI/Métier

Modéliser n'a jamais été aussi abordable et productif

- Ergonomie simple et productive
- Support intégré de UML, BPMN, BPM, l'Architecture d'Entreprise, L'analyse des besoins, ... dans un seul référentiel
- Travail de groupe distribué, intégré à SVN/Subversion
- Des générations sur étagère adaptables pour toutes les cibles
- Support MDA : transformation, extensibilité et adaptabilité



Modelio est disponible en trois éditions

- **Free** : Un outil complet de modélisation gratuit !
- **Express Java** : Un outil de développement Java guidé par le modèle de haute performance pour seulement 100€ !
- **Enterprise** : La solution de modélisation complète, supportant le travail de groupe, extensible avec une riche palette de modules de modélisation et de génération disponibles sur étagère

PILOTER L'ORDINATEUR PAR LA PENSEE !

Crédit : Programmez.J. Kaminsky



OpenVibe, la première ICO (Interface Cerveau-Ordinateur) est disponible, en Open Source !

Neuf années-homme pour développer OpenVibe. L'équipe de Rennes de l'INRIA était fière de présenter à Paris, le 13 mai, OpenVibe, le premier logiciel français permettant d' " agir par la pensée ". Il est gratuit et open-source, vous trouverez le lien sur programmez.com.

Il est rare que la présentation d'un logiciel attire de nombreuses caméras de télévision. C'était pourtant le cas, ce 13 mai, au siège de l'Inserm (Institut National de la Santé et de la Recherche Médicale). Il est vrai que la promesse attirait légitimement la curiosité des médias et de la communauté scientifique. Nous étions là et n'avons pas été déçus. Aujourd'hui, l'équipement nécessaire est un système simple d'électrodes au niveau du cuir chevelu, permettant de capter l'activité électrique cérébrale, analogue au dispositif de l'électro-encéphalogramme.

Plusieurs expériences ont été présentées, dans des domaines de réalité virtuelle et de santé.

REALITE VIRTUELLE

L'expérience a permis de montrer que l'application détectait que le sujet avait l'intention de déplacer sa main gauche ou sa main droite. L'application passe par une phase d'apprentissage, correspondant à celui nécessaire pour un logiciel de reconnaissance de la parole. Après une série de traitements des signaux, il est possible de savoir quelle était la main utilisée pour le mouvement imaginé par exemple, l'onde cérébrale provoquée par l'imagerie mentale du sujet est détectée et commande un déplacement sur l'écran. On reconstitue en 3D, et en temps réel, l'activité cérébrale. La démonstration était dans le domaine du jeu vidéo. Les handicapés moteur pourront ainsi plus facilement accéder à l'univers virtuel. L'application peut également être un outil d'entraînement cérébral, souligne Anatole Lécuyer, responsable du projet à l'Inria

NEURO-FEEDBACK

L'apport de l'Inserm a été notamment le repérage des " marqueurs électro-encéphalographiques " : j'écoute avec l'oreille gauche ou l'oreille droite. " Nous explorons actuellement

la possibilité de communiquer avec une personne dans le coma, afin d'essayer de la sortir de son état végétatif ", explique Jean-Philippe Lachaux, un des responsables du projet, à l'Inserm. Le concept de neuro-feedback ouvre par ailleurs des champs thérapeutiques infinis. La personne essaie de modifier le " signal " : les applications concernent la douleur, ou des gênes de type accouphènes. Ou encore des troubles de l'attention chez l'enfant hyperactif ou encore la rééducation post-traumatique.

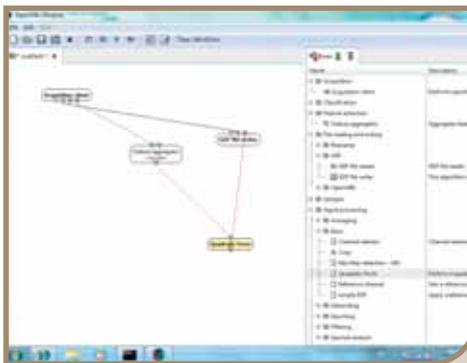
Le P300 speller : L'écriture par la pensée

Grâce au logiciel OpenVIBE, l'Inserm a développé une interface simple et conviviale qui offre la possibilité d'écrire des phrases en sélectionnant, simplement par la pensée, des lettres présentées sur un écran. L'expérience a montré que, au cours de l'affichage successif de lignes et colonnes de lettres, surlignées, la personne focalise son attention sur une lettre à épeler. Lorsque la ligne ou la colonne contient la lettre choisie, une réponse cérébrale particulière est générée. Cette réponse, connue des chercheurs, est déclenchée quand l'individu a détecté un stimulus attendu

OpenVibe : 9 années/homme

Yann Renard est l'ingénieur responsable du développement d'OpenVibe, à l'Inria. Il a œuvré avec deux autres développeurs pendant 3 ans. Il nous a livré quelques détails sur cette application de 9 années/homme : " Le code est en C++. L'application comporte plus de 150 000 lignes de code. Elle est très modulaire, à base de composants connectés par une interface graphique drag&drop. L'architecture est planifiée pour le multi-cœur, et prête pour les clusters, permettant d'agencer les modules existants et ceux qui seront ajoutés " Anatole Lécuyer, directeur du projet, l'a initié en 2005, dans le cadre d'une unité de l'Inria dédiée aux mondes virtuels, basée à Rennes. L'ouverture de la forge de OpenVibe représentait pour lui et pour l'équipe le commencement dans la vie réelle d'un projet qui ira loin.

qui apparaît de manière imprévisible. Elle survient environ 300 ms après la stimulation, d'où son nom de P300. Il est de cette manière, possible de savoir sur quelle lettre la personne focalisait son attention.



CONTRIBUEZ au Projet OpenVibe !

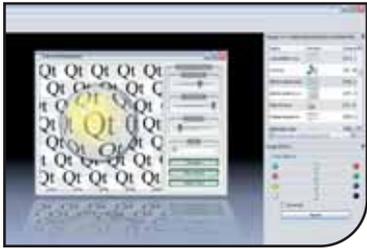
Sur le site officiel, les développeurs disposent de plusieurs outils. Un Designer permet de modéliser et de tester son Interface Cerveau-Ordinateur. Acquisition Server permet de communiquer entre le matériel d'acquisition (= casque) et l'application OpenVibe.

Disponible en open source, OpenVibe évoluera et se complètera grâce à vous, les développeurs. Programmez ! suivra régulièrement les évolutions ! Le code source et les binaires (137 Mo) sont disponibles sur Linux et Windows. Le projet est sous licence LGPL 2. A vous de jouer. Site : <http://openvibe.inria.fr>. (Rendez-vous également sur www.programmez.com !)

■ **Windows 7** est désormais disponible en version RC. La version finale devrait être disponible vers octobre ou novembre si tout va bien. Pour débiter dès maintenant les tests de compatibilité des applications, Microsoft France a ouvert un site spécifique : Windows 7 Applications et Compatibilité. Cette excellente initiative concerne aussi bien les développeurs, éditeurs que les utilisateurs. Site : <http://www.microsoft.com/france/windows/windows7/compatibilite/default.aspx>

■ **Kapitec**, distributeur exclusif de Web Performance Load Tester, annonce l'arrivée de la version 3.6. Cette version intègre de nombreuses nouveautés : nouvelles analyses utilisateurs, nouveau module des cas de tests, reporting amélioré, nouvelles méthodes pour les tests de charge.

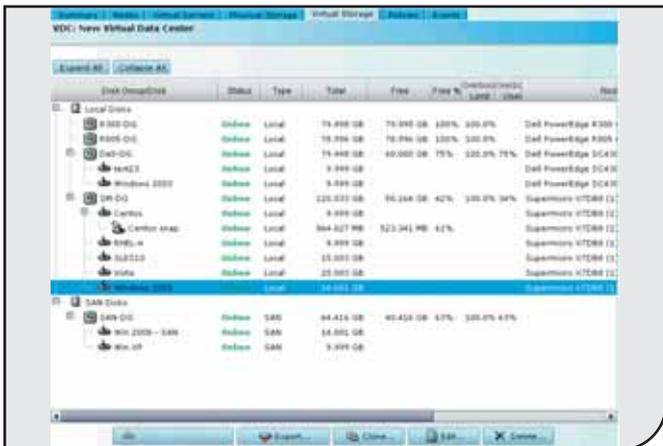
■ **Rasdaman 8** est désormais disponible. Il s'agit d'une extension destinée à PostgreSQL dédié au stockage maillé dans une base. Cela signifie qu'il traite des données de plusieurs dimensions en très gros volumes comme peut le faire PostGIS pour les données géospatiales. Des API sont disponibles en C++ et Java.



■ **Qt** souhaite mieux fédérer les développements de la communauté autour de Qt et de ses projets. Pour ce faire, l'éditeur vient d'ouvrir une plate-forme collaborative pour la communauté : Gitorious. Une excellente initiative ! site : <http://gitorious.org/>

■ **ABBYY**, fournisseur de technologies de reconnaissance de documents, d'extraction de données et de linguistique, annonce la disponibilité de FlexiCapture Engine 8.0, son nouveau SDK pour l'intégration des technologies d'extraction de données et de documents aux applications Windows.

■ **Quest Software** organise une Journée de la Performance IT, le 10 juin au NCI Com-Square Paris-La Défense. Dédiée à l'amélioration de la performance des systèmes d'information, la journée comprendra 3 salles et plus de 15 ateliers sur les thèmes favoris de Quest : gestion des environnements Windows, gestion des applications, gestion de bases de données et gestion des environnements virtuels. Site : <http://www.performanceit2009.com/>



■ **Après Sun**, Oracle a annoncé le rachat de Virtual Iron Software spécialisé dans l'administration des serveurs virtualisés et l'allocation dynamique des ressources. L'administration des environnements virtualisés devient l'élément stratégique de toute stratégie de virtualisation et la plus value des éditeurs alors que les couches de virtualisation (hyperviseur, player...) deviennent des commodités pour l'utilisateur.

Alter Way, intégrateur Open Source de référence, couvre les besoins de l'ensemble du système d'information avec ses quatre offres

CONSULTING

- Conseil en architecture
- Choix d'outils
- Industrialisation
- Best practices
- Audits

SOLUTIONS

SOLUTIONS APPLICATIVES

Applications Web, ECM, E-commerce, Intégration de données, Business intelligence, Développements spécifiques

SOLUTIONS D'INFRASTRUCTURE

Supervision, Serveur de messagerie, Serveurs d'infrastructure, Bases de données, Management IP, Filtrage et partage des connexions Internet, Sécurité

FORMATION

Un catalogue de plus de 60 formations

Des programmes de certifications : PHP, MySQL, PostgreSQL, Java

1200 personnes formées par an

Près d'une centaine d'intervenants sur tous les sujets dont quinze internes certifiés

Des partenariats forts : SUN, MySQL, Talend, Ingres...

HOSTING

Hébergement à haute disponibilité 24/7

Exploitation et infogérance

Solutions SaaS

Solutions de stockage

Solutions d'e-commerce

Des solutions techniques performantes et stables

TECHNOLOGIES

PHP / Python / Java / ECM
JasperSoft / Talend / Bases de données
Open-Xchange...



www.alterway.fr

OPEN SOURCE

Le casse-tête du multi-plate-forme

Aujourd'hui, la programmation est de moins en moins mono plate-forme (= mon application fonctionne sur un seul système) mais de plus en plus multi-plate-forme. Le site web en est le meilleur exemple. L'arrivée du cloud computing et des services en ligne (le SaaS) transforme radicalement la manière de concevoir et d'utiliser les logiciels. Demain, le développeur se souciera de moins en moins du système cible (sur quoi tourne mon application) pour se concentrer sur l'interface, les fonctions.

Dans ce dossier nous allons aborder différents thèmes car sous le vague terme multi-plate-forme, se cachent de nombreuses réalités. La cross compilation, ou compilation croisée, est un de ces aspects, peut être le plus intéressant à connaître et surtout à maîtriser. Il facilite grandement la création d'applications exécutables pour différents systèmes. Cette compilation multiple peut prendre différentes formes, comme nous le verrons. Elle peut être directement intégrée à l'environnement de développement ou alors passer par des outils tiers. Des environnements de développement sont conçus pour être multi-plates-formes et réaliser des logiciels pouvant fonctionner sur plusieurs systèmes.

Java est le langage le plus connu même si des limitations, des problèmes surgissent souvent. Des langages comme C++, les langages dynamiques et certains basic le peuvent aussi.

Avec l'interopérabilité omniprésente, le fait d'avoir un langage portable partout ou presque est devenu un réel enjeu. Si Windows demeure le système le plus utilisé, le paysage change rapidement et MacOS X et Linux prennent tous les ans des parts de marché.

Le multi-plate-forme apparaît crucial quand on parle de web, de web 2.0, de RIA. Et il est aussi source de cauchemars pour le développeur web. Nous verrons que cette problématique reste complexe dans une approche multi navigateur.

Dans les prochains numéros, nous continuerons à aborder cette épineuse question que l'on oublie trop souvent !

■ François Tonic

Multi-plate-forme : où, quand, comment, pourquoi ?

Le multi-plate-forme n'a jamais été un modèle très prisé des développeurs. Par contre le développeur web se trouve tous les jours, en principe, confronté à la variété des navigateurs et des technologies à supporter pour être le plus universel possible. Mais rien n'est jamais simple en informatique et vous verrez dans ce dossier que le multi-plate-forme demande rigueur et structure.

Comme dans tout développement, le multi-plate-forme ne s'improvise pas ! Il faut déjà connaître le contexte de l'application à développer, les technologies, langages envisagés, le type d'utilisateur, et le ou les systèmes cibles possibles.

Les langages et le problème du RIA

Qui dit multi-plate-forme, dit aussi langage de programmation. Dans un contexte web, privilégiez les standards reconnus : HTML, XHTML, CSS, REST, Atom, Javascript, Ajax. Cependant attention, sur HTML, évitez de trop vous focaliser sur HTML 5 tant que les navigateurs ne le supporteront pas entièrement et de façon optimale. Sur Ajax, le choix du framework ne sera pas non plus anodin. Dans tous les cas, une grosse phase de test sera nécessaire (voir plus loin dans ce dossier). Par contre on peut rencontrer des soucis sur un développement RIA (Rich Internet Application). Le plus universel reste Adobe Flash / Flex fonctionnant aussi sur Windows, MacOS X et Linux. Sa déclinaison desktop, AIR, fonctionne aussi sur les trois systèmes. Donc peu de souci à se faire. Par contre avec Silverlight, si Windows et MacOS X sont bien supportés, Linux reste en retard malgré l'arrivée du projet Moonlight. Et sur l'utilisation de JavaFX, le projet demeure trop immature et peu reconnu. XUL sera aussi une bonne solution mais limitée aux univers Mozilla. Sur le RIA, il n'existe pas de réponse universelle.

C++ (ANSI) est le langage le plus portable même s'il demande de la rigueur et une bonne compétence. Cependant, aujourd'hui, C++ est LE langage référence du multi-plate-forme (voir article sur Omake). C# commence lui aussi à se faire une

belle place, notamment grâce à la pile open source Mono qui permet de développer un peu partout avec le même langage (voir article Delphi Prism). Nous n'oublions pas Java qui fonctionne partout. Si la portabilité s'améliore constamment, attention tout de même aux disparités des JVM et aux différences de comportement selon les systèmes. Notamment sur tout ce qui est interface native et appels natifs. Reste l'incontournable basic. Et oui il demeure bien vivant ! Plusieurs environnements permettent de faire de la compilation multi cible à partir du même code basic. Le plus connu dans ce domaine est Real Basic de Real Software. Ensuite vous avez un grand nombre de langages, de langages dynamiques et de scripts faisant parfaitement du multi-plate-forme, souvent dans le domaine web et non desktop : PHP, Python, Ruby, Rails, etc.

Et les outils ?

L'outillage pour le multi-plate-forme est vaste. Encore une fois, tout va dépendre du projet envisagé. Aujourd'hui de nombreux outils, notamment open source, sont disponibles sur plusieurs plates-formes. Si on prend les classiques Eclipse et Netbeans par exemple mais aussi dans les outils Qt, Adobe, etc. En dehors de l'éditeur, de l'IDE, il ne faut pas omettre les outils de tests, l'intégration continue, etc. Et les choix sont nombreux : Maven, Ant, JUnit (ou autres), etc. On peut aussi opter pour des environnements totalement intégrés et multi-plates-formes, orientés données ou non tels que 4D, Omnis.

Tester, tester, tester, tester !

Pour réellement vérifier que son code fonctionne de la manière attendue sur les environnements cibles, il faut donc constamment réaliser des build et tester

sur toutes les cibles. Il ne faut surtout pas attendre la fin du développement pour le faire. Car en cas de bugs importants, la correction sera plus longue et plus difficile, surtout si la structure de l'application n'est pas réalisée en couches strictement distinctes, au moins séparer l'interface du code fonctionnel. Et plus un code sera structuré, modulaire, plus il sera aisé de corriger les imperfections de portage.

Les environnements de virtualisation apportent aujourd'hui une souplesse inégalée pour le développeur. Il peut faciliter créer des machines virtuelles spécifiques à chaque scénario d'utilisation.

■ François Tonic

Bonnes pratiques dès le départ

Sogeti met en avant 3 clés :

- ne pas se jeter directement sur le code, regarder les besoins, l'environnement sur lequel tournera l'application.
- Avoir un code lisible, modulaire, facile à maintenir
- Quand la compilation est terminée, le travail n'est pas terminé !
- A cela se rajoute bien entendu la séparation des couches (interface, code fonctionnel).

Pour les aider, Sogeti utilise un modèle de type MDA pour modéliser les couches et réaliser le découpage des applications. Les développeurs utilisent des langages et spécifications matures du web.

Développer multi-plate-forme en .Net avec Delphi PRISM

Il n'y a pas si longtemps, lorsqu'il était question de développement cross-platform, on pensait surtout au couple Windows/Linux. Aujourd'hui MacOS X —qui est un système UNIX— devient de plus en plus une cible de prédilection.

Petit problème cependant : offrez à un utilisateur Mac une application au look GTK ou WinForms, et voyez ses réactions ! L'effort d'intégration graphique à l'OS ne doit surtout pas être négligé. C'est à ce stade que Delphi PRISM entre en jeu. Grâce à Mono et Cocoa#, vous pouvez utiliser PRISM pour développer des applications Delphi.NET se comportant en tout point comme une application Cocoa (le framework fourni par MacOS). Cet article vous mettra sur les rails de tels développements, en vous présentant les bases de la coopération entre Delphi PRISM et MacOS X.

Prérequis

MacOS X 10.4 (ou supérieur)

- Les Apple Developer Tools : <http://developer.apple.com> permettent de disposer d'*Interface Builder*, l'éditeur d'interface graphique de MacOS X.
- Mono 1.2.6 ou plus récent : <http://www.mono-project.com>

Windows XP (minimum)

- Delphi PRISM
- Mono 1.2.6 ou plus récent

Pour des questions pratiques, il est préférable que MacOS et Windows puissent communiquer en réseau, afin de partager un répertoire de travail. Bien sûr, vous pouvez utiliser une solution de virtualisation comme VMWare ou Parallels pour exécuter Windows sur votre Mac, par exemple.

 Vous rencontrerez peut-être des problèmes de permissions entre MacOS et Windows, suivant la configuration de votre partage de fichiers. Assurez-vous que votre dossier de travail est accessible en écriture à votre compte Mac et au compte utilisé par Windows par le réseau (ou inversement). Si l'application MacOS générée (fichier .app, dans " bin/Debug " ou " bin/Release ") se termine immédiatement après son lancement, ajoutez-lui les droits d'écriture et d'exécution : `sudo chmod -R a+wx monapplication.app`. Vous devrez alors refaire cette opération à chaque généra-

tion. Si lors de la génération vous obtenez une erreur de la tâche MacPack disant que " le répertoire n'est pas vide ", supprimez à la main le contenu du répertoire " bin " de votre solution et recommencez.

Premier projet PRISM

Vous êtes maintenant prêt à créer votre première application MacOS X avec Delphi PRISM. Créez un nouveau projet Mono (Fichier > Nouveau > Projet) de type " Application Cocoa (Leopard)" ou " Application Cocoa (Tiger) ", suivant votre version de MacOS. Créez le répertoire de la solution dans votre répertoire de travail partagé entre Windows et MacOS. Plusieurs fichiers sont générés. Seuls deux d'entre eux retiendront ici notre intérêt :

- **Interface.nib** : il apparaît comme un répertoire contenant, entre autres, le fichier " designable.nib " dans l'explorateur de solution de Visual Studio. Il apparaîtra comme un fichier sous MacOS. Il s'agit du fichier que vous éditez avec *Interface Builder* pour créer votre interface utilisateur.
- **ApplicationController.pas** : il s'agit du contrôleur de votre application, que vous implémenterez avec Delphi PRISM et qui

dialoguera avec l'interface graphique par le biais d'actions —définies comme des méthodes et correspondant aux événements utilisateur— et d'outlets, points d'accès aux composants de votre interface, définis comme des champs de votre classe. Nous y reviendrons.

Design de l'interface graphique

Laissez de côté Delphi PRISM et, sous MacOS, rendez vous dans le répertoire de votre projet nouvellement créé. Ouvrez le fichier **Interface.nib** avec *Interface Builder*. Si vous ne connaissez pas encore le logiciel, vous serez peut-être dérouté au premier abord par ses nombreuses fenêtres. Voici une rapide présentation : [Fig.1]

Dans l'idée, cette fenêtre montre ce que contient votre fichier NIB. Pour faire un parallèle avec Delphi, les objets qui apparaissent ici pourraient être comparés aux composants non visibles que vous placerez dans un *DataModule*. Cette comparaison n'est qu'approximative, puisqu'apparaît également un objet pour chaque fenêtre de votre NIB. Le dernier objet, nommé " Appli-

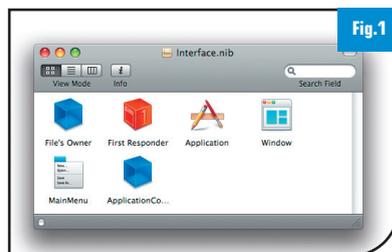


Figure 1 - Contenu du fichier NIB

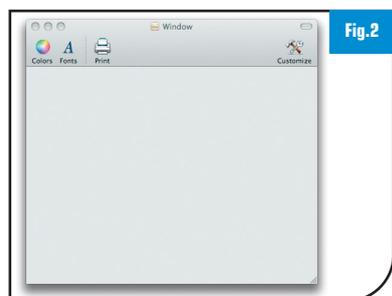


Figure 2 - Fenêtre par défaut

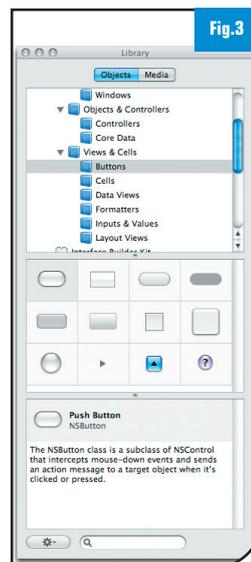


Figure 3 - Inspecteur d'objets

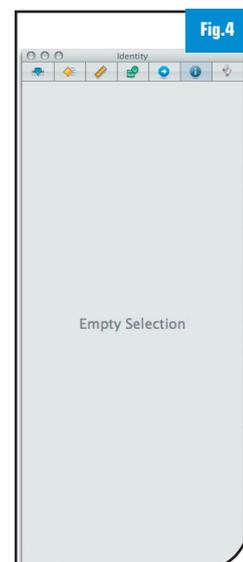


Figure 4 - Bibliothèque de contrôles



Microsoft®
Visual Studio® 2008

Montez en gamme à tarif réduit!

**jusqu'à
30%
de remise**



Vous possédez un outil de développement de la gamme Microsoft Visual Studio ?

4 promotions exceptionnelles

Si vous possédez Visual Studio Professionnel, Visual Studio avec MSDN ou encore Visual Studio Team System, vous avez la possibilité d'accéder au produit supérieur de la gamme à un prix préférentiel.



**Visual Studio
Professional**



**Visual Studio
avec MSDN**



**Visual Studio
Team System**



**Visual Studio
Team Suite**

Vous profitez ainsi à prix réduit (selon votre produit) :

- Des serveurs Microsoft
- Des systèmes d'exploitation Microsoft
- De logiciels de productivité : Microsoft Office, Project, Visio...
- D'outils de test : Montée en charge, fonctionnels, manuel...
- Logiciels de modélisation : diagrammes d'applications, de développement...
- Jusqu'à 4 incidents de support technique,
- Ressources techniques
- Toutes les mises à jour pendant votre abonnement : Visual Studio 10 et Windows 7 dès leur disponibilité

Exemple : Visual Studio 2008 avec MSDN Premium : 1137 € HT / an / par dev**

*Valable jusqu'au 26 juin 2009 - Tous les détails des promotions sur notre site internet.

** (abonnement 2 ans obligatoire - licence Open Business)

au lieu de 1395 € HT

COMSOFT direct

Bechtle's Software Specialist

**SOS
DEVELOPERS**

**Microsoft
GOLD CERTIFIED**

Partner

1^{er} revendeur en France en outils de développement Microsoft.

Contactez nos spécialistes MSDN au 04 97 21 58 65

msdn@comsoft.fr – www.sosdevelopers.com/msdn3.htm

cationController ", est un pont vers le fichier " ApplicationController.pas ". Nous y définirons les actions et outlets nécessaires au fonctionnement de notre interface. [Fig.2]

Cette fenêtre est tout simplement celle sur laquelle vous travaillerez, à l'instar des outils présents dans Delphi ou Visual Studio. Par défaut, elle comporte une simple barre d'outils contenant quelques boutons. Nous la modifierons sous peu en y déposant des contrôles depuis la bibliothèque.

Les deux fenêtres suivantes sont, respectivement, la bibliothèque de contrôles (visuels ou non) et de médias disponibles pour votre interface, et ce qui se rapproche de l'inspecteur d'objets. Cet inspecteur propose plusieurs vues permettant d'éditer les propriétés de l'objet sélectionné dans l'une des deux fenêtres précédentes. [Fig.3]

Pour notre exemple, nous allons modifier la fenêtre pour obtenir ceci : [Fig.5]

- Cliquez sur la barre d'outils incluse par défaut dans la fenêtre et supprimez-la.
- Redimensionnez la fenêtre à la taille voulue
- Dans la bibliothèque, trouvez le *label*. Il s'agit du premier composant dans Cocoa > Views & Cells > Inputs & Values. Déposez un label dans votre fenêtre grâce à un drag&drop et double-cliquez dessus pour en changer l'intitulé. Validez par " Entrée ". Vous remarquerez que des guides vous aident à placer correctement le composant dans son conteneur lorsque vous le déplacez.
- Ajoutez un *TextField* en dessous du label.
- Enfin, ajoutez un bouton. Tous les types de boutons se trouvent dans Views & Cells > Buttons. Les différents types de boutons sont parfois destinés à des utilisations diffé-



Figure 5 - Fenêtre finale

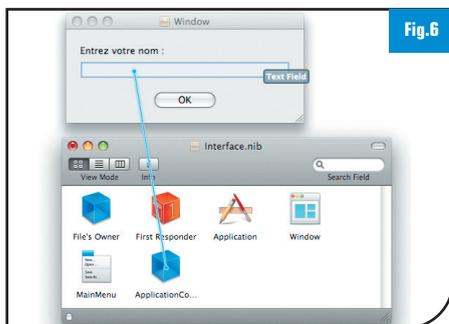


Figure 6 - Création d'un lien entre deux composants

rentes, parfois seulement à des design différents. Ici il s'agit d'un simple *Push Button*. Le but de l'application sera très basique : afficher une boîte de dialogue avec un message incluant le nom entré dans le *TextField* lorsque l'utilisateur clique sur le bouton.

Le comportement de votre application est implémenté dans votre objet *ApplicationController*, défini dans le fichier " ApplicationController.pas ". Afin d'interagir avec votre interface, la classe est en réalité déclarée de manière partielle dans un fichier " designable.pas ", lui-même inclus dans " interface.nib ". Ainsi, toute modification apportée dans *Interface Builder* à l'objet " ApplicationController " sera répercutée dans cette classe partielle, et vous pourrez coder en toute tranquillité dans " ApplicationController.pas " en profitant de ce pont vers votre interface.

Continuons justement notre chemin vers l'implémentation. Pour que la classe *ApplicationController* puisse agir comme nous le souhaitons, il faut lui indiquer deux choses :

- Un outlet vers le *TextField*, grâce auquel nous pourrions récupérer le texte entré par l'utilisateur.

- Une action, nommant la méthode à exécuter lorsque le bouton est cliqué.

Sélectionnez l'objet " ApplicationController " et, dans l'inspecteur, affichez l'onglet " identity " (l'avant-dernier onglet). Vous pouvez voir une section " Class Actions " et une section " Class Outlets ". Dans la première, ajoutez une action *onClick*: dont vous laisserez le type à *id*.

Dans la seconde, ajoutez un outlet que vous nommerez *edName* et qui doit avoir pour type *NSString*. Pour connaître le type d'un objet, sélectionnez-le et vérifiez le champ " Class " dans l'onglet " identity " de l'inspecteur. Reste maintenant à indiquer à la classe à qui (ou quoi) correspondent ces éléments dans l'interface. Pour cela, il faut créer des connexions entre les objets grâce à un drag&drop effectué avec un clic droit. Cliquez droit sur l'objet " ApplicationController " et en maintenant le bouton pressé, glissez le lien bleu qui s'affiche jusqu'au *TextField*. Lâchez alors le bouton : s'affiche une liste des outlets disponibles et de type tout à fait sérieuse, tout en conservant la possibilité d'un portage vers d'autres plates-formes grâce au framework .NET.

Effectuez la même opération en partant du bouton vers l'objet " ApplicationController ". Cette fois, une liste des actions disponibles s'affiche. Choisissez *onClick*: que vous venez

de créer. Le selector du bouton, qui est l'action exécutée lorsqu'il est cliqué, est maintenant dirigé vers la méthode *onClick*: que vous allez implémenter dans votre *ApplicationController*. [Fig.6]

Vous pouvez vérifier toutes ces connexions en examinant l'onglet " connections " de l'inspecteur (cinquième onglet) et en sélectionnant les différents objets affectés.

Implémentation de l'application

Tout est maintenant prêt pour implémenter l'application. Sauvegardez le fichier NIB et revenez à Delphi PRISM sous Windows. Afin de s'assurer que tout est bien synchronisé, effectuez un clic droit sur le fichier " designable.nib " (enfant de " Interface.nib ") dans l'explorateur de solution, et cliquez sur " Exécuter un outil personnalisé ". Cela régénère le fichier " designable.pas " avec les dernières modifications apportées au fichier NIB. Dans " ApplicationController.pas " ajoutez la déclaration de la méthode *onClick* en public de la classe *ApplicationController* :

```
public
    method onClick(sender: Object); partial;
end;
```

Puis implémentez la méthode :

```
method ApplicationController.onClick
(sender: Object);
begin
    Alert.AlertWithMessage('Bonjour !', 'OK',
        nil, nil, 'Bonjour à vous, ' + edName.
        Value + ' !').RunModal;
end;
```

Sauvegardez et générez l'application. Si tout se passe correctement, vous devriez avoir dans le dossier " bin/Debug " ou " bin/Release " un fichier " CocoaApplication1.app " (suivant le nom de votre projet bien sûr). Lancez l'application depuis MacOS : vous avez alors sous les yeux une véritable application MacOS X, s'intégrant parfaitement à l'OS. Vous avez maintenant entre les mains un outil qui vous permettra de cibler un OS en vogue tel que MacOS X de manière tout à fait sérieuse, tout en conservant la possibilité d'un portage vers d'autres plates-formes grâce au framework .NET.

■ Olivier Lance

Etudiant ingénieur SI

www.codegearfrance.com

La compilation croisée entre Linux et Windows

La compilation croisée est une opération de compilation qui, effectuée sur un système produit un exécutable ou une librairie pour un autre système. Voyons comment cela peut se passer entre Linux et Windows.

Compiler depuis un système A pour produire un exécutable, ou une librairie, pour un système B est souvent bien pratique. Supposons que l'on développe une application en C++ avec Qt ou wxWidgets en guise de toolkit graphique. Supposons que l'on travaille seulement avec les plates-formes Linux et Windows, mais le raisonnement est bien sûr valable avec un autre choix de plates-formes. C++ permet d'écrire du code source portable. Très bien. Qt ou wxWidgets, ou même d'autres toolkits permettent d'écrire du code portable. Toujours très bien. Mais compiler un projet sur chacune des plates-formes risque fort d'obliger à travailler avec des outils de développement différents, éventuellement chacun avec leurs spécificités et gérer de multiples projets. La compilation croisée est une réponse à ce problème. On écrit le code et on le compile sur une plate-forme seulement, en produisant des executables pour chaque plate-forme cible. Voyons comment pratiquer cela de Linux vers Windows et réciproquement, et déjouer quelques pièges.

1 DE LINUX VERS WINDOWS, LES OUTILS

Un compilateur croisé "qui va bien" est par exemple mingw32, généralement connu comme compilateur libre sous Windows, mais il existe aussi sous Linux. Vous pouvez l'obtenir à <http://www.libsdl.org/> le site de la librairie SDL, une librairie pour écrire des jeux portables entre Linux et Windows. La présence d'un compilateur croisée n'y est donc pas surprenante. En théorie on télécharge les sources du compilateur, puis on construit celui-ci "à sa main" à partir de script. Notre but d'aujourd'hui étant simplement la compilation croisée, nous téléchargerons plutôt des binaires tout prêts sur le même site, à <http://www.libsdl.org/extras/win32/cross/>. Téléchargez l'archive dont le nom contient linux-x86. Désarchivez dans le répertoire de votre choix. Par exemple sur ma machine: /home/fred/cross-tools. Notre compilateur croisé est un environnement complet dont les éléments sont répartis dans une arborescence. Un sous-répertoire *bin* pour les executables. Les noms de ceux-ci sont tous préfixés pour éviter des conflits avec les outils classiques. Ainsi g++ devient i386-mingw32-g++. Un sous-répertoire *lib*, un sous-répertoire *include* bien évidemment et aussi les sous-répertoires *info* et *man* qui contiennent les fichiers de documentations des mêmes noms. Pour une utilisation confortable, le mieux est d'intégrer le tout à votre environnement, en ajoutant quelques lignes à votre fichier .bashrc

```
export PATH=/home/fred/cross-tools/bin:$PATH
export MANPATH=/home/fred/cross-tools/man:$MANPATH
export INFOPATH=/home/fred/cross-tools/man:$INFOPATH
```

2 CONSTRUIRE UN EXÉCUTABLE

Voici un code d'une originalité folle :

```
// fichier hello.cpp
#include <iostream>
```

```
using namespace std;
int main(){
    cout << "Programmez!" << endl;
    return 0;
}
```

Sa compilation est non moins simple

```
i386-mingw32-g++ -o hello.exe hello.cpp
```

et produit un exécutable Windows. Nous ne devons pas omettre de mentionner l'extension .exe au fichier produit. Celui-ci est plutôt volumineux. La raison est que l'édition de liens avec le runtime a été effectuée statiquement par défaut, le but étant de produire un exécutable totalement autonome sur la plate-forme cible.

3 CONSTRUIRE ET UTILISER UNE LIBRAIRIE

Une grosse application est rarement monolithique, mais utilise des librairies. Librairies dont les formats sont différents selon les plates-formes. En outre nous voulons :

- 1) écrire un code qui compile aussi bien avec le compilateur de la plate-forme hôte qu'avec le compilateur croisé.
- 2) être capable de faire une édition de liens statique avec la librairie pour produire une application autonome sur la plate-forme cible.

Le premier point se règle simplement en traitant les particularités des librairies partagées ou dll sous Windows. Celles-ci doivent comporter un point d'entrée bien défini et toutes les autres fonctions doivent être exportées explicitement. On obtient cela avec une directive de compilation conditionnelle :

```
// fichier madll.cpp
#ifdef __WIN32__
#include <windows.h>
#define EXPORT extern "C" __declspec (dllexport)

// Point d'entrée dll Windows.
int WINAPI DllMain(HINSTANCE hInstance, DWORD fdwReason, PVOID
    pvReserved) {
    return TRUE;
}
#else
#define EXPORT
#define CALLBACK
#endif
// Une fonction de notre librairie
EXPORT int CALLBACK fois2(int valeur) {
    return valeur*2;
}
```

Pour Linux on compile simplement par

```
g++ -c madll.cpp
```

Et pour Windows

```
i386-mingw32-g++ -D__WIN32__ -c madll.cpp
```

4 EDITION DE LIENS STATIQUE

Abordons maintenant le deuxième point. Soit un code qui utilise notre librairie :

```
#include <iostream>
#include <windows.h>
extern "C" int CALLBACK fois2(int valeur);
using namespace std;

int main() {
    cout << fois2(3) << endl;
    return 0;
}
```

Nous voulons à partir de ce code produire une application autonome. Pour cela nous devons travailler un peu notre librairie qui pour l'instant n'est qu'un fichier objet à l'issue de la compilation faite plus haut. Commençons par en faire une archive :

```
ar -rs libmadll.a madll.o
```

A ce point, si nous tentions une édition de liens pour générer la librairie nous aurions un message d'erreur :

```
./libmadll.a: could not read symbols: Archive has no index;
run ranlib to add one
```

Nous devons donc utiliser *ranlib* comme nous dit le message, mais attention ici, piège classique ce n'est pas le *ranlib* du système hôte que nous devons utiliser mais celui du compilateur croisé. Nous faisons donc :

```
i386-mingw32-ranlib libmadll.a
```

et maintenant nous pouvons faire l'édition de liens de la librairie proprement dite :

```
i386-mingw32-ld -dll -kill-at -o madll.dll madll.o
```

La directive *-dll* parle d'elle-même. Quant à *-kill-at* nous y reviendrons plus loin. Il est maintenant possible de construire notre application autonome en compilant *use-madll.cpp* :

```
i386-mingw32-g++ -L./ -o use-madll.exe use-madll.cpp -lmingw32 -lmadll
```

Il y a ci-dessus une petite subtilité. Pourquoi ajoutons-nous la librairie *mingw32* qui est liée automatiquement de toute façon ? Ici nous n'avons pas du tout besoin de la faire en fait. Mais c'est une bonne habitude à prendre que de **forcer** la lecture de cette librairie en **premier**. Sans trop entrer dans les détails dans ce court article, disons qu'il est possible, dans de grosses applications, que vous rencontriez un message d'erreur agaçant disant que pour une application GUI, la fonction *WinMain* n'est pas définie, alors qu'elle l'est pourtant bel et bien dans votre code. Casse-tête courant qui en a éterné plus d'un :) Une particularité de l'éditeur de liens *ld* est de chercher un symbole requis par une librairie dans les librairies qui la suivent dans l'ordre d'inclusion. Or il se trouve que *mingw32* a besoin de

WinMain, et que par défaut *mingw32* est incluse automatiquement en dernier et ne peut profiter de votre définition de *WinMain*. Voilà pourquoi nous recommandons, avec la ligne de compilation ci-dessus, de toujours forcer le bon ordre d'inclusion des librairies.

5 EDITION DE LIENS DYNAMIQUE

Sous Windows une librairie peut être chargée à la demande et il en va de même pour les adresses des fonctions exportées. Voici un code, écrit et compilé sous Windows qui fait cela :

```
#include <iostream>
using namespace std;
#include <windows.h>
const LPWSTR la_librairie = L"madll.dll";

int main(int argc, char* argv[]) {
    HMODULE hmodule = ::LoadLibrary(la_librairie);
    if(!hmodule) {
        cout << "Impossible charger librairie" << endl;
        exit(1);
    }

    typedef int (CALLBACK *type_pf)(int);
    // sans -kill-at
    // type_pf pf = (type_pf)::GetProcAddress(hmodule, "fois2@4");
    type_pf pf = (type_pf)::GetProcAddress(hmodule, "fois2");
    if(!pf) {
        cout << "Impossible charger fonction" << endl;
        exit(1);
    }
    cout << pf(3) << endl;
    FreeLibrary(hmodule);
    return 0;
}
```

Avec l'API Windows *GetProcAddress* nous obtenons un pointeur sur la fonction à partir du nom de celle-ci. C'est le moment de constater l'effet pernicieux de l'omission de la directive *-kill-at* dont nous avons parlé plus haut. Sans cette directive le nom de notre fonction est décoré et devient "fois2@4" comme on le voit dans la capture ci-contre, obtenue avec l'utilitaire *Dependency Walker* de Microsoft. Bien sûr si nous utilisons ce nom (cf. la ligne de code en commentaire ci-dessus) cela fonctionne, mais ce n'est pas l'esprit de la programmation Windows. La bonne manière est d'employer *-kill-at* qui comme son nom l'indique supprime le symbole at (@)

6 DE WINDOWS VERS LINUX

Le plus simple est d'installer un environnement *Cygwin* (<http://www.cygwin.com/>) sous Windows, puis d'installer le compilateur croisé *mingw32*, encore lui :) Vous le trouverez à http://sourceforge.net/project/showfiles.php?group_id=135860&package_id=153563. Son utilisation ne pose aucune difficulté.

■ Frédéric Mazué
fmazue@programmez.com



Infragistics®



Introduisant NetAdvantage pour .NET 2009 Volume 1

Des composants interface utilisateurs supérieurs pour Windows Forms, WPF, ASP.NET et Silverlight

Pour de plus amples infos : infragistics.com

 **NetAdvantage®**
.NET ASP.NET, WinForms, WPF, Silverlight

Four Platforms. One Package.

 **N°Vert** 0800 667 307

grids

tree

menus

navigation

charts

& more!

Qmake, le système de build multi-plate-forme qu'il vous faut.

Qmake est l'outil de build fourni avec le framework Qt. La puissance de ce dernier repose, entre autres, sur la grande portabilité du code Qt. Cette portabilité nécessite absolument un système de construction totalement multi-plate-forme, c'est le rôle de Qmake. Voyons ce qui en fait un outil attrayant.

Qmake n'est pas, à proprement parler un système de build. C'est un outil qui permet de simplifier considérablement le processus de création des fichiers de build. Typiquement, Qmake va être utilisé pour générer les fichiers Makefile nécessaires à la construction d'un projet. Actuellement, Qmake permet de générer des Makefiles pour toutes les plates-formes supportées par Qt. Il est aussi capable de générer les fichiers projet pour Microsoft Visual studio ou Xcode permettant ainsi d'utiliser les outils propres à certaines plates-formes. L'intérêt de cet outil réside dans son utilisation pour des projets écrits ou non en Qt. Pour les projets Qt, Qmake génère automatiquement les cibles pour Moc (Meta Object Compiler) et Uic (User Interface Compiler).

Les bases...

Commençons par les bases, Qmake se repose sur un fichier texte (UTF8 pour bien faire) dit "fichier projet". Ce fichier est une suite de variables ou de directives qui seront utilisées pour générer les instructions de compilation. La façon la plus simple d'obtenir un squelette de fichier projet est de faire appel à Qmake dans le répertoire racine de votre projet :

```
qmake -project
```

Prenons un premier exemple simpliste. Créez un répertoire exemple1. Dans celui-ci, créez un répertoire src.

```
mkdir -p exemple1/src
```

Créez dans le répertoire src, un fichier main.cpp contenant le " programme " suivant :

```
#include <iostream>

int main(){
    std::cout << "HelloWorld\n" ;
    return 0;
}
```

Maintenant, dans le répertoire exemple1, lancez " qmake -project ". Celui-ci va créer un fichier appelé exemple1.pro contenant ceci :

```
TEMPLATE = app
TARGET =
DEPENDPATH += . src
INCLUDEPATH += .
# Input
SOURCES += src/main.cpp
```

Nous reviendrons sur ce fichier, pour le moment transformez-le en fichier de build en tapant " qmake exemple1.pro ". Cela aura pour

effet de générer le Makefile, lancez-le en tapant " make ", vous verrez alors votre programme se construire.

Vous constaterez aussi que votre programme se retrouve lié à Qt (et à toutes ses dépendances)... cela fait beaucoup pour notre " HelloWorld "... D'autant plus que nous n'utilisons pas Qt ! C'est donc le moment d'expliquer à Qmake que nous ne voulons pas qu'il lie notre application à Qt. Revenons donc sur le fichier projet. Nous constatons d'abord qu'il ressemble énormément à un fichier de configuration et que chaque ligne présente un schéma identique :

```
VARIABLE = VALEUR
```

D'autres types d'instructions existent mais la majeure partie d'un fichier projet est constituée de déclarations de variables. La première disponible est la variable TEMPLATE, celle-ci contient le template qui sera utilisé pour générer le projet et peut prendre les valeurs suivantes :

- app : crée un Makefile pour générer une application ;
- lib : crée un Makefile pour générer une librairie ;
- subdirs : crée un Makefile par répertoire déclaré dans la variable SUBDIRS ;
- vcapp : spécifique à Windows, crée un fichier de projet Visual studio pour une application ;
- vclib : spécifique à Windows, crée un fichier de projet Visual studio pour une librairie.

Dans notre cas nous utilisons le template app. La variable suivante est TARGET, qui est, par défaut, vide (la cible générée prend alors le nom du répertoire). Viennent ensuite :

- DEPENDPATH : la liste des répertoires qui vont être scrutés pour résoudre les dépendances ;
- INCLUDEPATH : la liste des répertoires contenant les fichiers d'en-têtes ;
- SOURCES : la liste des fichiers sources du projet ;
- si nous avons des fichiers d'en-têtes, ils seraient listés dans la variable HEADERS.

Lors de la construction précédente du binaire vous avez pu constater que celui-ci était compilé et lié contre Qt alors que nous ne l'utilisons pas. Ceci n'est pas anodin et un " ldd exemple1 " vous fera constater l'étendue des dégâts ! Pour remédier à ce problème rien de plus simple : il suffit d'ajouter la définition suivante à notre fichier projet :

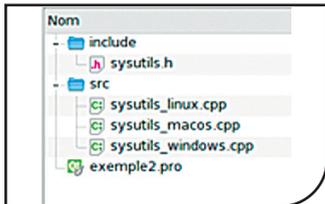
```
CONFIG -= qt
```

Relancez la compilation. Le Makefile est automatiquement régénéré (Qmake prend soin de mettre en place la règle adéquate). Constatez aussi que, cette fois, aucune librairie superflue n'est incluse, et ldd vous dit merci ! Qu'avons-nous fait avec la variable CONFIG ? Nous avons simplement demandé à Qmake de ne pas générer les règles qui lient Qt à notre application.

...pour s'amuser !

Attelons-nous maintenant à construire un projet multi-plate-forme avec du code dit " platform specific ". Nous allons, pour cela, construire une librairie partagée d'utilitaires systèmes. Cette librairie contient un ensemble de fonctions telles que `forceRemoveDirectory(const QString &)`, par exemple, qui font appel aux utilitaires propres à chaque système (`del` ou `rm` dans le cas présent).

Notre librairie va compiler contre QtCore car nous souhaitons bénéficier des types de Qt (entre autres), mais nous ne souhaitons pas compiler contre la partie GUI, nous verrons comment faire. Les



librairies partagées sont évidemment spécifiques à une plate-forme : `share object (.so)` sous GNU/Linux, `DLLs` sous Windows et `dylib` sous Mac OS. Il va falloir régler ce problème qui pourrait être épineux si nous ne disposions pas de Qmake.

Dans un premier temps voici l'arborescence de notre projet figure ci-contre. Tout d'abord, générez le squelette de base du fichier projet en tapant :

```
qmake -project -t lib
```

Dans un premier temps, voyons comment gérer la compilation des sources spécifiques à chaque OS. Qmake met à notre disposition un système de " scopes " très pratique. Les scopes sont des conditions qui permettent de prendre en compte les spécificités de chaque situation. Une des utilisations vraiment intéressante des scopes est la configuration du build en fonction de l'OS. Ainsi il suffit, pour ne compiler que le fichier source relatif à un OS, d'ajouter les définitions suivantes au fichier projet :

```
win32 {
    SOURCES += src/sysutils_windows.cpp
}
macx {
    SOURCES += src/sysutils_macos.cpp
}
unix{
    SOURCES += src/sysutils_linux.cpp
}
```

Ce squelette nous permet aussi d'ajouter, pour chaque système d'exploitation, l'option adaptée aux librairies partagées. Il suffira de rajouter : `CONFIG += shared` pour les Linux/Unices, `CONFIG += dll` pour Windows et `CONFIG += dylib lib_bundle` pour Mac OS X. Dans le dernier cas, l'option supplémentaire permet de créer le bundle particulier aux librairies de Mac OS X.

Pour utiliser les types de Qt, nous devons utiliser QtCore. En revanche, comme nous désirons nous passer de QtGui, il faut le préciser à Qmake (ces 2 modules étant inclus par défaut) :

```
QT -= gui
```

La variable QT permet de configurer les modules de Qt utilisés par notre projet. La déclaration par défaut (lorsque rien n'est spécifié) est :

```
QT = core gui
```

Donc quand nous écrivons :

```
QT -= gui
```

C'est équivalent à :

```
QT = core
```

Les opérateurs `+=` et `-=` se comportant comme nous l'attendons en concaténant et en supprimant une valeur de la liste. Le signe `=` représentant, bien sûr, l'affectation.

Les scopes de Qmake deviennent indispensables une fois couplés aux possibilités des objets qui peuvent être déclarés dans le fichier projet. Un objet est une variable qui contient plusieurs membres. Certaines variables prédéfinies du fichier projet font une bonne utilisation de ces objets si les membres requis sont remplis. Prenons l'exemple de la variable prédéfinie `INSTALLS`. Celle-ci prend en argument une liste d'objets contenant au minimum le chemin d'installation. La forme la plus simple est donc la suivante :

```
target.path = /usr/lib
INSTALLS = target
```

Ici, nous spécifions que les cibles produites lors de la construction doivent être installées dans le répertoire `/usr/lib`. Il peut arriver que nous ayons plus que les cibles de construction à installer. Dans le cas présent nous voulons installer aussi le fichier d'en-tête. Pour cela, un nouvel objet est nécessaire :

```
# Les fichiers concernés par cet objet sont les .h du répertoire
include/
includes.files = include/*.h
# ceux-ci seront installés dans le répertoire /usr/include
includes.path = /usr/include
Une fois ceci fait, il est nécessaire d'ajouter l'objet includes
à la variable INSTALLS :
INSTALLS += includes
```

Les objets de la variable `INSTALLS` peuvent posséder un autre membre : `extra`. Celui-ci permet d'exécuter un certains nombre de commandes lors du build. Par exemple, si nous désirons installer les sources compressées de notre librairie :

```
sources.files = src/sysutils_linux.cpp.gz
sources.path = /usr/src/example2
sources.extra = gzip -c src/sysutils_linux.cpp > src/sysutils
_linux.cpp.gz
INSTALLS += sources
```

Ces possibilités sont, bien entendu, à mettre en relation avec les scopes conditionnels concernant les systèmes d'exploitation. Pour terminer ce rapide tour d'horizon des possibilités de Qmake, ajoutons qu'il est possible de déclarer des liaisons avec d'autres librairies si celles-ci supportent `pkgconfig`, en utilisant la syntaxe suivante :

```
CONFIG += link_pkgconfig
PKGCONFIG += ogg dbus-1
```

Nous n'utilisons pas cette possibilité dans notre exemple.

J'espère que cet article d'introduction vous aura convaincu que la construction multi-plate-forme n'est pas l'enfer fréquemment décrit, du moment que l'on choisit les bons outils...

■ Arnaud Dupuis
Uperto/Devoteam

Le multi navigateur : le guide de survie du développeur web

Autant le dire tout de suite : il n'existe aucune méthode miracle pour faire fonctionner un site ou une application web sur les principaux navigateurs du marché. La quantité de problèmes engendrés par le seul Internet Explorer 6 est d'ailleurs effrayante... mais elle est aussi incontournable quand on sait que ce navigateur représente encore 18% des utilisateurs (et plus encore en contexte d'entreprise). La démarche et les quelques astuces qui suivent vous permettront donc d'être mieux armés face à ces petites galères quotidiennes.



La fonction d'inspection de Firebug en action

Munissez-vous d'abord d'un bon éditeur HTML : vous éviterez ainsi quelques erreurs silencieuses sous certains navigateurs et très gênantes ailleurs. Pour bien visualiser ce que vous codez, Firefox et son extension Firebug s'avèrent indispensables.

Les fonctions d'inspection et d'édition en ligne de Firebug vous permettront de mieux comprendre ce qui se passe dans votre page et faciliteront considérablement votre travail de débogage – y compris en dehors de Firefox.

Il vous faut aussi de quoi tester régulièrement vos pages web sur les trois navigateurs les plus utilisés par les internautes : IE 7, Firefox 3 et IE 6.

C'est là que cela se complique : si vous ne travaillez qu'avec un seul poste de travail, il vous faudra mettre en place une ou plusieurs machines virtuelles pour faire vos tests à la fois sur les versions 6 et 7 d'IE qui ne peuvent fonctionner sur le même système. Une autre option est l'utilisation de versions émulées d'Internet Explorer, mais elle est loin d'être parfaite. Notons enfin qu'IE 8 permet de se passer d'IE 7 grâce à son mode de compatibilité très pratique et apporte aussi quelques bons outils de débogage. La bonne nouvelle, c'est que les autres navigateurs vous poseront relativement peu de problèmes une fois la compatibilité IE6/IE7/Firefox assurée.

Rappelez-vous tout de même que quelques tests sur Safari (7% des visiteurs) doivent être effectués, au moins en fin de projet.

Avoir la bonne méthode

Même si un code conforme aux standards et propre ne garantit pas en soi la compatibilité entre navigateurs, vous imposer cette rigueur vous facilitera les choses. Commencez donc par réaliser votre squelette XHTML, sans style ni javascript, mettant en place la structure logique et sémantique de la page. Positionnez ensuite vos blocs à l'aide des différentes propriétés CSS en séparant scrupuleusement les responsabilités : toute la mise en forme doit se trouver dans des fichiers CSS (pas de style " inline " directement dans le HTML). C'est à cette étape que l'essentiel des problèmes va se poser, le positionnement sous IE suivant des règles assez différentes de la norme (voir les astuces plus bas à ce sujet) : contrôlez intensément les différences IE6/IE7/Firefox à cette étape.

Occupez-vous ensuite de la décoration et du javascript. Un site qui utilise beaucoup de javascript peut poser de gros problèmes de compatibilité, mais l'utilisation adéquate d'une bibliothèque JS de référence (Prototype, JQuery, Dojo, Mootools) vous en évitera l'essentiel.

Les astuces à connaître

Un bon doctype

Mettez toujours un doctype valide au début de vos pages et n'écrivez surtout rien avant (même pas le prologue xml pourtant autorisé par XHTML), dans le cas contraire IE passe en mode de rendu " quirks " qui rend l'affichage tout simplement absurde.

Redéfinir le style des balises par défaut

Les balises ne sont pas interprétées par défaut avec le même style suivant le navigateur. Forcer le style de toutes les balises (en particulier *border*, *margin* et *padding*) vous économisera quelques migraines.

Utiliser les commentaires conditionnels pour les styles spécifiques à IE

Il y a de très fortes chances que vous deviez utiliser les styles spécifiques à telle ou telle version d'IE pour fixer votre mise en page. Les commentaires conditionnels sont faits pour ça :

```
<!--[if IE]>
    <link type="text/css" rel="stylesheet" href="ie.css" />
<![endif]-->
```

La transparence des PNG

La transparence des PNG n'est pas gérée nativement par IE6, mais plusieurs hacks permettent de contourner cette limitation. Par exemple : www.twinhelix.com/css/iepngfix

Les bugs de positionnement IE6

IE6 comporte de nombreux petits bugs (ou différences avec les standards) dans l'interprétation du positionnement des blocs, en particulier avec les flottants. Un excellent site en fait le recensement et fournit les solutions adéquates : www.positioniseverything.net/explorer.html

Et pour finir, rappelez-vous les trois commandements de l'intégrateur web : tester, tester, tester !

■ Jean-Baptiste Boisseau
Eutech SSL

Développez 10 fois plus vite

WEBDEV

Ajax en 1 clic

WEBDEV 14 est l'environnement de développement professionnel qui permet de développer jusqu'à **10 fois plus vite** tous les types de sites et d'applications reliés aux données de votre entreprise.

Le **WEB 2.0** est facile: l'activation d'**AJAX** dans vos sites s'effectue naturellement; un simple clic dans l'éditeur de code indique que le code à exécuter est de type «**Ajax**». **WEBDEV 14** est certainement le seul environnement au monde à proposer autant de souplesse et de puissance.



Sous l'éditeur de code de **WEBDEV 14**: un clic et le traitement programmé devient «**Ajax**»

WEBDEV 14 gère le cycle complet de développement et d'administration: langage L5G, générateur de code **PHP**, débogueur, Webservices, gestionnaire de sources, installateur, base de données SQL intégrée et lien avec toutes les bases du marché, composants, éditeur d'états PDF et code-barres, règles métier, dossier, outils de déploiement et d'administration...: tout est inclus, en français.

Vous aussi, réalisez vos sites WEB 2.0 10 fois plus vite... avec WEBDEV 14.

UN OCEAN DE WEB 2.0

VOTRE CODE EST MULTI-PLATEFORMES:

Windows, .Net, Java, PHP, J2EE, XML, Internet, Ajax, Pocket PC, SmartPhone, Client riche ...



DEMANDEZ LE DOSSIER GRATUIT

252 pages + DVD + 112 Témoignages.
Tél: **04.67.032.032** ou **01.48.01.48.88**
info@pcsoft.fr

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr



Développeur AGILE

Pour retrouver le goût du développement

Les méthodes agiles sont des méthodologies de travail et d'organisation aidant à être plus efficace dans le développement et l'organisation d'un projet informatique. Aujourd'hui, l'agilité n'est pas un mot en l'air, tout particulièrement dans les SSII, et chez les équipes de développement. Deux méthodes, complémentaires, se détachent assez nettement : Scrum et le eXtreme Programming. L'une concerne plutôt l'organisation des équipes et la manière de mener un projet, l'autre se concentre sur le développement des fonctionnalités. Combinées, cela donne un cocktail détonnant ! De nombreuses équipes, de

petites tailles, adoptent et adaptent ces méthodes et clairement, cela redonne l'envie de développer car le développeur retrouve une place de choix dans l'organisation. Il n'est plus un simple exécutant, un " pisseur de code " comme on a souvent cherché à le limiter. Mais cela s'accompagne aussi d'un développement moins solitaire et avec de nouvelles responsabilités. Pour un développeur habitué à coder dans son coin, à avoir entre les mains des spécifications, un cahier des charges figés dans le marbre, le couple Scrum / XP déroute. Car la visibilité sur le projet est à court terme, avec des itérations courtes et des développements de petites tâches où savoir communiquer, travailler, échanger avec le reste de l'équipe est nécessaire. Le développeur doit aussi s'habituer à l'art délicat de l'estimation. C'est-à-dire que c'est à lui d'évaluer la difficulté de la fonction à développer et surtout de donner un temps de développement. Ces contraintes et libertés obligent à avoir des équipes relativement homogènes et à encadrer les nouveaux venus. Une bonne maîtrise des technologies est également indispensable.

Le développeur agile est une réalité même si cela se limite essentiellement aux développements en équipe car pour le développeur indépendant ou occasionnel, l'agilité aura peu d'influence, sauf peut être sur son organisation et la manière de découper un développement.

Dans ce dossier nous allons explorer les méandres de Scrum, de XP et voir comment se déroule une journée type d'un développeur agile, sans oublier les meilleurs conseils pour bien démarrer dans l'agilité.

■ François Tonic



Les méthodes agiles pour mieux développer

Le "Chaos Report" du Standish Group publié en 1994 met en évidence que 31% des projets informatiques sont arrêtés en cours de route, 52% n'aboutissent qu'au prix d'un important dépassement des délais et du budget (189%) et en offrant moins de fonctionnalités qu'il n'en était demandé. Seuls 16% des projets peuvent être considérés comme des succès. Depuis 1994 et malgré une volonté farouche de progresser dans ces domaines, force est de constater que nous (développeurs) sommes toujours confrontés aux mêmes enjeux et challenges.

Seuls 16% des projets peuvent être considérés comme des succès. Depuis 1994 et malgré une volonté farouche de progresser dans ces domaines, force est de constater que nous (développeurs) sommes toujours confrontés aux mêmes enjeux et challenges.

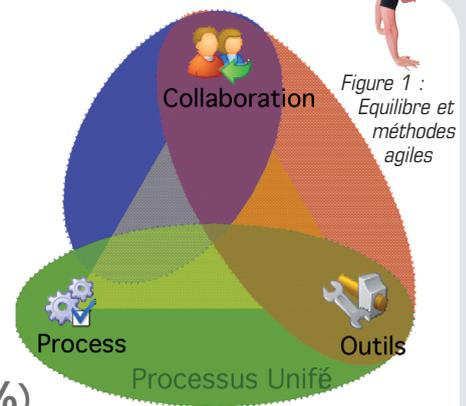


Figure 1 : Equilibre et méthodes agiles

A lors... implication des utilisateurs, soutien de la direction, exigences clairement définies, planning bien adapté, jalons plus fins, compétences des équipes et leur engagement, vision et objectifs clairs, sont des facteurs clés de succès connus et partagés d'un projet informatique et pourtant... Aujourd'hui, ils ne peuvent être appliqués qu'en prenant en compte le meilleur des méthodes "Agiles" pour appréhender toutes les dimensions d'un projet de développement. [Fig.1]

Comment choisir la bonne méthodologie et les impacts / conséquences de celle-ci

La dimension organisationnelle est celle qui est très, trop ?, souvent mise en avant pour aborder les méthodes agiles dans le développement. Parlons tout d'abord process, je sais c'est iconoclaste, mais le choix et la bonne utilisation d'un langage de modélisation et de spécifications basé sur UML et centré sur l'architecture permettent une définition et une gestion rigoureuse des cas d'utilisation et exigences. Le critère de la force est alors illustré.

Poursuivons par la dimension collaboration, la méthode Scrum devient incontournable en focalisant l'ensemble des équipes MOA/MOE sur la poursuite des objectifs, avec en premier le respect du délai pour les fonc-

ctions prioritaires et essentielles pour l'utilisateur final. Elle incarne les dimensions coordination et puissance ; la référence au rugby n'est pas innocente... Enfin, la mise en oeuvre des pratiques d'eXtreme Programming adresse complètement les notions de développement piloté par les tests, intégration continue, règles et normes de codage avec mesure de la qualité en continu. Elle symbolise les réflexes à acquérir ...

Quelles sont donc leur condition de mise en oeuvre ? L'organisation "Unité de temps, unité de lieu et unité d'action" chère aux écrivains classiques est elle l'unique solution ? La caractéristique d'une équipe (d'une bonne...) au delà de la valeur individuelle de chacun se traduit par les valeurs essentielles partagées par ses membres : la communication, le retour d'information, la simplicité et le respect. Ces valeurs ont toutes pour objectifs d'améliorer l'équipe et de développer le bon produit.

L'espace partagé est un facteur important, il peut être physique ou virtuel suivant la répartition des activités entre différentes localisations, la plate-forme collaborative prend tout son sens en termes de référentiels techniques, données, normes...

Dans ces conditions, le mode de production distribué et/ou centralisé est alors complètement pris en compte et organisé ; la réalisation d'un produit sur des lieux géographiques

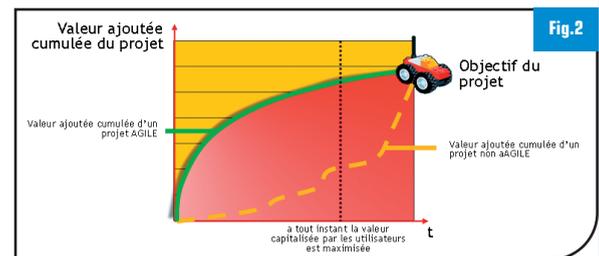


Figure 2 : Valeur ajoutée du produit

différents n'est plus un obstacle. Les impacts sur le mode de développement du produit sont par ailleurs multiples. [Fig.2]

La relation MOA/MOE est modifiée pour produire la plus grande valeur métier au plus tôt. Les choix et arbitrages sont partagés au début et tout au long du projet pour absorber les perturbations extérieures (difficultés techniques, évolutions du besoin...)



Figure 3 : la communication et la transparence

et apporter ainsi aux utilisateurs la garantie de disposer des fonctionnalités indispensables. [Fig.3]

Pour l'équipe de développement, chacun, même si il n'est pas directement impliqué, doit être à tout moment informé des activités en cours. De ce

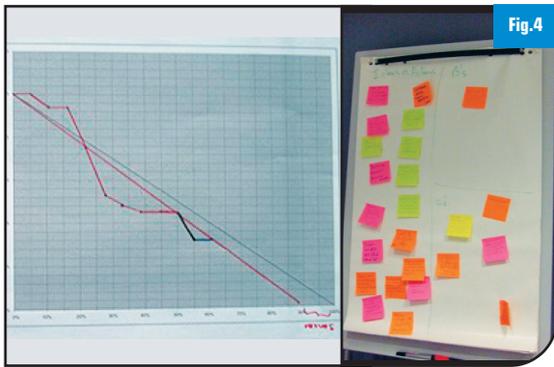


Figure 4 : ce qui est important est et reste visible

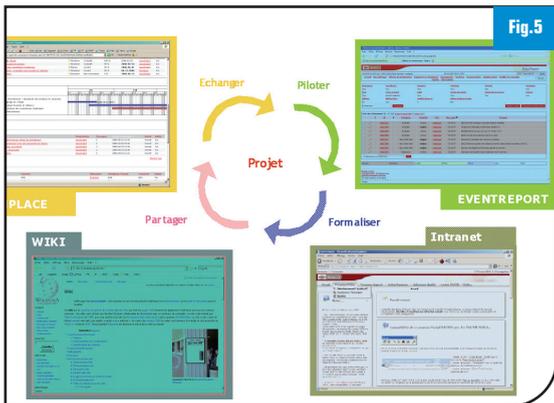


Figure 5 : Plate-forme collaborative en support

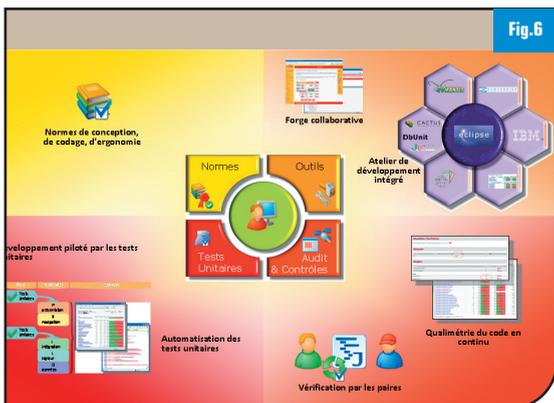


Figure 6 : Outillage pour la production (ex environnement Java)

fait, tout intervenant peut contribuer à la résolution d'un problème naissant et bénéficie d'une vision d'ensemble. Le Chef de projet joue un rôle essentiel. Il doit adapter son management à l'esprit et aux pratiques de l'équipe afin de constituer un environnement propice à la " fusion " des développeurs et à la libre expression de leur savoir-faire. Il s'agit alors plus de leadership que de relation hiérarchique c'est à dire plus d'animation, de facilitation et de coordination que de contrôle. Cependant, ce n'est pas non plus la porte ouverte à l'auto gestion des équipes de développement

mais bien le maintien dans la bonne direction de tous les intervenants et cette mission là est éminemment importante.

Pour assurer la visibilité du projet : une règle simple : " ce qui est important pour le projet est et reste visible ". L'objectif est d'être en mesure de présenter son avancement à tout moment en se basant sur des informations, pertinentes et à jour, que ce soit pour le produit, le processus de développement, l'architecture, les difficultés, les bonnes idées, les axes d'amélioration ou les tâches à réaliser. [Fig.4]

Pour ce dernier point, trois horizons sont définis: la journée, la semaine, le mois. Chaque matin, une courte réunion de l'équipe fixe les objectifs du jour, le reste à faire (en heures) d'un sprint, d'une livraison, est réévalué tous les soirs au vu des travaux effectués, et partagé par tous. Les développements sont testés au fil de l'eau par une équipe distincte dite d'assemblage sur la production de la semaine. La correction de toutes les anomalies détectées est inscrite dans le plan de développement général pour la planification mensuelle.

Comment ensuite l'utiliser dans son développement, les outils à mettre en oeuvre

L'outillage doit faire l'objet d'une attention particulière. A chaque phase du cycle de vie du projet correspond un outil ou une suite d'outils spécifique. Pour appréhender la dimension collaborative, le partage des informations au sein de l'équipe de développement peut être réalisé manuellement. Cependant, la complexité toujours plus grande des projets implique la mise en œuvre d'un portail qui permet de couvrir les quatre fonctions suivantes : échanger , partager, formaliser et piloter. [Fig.5] Pour la conception, il s'agit alors de s'appuyer sur des outils de modélisation et de traçabilité des exigences. Les scénarios de tests techniques et utilisateurs sont spécifiés et viennent enrichir un référentiel de manière automatique et réutilisable.

Enfin il convient de normaliser et contrôler la production technique, au fil de l'eau et prévenir les erreurs. [Fig.6]

Une forge logicielle est nécessaire pour instrumenter les normes de développement internes et/ou clients. Des ateliers de développement intégré facilitent la génération et le contrôle du code. Ces ateliers sont interfacés avec des outils de gestion de configuration au sens ITIL du terme et d'un outillage de qualimétrie. Une démarche " performances " basée sur des outils de tests, d'analyse et de traçabilité, ainsi qu'une plateforme de développement et de tests permettent de reproduire le plus fidèlement possible l'environnement cible même, si l'échelle n'est pas identique en termes de volumétrie de données.

Les problèmes de formation, de compétences

Pour maîtriser concepts, outillages et organisation, la courbe d'apprentissage des équipes doit être prise en compte. Le vocabulaire utilisé, la gestion des priorités, l'équilibre permanent entre la dimension technologique et les livraisons de produits ne s'acquièrent pas immédiatement. L'initialisation du projet, coaching et accompagnement dans la montée en charge sont les maîtres mots de cette courbe d'apprentissage. Ces thèmes concernent autant les équipes de développement que, pour certains, les équipes MOA. En effet, pour bien se comprendre il faut partager les " mots ". [Fig.7] Le coaching des développeurs est réalisé lors des séances de " pair-programming ", le développeur est au clavier et à la souris. Il s'occupe principalement de la syntaxe du code, de la compilation et du rejeu des tests. Le coach, quant à lui, s'intéresse à ce que le code doit faire, la conception et la revue associée. L'élaboration de la solution technique ou sa mise au point émerge du travail mis en commun des deux équipiers. Cette pratique facilite le partage des connaissances et garantit l'homogénéité des développements et la réso-



FarPoint Spread for .NET Bundle

à partir de € 1 194

FarPoint

Composant de feuille de calcul ASP.NET et WinForms haute performance personnalisable.

- Nouvelles fonctions ASP.NET: extensions AJAX, impression vers PDF, éditeur de modèle de ligne, assistant de démarrage rapide, nouveaux types de cellules, etc.
- Nouvelles fonctions WinForms: export PDF, groupements, barre de formules, Excel 2007 XML
- Contrôle unique 2 milliards de feuilles avec 2 milliards de lignes et 2 milliards de colonnes
- Inclut une version pour .NET 2.0 et .NET 3.5 (Visual Studio 2008)



TX Text Control for .NET

à partir de € 413

TX
TEXT CONTROL

Le traitement de texte pour Visual Studio .NET.

- Le traitement de texte professionnel pour vos applications
- Zones de texte Windows Forms hors droits
- WYSIWYG, tableaux imbriqués, cadres, en-têtes, pieds de pages, images, puces, listes numérotées, zoom, sauts de section, etc.
- Opérations aux formats DOCX, DOC, RTF, HTML, TXT et XML



DXperience

à partir de € 943

DevExpress®

Tous les outils DevExpress ASP.NET, WinForms et IDE Productivity en un.

- Abonnement de 12 mois pour tous les produits et mises à jour Developer Express et accès aux versions bêta en développement actif
- Composants et outils : grilles, entrée de données, outils d'écriture de code, analyse de données, graphiques, navigation/disposition, planification, solutions reporting, bibliothèques d'impression, outils de remaniement, bibliothèques ORM



ReSharper

à partir de € 149

JetBRAINS

Outil de productivité enfichable et intelligent pour Visual Studio 2005/2008.

- Assistance intelligente au codage, signalement des erreurs en cours de travail et correction rapide
- Refonte du code, test des unités, navigation et recherche, édition de scripts NAnt et MS Build, édition ASP.NET, etc.
- Toutes ces fonctions avancées sont disponibles dans Visual Studio
- Analyse et signale les erreurs de code C# (jusqu'à C# 3.0) en cours de frappe

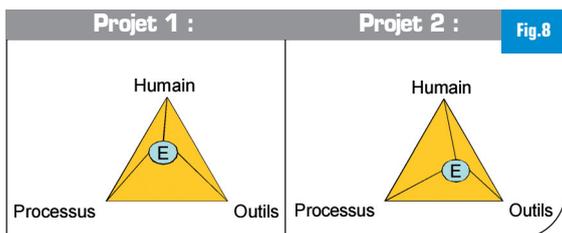
lution des bugs (si si, il y en a !!). Même si cette organisation du travail semble gourmande en énergie, le ROI est, croyez le, rapide.

La montée en charge de l'équipe pour la fabrication du produit doit faire l'objet d'une planification relative aux cadences des itérations. L'intégration d'un développeur par mois pour une équipe de moins de neuf personnes est un bon rythme.

Abordons maintenant les aspects gains et axes d'amélioration... Les retours d'expérience peuvent être illustrés par deux exemples de projets qui ont privilégié des axes différents lors de la mise en œuvre des méthodes agiles. Les schémas ci-dessous présentent pour chaque projet le point d'équilibre. [Fig.8]

Les choix réalisés apparaissent clairement lors du développement de ces projets. Les avantages et points d'amélioration qui en découlent sont présentés dans le tableau ci-dessous. A partir de ces analyses, les équipes positionnent le point de départ et défi-

Figure 8 :
Position du point
d'équilibre sur 2
projets



Retours d'expériences projets

	Projet 1	Projet 2
Avantages	<ul style="list-style-type: none"> • Process : La démarche évite l'effet tunnel propre au développement en " nouvelle technologie " : cela améliore la qualité du reporting et rassure ainsi le client qui " voit " le logiciel se monter sous ses yeux au fil des semaines • Humain : C'est un élément de motivation et de responsabilisation de l'équipe de développement. La productivité de l'équipe est allée en s'améliorant de sprint en sprint. 	<ul style="list-style-type: none"> • Process : Tenue du planning initial Points de visibilité réguliers Qualité des livraisons Prise en compte au fil de l'eau de nouvelles demandes et sans remise en cause de la date de fin de projet. • Outillage : mesure permanente de la qualité technique Evolutivité de l'application : référentiel de tests fonctionnels, scénarios de tests techniques, couverture de tests unitaires, point d'amélioration du code
Points d'amélioration	<ul style="list-style-type: none"> • Outillage : Outiller la méthode de gestion des exigences Connecter la méthode à la plate-forme d'intégration continue de manière à gagner en productivité sur l'automatisation des tests en environnement Microsoft • Humain : Suivi de la courbe d'apprentissage de la méthode 	<ul style="list-style-type: none"> • Process : Mobilisation importante des équipes pour les tests



Figure 7 : Vocabulaire méthodes Agiles

nissent les actions pour conduire l'équipe, l'organisation et les futurs développements au barycentre de ce triangle.

Les recommandations.

Ainsi trois erreurs sont couramment commises.

1 Se jeter dans la mêlée : la propension à penser que plus vite les développements sont lancés, plus vite les objectifs de délais sont atteints, est un élément de perturbation dont on ne perçoit les impacts que lors des premières livraisons. Il est alors souvent très tard pour corriger.

2 La complexité des architectures est la plupart du temps due au phénomène bien connu des développeurs, l'aspiration technologique.

Cette aspiration fait alors oublier le principe fondamental d'adéquation de l'architecture au problème à résoudre et non le satisfecit à développer une superbe technologie qui ne servira pas.

3 L'outillage insuffisant ou peu utilisé conduit à des débauches d'énergie tant dans le cadre du suivi du projet que dans l'élaboration du produit lui-même par l'impact des régressions à chaque livraison.

Afin de ressortir les bonnes pratiques, retenons quatre éléments clés :

La rigueur. Si les méthodes agiles sont génératrices de créativité dans la production des produits aussi bien que dans l'organisation des développements, il importe que la rigueur d'exécution sous-tende toutes les opérations.

La simplicité. Elle agit comme corollaire à la mise en œuvre d'architectures et de solutions technologiques pour la qualité du produit à développer.

La transparence. Partager et atteindre des objectifs communs aux équipes MOA/MOE, nécessite une transparence totale des difficultés et des arbitrages à réaliser aussi bien que de la bonne estimation des fonctions dites " essentielles ".

L'accompagnement. Cet accompagnement doit être orienté développeurs, comme nous l'avons exprimé précédemment car rien n'est inné, mais également vers les autres membres de l'équipe qui concourent à la réussite du projet.

Arrêtons-nous quelques instants sur ces éléments clés... Ces points sont universels et régissent notre activité de développement depuis toujours. Les méthodes agiles sont un levier, maintenant accessible et opérationnel, pour les mettre en œuvre au service du métier de nos utilisateurs.

■ Jacques Mezhrhid
Responsable National du développement des applications et des services managés chez Sogeti

ihm

EN TOUTE SIMPLICITÉ !

Pour implémenter des applications riches, accédant à des données hétérogènes (applicatifs métier, BD, et depuis la V4.1, Hibernate, GED et silos documentaires), déployées en Swing, plugin Eclipse ou DHTML/Ajax et s'insérant naturellement dans les processus d'entreprise...

Adoptez la puissance et l'agilité de l'approche Model-Driven

**NOUVELLE
VERSION
LEONARDI
V4.2
open source**

Concentrez-vous sur votre métier et dotez votre entreprise d'avantages compétitifs durables: amélioration du cycle de vie du logiciel, démarche itérative par prototypage pour coller aux besoins, découplage technologie/métier, évolutivité, agilité et évolutivité, le tout sans expertise technique pointue !



eXtreme Programming :

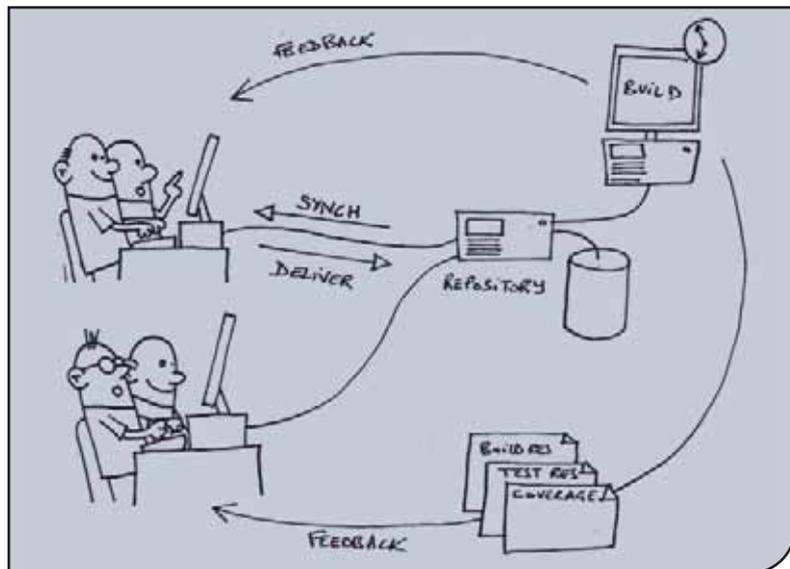
les fondamentaux du développeur agile

L'eXtreme Programming est la première méthode de développement agile appliquée en France. Contrairement à Scrum, elle prescrit des pratiques d'ingénierie logicielle. Nous allons présenter ici les plus marquantes.

Les experts de Scrum l'affirment : les pratiques d'ingénierie décrites dans l'eXtreme Programming sont indispensables pour réussir de manière durable un projet de développement incrémental, en voici quelques-unes :

Tests automatisés : comment s'assurer que le logiciel livré à chaque fin d'itération fonctionne correctement ? Il n'est pas question de tester à la main toutes les fonctions du logiciel à chaque fin d'itération : cela finirait par prendre plus de temps de tester que de développer, jusqu'à un moment où la totalité du temps de l'itération ne suffirait plus pour tester suffisamment l'application. La solution préconisée par l'eXtreme Programming est d'automatiser autant de tests que possible, et pour tous les types de tests. Pour chaque type, il existe des outils en facilitant l'automatisation, par exemple : les tests unitaires (Xunit :: junit, nunit...); les tests d'interface web (Selenium, webdriver...), ou encore les spécifications exécutables (fitnesse, greenpepper). Ainsi, le développement et le paramétrage du harnais de test deviennent partie intégrante du travail de développement.

Intégration continue : le système en cours de construction est intégré en permanence. En pratique, les sources sont compilées, les tests déroulés automatiquement, aussi fréquemment que possible. Ainsi, l'équipe est informée très rapidement dès qu'une erreur s'introduit dans le code, et les corrections rendues plus faciles puisque les modifications sont encore "fraîches". On peut s'appuyer sur des outils comme Hudson, Continuum ou Bamboo pour mettre en place un tel mécanisme.



© Emmanuel Chenu

Développement Piloté par les Tests (ou TDD)

Pour chaque fonctionnalité que je veux ajouter dans le code :

- 1) je code le test (automatisé) qui montrera que j'ai réussi,
- 2) je fais le développement le plus simple possible pour que ce test passe,
- 3) je remanie (refactoring) le code pour réduire la duplication.

Appliquer systématiquement et rigoureusement ce principe permet de produire du code couvert par des tests automatisés. Mais les développeurs témoignent que les bénéfices vont au delà - " le TDD pousse à bien coder " (Laurent Cobos, développeur chez Pyxis Technologies France) : des principes tels que le découplage, la cohérence forte, où la responsabilité unique sont encouragés, les API développées sont faciles à utiliser, le code et les tests sont expressifs, faciles à lire et à comprendre, auto-documentés. " l'approche du design est inversée " (Bruno Thomas, déve-

loppeur; Orange Business Services - Online Multimedia) : on se place en premier lieu comme utilisateur des API en cours de développement, et le design n'est plus une phase préalable au développement : il émerge en permanence pendant le développement. Au début, on a l'impression de perdre beaucoup de temps C'est difficile de définir la suite de tests qui permettra de coder ce qu'on a en tête, et même nommer un test semble fastidieux. Puis on perçoit graduellement les bénéfices, on modifie sa tournure de développement, et on comprend que passer plus de la moitié du temps de codage sur les tests est efficace pour le projet. " Si on arrête le TDD, je quitte l'entreprise ! " (Jean-Philippe Caruana, développeur, Orange).

Remaniement ou refactoring

Remanier le code, c'est le modifier sans en changer les fonctionnalités, dans l'objectif de le rendre plus facile à comprendre et à faire évoluer. L'eX-



trème Programming encourage des activités de remaniements fréquentes, à plusieurs échelles (méthode, objet, composant, système complet), y compris dans le harnais de test lui-même. C'est une hygiène à respecter pour lutter contre l'entropie dans le code.

Sans remaniement, le même cycle se reproduit toujours. Au fur et à mesure que le logiciel évolue, il devient si gros et si complexe qu'il s'écroule sous son propre poids : il est extrêmement long pour tout nouvel arrivant d'en comprendre la structure; et très coûteux de le faire évoluer pour répondre aux nouvelles demandes de ses utilisateurs. A ce stade, on décide de lancer une refonte " from scratch ", ou bien le logiciel finit par être oublié au profit d'un concurrent plus jeune et plus performant.

Grâce à un refactoring régulier et systématique, on peut enfin faire la promesse que le logiciel pourra vivre pendant 10 ou 20 ans.

Ce sont les tests automatisés et l'intégration continue qui le rendent possible : on n'a plus peur de " casser " le code quand on y touche. Si cela arrive, un test automatique échoue, l'alerte est levée immédiatement, et il est encore facile de revenir en arrière. Le remaniement est l'activité la plus complexe du développement, mais c'est aussi la plus gratifiante. " *Le refactoring, c'est la récompense* " (Antoine Contal, coach xp, Orange).

Le binômage ou pair programming

Le binômage, c'est la plupart du temps, un clavier, un écran, et 2 développeurs derrière! Le travail de développement devient un dialogue permanent. Le bénéfice immédiat est la qualité du code produit, et la diffusion des compétences dans l'équipe.

Les développeurs qui ont adopté cette pratique ne tarissent pas d'éloges : " *Chaque fois que je binôme, j'apprends* " (Radhouane Gourchene; développeur, Orange), " *Cela m'a permis de m'intégrer rapidement au projet* " (Laurent Cobos), " *Tu nivelles la compétence de l'équipe par le haut* " (Bruno Thomas), " *Tu n'es*

plus tout seul face au problème devant la machine " (Jean-Philippe Caruana), " *quand on code, on peut s'égarer, l'autre, c'est ta boussole* " (Thomas Dufourd), " *coder à 2, c'est diviser le nombre d'erreurs par 5* " (Radhouane Gourchene).

Quelques conseils pratiques : faire circuler fréquemment les individus dans les binômes, jouer au ping pong (" *J'écris un test, tu le fais passer, tu écris un test, je le fais passer, j'écris un test...* "), ne pas hésiter à se séparer quand être en face à face n'est pas efficace (pour les activités exploratoires par exemple, explorer en parallèle et faire le point régulièrement). Pour certains, binômer, c'est agréable et plutôt naturel, cela permet de faire connaissance, et même les tâches les plus ingrates deviennent motivantes. Mais ce n'est pas toujours facile, surtout quand on ne connaît pas son binôme : au début, c'est un bras de fer, on veut démontrer sa compétence et cacher ses lacunes, on a peur de ralentir l'autre. Puis on apprend à se faire confiance, à exposer ses faiblesses et à s'appuyer sur les forces de l'autre, et le travail devient un enrichissement mutuel.

Binômer rend l'activité de développement très intense : face à un problème difficile, pas d'échappatoire possible, pas question de l'éviter, ni de faire des compromis sur la qualité. Quand on est deux, on est plus discipliné, et le regard de l'autre pousse à donner le meilleur de soi-même, en permanence. A la fin de la journée, on est épuisé. Heureusement, l'eXtreme Programming engage par ailleurs à travailler à un rythme soutenable indéfiniment : les heures supplémentaires doivent rester exceptionnelles.

Pour quel projet ?

L'eXtreme Programming est un mode de développement demandant rigueur et discipline. Il est recommandé pour les développements logiciels réalisés en équipes colocalisées, où la qualité du code (en particulier son évolutivité à long terme) est la préoccupation première. C'est cette qualité qui permet de s'assurer que le logiciel

pourra évoluer longtemps, tout en apportant régulièrement de la valeur à ses utilisateurs.

Un art martial du développement ?

L'eXtreme programming est une méthode agile complète, applicable sans référence à Scrum. Elle prescrit en tout 13 pratiques et s'appuie sur le respect de 4 valeurs. Pour chaque individu, comme pour l'équipe dans son ensemble, les adopter demande courage et persévérance.

Au tout début, cela peut être un choc, les habitudes sont bousculées, les automatismes remis en question. Les premières semaines peuvent être difficiles, avec l'impression que le projet piétine. Après le premier apprentissage, les résultats apparaissent. Enfin, l'investissement s'avère payant, de manière durable : le logiciel développé continue à évoluer à un rythme quasi constant que l'équipe peut soutenir indéfiniment; les développeurs prennent plaisir à travailler ensemble et sont fiers du code qu'ils produisent. Pour chaque individu, ça passe ou ça casse : des développeurs rejettent en bloc certaines pratiques. Notamment, le binômage ne convient pas à certains tempéraments. En revanche, ceux qui ont adopté l'eXtreme Programming ne jurent que par lui : " *J'ai l'impression d'avoir changé de métier* " (Thomas Dufourd) " *XP m'a redonné le goût du développement* " (Jean-Philippe Caruana), " *c'est l'avenir du développement* " (Bruno Thomas).

Ressources

Test Driven Development: By Example
Kent Beck

Refactoring
Martin Fowler

L'Extreme Programming - Avec deux études de cas, 2002
Jean-Louis Bénard, Laurent Bossavit, Régis Médina et Dominic Williams



■ Raphaël Pierquin
coach d'équipes de développement chez Pyxis Technologies France
rpierquin@pyxis-tech.com

La journée d'un développeur agile

Les méthodes agiles sont des pratiques itératives de développement logiciel qui répondent à des besoins de changement pouvant être fréquents, et qui privilégient l'interaction humaine en visant la satisfaction réelle du client. Au travers de cet article, nous racontons la journée type de Martin, devenu développeur agile, et les grands changements que cela a engendré dans son travail. Si vous êtes développeur professionnel, vous comprendrez vite les bénéfices à utiliser de telles méthodes !

9H00

Nightly Build et rapports journaliers

Martin arrive au bureau. Il travaille comme développeur sur différents projets et, aujourd'hui, il est plus motivé que les autres jours. Il repense au jour où sa société a décidé de gérer les projets à l'aide des méthodes Agiles. Depuis, il a retrouvé un intérêt pour son travail qu'il n'avait plus eu depuis quelques années. Le temps de poser ses affaires, il jette un rapide coup d'œil à son mail : le rapport de génération de build automatique, qui s'exécute à 2:00 du matin, lui indique que tout s'est bien passé et que la build a été livrée aux testeurs sur un partage réseau.



9H15

Les individus et les interactions

Martin rejoint l'équipe à la machine à café. Leur chef de projet arrive un peu après et en profite pour donner un feedback rapide sur les tests menés la veille : tout est au vert, les testeurs sont très satisfaits des ajouts remontés hier. Les priorités

de la journée sont rapidement balayées pour que toute l'équipe soit alignée sur les mêmes objectifs. Comme d'habitude le point se conclut par un petit tour de table : Qu'est ce qui se passe bien ? Moins bien ? Que peut-on améliorer ? Frédéric fait part au reste de l'équipe d'une optimisation qu'il a trouvée pour accéder aux données SQL. Il partagera l'info sur le wiki interne de l'équipe.

9H30

Collaboration entre le métier et la technique

De retour à l'open space, Martin lance son Visual Studio, met à jour sa solution en récupérant les dernières versions des codes sources depuis la branche principale du projet sous Team Foundation Server (TFS). Il reprend les documents décrivant les fonctionnalités à implémenter. Ces documents ne sont pas exhaustifs et se présentent sous la forme de " use case " : le besoin métier et les solutions techniques y ont été élaborées en collaboration entre les fonctionnels et les membres de l'équipe. Ce document de travail évolue tout au long du projet grâce à l'effort de chacun pour contribuer à son écriture. Martin a justement prévu de voir Sacha à 11h00 pour discuter d'une fonctionnalité. Les réunions entre le fonctionnel et la technique ont permis d'améliorer la cohérence globale du projet mais aussi de mieux faire comprendre et revaloriser le rôle de chacun.

9H45

Check-in réguliers et intégration continue

Martin récupère ses tâches depuis



TFS (Work Items) et démarre ses développements dans l'ordre des priorités. C'est véritablement dans la manière de développer que Martin a changé ses habitudes afin de se caler au rythme des itérations : l'objectif est de faire simple, pragmatique et évolutif pour apporter de la valeur fonctionnelle le plus rapidement possible.

Martin vient de finir les évolutions qu'il avait prévues la semaine dernière. Il fait une dernière mise à jour de ses codes sources en local, compile et intègre ses modifications dans le référentiel. Une build de validation est automatiquement mise en file d'attente sur le serveur de build qui exécutera par ailleurs les tests unitaires. Martin recevra ensuite un rapport lui permettant de s'assurer que tout s'est bien passé.

11H00

Les changements, même de dernier moment, sont les bienvenus !

Sacha vient retrouver Martin à son poste de développement et lui tend un document de demande d'évolution. Il faut revoir 25% d'une fonctionnalité développée la semaine dernière et faire des modifications



plus ou moins mineures. Pas de problème pour Martin, ce sera un développement de quelques heures. Le tout se faisant dans un flux continu, les changements arrivant au fur-et-à-mesure que les tests sont faits, les modifications sont finalement relativement mineures.

En fait, c'est devenu très simple pour Martin : il développe rapidement des choses testables sur la base des use cases émis par les fonctionnels.

Plus il met rapidement à disposition des testeurs quelque chose d'utilisable, plus vite les testeurs reviennent vers lui pour demander des changements en adéquation avec le métier.

14H00

Réunion hebdomadaire et mesures de progression

Après le déjeuner, comme tous les mercredis, l'équipe se retrouve dans l'open space pour faire le point sur l'avancement du projet. L'élément de mesure principal est la complétude des fonctions livrées et testées. Ces réunions hebdomadaires sont aussi l'occasion d'inviter d'autres membres des équipes de test ou des fonctionnels. Le chef de projet donne un retour sur la progression, mais aussi la satisfaction remontée.

On en profite aussi pour passer en revue les bugs les plus importants, ceux qui sont en priorité 1 (à faire le plus rapidement possible) et sévérité 1 ou 2 (bug critique ou bug majeur). Grâce aux différents indicateurs remontés par TFS et le cube associé au projet, le chef de projet a pu mettre en place une feuille Excel permettant de piloter le rythme des développements, la qualité des livrables et la progression du travail

réalisé, tout cela dans le souci de gérer la bonne priorité au bon moment.

14H30

Tests unitaires et couverture de code

Retour au développement, Martin implémente les tests unitaires des développements réalisés le matin même. Il trouve intéressant la pratique du développement piloté par les tests (TDD) mais son équipe n'a pas choisi de suivre cette pratique.

Néanmoins, les tests unitaires permettent de vérifier qu'il n'y a pas de régression engendrée par les nouveaux développements, et la couverture de code associée aux tests unitaires permet de vérifier l'exhaustivité des tests et ainsi la qualité qui en découle. Cette méthode a permis à Martin de s'améliorer comme développeur professionnel : du code plus facile à maintenir, plus lisible et surtout plus fiable.

Au final, c'est toute l'équipe qui a bénéficié de cette pratique et les compétences s'en sont trouvées nettement améliorées, pour le plus grand bien de tous. Tirer l'équipe vers l'excellence technique est l'un des principes des méthodes Agiles.

18H30

Conclusion

Martin a fini sa journée et est satisfait de son travail : il obtient du feedback des testeurs et aide les fonctionnels à formaliser leurs besoins lors de leurs réunions de travail. L'ambiance au sein de l'équipe n'a jamais été aussi bonne et constructive : tout le monde s'entraide et la communication est efficace. Il sait que grâce au choix des méthodes Agiles, sa carrière de développeur professionnel reprend de l'intérêt, qu'il est valorisé par le feedback des utilisateurs finaux qui obtiennent des réponses concrètes à leurs besoins, et que son avis est respecté car l'autonomie dont il bénéficie dans la conception technique prouve que ses idées sont bonnes. Pour rien au monde, il ne voudrait revenir aux méthodes précé-

Mpoware lance l'Agile Training Center

Pour les développeurs, testeurs, chefs de projet, architectes ou MOA, le workshop d'une journée, que propose Mpoware au travers de l'Agile Training Center, permettra aux équipes projets de comprendre et pratiquer les valeurs et principes des méthodologies Agiles appliquées à la plateforme Microsoft Visual Studio Team System 2008. La plupart du temps, les équipes projet déploient VSTS mais utilisent seulement la plateforme pour versionner leurs codes sources, passant ainsi à côté des avantages et du retour sur investissement qu'ils gagneraient à suivre un processus Agile. Ce n'est pas la plateforme qui rend les équipes agiles, c'est la méthodologie et ses valeurs appliquées à la plateforme. Mpoware propose un workshop d'une journée avec mises en situation projet pour vos équipes : participez à toutes les étapes d'un projet Agile et pratiquez les méthodologies Agiles alliées à la plateforme VSTS pour améliorer concrètement la performance et la productivité de vos équipes projets.

Contactez nous vite pour réserver votre journée via contact@mpoware.com.

dentes dans lesquelles le chef de projet lui demandait de lire des spécifications techniques détaillées qui n'étaient jamais exhaustives et qui ne laissaient aucune place à sa créativité de développeur.

Références

Manifesto for Agile Software Development : <http://www.agilemanifesto.org/>
 Guidelines for Test-Driven Development : <http://msdn.microsoft.com/en-us/library/aa730844.aspx>
 Continuous Integration : <http://martinfowler.com/articles/continuousIntegration.html>



■ Frédéric
QUEUDRET

MVP Client Application & CTO de la société Mpoware, accélérateur d'innovation. Société française d'édition de logiciels et de prestations de services sur les technologies .NET. Mpoware est signataire du manifeste Agile et accompagne ses clients dans la mise en place des méthodes Agiles au sein de leurs équipes.

<http://www.mpoware.com/>

Avis d'expert

Standardiser la résolution de problèmes dans une équipe agile

Jeff Sutherland a créé la première équipe Scrum en mettant en oeuvre un article écrit par deux japonais. Des démarches Lean venues du Japon, nous pouvons tirer des outils pour aider à améliorer la qualité dans une équipe agile. Voyons comment une équipe XP/Scrum utilise un de ces outils pour la résolution de problèmes.

Dans l'équipe eXtreme Programming / Scrum où je travaille, lorsqu'un problème survient, nous utilisons depuis quelques mois un format standardisé issu du Lean : le **QRQC** (Quick Quality Response Check). De quoi s'agit-il ? En présence d'une situation insatisfaisante, un membre de l'équipe prend une feuille de papier et nomme le problème en haut à gauche. Par exemple : nous avons mis 5h pour ajouter deux options au lieu de 1h30 estimée initialement. On indique ensuite l'impact pour le client : risque de non-respect de l'engagement du sprint en cours. Vient ensuite l'impact pour l'équipe de développement : mon binôme écrit "maux de tête". Puis on passe à la description du problème (que s'est-il passé ?) :

- Première difficulté : nous avons tenté pendant une heure d'écrire un test pour soutenir l'ajout d'une option dans le fichier de configuration.
- Deuxième difficulté : pour l'une des options, nous avons passé du temps à trouver le lien entre l'interface homme-machine et sa représentation dans le code.

On note ensuite les causes probables :

- 1/ Méconnaissance de l'utilisation combinée de JNDI et Spring.
- 2/ Utilisation inutile de JNDI dans ce contexte
- 3/ Nom de variable décorrélé du terme employé dans l'interface graphique

Pour chacune des causes, on regarde s'il existe un standard explicite et s'il a été respecté. Dans le cas de cet exemple, nous n'avons aucun standard. Ensuite, pour chacune des trois

causes, nous énumérons des actions possibles :

- 1/ Faire un *how to* sur comment écrire des tests conjuguant Spring, JNDI et l'ajout d'option.
- 2/ Remplacer JNDI par un fichier de propriétés.
- 3/ Choisir des noms de variables plus cohérents.

Enfin, on indique la date et le temps passé (un chronomètre avait été déclenché en prenant la feuille). Nous y avons consacré trop de temps : 20 minutes au lieu des 10 minutes du QRQC précédent. Avec mon binôme, nous concluons qu'il vaut mieux faire de plus petits QRQC au fur et à mesure que les problèmes sont rencontrés plutôt que d'attendre la fin de la tâche. Le lendemain, cette stratégie se révélera payante.

Un écart par rapport à un standard existant

Pour notre problème de dépassement de temps, nous n'avons aucun standard, mais ce n'est pas toujours le cas. Dans une équipe agile, des standards explicites existent dont nous n'avons pas conscience, comme la définition de *done* (aussi appelée fini-fini).

Au lendemain d'une démonstration de fin d'itération difficile, un QRQC a lieu pendant la rétrospective. L'une des causes est le non respect de la définition de *done*. Enoncer en équipe cet écart par rapport au standard permet alors à un des développeurs de prendre conscience qu'il ne respectait plus cette définition de *done*. Il décide de se montrer plus rigoureux et un autre développeur se propose

Problème	Impact Client	Impact Dev.
Démo difficile	Des stories non validées	Stress, non respect de l'engagement
Décrire le problème : Que s'est-il passé ?		
A 13h40 la recette ne fonctionnait pas. La démo a commencé en retard. Elle a été faite sur un poste de développement.		
Cause Probable	Standard Existe / Respecté	Action Possible
Deux instances de tomcat tournaient sur la recette	OUI / OUI	Cabler stop sur force-stop
Non respect de la définition de done	OUI / NON	Passer en revue les stories chaque matin
Date : 10/12/2042		Temps passé : 10 min 40 sec

pour vérifier les critères du fini-fini chaque matin.

Un standard à améliorer

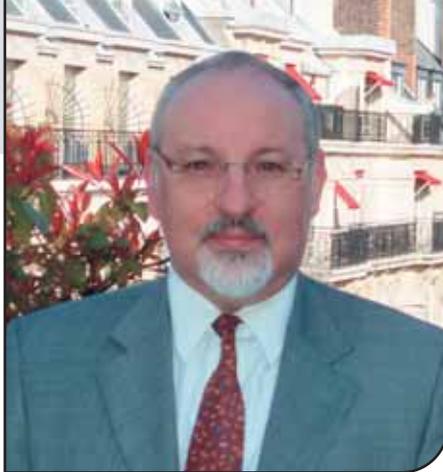
Une autre cause de la démo difficile dont je viens de parler était que deux instances de Tomcat tournaient sur la machine de recette. C'est la mauvaise instance de Tomcat qui répondait lors de la démo.

Nous nous posons alors la question des standards :

- Un standard existe-t-il ? Oui : un script centralise l'accès aux serveurs d'application.
- Le standard a-t-il été respecté ? Oui. On a bien invoqué `application_server stop`, mais seule l'option `force-stop` tue Tomcat à coup sûr.
- Solution possible : changer le standard en câblant l'option `stop` sur l'option `force-stop`

Se poser plus systématiquement la question des standards au sein d'un QRQC permet donc d'en prendre conscience, aide à les vérifier et surtout à les améliorer. Le QRQC étant lui-même un standard, sans doute est-il possible de l'améliorer !

■ Philippe Blayo, Développeur
Orange Business Services



Entretien avec Claude Puppatti, directeur général de Valtech France, spécialiste des Méthodes Agiles.

“L’agilité, une révolution culturelle”

Claude Puppatti dirige Valtech pour la France. Le groupe intègre 4 entités, dont Valtech Technology, conseil et intégration de projets, spécialiste des méthodes agiles. Il est installé dans 7 pays, dont les Etats-Unis et l’Inde. Il compte 1130 employés, dont 266 en France. 39% du CA de 100,6 M€ est réalisé par la France (chiffres 2008).

Programmez : Les équipes techniques acceptent davantage ces méthodes ?

Claude Puppatti : Le fonctionnement est différent, les équipes du client et du prestataire sont étroitement mêlées, ce qui demande à ré-inventer les contrats : la logique maître d’ouvrage/maitrise d’ouvrage est dépassée, on est dans une relation de co-responsabilité. Le mode de communication devient très direct, les équipes opérationnelles ont davantage de responsabilités. Les reporting doivent être organisés autrement afin de ne pas peser, comme c’est souvent le cas, sur le cycle.

P : Le concept de “ méthodes agiles ” est-il aujourd’hui bien compris des entreprises ?

CP : Le terme est galvaudé, mais surtout, la difficulté réside dans le fait qu’il s’agit d’un rude changement de mentalités, d’une révolution culturelle ! Cela demande un effort, alors que demeurer dans le classique “ effet tunnel ” des projets est si confortable ! Nous sommes dans des cycles courts, de 4 à 6 semaines, et il faut

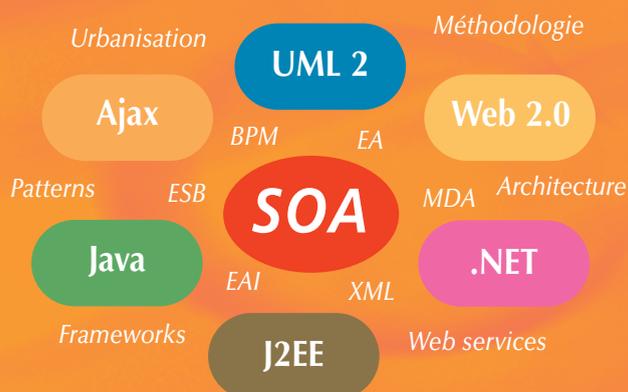
disposer très vite des composants. Les méthodes agiles induisent une conduite du changement au sein de l’entreprise. Les cycles de développement ne suivent pas les cycles de décision de l’entreprise. On avait 3 ans pour un projet, aujourd’hui, on a 15 mois. Les priorités changent : installer un ERP ne suffit plus forcément à donner un avantage stratégique.

P : Etre spécialiste des Méthodes agiles est un atout ?

CP : Finalement, notre mission est de plus en plus souvent de modifier la culture de l’entreprise. Nous allons constituer au niveau de Valtech une task-force capable de vendre une offre globale, incluant la conduite du changement, au sein de La Direction Informatique, et dans le fonctionnement des services. Notre maîtrise des méthodes agiles nous confère une avance au niveau des SSII. Une fenêtre d’avance de 15 à 24 mois ? Les choses vont vite !

Propos recueillis par Jean Kaminsky

Soyez prêts pour les nouvelles architectures SOA et Web 2.0 !



SOA est devenu en peu de temps le mot-clé des développements logiciels. SOA est une nouvelle façon de faire qui s’appuie sur un ensemble de technologies existantes : UML, J2EE, .Net, XML, etc. Maîtriser SOA implique de maîtriser ces technologies pour les associer efficacement au sein d’une nouvelle approche.

SOFTEAM Formation, forte de son expérience en Méthodologie, Architecture et Développement, a construit un cursus complet de formation SOA qui vous permet de débiter dès les phases amont, de poursuivre en architecture, et d’aller jusqu’à la réalisation dans le langage de votre choix.

Nouveau catalogue Formation 2009 :

UML pour la maîtrise d’ouvrage	2 j
Analyse et conception avec UML	4 j
SOA Alignement Métier du Système d’Information	2 j
SOA Architecture d’Entreprise (EA)	2 j
SOA Méthodologie pour SOA	2 j
SOA Architecture technique SOA	2 j
SOA Développement de Web Services en Java	3 j
SOA Développement de Web Services en C#	3 j
Architecture distribuée : la synthèse	2 j
Programmation orientée objet avec Java	4 j
Développement d’applications JEE 5	5 j
Développement d’applications JEE 5 Front End	4 j
Développement d’applications JEE 5 Back End avec EJB 3	3 j
Maîtrise du framework (Struts / JSF / Spring / Hibernate)	3 j
Développement d’applications .NET / C#	4 j
Développement d’applications RIA avec	2 j
Web 2.0 (Ajax / Dojo / GWT / FLEX3)	4 j

Convergence SOA, UML2, BPMN, EA

Modélisation EA, BPMN, SOA avec Objecteering SOA Solution	2 j
Analyse et Conception UML2 avec Objecteering Modeler	5 j
Expression de besoins en UML2 avec Objecteering Scope Manager	1 j
Architecture MDA avec Objecteering MDA Modeler	2 j
Génération de code Java, .NET C#, C++ avec Objecteering Developer	1 j



Vendre SCRUM à une équipe en cycle V

Passer à la méthode SCRUM pour une équipe de développement de logiciel appliquant une méthode classique (de type cycle en V) est un challenge en vue d'une bien meilleure efficacité, mais nécessite de grands changements organisationnels et une remise en question des pratiques de l'équipe. Nous présentons dans cet article un bref résumé de la méthode, ses avantages, ses difficultés, et quelques recettes simples pour aider à franchir le pas.

Pour rappel, la méthode de travail doit être adaptée à la réalisation d'un nouveau produit complexe pour lequel il n'est pas possible de faire des spécifications fonctionnelles et techniques détaillées et exhaustives. Les tâches sont découvertes progressivement, les changements sont la norme et doivent être considérés comme une opportunité: un besoin, une solution technique qui n'ont pas été identifiés lors du lancement du projet, mais qui sont pertinents, doivent être pris en compte. A l'inverse, un choix non adapté doit être remis en question et retiré si nécessaire.

Ainsi, il importe que la méthode de travail permette à l'équipe de s'adapter aux changements et imprévus, qu'ils soient techniques, fonctionnels ou organisationnels (par exemple un changement de la constitution de l'équipe). Il est donc nécessaire d'utiliser des outils et une organisation qui permettent :

- D'optimiser le travail du groupe: faire ce qui est nécessaire et suffisant pour l'utilisateur, favoriser le partage des savoirs et des compétences et mettre en avant les compétences de chacun.
- D'établir un dialogue de qualité avec l'utilisateur: comprendre ce qu'il dit, parler un langage qu'il comprend et l'impliquer dans la réussite du projet
- De s'organiser en s'appuyant sur des estimations fiables: un plan précis, mais faux, est plus qu'inutile: il est dangereux! Il est important d'estimer la qualité de l'information dont on dispose.

Les méthodes de travail qui respectent ce besoin de grande réactivité sont appelées les méthodes agiles. Le " manifeste agile " formalise la notion d'agilité en la déclinant sous la forme de 4 valeurs et de 12 principes. Il est intéressant de présenter et parcourir avec les membres de l'équipe ces valeurs et principes afin

de les sensibiliser sur les enjeux et réalités pratiques auxquels il est nécessaire de faire face dans le cadre du développement de logiciel.

Quels sont les principes de scrum ?

Ils sont simples.

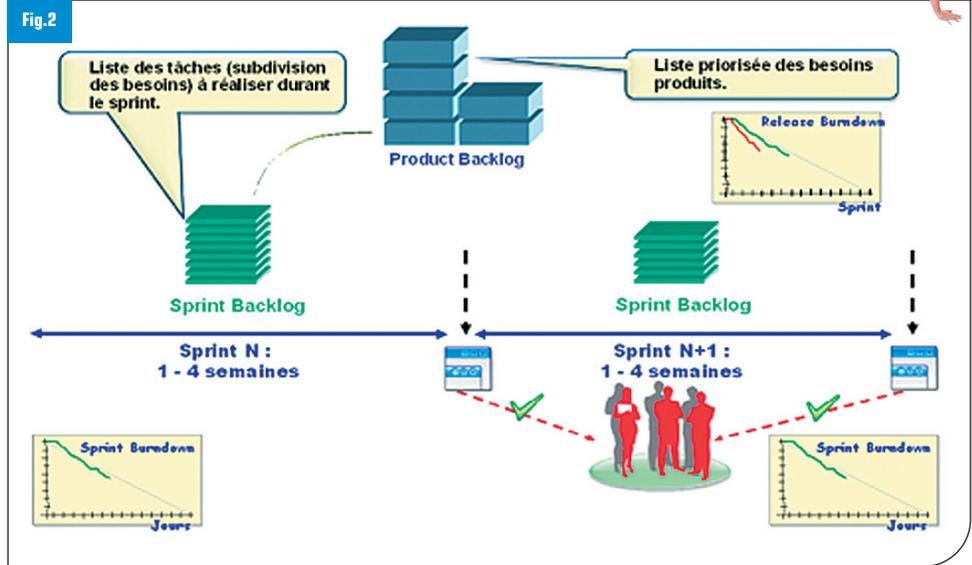
- Les personnes qui interviennent sur le projet sont réparties en seulement 3 rôles :
 - Le Product Owner / directeur de produit : il représente les utilisateurs, définit et priorise les demandes produit.
 - L'équipe SCRUM: avec moins de 7 personnes regroupant tous les rôles traditionnels (architecte, développeur, testeur, administrateur). Cette équipe développe le produit et se gère en toute autonomie.
 - Le Scrum master, qui n'est pas le chef de projet mais qui a un rôle de coach: sa mission est de tout mettre en oeuvre pour que l'équipe travaille dans de bonnes conditions et se concentre sur l'objectif du projet.
- Le Product BackLog (cf figure 1) liste les besoins produits. Il est géré par le Product Owner mais est partagé avec l'ensemble du groupe. Les besoins sont priorisés et l'accent est mis sur les besoins les plus prioritaires et devant être adressés à court terme. Ils sont associés à des critères d'acceptance qui permettent de définir, sans ambiguïté, les critères qui permettront de considérer que le besoin est couvert.

Gestion des besoins et des tâches.





- Le Sprint BackLog contient les tâches à réaliser durant une itération appelée Sprint.
- Les Burndown Charts sont des graphiques permettant de suivre visuellement l'avancement du sprint et de la release (composée de plusieurs sprints).
- Les sprints ont une durée fixe et se déroulent de la façon suivante: (cf figure 2)



- Chaque sprint commence par une réunion de planification appelée Sprint planning. A l'issue de cette réunion, l'ensemble de l'équipe connaît et comprend l'objectif du sprint et chaque besoin, qui doit être adressé dans le sprint, est décomposé en tâches.

- Durant le sprint, chaque membre de l'équipe prend une et une seule tâche dans le pool de tâches les plus prioritaires et effectue celle-ci.

- A heure fixe, un daily scrum (mêlée quotidienne) est effectué debout. Chaque membre prend la parole à tour de rôle pour indiquer ce qu'il a fait, ce qu'il va faire et les problèmes qu'il a rencontrés. L'objet de ce meeting est de faire un statut mais pas de résoudre les problèmes.

- En fin de sprint, les besoins réalisés durant le sprint sont montrés à travers l'utilisation du logiciel.

- La réunion de rétrospective est faite après la démonstration. Cette réunion a pour objet de lister les points qui ont bien fonctionné, ceux qui peuvent être améliorés et de convenir d'un ou 2 axes d'amélioration pour le prochain sprint.

incrémentale avec des itérations time boxées. Ainsi, SCRUM permet de satisfaire le client en livrant tôt et régulièrement des logiciels utiles et de qualité (puisque composés de fonctionnalités terminées).

On retrouve dans SCRUM l'importance de la cohésion du groupe pour faire face aux difficultés qui vont être rencontrées tout au long de la vie du projet.

Dans l'équipe Scrum, chaque membre est impliqué et a sa place dans le projet : les compétences, connaissances et avis de chacun sont importants et complémentaires. Cette organisation permet à chaque membre non seulement de remonter des besoins qui n'ont pas forcément été identifiés, mais également de mieux comprendre les besoins et difficultés des autres membres de l'équipe : par exemple, l'implication d'une personne en charge de l'exploitation de l'application permet de mieux comprendre et appréhender les besoins de monitoring ou de déploiement de l'application.

La responsabilité n'incombe plus à une seule et même personne mais est partagée entre le product owner, l'équipe scrum et le scrum master.

Quel est l'intérêt de cette approche ?

L'équipe est très efficace :

- Les membres de l'équipe sont impliqués donc amenés à collaborer

ensemble, ce qui permet de partager la connaissance et d'éviter ainsi que le groupe ne devienne trop dépendant d'une seule personne. Plusieurs personnes sont en mesure de prendre une tâche hautement prioritaire et sont ainsi capables de terminer une fonctionnalité.

- Les livraisons régulières motivent les membres et les impliquent dans la réussite du projet.
- L'équipe travaille dans de bonnes conditions puisque le scrum master fait en sorte qu'elle ne soit pas perturbée par des éléments extérieurs au projet.

L'équipe livre régulièrement des fonctionnalités terminées, ce qui permet :

- D'utiliser le logiciel en l'état.
- De mesurer la capacité de production (appelée vélocité) du groupe et ainsi d'ajuster la planification des fonctionnalités qu'il reste à implémenter.

Comment passer en mode scrum ?

- Former l'équipe à SCRUM.
- Rassembler l'équipe projet dans une seule et même pièce.
- Mettre en place des outils et pratiques pour travailler efficacement en mode incrémental. SCRUM ne précise pas les techniques d'ingénierie du logiciel à utiliser. Une bonne pratique consiste à utiliser celles préconisées par XP notamment l'utilisation d'une plate-forme

Déroulement d'un sprint

Qu'est ce qui différencie scrum des autres méthodes agiles ?

A la différence des méthodes UP ou XP, SCRUM met l'accent sur l'implication de chaque membre de l'équipe pour atteindre un objectif et non pas sur des pratiques de développement de logiciel.

Le product owner est fortement impliqué : il revoit les priorités et besoins à chaque sprint et se focalise sur leurs valeurs business.

L'équipe Scrum travaille de façon

d'intégration continue et le Test Driven Development (TDD)

- Initialiser un product backlog en se concentrant sur les besoins hautement prioritaires. Mettre l'accent sur les critères d'acceptance de chaque besoin. Le découpage est un exercice difficile et pour lequel il est important de consacrer du temps puisque c'est à partir de ce découpage que l'équipe va travailler.
- Se poser la question de la nécessité d'opérations faites habituellement en mode cycle en V tant d'un point de vue technique qu'organisationnel (nécessité d'un rapport, d'un document...).
- Si le projet est en cours et a commencé en mode cycle en V, arrêter toutes les nouvelles tâches de développement et de spécifications et terminer en testant et intégrant les fonctionnalités commencées.

Les difficultés qui peuvent être rencontrées lors de la mise en place de scrum

SCRUM remet en question :

La façon de travailler de l'équipe :

- Les fonctionnalités doivent être exprimées sous forme de besoins qui ont de la valeur pour l'utilisateur et qui sont adressables en un sprint: trouver le bon niveau de granularité et le bon découpage demande de la pratique.
- L'équipe doit être capable de délivrer des fonctionnalités avec un coût de tests, de packaging et de déploiement acceptables. La mise en place de l'outillage peut nécessiter un effort important pour des projets existants ayant peu ou pas de tests automatisés ou des procédures de déploiement manuelles et fastidieuses.
- Des critères d'acceptance doivent être définis avant d'adresser le besoin. L'équipe doit s'habituer à terminer (en accord avec les critères d'acceptance définis) les fonctionnalités adressées avant d'en commencer d'autres.

L'organisation de l'équipe :

- Il faut faire accepter aux personnes

la nouveauté. Elles ne travaillent pas exclusivement sur leur domaine de compétence ou selon leurs aspirations: par exemple, un développeur va être amené à contribuer aux tests pour atteindre l'objectif du sprint. Son rôle ne se limite donc plus simplement à produire du code.

- Chaque membre n'a pas une liste de tâches affectées mais prend des tâches parmi les plus prioritaires tout au long du sprint. Toutefois, il existe des limites puisque certaines tâches sont liées à un corps de métier et ne peuvent être réalisées par n'importe quel membre de l'équipe. C'est le cas, par exemple, de certaines tâches de déploiement en production.
- Les membres de l'équipe qui avaient un rôle décisionnel (notamment le chef de projet) ou qui avaient obtenu un " titre " convoité depuis longtemps peuvent ressentir un sentiment de frustration.
- A l'inverse, les personnes qui ne sont pas habituées à se prononcer auront besoin d'une phase d'adaptation.

De ce fait, du temps et de l'adaptation à plusieurs niveaux peuvent être nécessaires pour obtenir une cohésion optimale de l'équipe.

Pour passer outre ces difficultés, le rôle du scrum master est essentiel. Il doit pour cela :

- Être à l'écoute de l'équipe et attentif aux obstacles rencontrés qui peuvent être de nature très variée (conflits internes, problèmes d'infrastructure, administratifs...),
- Faire appliquer les valeurs et les pratiques de SCRUM,
- Superviser l'activité des membres,
- Communiquer au client l'avancement et les problèmes,
- Protéger l'équipe de tous les éléments perturbateurs externes,
- Encourager la coopération entre les personnes,
- Aider le product owner à sélectionner les fonctionnalités qui ont la plus-value maximale.

Quelles solutions peuvent être mises en place ?

Il peut être nécessaire de procéder par étapes, notamment lorsque l'équi-

pe a un historique de cycle en V. Par exemple, la non affectation de tâches lors du sprint planning est sujet à discussion. Si, malgré une explication, la majorité n'est pas convaincue de l'intérêt de cette approche, il est possible dans un premier temps d'affecter les tâches lors du sprint planning. On démontre ainsi à l'équipe qu'en se focalisant sur les objectifs du sprint, le turn over des tâches est tel qu'il n'y a pas d'intérêt à les assigner. Après quelques sprints, il est plus facile de les convaincre d'essayer (et d'adopter :) le fait de ne pas affecter les tâches.

Il est important d'essayer de nouvelles pratiques et de se faire un jugement après les avoir testées: l'utilisation du tableau scrum (qui contient l'objectif du sprint, les tâches à faire, en cours et terminées) et du planning poker pour faire des estimations font partie des pratiques les plus courantes pour appliquer rapidement les valeurs et règles de SCRUM.

Pour conclure, bien que la méthode SCRUM réponde à des règles relativement précises, il est important de bien garder à l'esprit que chaque équipe, chaque projet et chaque contexte sont différents. La solution retenue sur un projet n'est pas forcément celle qui doit être appliquée sur tout autre projet. Le cadre de SCRUM est suffisamment large pour permettre à chaque équipe de trouver l'organisation et les solutions adéquates.

■ Laurent Laslaz
Scrum Master
Objet Direct

A propos de Objet Direct

Objet Direct (filiale de Homsys Group) développe depuis plus de 10 ans son expertise dans la modélisation, les architectures objet (Java, .NET, RIA) et les Méthodes Agiles, pour accompagner ses clients dans le pilotage de leurs projets, depuis la conception jusqu'au déploiement des applications, et optimiser leurs systèmes d'information, leurs méthodes de développement logiciel et leurs processus métiers.

www.objetdirect.com



Adopter le mode agile en début d'un projet ou en cours de route

Faire fonctionner un projet informatique en mode agile reflète la volonté de créer une dynamique de travail à la fois efficace et valorisante pour les équipes impliquées. Quel que soit le moment de la vie d'un projet où l'Agilité est adoptée, cette décision est un acte novateur, porteur de changements dans les habitudes des développeurs, des managers, des utilisateurs.

Pour démarrer un projet agile ou pour " rendre agile " un projet en cours de développement, il faut agir sur les mêmes axes : organisation spécifique de l'équipe, mise en place de l'usine de développement nécessaire pour les tests, gestion de la configuration de travail itératif, avec comme objectif l'amélioration continue. Définir une organisation en début de projet présente bien des avantages: si l'équipe choisie connaît les principes de l'Agilité elle fonctionnera facilement en mode agile (SCRUM, XP, Lean ou autre mixage opportun de méthodes). Si elle n'est pas encore à l'aise avec les pratiques agiles, elle sera motivée d'explorer des nouvelles voies qui –selon les règles de l'art de l'agilité– devraient valoriser son travail.

De la même façon, l'adoption des méthodes agiles en cours de route, est globalement perçue par les équipes de développement comme une aide à l'amélioration. Sur l'axe organisationnel, la difficulté la plus fréquente en cas de mise en place de l'Agilité sur un projet déjà démarré, est l'intégration des profils préférant travailler en solitaire.

De la " MOA " au " Product Owner " et des spécifications fonctionnelles au backlog

Un acteur majeur dans la dynamique Agile est le représentant des utilisateurs (Product Owner). C'est lui qui est mandaté pour définir les priorités métier et remplir la " feuille de route "

de l'équipe de développement pour chaque nouvelle itération. Il est bien plus facile d'intégrer la nécessité de trouver le profil adéquat comme pré-requis pour un démarrage, que d'identifier ce profil dans un dispositif avec des habitudes de travail installées. Un mauvais casting du Product Owner peut compromettre un projet Agile. Ce risque est d'autant plus important dans le cas de l'agilisation d'un projet existant.

Le Product Owner est le responsable et le principal contributeur du contenu du backlog. Ce terme SCRUM est une liste non hiérarchisée des fonctionnalités simples (" stories "). L'équipe de développement, qui interprète et décline techniquement les fonctionnalités, alimente aussi le backlog. Sa constitution est consentie comme un élément d'entrée pour le démarrage d'un projet agile, et peut être perçue comme une surcharge de travail bureaucratique sans bénéfice immédiat dans le cas de l'adoption de l'agilité en cours de route. Dans ce 2e cas, le backlog risque de ne pas avoir assez de profondeur et de ne pas donner assez de visibilité sur l'avancement du projet. De leur côté, les équipes de développement peuvent perdre en motivation et confiance, car elles ne peuvent pas se projeter suffisamment dans l'avenir.

Le rythme itératif

C'est la pierre angulaire des projets agiles. Le respect de ce rythme et des bonnes pratiques énoncées par les méthodes agiles (la stand-up mee-

ting, le TDD, le " Kanban ", etc.) est essentiel.

Adopter ces nouvelles pratiques pour un nouveau projet est " naturel ". En effet, il est plus facile de définir des règles du jeu sur " une feuille blanche " que de changer les habitudes et le processus de gestion d'un projet en cours. Plus le projet est important et le processus mature, plus il sera difficile de changer. A ce moment, une question s'impose: pourquoi adopter le mode de fonctionnement agile pour un projet déjà démarré ? Souvent la réponse est liée au dysfonctionnement du groupe, un manque de visibilité sur l'avancement, une motivation faible des équipes de développement. Ces dernières subissent une pression selon des urgences qu'elles ne comprennent pas. Si le mode urgence est instauré, la communication entre les divers acteurs du projet censés travailler ensemble est faible, voir inexistante.

Le Planning Game : un outil de responsabilisation

Il s'agit d'un instrument de régularisation du rythme de réalisation. Il consiste à définir la liste des fonctionnalités à développer pendant la prochaine itération (le " sprint backlog "). Elle est évaluée sur la base d'une charge de travail soutenable, définie par l'ensemble de l'équipe, et mesurée en " complexité ". Le système de planification collaborative responsabilise l'équipe de développement et la motive davantage à tenir un engagement qu'elle a pris elle-même. Au

contraire, le planning estimé et imposé par le management est souvent subi par l'équipe de développement, qui se désolidarise des objectifs qu'elle estime loin de la réalité du terrain. Adoptée en début de projet ou intégrée dans un projet en cours, cette pratique est appréciée par les développeurs, qui ont l'opportunité nouvelle de remonter régulièrement les retours " du terrain ". La mécanique met en évidence la capacité réelle de production.

L'usine de développement, les tests et la qualité logicielle

Les principes de l'Agilité s'appuient sur des pratiques (le TDD, l'intégration continue, etc.) à outiller. Pour cela, l'installation de l'usine de développement est une action clé du démarrage d'un projet.

Qu'est ce que la " Software Factory " doit inclure ? Tout d'abord il faut définir la solution d'intégration continue pour automatiser le build. Le monde Unix le fait depuis très longtemps, la communauté Java a développé Ant et celle .net, Nant et MSBuild. L'automatisation permet d'implémenter deux pratiques de l'eXtreme Programming : le " commit " quotidien et la construction d'un build " auto-testable ". L'essor du TDD (test driven development) a permis de créer des techniques d'intégration des tests unitaires dans le code. Les serveurs d'intégration peuvent gérer les tests automatisés du monde Java (JUnit) et .net (NUnit), la construction du build comprend l'exécution de ces tests. Parmi les plates-formes d'intégration continue les plus utilisées, citons Hudson, Cruise Control, Continuum. Dans la démarche agile, le contrôle de la qualité logicielle passe aussi par l'automatisation des tests fonctionnels (recette). La " XU family " met à dis-

position divers outils open source à cet effet : FitNesse, Selenium, Sahi, Greenpepper, etc. Les serveurs d'intégration continue peuvent aujourd'hui intégrer des outils d'automatisation de la recette, mais pour cela il faut tenir compte que la machine de tests automatisés peut être utilisée seulement si l'architecture de l'application est compatible avec le codage des tests unitaires.

Pour un projet à concevoir, il est probable qu'il y a peu ou pas de contraintes d'architecture. Pour un projet en déroulement, la revue du code, l'ajout des tests unitaires, et l'installation d'une nouvelle configuration de développement, peuvent s'avérer complexes. Selon le contexte réel du projet, les réponses aux questions suivantes sont déterminantes : l'application a-t-elle été conçue pour être testable ? Le " product owner " a-t-il la possibilité d'assumer la définition des tests fonctionnels automatisés ? L'avancement du projet permet-t-il d'absorber l'investissement nécessaire pour le développement de tests de tout type ? Plus le projet est avancé, plus l'effort de " refactoring " du code existant - le remboursement de la " dette technique " - risque d'être grand. Parfois l'écart entre l'existant et la cible est tellement important que le " refactoring " devient plus coûteux que la refonte de l'application. Dans ce cas extrême, devant un logiciel vraisemblablement non maintenable, une revue des objectifs s'impose.

Le travail collaboratif et l'amélioration continue

La mise en œuvre des pratiques liées à l'amélioration continue est une autre action clé d'un projet Agile. L'Agilité s'appuie massivement sur " l'intelligence du groupe ". La " programmation en binôme " (coder à deux), la " programmation par l'équipe "

(toute l'équipe se met autour de la table pour programmer), les tests croisés (les membres de l'équipe testent réciproquement le code) sont des pratiques qui s'appuient sur la progression par le retour des pairs.

Autres pratiques importantes qui facilitent l'échange : les points d'analyse commune comme le " Stand-up Meeting ", mise au point quotidienne d'une durée de 10 minutes environ, et la " rétrospective ", réunion régulière (par exemple en fin de chaque itération) qui permet d'analyser le travail d'équipe et d'identifier les axes d'amélioration. Bien sûr, ces activités demandent un investissement de temps que parfois ni l'équipe ni les managers ne sont prêts à faire, mais dont à l'usage, l'ensemble des acteurs reconnaît les bénéfices. Dans le cas d'un projet qui démarre, définir ce type de fonctionnement fait partie du " plan d'assurance qualité " du projet et il sera accepté, même si la réticence sur le " temps passé en réunion " par l'équipe de développement reste présente. A l'opposé, le changement d'un fonctionnement connu, impliquant la participation à des " nouvelles réunions ", est plus difficile à assumer. L'équipe et les leaders ne sont pas prêts à renoncer au mode de travail habituel, il a un côté rassurant, car connu, même si un accord existe sur le besoin d'accroître son efficacité. Pour ces raisons, il semble judicieux de rendre agile un projet en mode " échantillon " plutôt qu'en mode " Big-bang ". Autrement, un phénomène de rejet peut s'installer et mettre en péril la bascule Agile.



■ Oana Juncu
Direction
de Projets - SFEIR
www.sfeir.com

Abonnez-vous à **e-PROgrammez!** Le magazine du développement

2,7 € seulement par numéro au format PDF (pour le monde entier)

et pour 0,84 € de plus par mois : abonnement illimité aux archives
(numéros du magazine, et articles, en PDF)

Abonnez-vous sur www.programmez.com

Formations certifiantes en informatique & management

- ✓ **Certifications comprises** avec toutes nos formations
- ✓ **Ordinateur portable** offert avec les supports
- ✓ **Abonnements** : *L'Informaticien, Hakin9, Management, Programmez, Solutions & Logiciels...* offerts !
- ✓ **30 jours de coaching** (support technique)
- ✓ Formations éligibles **DIF, FONGECIF, OPCA...**
- ✓ Garantie **“Enchanté ou Invité”**
- ✓ **Votre carte personnelle d'accès à SMARTcenter** et à la SMARTLibrary (accès à vie à tous les supports de formation EGILIA)
- ✓ **Certificate of Excellence** en fin de formation...



Paris Lyon Lille Aix-en-Provence Strasbourg
Bordeaux Toulouse Rennes Bruxelles Genève

Découvrez les
nombreux avantages
sur www.egilia.com !

CONTACTEZ NOS CONSEILLERS FORMATION

 **N° National 0 800 881 558**

APPEL GRATUIT DEPUIS UN POSTE FIXE

www.egilia.com

Bien débuter en Silverlight

Silverlight gagne du terrain dans le domaine de la création d'applications internet riches et devient une technologie incontournable.

La simplicité et son intégration avec la plate-forme .NET représentent les principales raisons de son adoption par un grand nombre de développeurs à travers le monde. De plus, on ne manque pas d'outils, il y a des environnements adaptés aux besoins spécifiques des développeurs et des designers, et des mécanismes de collaboration fluides.

Silverlight 2.0 partage la technologie de WPF en se basant sur le langage XAML " Extensive Application Markup Language ". Le langage XAML est un dialecte XML qui peut être utilisé pour spécifier de manière déclarative l'interface utilisateur d'une application Silverlight ou WPF. XAML peut également servir plus largement pour représenter des objets .NET.

Silverlight reprend aussi le modèle ASP.NET en intégrant les fichiers de classe " Code-behind " pour gérer la séparation du code et la présentation. Ces fichiers peuvent être écrits en plusieurs langages dont C#, VB.NET, Python et Ruby.

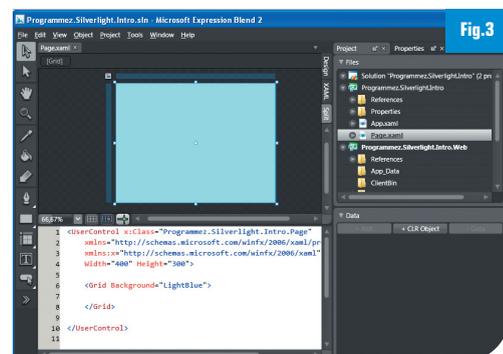
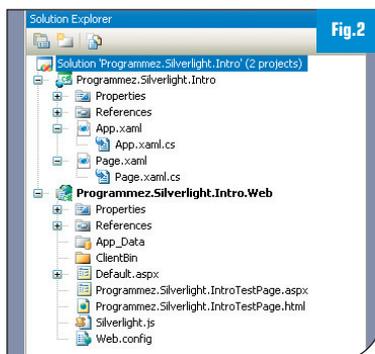
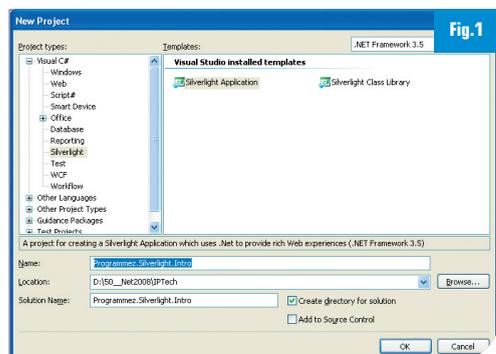
Préparez l'environnement de développement

- 1- **Visual Studio 2008 SP1** ou **Visual Web Developer Express with SP1** (gratuit).
- 2- **Silverlight Tools for Visual Studio 2008 SP1** : extension pour Visual Studio 2008 SP1 ou Visual Web Developer Express with SP1 qui va mettre à jour Visual Studio.
Ajouter le support des projets Silverlight, les runtime de développement, et le SDK.
- 3- **Microsoft Expression Blend 2**. Et son Service Pack 2 : Un nouvel environnement de développement orienté designers qui permet de créer des interfaces graphiques pour les applications Silverlight 2.
- 4- **Deep Zoom Composer** : outil permettant de préparer vos images pour les utiliser avec la fonctionnalité Deep Zoom.
- 5- **Silverlight Toolkit** : projet open source Microsoft project contenant des contrôles et des composants Silverlight pour créer des applications.

Créez votre premier projet Silverlight

Nous allons utiliser Visual Studio pour créer notre première application et commencer à explorer Silverlight : [Fig.1]

Par défaut, un projet d'application Silverlight nouvellement créé contient un fichier Page.xaml et App.xaml, ainsi que les fichiers de classe code-behind (qui peuvent être écrits en VB, c#, Ruby ou Python) qui y sont associés : [Fig.2]



Le fichier App.xaml est généralement utilisé pour déclarer des ressources, comme les pinceaux (Brush) et les styles qui sont partagés entre les composantes de l'application. La classe code-behind du fichier App.xaml peut être utilisée pour gérer les événements de niveau application - comme Application_Startup, Application_Exit et Application_UnhandledException.

Après compilation, le projet est transformé en fichiers ".xap" associés à des assemblies .Net standard. Les fichiers ".xap" utilisent l'algorithme de compression zip standard pour réduire la taille de téléchargement de client.

Pour héberger et exécuter une application Silverlight 2, vous pouvez ajouter une balise <object> dans toute page de HTML standard qui pointe vers le fichier .xap. Ce fichier sera alors automatiquement téléchargé et instancié par le plugin Silverlight du navigateur :

```
<div id="silverlightControlHost">
  <object data="data:application/x-silverlight,"
    type="application/x-silverlight-2"
    width="100%" height="100%">

    <param name="source"
      value="Programmez.Silverlight.Intro.xap"/>

  </object>
</div>
```

Ajoutez des contrôles utilisateur avec Blend

Vous pouvez charger le projet créé avec Visual studio avec Expression Blend : [Fig.3]

Silverlight offre trois possibilités pour l'agencement des éléments d'interface :

- **Le Canvas** : Permet le positionnement explicite des contrôles en utilisant des coordonnées.
- **Le StackPanel** : Permet le positionnement horizontal ou vertical d'un groupe de contrôles.
- **Grid** : Similaire aux tables Html. Permet le positionnement en multi-lignes et multi- colonnes.

Nous allons utiliser Blend pour créer une grille avec trois lignes et trois colonnes. Pour cela il suffit de la zone d'édition graphique du concepteur. L'étape suivante sera de glisser-déposer des contrôles utilisateurs sous Blend. Et le code XAML associé généré par Blend : (voir listing A)

Retourons dans Visual Studio pour ajouter un peu de logique C# à notre application. Ajoutons un gestionnaire d'évènement click au bouton btnSearch : (voir listing B)

(listing A)

```
<Grid Background="LightBlue">
  <Grid.ColumnDefinitions>
    <ColumnDefinition Width="0.247**"/>
    <ColumnDefinition Width="0.533**"/>
    <ColumnDefinition Width="0.22**"/>
  </Grid.ColumnDefinitions>
  <Grid.RowDefinitions>
    <RowDefinition Height="0.265**"/>
    <RowDefinition Height="0.148**"/>
    <RowDefinition Height="0.587**"/>
  </Grid.RowDefinitions>

  <TextBlock Margin="5,5,5,5" Grid.Column="1"
    TextWrapping="Wrap"
    HorizontalAlignment="Stretch"
    VerticalAlignment="Center"
    FontSize="11" FontFamily="Verdana">
    <Run FontFamily="Verdana" FontSize="18"
      FontWeight="Bold" Foreground="#FFC24444"
      TextDecorations="Underline"
      Text="Recherche Utilisateurs"/>
  </TextBlock>

  <TextBox Margin="8,8,8,8" Grid.Column="1"
    Grid.Row="1" Text="" TextWrapping="Wrap"
    x:Name="txtSearch"/>

  <TextBlock Margin="8,8,8,8" Grid.Row="1"
    TextWrapping="Wrap" FontSize="12">
    <Run FontSize="12" FontWeight="Bold" Text="Nom : "/>
  </TextBlock>

  <Button Margin="8,8,8,8" Grid.Column="2"
    Grid.Row="1" Content="Ok" x:Name="btnSearch"/>
  <TextBlock Margin="8,8,8,0" TextWrapping="Wrap"
    Height="28" x:Name="Résultats"
    VerticalAlignment="Top" Grid.Row="2"
    Text="Résultats : " FontWeight="Bold"
    FontFamily="Portable User Interface"
    FontSize="12"/>
  <ListBox Margin="8,5,8,8,8" Grid.Column="1"
    Grid.Row="2" x:Name="lstResults"/>
</Grid>
```

(listing B)

```
<Button Margin="8,8,8,8" Grid.Column="2"
  Grid.Row="1" Content="Ok" x:Name="btnSearch"
  Click="btnSearch_Click"/>
```

Et le code de la méthode btnSearch_Click :

```
13 namespace Programmez.Silverlight.Intro
14 {
15     public partial class Page : UserControl
16     {
17         string[] users = { "Scott Guthrie",
18                           "Bill Gates",
19                           "Steve Ballmer",
20                           "Scott hanselman"};
21
22     public Page()
23     {
24         InitializeComponent();
25     }
26
27     private void btnSearch_Click(object sender,
28     RoutedEventArgs e)
29     {
30         lstResults.Items.Clear();
31         foreach (string userName in users)
32         {
33             if (userName.Contains(txtSearch.Text))
34             {
35                 lstResults.Items.Add(userName);
36             }
37         }
38     }
39 }
40 }
```

Le Code C# utilisé est des plus simples, mais l'objectif est de monter l'accessibilité des contrôles Silverlight à partir du code et la facilité de leur manipulation (exactement identique à un développement WinForms ou ASP.NET).

Utilisez les styles

Silverlight offre des fonctionnalités de style similaires aux feuilles CSS permettant d'encapsuler les valeurs de propriétés de contrôles comme une ressource réutilisable. Ces styles peuvent être stockés dans des fichiers séparés et seront utilisés dans l'intégralité de notre application. Ajoutons un style au fichier App.xaml : (voir listing C) La définition de ce style " Label " peut être référencée dans le reste de notre application en utilisant la propriété style : (voir listing D)

(listing C)

```
<Application.Resources>
  <Style x:Key="Label" TargetType="TextBlock">
    <Setter Property="FontSize" Value="12">
    </Setter>
    <Setter Property="FontWeight" Value="Bold">
    </Setter>
    <Setter Property="TextDecorations" Value="Underline">
    </Setter>
    <Setter Property="Foreground" Value="Navy">
    </Setter>
    <Setter Property="Margin" Value="8,8,8,8">
    </Setter>
  </Style>
</Application.Resources>
```

(listing D)

```
<TextBlock Style="{StaticResource Label}" Grid.Row="1"
  TextWrapping="Wrap" Text="Nom : ">
</TextBlock>
```

Et le résultat :



Changez l'apparence des contrôles

Avec Silverlight la personnalisation de l'apparence standard des contrôles est un jeu d'enfant. L'une des approches est d'utiliser les styles pour redéfinir la propriété " Template " qui définit l'apparence standard d'un contrôle (le bouton par exemple) :

```
<Style x:Key="RoundButton" TargetType="Button">
  <Setter Property="Template">
    <ControlTemplate>
      <Border x:Name="brd1" Width="60"
        Height="28" CornerRadius="15">
        <TextBlock Foreground="#222"
          TextAlignment="center"
          Text="Ok" FontSize="11"
          VerticalAlignment="center"
          FontFamily="Arial"/>
      </Border.Background>
      <RadialGradientBrush
        GradientOrigin=".3, .3">
        <GradientStop
          Color="#FFF" Offset=".15"/>
        <GradientStop
          Color="#777" Offset="1"/>
      </RadialGradientBrush>
      </Border.Background>
    </Border>
  </ControlTemplate>
</Setter.Value>
</Style>
```

Ajoutons la référence de ce style à notre bouton :

```
<Button Margin="8,8,8,8" Grid.Column="2"
  Style="{StaticResource RoundButton}"
  Grid.Row="1" Content="Ok" x:Name="btnSearch"
  Click="btnSearch_Click"/>
```

Et le tour est joué :



Conclusion

Le précédent exemple nous a permis de faire une rapide introduction au développement Silverlight et à ses outils. Vous pourrez trouver des ressources complémentaires sur le site de Silverlight <http://silverlight.net/> et d'autres contrôles utilisateur plus avancés dans le Silverlight Toolkit : <http://www.codeplex.com/silverlight>

■ Hassen HARRATH

Spécialiste .NET chez IP-Tech, SSI Tunisienne spécialisée dans les développements informatiques pour le marché Français (www.ip-tech-offshore.com).

Microsoft Expression Studio 3

Dans cet article, nous allons passer en revue les principales nouveautés annoncées par Microsoft durant le MIX09 qui s'est tenu du 18 au 20 Mars dernier à Las Vegas. Lors de cet évènement, le focus avait été mis notamment sur Expression Web 3 et Expression Blend 3. Pour plus d'informations sur les éléments vus ou cités, je vous invite à vous reporter aux liens disponibles à la fin de cet article.

Expression Web 3

Mix 09 ne nous aura pas beaucoup plus éclairés sur les nouveautés qui seront intégrées dans la prochaine mouture d'Expression Web. On peut néanmoins s'attendre à des améliorations sur le support des CSS dans la fenêtre de Design. Silverlight 2 et MVC devraient venir aussi s'ajouter à la liste des technologies supportées par Expression Web. Il a également été dit que le support de systèmes de gestion de versions comme Microsoft Visual SourceSafe, ou plutôt Team Foundation Server, n'était pas prévu à l'heure actuelle dans la version 3 d'Expression Web. L'une des raisons étant que la priorité a été donnée au support des technologies citées plus haut. Mais rien n'empêche Microsoft de revenir sur certains éléments avant sa sortie en version finale. Contre toute attente et malgré le fait qu'ils aient été fortement sous-entendus lors des sessions, ce n'est pas ce genre de nouveautés qui ont été dévoilées lors de l'évènement de Las Vegas. Outre le fait de revenir sur des possibilités déjà existantes (affichage simplifié des CSS, utilisation de Seadragon Ajax dans votre site Web, etc.), Microsoft a plutôt insisté sur un nouvel outil qui fera son apparition dans Expression Web 3 : SuperPreview.

Super Preview : un afficheur multi-navigateur

L'une des principales nouveautés mise en avant par Microsoft sur la prochaine version d'Expression Web est sans aucun doute Super Preview. Pour expliquer rapidement cette nouvelle fonctionnalité incluse dans Expression Web 3, on pourrait dire grossièrement qu'il s'agit d'un afficheur multi-navigateur de site Internet, accompagné de fonctions de débogage très utiles. Super Preview ne vient pas remplacer la fenêtre de Design d'Expression Web, mais c'est une fonctionnalité qui peut être utilisée en parallèle avec celle-ci. Examinons donc plus précisément ce que nous pourrions faire avec [Fig.1].

Aperçus sous divers navigateurs locaux ou distants

Tout d'abord, comme dit plus haut, celui-ci permet d'avoir un affichage des pages d'un site Internet sous différents moteurs de navigateur Web. Une des principales qualités de Super Preview est d'embarquer avec lui une version portable d'Internet Explorer 6. En effet, cela permet de bénéficier des éléments de débogage que propose Super Preview pour les appliquer sur un site Internet sous Internet Explorer 6. Donc, si vous

avez installé une version plus récente d'Internet Explorer sur votre machine - puisque celle-ci fonctionne comme une mise à jour Windows-, vous avez désormais la possibilité avec ce nouvel outil d'avoir un aperçu de votre site Internet depuis Super Preview sous le moteur de rendu d'Internet Explorer 6.

Bien sûr pour l'accompagner, vous pouvez intégrer d'autres navigateurs Web dans Super Preview [Fig.2]. Dans un premier temps, vous avez à disposition les navigateurs installés sur votre ordinateur (Internet Explorer 7 ou 8, Firefox et Safari). Mais il est également possible de pouvoir relier Super Preview avec des navigateurs présents sur d'autres machines - comme un Mac- pour bénéficier de leur aperçu dans celui-ci. La navigation entre ces prévisualisations se fait d'une manière instantanée après le chargement de la page du site Internet. Il est possible aussi de faire une comparaison avec une image de votre site Internet, pour correspondre au mieux avec votre maquette graphique.

Quelques outils en plus

Super Preview met également à votre disposition quelques éléments de débogage. Pas de quoi le comparer

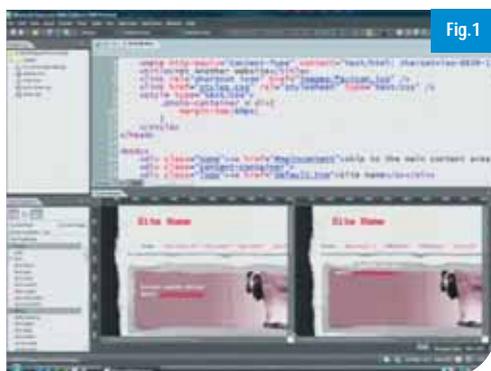


Fig.1

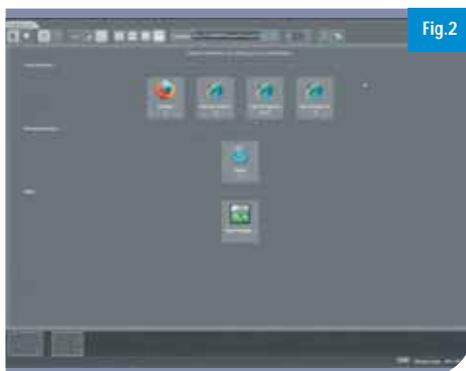


Fig.2

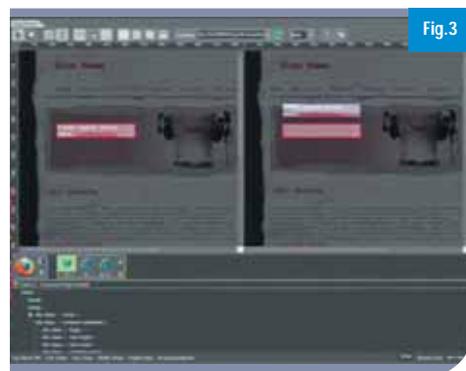


Fig.3

Testez facilement les performances de vos applications Web
et Déployez en toute confiance

DÉCOUVREZ LA VERSION 3.6



Développement automatisé
de Cas-Tests et Test de Charge

Rapports d'analyse des Tests
de performance et de charge

Analyse des Serveurs et de
l'impact sur les performances

Interface utilisateur et Rapports
en Français

Evaluation gratuite sur <http://www.kapitec.com/Pub/WP?id=120>
Bénéficiez de notre assistance technique pendant les 15 jours d'évaluation.

équitablement avec Firebug ou le débogueur intégré dans Internet Explorer 8, mais il reprend néanmoins les bases de ces derniers. La sélection d'éléments, pour les comparer entre eux, peut se faire directement depuis les prévisualisations ou encore depuis un arbre DOM du site Internet visualisé [Fig.3].

En effet, Super Preview ne fait pas qu'afficher une prévisualisation, il en reprend aussi les informations générales HTML. En sélectionnant un élément sur la prévisualisation, on retrouve des informations comme : son type de balise, sa position, ses dimensions, sa classe CSS ou son ID directement depuis la barre d'état.

Parmi ces outils, on peut noter la possibilité d'ajouter une règle autour des visualisations ou encore d'ajouter des repères de la même façon que sur Photoshop ou Expression Design. SuperPreview nous donne la possibilité de visualiser notre site Internet sous différentes méthodes. Cela peut se faire côte à côte, l'une en-dessous de l'autre ou via une superposition des prévisualisations.

Quelques limitations également

Dans cette version bêta de Super Preview, on sent tout de même qu'il reste encore beaucoup à faire pour la version finale.

Comme indiqué précédemment, seuls Firefox et Safari sont pris en compte par Super Preview. Qu'en est-il alors d'Opera ou encore de Chrome ? Microsoft promet néanmoins qu'ils seront intégrés prochainement dans cette fonctionnalité.

Un autre point gênant reste que les filtres et transitions ne sont pas disponibles sur le moteur d'Internet Explo-

rer 6 présent dans Super Preview. Par exemple, cela ne permet pas de traiter les PNG transparents avec DXImageTransform. On peut également noter que certains éléments HTML, comme les balises object, embed et applet, peuvent ne pas être affichés avec les bonnes dimensions si IE8 n'est pas installé. Une autre utilisation de Super Preview, qui aurait pu être d'une très grande utilité, aurait été le support de moteurs de rendu de messagerie mail comme Outlook 2007.

En effet, Microsoft n'utilise plus le moteur d'Internet Explorer pour afficher les mails HTML dans Outlook 2007, ce qui peut générer quelques soucis d'affichage avec une structure HTML/CSS classique. Il faut cependant souligner que cette version est une Bêta et que des modifications sont encore à venir. Il s'agit là de quelques limitations, mais qui ne devraient pas empêcher Super Preview de devenir un outil aussi incontournable que Firebug ou encore le débogueur d'Internet Explorer 8.

Expression Blend 3

Les nouveautés annoncées pour Blend sont, quant à elles, beaucoup plus nombreuses que pour Expression Web 3. Outre une interface légèrement revue mais qui ne changera en rien vos habitudes, on peut souligner la présence d'éléments comme le support de Visual Studio Team Foundation Server ou encore l'ajout de nouveaux éditeurs XAML, C# et VB, avec notamment la présence ou l'amélioration de l'IntelliSense pour ces langages. Ceci évite désormais l'emploi de Visual Studio pour profiter de ces fonctionnalités avec ces langages-là.

SketchFlow : Donner vie à vos concepts et vos prototypes

SketchFlow vous permet avec une seule et même application intégrée à Expression Blend 3 de réaliser vos maquetages ou prototypages. En effet, ce nouvel outil permet aux designers de dessiner des esquisses, les mettre en forme, les simuler, leur donner vie pour enfin les déployer, et cela d'une façon beaucoup plus interactive que via PowerPoint ou d'autres méthodes.

Avec SketchFlow vous avez la possibilité de donner de l'interactivité à vos croquis de prototypes car celui-ci permet à tout moment d'y ajouter des contrôles WPF. Pour faire simple, il permet la même chose que ce que vous pouvez faire dans un projet WPF normal. Et afin de rester dans l'esprit de prototype et de maquetage, ces contrôles ont droit à une apparence spécifique se rapprochant d'une esquisse [Fig.4]. Cela permet de se focaliser principalement sur le fonctionnement de votre application plutôt que de s'attarder sur des détails graphiques. Néanmoins, il vous est à tout moment possible de basculer d'apparence depuis les propriétés de vos contrôles.

Vos esquisses peuvent être lancées comme une application WPF depuis le SketchFlow Player [Fig.5]. Ce lecteur, inclus automatiquement dans votre projet de maquetage, donne la possibilité à d'autres personnes de naviguer facilement entre vos différents écrans. Si vous le souhaitez, des animations peuvent être jouées depuis ce lecteur. Cela permet de se faire une idée sur la façon dont les éléments interagissent entre eux. SketchFlow Player permet aussi à ses

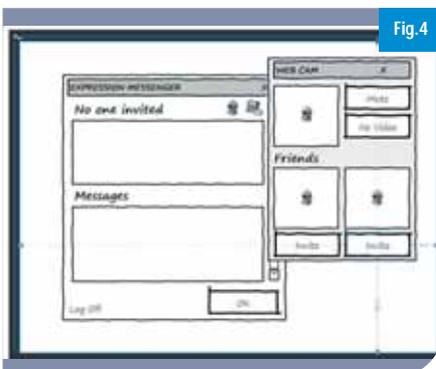


Fig.4

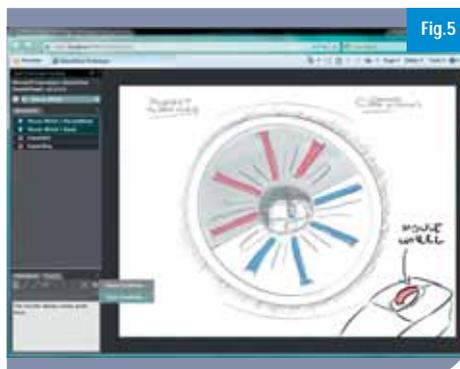


Fig.5



Fig.6

utilisateurs d'ajouter des commentaires ou des annotations sur les croquis de vos écrans. Une fois enregistrés sous forme de fichier, vous pouvez ouvrir ces commentaires et annotations dans votre projet Expression Blend 3.

SketchFlow permet d'associer facilement des données à vos maquettes pour leur fournir plus de contenu. Ces données et leur modèle d'affichage sont éditables de la même manière qu'un projet WPF de base. Il est intéressant de noter que votre projet SketchFlow est exportable sous forme de fichier Word. Durant l'opération d'export, Expression Blend 3 vous crée un document complet comprenant une table de contenu, des images de vos écrans, les annotations associées à ceux-ci, etc.

Créer des comportements et des composants sans code

Expression Blend 3 introduit désormais une bibliothèque de comportements de base pour vos contrôles. Cela permet notamment au graphiste de ne pas avoir besoin d'un développeur pour créer des interactions utilisateurs. Néanmoins, un développeur aura toujours la possibilité de créer des comportements supplémentaires pour que le graphiste puisse les utiliser de son côté depuis sa bibliothèque.

Cette nouvelle version d'Expression Blend permet aussi une création de composants beaucoup plus aisée. En effet d'un simple clic, vous avez désormais la possibilité de créer des composants à partir d'éléments graphiques existants. Il introduit par la même occasion la notion de *ControlParts* : Vous pouvez attribuer un élément graphique comme une partie de contrôle. Si on prend comme exemple le Slider, vous pouvez facilement définir un élément graphique existant comme curseur.

De ce fait, une nouvelle fenêtre fait son apparition pour vous permettre de choisir simplement un élément comme étant une partie du composant. Cette fenêtre vous donne accès à une liste prédéfinie du composant que vous souhaitez créer. L'élément

choisi adoptera de cette manière un comportement logique associé à la partie du composant sélectionnée.

Importer un fichier Adobe Photoshop ou Illustrator

Une autre nouveauté qui a déjà pu faire son apparition en partie dans la version 2 d'Expression Web est l'import d'un fichier de type Adobe Photoshop ou Illustrator. A la différence qu'ici, son utilisation est beaucoup plus poussée et surtout beaucoup plus utile. Dans ce qui suit, je décrirai principalement l'import de fichier Photoshop car le fonctionnement de l'import de fichier Illustrator est globalement le même que l'import de fichier d'Expression Design.

Lorsque l'on importe un fichier Photoshop, Expression Blend 3 permet de sélectionner les calques à inclure dans les données que l'on désire ajouter à son projet [Fig.6]. Il est également possible de créer de nouveaux groupes d'éléments depuis l'arborescence de sélection des calques à importer. Pour ce qui est du respect des éléments Photoshop, Expression Blend 3 est assez généreux pour cette nouvelle fonctionnalité. En effet, il respecte les calques, leur arborescence, leur position, les bitmap, les formes vectorielles, les textes, les masques et les formes ayant un remplissage solide ou dégradé. Expression Blend 3 s'occupe aussi de générer le XAML qui correspond à ce que vous lui avez importé. Pour ce qui est des masques et des formes vectorielles, dès qu'ils sont importés dans votre projet, ils sont tout à fait modifiables à votre guise.

La manipulation des éléments Photoshop importés ne diffère pas des éléments déjà présents et il vous sera également possible de créer des composants directement depuis un élément ou un groupe d'éléments. Pour pallier au problème de la taille de pixels sous Adobe Photoshop (72 DPI) et Expression Blend (96 DPI), celui-ci vous propose de les redimensionner si nécessaire. Dans le cas où des problèmes de compatibilité se présentent, il est possible de sélectionner un calque qui fusionne l'en-

semble des éléments Photoshop sous sa bonne version. Ceci étant dû au fait que certains éléments ne sont pas encore supportés, comme par exemple l'opacité d'un élément ou les effets de Photoshop. Le but de Microsoft étant que le support des éléments d'un fichier Photoshop soit total.

Une version Bêta disponible

Pour couronner le tout et pour un poids de plus ou moins 289.4 MB, vous avez désormais la possibilité de vous faire une idée de ces nouveautés avec la version Preview d'Expression Web 3 dont le lien est disponible à la fin de cet article. Néanmoins, veuillez noter quelques petites remarques avant de vous ruer sur cette version Preview !

Tout d'abord et le plus important à mon sens, si vous ouvrez un projet Silverlight 2 dans Expression Blend 3 Preview, celui-ci vous demandera à être converti en projet Silverlight 3. Une fois ceci fait, vous ne pourrez plus revenir en arrière.

Prenez en compte aussi le fait que SketchFlow n'est pas disponible dans cette version Preview. Donc, si vous comptiez voir d'un peu plus près cette fonctionnalité, il vous faudra prendre votre mal en patience !

Ressources

- Internet Explorer 8 : <http://www.microsoft.com/ie8/>
- Seadragon Ajax : <http://livelabs.com/seadragon-ajax/>
- Silverlight 3 Bêta :
 - <http://silverlight.net/getstarted/silverlight3/default.aspx>
- Expression Blend 3 Bêta : <http://www.microsoft.com/expression/try-it/blendpreview.aspx>
- Expression Studio 2 (Evaluations disponibles) : <http://www.microsoft.com/expression/>
- Sessions du MIXO9 sur Expression : <http://videos.visitmix.com/MIXO9/Tags/Expression>

■ Axel Peter

Intégrateur .NET & Web Designer
<http://www.wygwam.com>
<http://blogs.codes-sources.com/chronos>

Silverlight 3 envahit aussi le Bureau !

Alors que Silverlight 2 ne fonctionnait qu'en mode connecté (en ligne) et à l'intérieur d'un navigateur, la version 3 de la plate-forme ne respecte plus ces limitations et débarque (enfin) sur le bureau, à l'instar d'un Adobe AIR.

Dans cet article nous allons apprendre à :

- Rendre une application Silverlight 3 téléchargeable
- Détecter si l'application est :
 - Installée sur le bureau ou sous navigateur
 - Online ou offline.
- Récupérer des données à partir du site hôte de l'application

Avant de commencer, notez que les pré-requis pour cet article sont :

- Connaître les bases de la plate-forme .Net
 - Installer Visual Studio 2008 (dont la version express est gratuite)
- Attention, installez la version anglaise, les Tools et SDK existent seulement en anglais pour l'instant !
- Installer les Silverlight 3 Tools et SDK ainsi que les .Net RIA services (<http://www.silverlight.net/GetStarted/silverlight3>)

Bases de l'application

- **Démarrez Visual Studio 2008**
- Ouvrez le menu **File -> New Project ...**
- Dans la boîte de dialogue " **New Project** " qui apparaît, choisissez " **.Net Framework 3.5** " et sous le type de projet " **Silverlight** " : " **Silverlight Application** " [Fig.1].
- N'oubliez pas de lui donner un nom, dans ce cas-ci : " **OutOfBrowser** "
- Cliquez sur **Ok**.
- Une nouvelle boîte de dialogue apparaît : " **New Silverlight Application** ". Prenez soin de cocher l'option " **Link to ASP.Net server project** ". Cette option activera le lien .Net RIA Services entre l'application Silverlight et son site ASP.NET host [Fig.2].
- Cliquez sur **Ok**.
- Visual Studio va générer automatiquement toute la structure de la solution.
- Dans le fichier **MainPage.xaml** du projet Silverlight, ajoutez un texte " **Hello World** ".

```
<Grid x:Name="LayoutRoot" Background="White">
    <TextBlock>HelloWorld</TextBlock>
</Grid>
```

Voici le résultat obtenu lors d'un débogage de la solution créée ci-dessus : [Fig.3]

Rendre cette application téléchargeable

Grâce au bouton fourni par la plate-forme

Permettre à vos utilisateurs d'installer votre application sur leurs machines se fait dans son fichier manifeste. Ce fichier se trouve dans le dossier " **Properties** " de votre application. [Fig.4]. Pour procéder, décommentez le nœud " **Deployment.ApplicationIdentity** ".

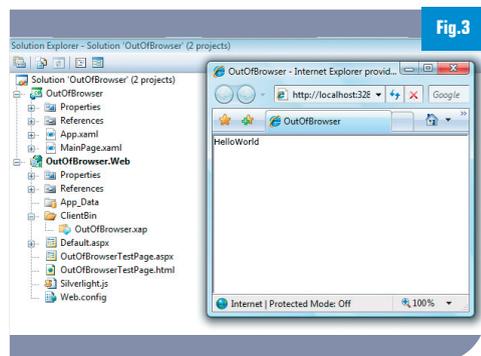
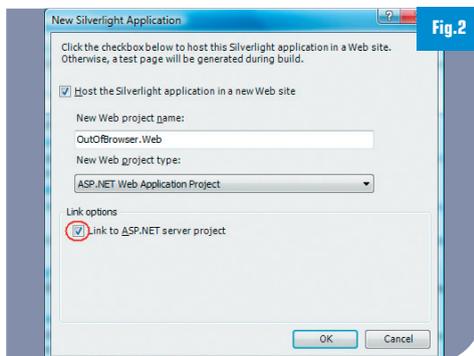
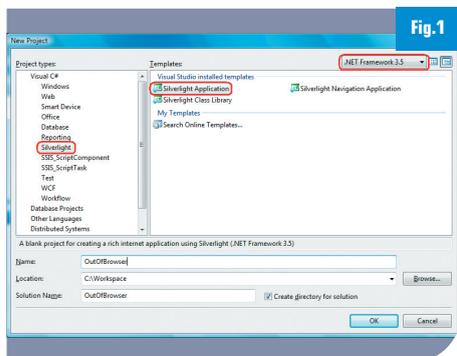
```
<Deployment xmlns="http://schemas.microsoft.com/client/2007/
deployment"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
>
    <Deployment.Parts>
    </Deployment.Parts>

    <!-- Uncomment the markup and update the fields below to make
your application offline enabled -->
    <Deployment.ApplicationIdentity>
        <ApplicationIdentity
            ShortName="Out of Browser Silverlight Application"
            Title="Window Title of Your Silverlight Application">
            <ApplicationIdentity.Blurb>Description of your
Silverlight application</ApplicationIdentity.Blurb>
        </ApplicationIdentity>
    </Deployment.ApplicationIdentity>
</Deployment>
```

Ce nœud contient 3 paramètres d'origine :

- **ShortName** : est le nom de l'application, c'est ce nom qui sera utilisé pour les icônes du bureau, les raccourcis, etc.
- **Title** : est le titre de l'application, ce titre sera utilisé en en-tête de fenêtre par exemple.
- **ApplicationIdentity.Blurb** : est la présentation de votre application. Cependant, un autre nœud est intéressant à ajouter : le nœud " **ApplicationIdentity.Icons** "

```
<ApplicationIdentity.Icons>
    <Icon Size="16x16">Icons/icon_16.png</Icon>
    <Icon Size="32x32">Icons/icon_32.png</Icon>
</ApplicationIdentity.Icons>
```



Développez 10 fois plus vite

WINDEV Mobile

DÉVELOPPER VOS APPLICATIONS POUR POCKET PC, SMARTPHONE & TERMINAL MOBILE : FACILE !



WINDEV Mobile 14 est l'environnement de développement professionnel qui permet de développer jusqu'à **10 fois plus vite** les applications sur mobile dont votre entreprise et vos clients ont besoin: gestion de stock, force commerciale, géolocalisation, saisies médicales, expertises, relevés de terrain, prise de commande temps réel, réglage de chaîne de production, etc...

La **puissance** et la **facilité** de développement de WINDEV Mobile 14 permettent un développement complet en quelques journées.

L'environnement est livré complet, le déploiement des applications réalisées est **gratuit** sans redevances (base de données incluse).

Toutes les fonctionnalités d'un AGL professionnel sont offertes. Tous les aspects de la mobi-

lité sont gérés: base de données, accès direct, réplication, WiFi, Bluetooth, 3G, Internet, socket, ActiveSync, réseau, J2EE, SMS, RFID, lien avec votre S.I., **codes-barres automatiques...**

Vous aussi réalisez vos applications mobiles 10 fois plus vite... avec WINDEV Mobile 14.

(Logiciel **professionnel**, Support Technique gratuit*)

VERSION EXPRESS GRATUITE
Téléchargez-la !

Un tableau de bord en temps réel sur son téléphone? Facile !



VOTRE CODE EST MULTI-PLATEFORMES:
Windows, .Net, Java, PHP, J2EE, XML, Internet, Ajax, Pocket PC, SmartPhone, Client riche ...

Logiciel **professionnel**

DEMANDEZ LE DOSSIER GRATUIT

252 pages + DVD + Version Express incluse + 112 Témoignages.

Tél: **04.67.032.032** ou **01.48.01.48.88**
info@pcsoft.fr

Fournisseur Officiel de la Préparation Olympique

www.pcsoft.fr



```
<Icon Size="48x48">Icons/icon_48.png</Icon>
<Icon Size="128x128">Icons/icon_128.png</Icon>
</ApplicationIdentity.Icons>
```

Ces icônes doivent faire partie de la solution Silverlight 3 pour être intégrées au fichier .xap qui sera téléchargé par le client [Fig.5]. Un nouveau menu est maintenant disponible sur le clic droit de l'application : " Install 'ShortName' ... onto this computer "[Fig.6].

Rendre cette application téléchargeable

Grâce à un bouton personnel

Bien que la plate-forme Silverlight offre un bouton de base pour proposer le téléchargement et l'installation d'une application au client, cette interface n'est pas très instinctive pour l'utilisateur lambda.

Pour pallier ce problème, il est possible de créer votre propre bouton (ou autre action) " télécharger cette application ".

- Ouvrez **MainPage.xaml** dans le projet Silverlight.
- Ajoutez-y un bouton.

```
<Grid x:Name="LayoutRoot" Background="White">
  <StackPanel>
    <Button
      Name="DownAndInstallBtt"
      Content="Télécharger et installer cette application
sur mon pc"
      Margin="5"
      Click="Button_Click"/>
    <TextBlock
      Margin="5"
      Text="HelloWorld"/>
  </StackPanel>
</Grid>
```

- Ouvrez **MainPage.xaml.cs**
- Ajoutez la fonction " **Button_Click** " :

```
public MainPage()
{
  InitializeComponent();
}

private void Button_Click(object sender, RoutedEventArgs e)
{
  bool allowed = App.Current.Attach();
  if (!allowed)
  {
```

```
    MessageBox.Show("L'application n'a pas été téléchargée
sur le bureau.");
  }
}
```

Attention, la fonction `App.Current.Attach();` générera une exception si l'application a déjà été installée sur l'ordinateur.

Détecter si l'application est installée sur le bureau ou sous navigateur

Pour adapter le comportement de l'application à son environnement il est bon de savoir si elle s'exécute dans un navigateur ou si l'utilisateur vient de lancer la version qu'il a préalablement installée.

- Ouvrez **MainPage.xaml** dans le projet Silverlight.
- Ajoutez une nouvelle zone de texte et nommez-la " **WebOrDesk-TextBox** "

```
<Grid x:Name="LayoutRoot" Background="White">
  <StackPanel>
    (...)
    <TextBlock
      Name="WebOrDeskTextBlock"
      Margin="5"
      Text="HelloWorld"/>
  </StackPanel>
</Grid>
```

- Ouvrez **MainPage.xaml.cs**
- Dans le constructeur de la page, juste après l'initialisation des composants, ajoutez ce code :

```
public MainPage()
{
  InitializeComponent();
  if (App.Current.RunningOffline)
  {
    WebOrDeskTextBlock.Text = "Bureau";
  }
  else
  {
    WebOrDeskTextBlock.Text = "Navigateur";
  }
}
```

Le booléen `App.Current.RunningOffline` vous donne cette information. [Fig.8]. Accessoirement, il est aussi possible de rendre le bouton " télécharger et installer cette application sur mon pc " invisible lorsque c'est déjà le cas.

```
public MainPage()
{
  InitializeComponent();
  if (App.Current.RunningOffline)
  {
    WebOrDeskTextBlock.Text = "Bureau";
    DownAndInstallBtt.Visibility = Visibility.Collapsed;
  }
  else
```

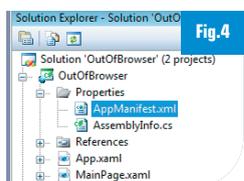


Fig.4

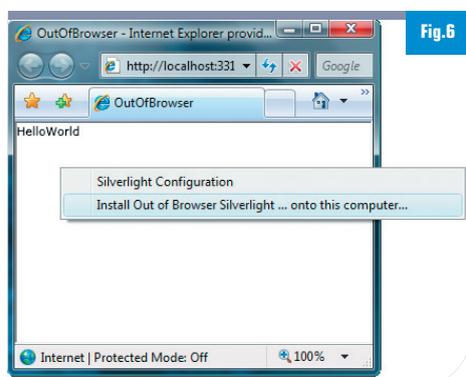


Fig.6

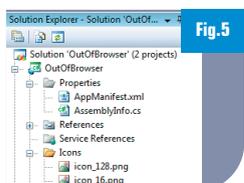


Fig.5

```

{
    WebOrDeskTextBlock.Text = "Navigateur";
    DownAndInstallBtt.Visibility = Visibility.Visible;
}
}

```

Détecter si le pc est online ou offline

Dans le même état d'esprit, il est utile de connaître l'état de la connexion internet de la machine sur laquelle tourne l'application.

Mais dans ce cas-ci, le travail est un peu plus complexe. En effet, cette connexion n'a pas un état statique, il est possible que son état change durant l'exécution de l'application. Un événement est envoyé à l'application par la plate-forme lorsque cela arrive. Il suffit donc de s'y abonner et de révérifier l'état de la connexion quand il est lancé.

- Ouvrez **MainPage.xaml** dans le projet Silverlight.
- Ajoutez une nouvelle zone de texte et nommez-la " **OnOrOff-TextBlock**"
- Ouvrez **MainPage.xaml.cs**
- Dans le constructeur de la page, juste après l'initialisation des composants, ajoutez ce code :

```

public MainPage()
{
    InitializeComponent();
    (...)

    OnOrOff();
    NetworkChange.NetworkAddressChanged +=
        new NetworkAddressChangedEventHandler(NetworkChange_
NetworkAddressChanged);
}

void NetworkChange_NetworkAddressChanged(object sender, Event
Args e)
{
    OnOrOff();
}

void OnOrOff()
{
    if (NetworkInterface.GetIsNetworkAvailable())
    {
        OnOrOffTextBlock.Text = "Online";
        OnOrOffTextBlock.Foreground = new SolidColorBrush
(Colors.Green);

```

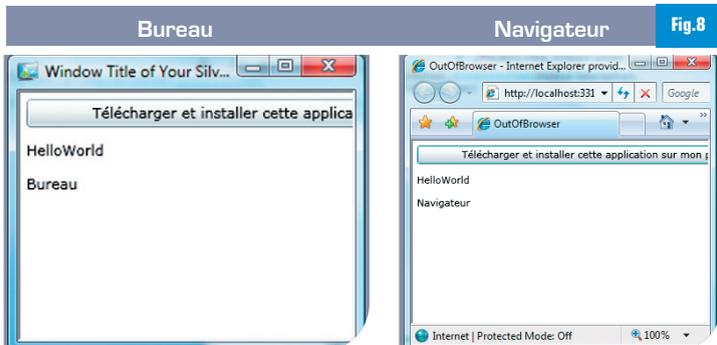
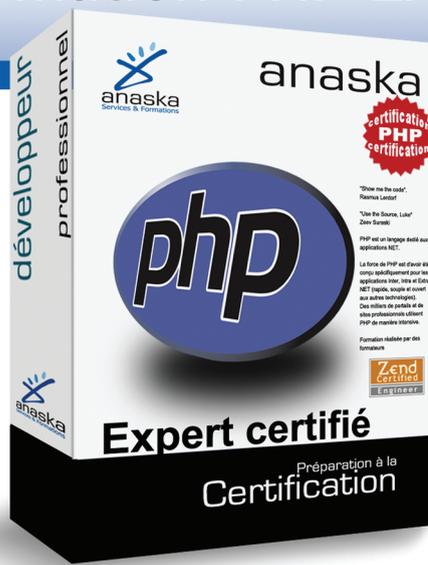


Fig.8

PRENEZ DE LA HAUTEUR

Web > PHP

Formation PHP Expert



"Montez en puissance sur PHP et préparez vous à passer la certification PHP"

Au Programme des 5 jours :

- * Certification PHP et rappels
- * La programmation orientée objet PHP 5
- * Les bases de données et PDO
- * XML et les services Web
- * Flux et réseau
- * Sécurité
- * Optimisation des performances
- * Debug et audit de code



Prochaines sessions

Paris 2009
8 Juin, 6 et 27 Juillet

Lyon 2009
6 et 27 Juillet

Tarif
1995 € HT

LE SPECIALISTE DE LA FORMATION POUR L'OPEN SOURCE

Informations

01 45 28 09 82

www.anaska.com

anaska

Alter Way GROUP



```
}
else
{
    OnOrOffTextBlock.Text = "Offline";
    OnOrOffTextBlock.Foreground = new SolidColorBrush
(Colors.Red);
}
}
```

Lorsque l'ordinateur client perd la connexion internet, l'application le remarque en quelques secondes.

Récupérer des données à partir du site hôte de l'application

Silverlight 3 et les .Net RIA Services simplifient grandement la vie fastidieuse des développeurs. Auparavant, pour consommer des données d'un serveur dans une application Silverlight, la méthode la plus couramment utilisée était de connecter cette application Silverlight à un service WCF (Windows communication foundation) lui-même correctement configuré pour que sa sécurité lui permette de dialoguer avec l'application. *gasp* Rien que de l'écrire, ça fait mal ! Aujourd'hui, le paradis nous ouvre ses portes. En effet, il est possible de créer une communication entre un site ASP.NET et une application Silverlight 3 qu'il héberge, et ce, en quelques clics, mais tout en gardant une grande liberté de configuration. Ainsi, la plate-forme nous offre de la simplicité, sans nous faire payer en généricité.

- Pour les besoins de la démo, créons rapidement une petite base de données des employés de Heode dont voici la structure : [Fig.9]
- Retournez dans **Visual Studio**
- Cliquez avec le bouton droit de la souris sur le projet "**OutOfBrowser.Web**" (le site web ASP.NET)
- Sélectionnez le menu Add new -> new Item ...
- Dans la boîte de dialogue "**New Item**", sélectionnez le **template** "LINQ to SQL Classes" et donnez lui le nom : **SuperHeroDB**.

Les classes LINQ to SQL sont un moyen simple et rapide de vous connecter à une base de données.

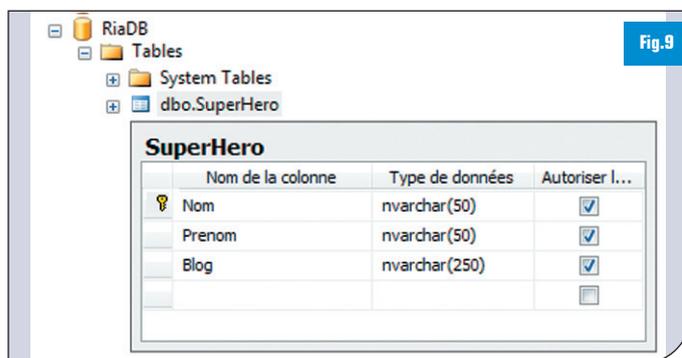


Fig.9

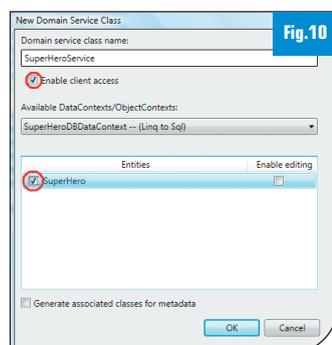


Fig.10

Les autres options vous permettent de demander à la plate-forme de générer automatiquement toute l'opération d'édition, d'insertion et de résolution de conflits, mais nous n'en aurons pas besoin dans cet exercice.

- Ouvrez le fichier SuperHeroDB.dbml
- Naviguez dans l'explorateur de serveur de Visual Studio jusqu'à trouver la table **SuperHero** et glissez-la sur la surface de dessin des classes **LINQ to SQL**.

- Compilez le projet ASP.NET
- Sélectionnez une fois de plus le menu **Add new -> new Item ...**
- Dans la boîte de dialogue "**New Item**", sélectionnez le **template** "**Domain Service**" et donnez lui le nom : **SuperHeroService**.

Un Domain Service est un service de données et d'opérations qui sera accessible au site web et à tous ses composants internes. C'est à travers ce service que le site web et l'application Silverlight vont communiquer.

- Cliquez sur Ok.
- La boîte de dialogue suivante vous demande de sélectionner à quel accès aux données vous voulez lier le Domain Service. Choisissez **SuperHeroDB**, ensuite cochez l'option devant la table **SuperHero**.

[Fig.10]

- Cliquez sur Ok.
- Compilez la solution au complet.

Vos données sont maintenant exposées par le site web aux clients Silverlight.

- Ajoutez la référence System.Windows.Controls.Data à votre projet Silverlight
- Ouvrez **MainPage.xaml** dans le projet Silverlight.
- Ajoutez une nouvelle grille de données et nommez-la "**MaDataGrid**"

```
<UserControl x:Class="OutOfBrowser.MainPage"
xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
xmlns:data="clr-namespace:System.Windows.Controls;assembly=System.Windows.Controls.Data"
Width="400" Height="300">
<Grid x:Name="LayoutRoot" Background="White">
<StackPanel>
<data:DataGrid Name="MyDataGrid" Margin="5"/>
(...)
</StackPanel>
</Grid>
</UserControl>
```

- Ouvrez **MainPage.xaml.cs**
- Dans le constructeur de la page, juste après l'initialisation des composants, ajoutez ce code :

```
public MainPage()
{
    InitializeComponent();
    (...)
    SuperHeroContext context = new SuperHeroContext();
    MyDataGrid.ItemsSource = context.SuperHerros;
    context.LoadSuperHero();
}
```

Cette méthode passionnante d'accès aux données est bien plus complète que le peu que je viens de vous montrer. Pour les lecteurs intéressés, je conseille grandement une lecture du document relatif aux .Net RIA Services disponible sur <http://www.silverlight.net/GetStarted/silverlight3>.

■ Simon Boigelot & Loic Bar - HEODE.com

WINDEV : VOUS AUSSI, DÉVELOPPEZ 10 FOIS PLUS VITE !



WINDEV : élu «Langage le plus productif du marché»
www.pcsoft.fr



DEMANDEZ LE DOSSIER GRATUIT

252 pages + DVD + Version Express incluse
+ 112 Témoignages.
Tél: 04.67.032.032 ou 01.48.01.48.88
info@pcsoft.fr

Cloud Computing : déploiement d'applications Java

Cette série d'articles a pour but d'expliquer comment effectuer un déploiement d'applications Java sur la plate-forme Amazon EC2. La première étape consiste à préparer votre compte Amazon pour disposer d'un environnement exploitable. Pour cela, connectez-vous à votre compte et inscrivez vous à EC2 si cela n'est déjà fait (<http://aws.amazon.com/ec2>).

Sur le site, cliquez sur l'onglet " Your Account ", puis le lien " Access Identifiers ". Notez bien votre " Access Key ID " et votre " Secret Access Key " car vous en aurez besoin plus tard. Dans la section " X.509 Certificate ", cliquez sur le bouton " Create New " et sauvez sur votre poste les 2 fichiers qui seront générés (cert-XXXXX.pem et pk-XXXXX.pem) dans un répertoire nommé ".ec2 " (attention à bien garder le caractère '.') à la racine de votre compte. Sous Windows le plus simple consiste à exécuter les commandes suivantes :

```
cd %USERPROFILE%
mkdir .ec2
```

Une fois ces fichiers récupérés, installez les outils proposés par Amazon dans le répertoire ".ec2 " précédemment créé : <http://developer.amazonwebservices.com/connect/entry.jspa?externalID=351&categoryID=88> Les outils Amazon nécessitent que plusieurs variables d'environnement soient initialisées :

```
set EC2_HOME=%USERPROFILE%\.ec2
set PATH=%PATH%;%EC2_HOME%\bin
set EC2_PRIVATE_KEY=%EC2_HOME%\pk-XXXXX.pem
set EC2_CERT=%EC2_HOME%\cert-XXXXX.pem
```

Attention à bien remplacer les " XXXXX " par les noms de fichiers que vous aurez récupérés précédemment.

Pour vérifier le bon fonctionnement de votre compte EC2, vous pouvez lancer la commande suivante :

```
ec2-describe-images -o amazon
```

Cette instruction liste toutes les images système préparées par Amazon (AMI). Vous allez utiliser l'image Fedora Core 8 proposée par Amazon (" ami-5647a33f " au moment de la rédaction de cet article). Pour vous connecter à la machine, vous devez disposer d'une " keypair " qui véhicule vos accès aux serveurs afin de vous permettre de vous connecter. Pour cela, lancez la commande :

```
ec2-add-keypair ma-keypair
```

Cette commande génère une clé privée de type RSA. Vous devez copier le résultat affiché (en incluant les lignes -- BEGIN RSA PRIVATE KEY -- et -- END RSA PRIVATE KEY--) dans le fichier %EC2_HOME%/ma-keypair. Vous allez maintenant démarrer un serveur sur l'infrastructure EC2 (n'oubliez pas que ce faisant, Amazon vous facturera 0.10 \$ par heure).

```
ec2-run-instances ami-5647a33f -k ma-keypair
```

Cette commande lance un serveur de type SMALL. Il vous est possible de lancer plusieurs serveurs en même temps ou de spécifier le type de machine à démarrer. Le démarrage d'une machine EC2 prend un peu de temps (jusqu'à quelques minutes). Vous pouvez consulter le statut de la machine via la commande suivante :

```
ec2-describe-instances
```

Quand le statut de votre machine indique " running ", elle est prête à l'emploi. Notez l'adresse de la machine (ec2-xx-xx-xx-xx.compute-1.amazonaws.com) et son identifiant (i-XXXXXXXX).

Afin de pouvoir se connecter à ce serveur, vous devez autoriser les connexions SSH (port 22) et Web (port 80) :

```
ec2-authorize default -p 22
ec2-authorize default -p 80
```

Cette commande n'est à faire qu'une seule fois car elle modifie les règles du pare-feu nommées " default " en ouvrant les ports 22 et 80. Si vous utilisez Putty pour vous connecter via SSH, vous devez convertir le fichier ma-keypair en un fichier exploitable par Putty. Pour cela, utilisez l'outil " PUTTYgen ". Ouvrez le fichier %EC2_HOME%/ma-keypair via cet outil et sauvez la nouvelle clé privée. Vous devez en suite définir correctement la clé à utiliser lors de la connexion au serveur (cf <http://docs.amazonwebservices.com/AmazonEC2/gsg/2006-06-26/putty.html>). SSH vous demandera si vous souhaitez vous connecter. Tapez " yes " et ce sera chose faite.

Maintenant installez le serveur Apache. Puis, démarrez-le :

```
[root@domU-12-31-39-83-48-C2 ~]# /etc/init.d/httpd start
Starting httpd:                               [ OK ]
```

Vous pouvez maintenant vous connecter via votre navigateur au site <http://ec2-xx-xx-xx-xx.compute-1.amazonaws.com> (en remplaçant bien sûr les 'xxx' par ceux qui correspondent à l'instance de votre machine tels qu'affichés par la commande ec2-describe-instances). Pensez à bien éteindre votre serveur une fois que vous n'en avez plus besoin via la commande lancée sur votre machine :

```
ec2-terminate-instances i-XXXXXXXX
```

Les articles à suivre expliqueront comment construire sa propre image Amazon EC2 (AMI) ainsi que comment mettre à jour des applications sur EC2. Enfin, une fois les premiers déploiements sur EC2 effectués, vous verrez comment le projet Open Source Elastic Grid vous permettra de maintenir une application en production selon des critères de qualité de service (QoS) que vous spécifierez.

■ Jérôme Bernard - Director, EMEA Operations, Elastic Grid, LLC.
Site: <http://www.elastic-grid.com> - Blog: <http://blog.elastic-grid.com>

NE MANQUEZ PAS LE PROCHAIN NUMÉRO

N°121 juillet-août 2009, parution 30 juin

✓ Développement : Attention le futur est déjà là !

Les outils de la Direction Informatique

Vous avez besoin d'info sur des sujets d'administration, de sécurité, de progiciel, de projets ? Accédez directement à l'information ciblée.



L'INFORMATION SUR MESURE

Actu triée par secteur

Cas clients

Avis d'Experts



Actus

Événements

Newsletter

L'INFORMATION EN CONTINU

www.solutions-logiciels.com



Pourquoi passer à la programmation parallèle

A l'occasion de la sortie d'Intel Parallel Studio et des nombreuses avancées que les éditeurs préparent pour 2009 et 2010, *Programmez !* vous propose de mieux comprendre la programmation adaptée aux processeurs à plusieurs cœurs. C'est ce que l'on appelle la programmation parallèle. C'est-à-dire capable d'exécuter plusieurs opérations, instructions, parallèlement sur plusieurs processeurs ou plusieurs cœurs. Complexe à maîtriser, le parallélisme est le mont Everest du développement. Nous vous proposons de l'aborder méthodiquement en posant les fondamentaux pour bien l'assimiler et éviter les erreurs les plus classiques. Vous verrez alors que l'on peut faire exploser les performances de certaines portions de son code ! Bonne découverte !

■ François Tonic

En 1965, Gordon Moore (co-fondateur d'Intel) observait que le nombre de transistors contenus dans un semi-conducteur disponible sur le marché, doublait tous les 18 à 24 mois. C'est de ce constat empirique qu'est née la loi de Moore. En d'autres termes, les capacités techniques des ordinateurs s'amélioraient régulièrement, permettant aux programmes informatiques

d'en tirer parti sur le plan de la rapidité d'exécution sans grandes modifications (souvent aucune). Ce constat a fait le bonheur de plusieurs générations d'informaticiens, au point que certains appelaient ce phénomène le " free

lunch ". A cette époque, les fondeurs basaient le plus gros de leur discours marketing sur une course au Giga-Hertz, chacun se vantant d'aller plus vite que l'autre

Depuis quelque temps, vous avez sans doute remarqué l'apparition de processeurs multi-cœurs allant de deux à quatre cœurs dans les offres constructeurs grand public. Cependant, vos applications ne tournent sans doute pas plus vite sur ce type de matériel. En effet les systèmes d'exploitation, les applications d'aujourd'hui ne sont pas adaptés pour tirer parti efficacement d'une archi-

tecture multi-cœur. Autrement dit, cette évolution matérielle n'est pas forcément vécue comme un gain important par les utilisateurs.

Pourquoi les fondeurs ont-ils changé de paradigme ?

Depuis quelques années les constructeurs ont été confrontés à des contraintes physiques fortes (chaleur & consommation) qui ne permettaient plus d'améliorer leurs offres en restant sur les mêmes bases techniques. Par exemple, Intel avait prédit dès 2006 que la température de ses puces reposant sur la microarchitecture Netburst (Pentium IV) atteindrait environ 6000 degrés en 2015, soit environ la température à la surface du soleil. Ne pouvant plus croître sur le plan des Hertz, les constructeurs ont choisi de répartir la puissance sur plusieurs cœurs au sein d'un même processeur (architecture multi-cœur).

La course aux Hertz est donc terminée, place à la course au nombre de cœurs ! Dès aujourd'hui, les constructeurs de microprocesseurs disposent de nouvelles plates-formes sachant grandir facilement en nombre de cœurs tout en améliorant les performances de l'infrastructure interne du processeur lui-même. Le tout dernier processeur Intel Core i7 (Nehalem) rencontre déjà un franc succès auprès de constructeurs d'or-

dinateurs. Les ordinateurs de demain seront donc massivement multi-cœurs (supérieur à 8 cœurs), les américains parlent du passage du Multi-core au Many-core, ce qui peut sembler une bonne nouvelle.

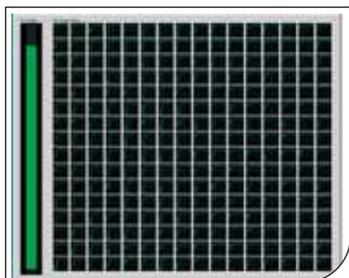
Les systèmes d'exploitation sont déjà en cours d'adaptation

De leur côté, les éditeurs de systèmes d'exploitation sont conscients que leurs produits ne sont plus adaptés pour exploiter correctement ces nouvelles architectures matérielles.

Les prochains systèmes profiteront de ces nouvelles offres qui aujourd'hui peuvent sembler exagérées au regard des besoins actuels, mais n'oublions pas que le monde change et que les progrès techniques ont toujours été porteurs d'innovations qui aujourd'hui nous apparaissent normales. A titre d'exemple, dans sa version 64 bits, le prochain système Windows sera capable de gérer pleinement 256 cœurs. Nous comprenons mieux avec cet exemple la nouvelle appellation Many-core pour qualifier des matériels regorgeant de cœurs.

The free lunch is over

Dès 2005, Herb Sutter, célèbre architecte C++ de Microsoft, annonçait "The free lunch is over". Autrement dit, les gains de performances de vos applications constatés sur les



Windows 7 en 64 bits sera pleinement capable de gérer 256 cœurs.

nouveaux matériels sont terminés. C'est donc aux développeurs de réviser leurs codes pour retrouver de la puissance. Dans ce contexte, pouvons-nous affirmer que le "free lunch" est définitivement révolu ? Pour enrayer cette catastrophe, devons-nous engager une révision radicale dans nos pratiques de développement ?

Conséquences pour les développeurs

Pourquoi peu de développeurs connaissent la programmation parallèle ? Pour les développeurs, la prise en compte effective de ce changement technologique peut se révéler très compliquée. Rappelons qu'il y a déjà une petite vingtaine d'années que les techniques de multithreading se sont intégrées dans un écosystème fortement séquentiel.

Néanmoins, les détracteurs du parallélisme peuvent trouver de nombreuses raisons pour ne pas s'adapter à cette nouvelle donne. La culture séquentielle à l'origine de l'informatique peut être considérée comme la plus légitime. Nous sommes aussi certains que le cerveau humain éprouve des difficultés à raisonner en multitâche ce qui peut être vu comme un frein pour le passage en mode parallèle.

Aujourd'hui, nous pouvons reconnaître que **les environnements de programmation** comme les outils, les bibliothèques, offrent peu d'adhérence avec ces concepts. Alors, comment s'assurer que la mise en parallèle de certaines parties du système est à la fois fiable et pertinente vis-à-vis du langage de programmation et de l'infrastructure matérielle ? Définir avec certitude que les notions d'isolation, d'immuabilité ou de synchronisation sont correctement exprimées vis-à-vis du problème posé au regard des environnements, des langages, des outils, **sont des préoccupations essentielles** que nous abordons difficilement, **faute d'un outillage adapté**.

Dans le cas des langages C et C++ où l'héritage d'une culture séquentielle se comprend facilement (le langage C est issu d'une philosophie très système inspirée par la plate-forme UNIX

AT&T originelle où la notion de multithreads n'existait pas) mais dont l'adaptation au multithreading est venue se superposer à un existant séquentiel donnant lieu à un jeu d'API rudimentaires difficilement compréhensibles pour le développeur non système. Initialement, la notion de thread a été introduite pour offrir aux développeurs un moyen de paralléliser du code sans avoir à créer plusieurs processus systèmes (très coûteux sur le plan des ressources du système d'exploitation). On constate avec le recul que de nombreux environnements se sont inspirés des interfaces vétustes originelles, plaçant le multithreading au rang des technologies réservées aux experts.

Les principaux écueils de la programmation parallèle ?

Après 20 ans de multithreading, nous pouvons reconnaître que les modèles parallèles sont à la fois peu répandus, peu connus, et difficiles à implémenter. Pour un novice en développement parallèle, il est très facile de provoquer de nombreux problèmes potentiels sans forcément les constater. En effet, l'exécution parallèle ne nous offre pas de certitude sur l'ordre d'exécution des instructions associées aux threads courants : le principe d'incertitude énoncé par Heisenberg défini en physique quantique a été renommé Heisenbug pour qualifier ce principe en informatique. En d'autres termes, toute ressource partagée par plusieurs threads doit faire l'objet d'une protection, sinon gare aux incohérences des résultats. Pour illustrer la dimension des difficultés rencontrées, nous pouvons citer les erreurs les plus courantes en programmation parallèle :

Race condition

Le code C# ci-dessous, pourtant très simple est tout de même faux dans le cas où la propriété Status est plongée dans plusieurs threads parallèles.

```
public class StatusManager
{
    private int _status;

    internal int Status
```

```
{
    get
    {
        return ++_status;
    }
}
```

Si de nombreux threads appellent simultanément cette propriété, l'exécution se déroulera souvent correctement mais parfois l'incrément sera faussé lorsque les threads se chevaucheront sur l'instruction d'incrément. De nombreux développeurs imaginent que l'opérateur d'incrément est atomique alors qu'il n'en est rien comme le montre le code assembleur.

```
MOVE EAX, [_status]
INC EAX
MOVE [_status], EAX
```

Le danger est que cette erreur est silencieuse et il est possible que personne ne la relève avant longtemps. Imaginez le cas où notre valeur représente le prix d'un produit financier, les conséquences pourraient être graves. La *race condition*, est sans doute l'erreur la plus populaire dans la programmation parallèle.

Pour corriger notre code il nous suffit de remplacer notre opérateur d'incrément par une méthode *InterLocked.Increment(ref _status)* qui nous assure une exécution atomique.

Cependant, les ressources à protéger ne sont pas toujours des entiers, mais sont souvent des types complexes. Dans ce cas, nous ferons appel à un outillage de synchronisation plus polyvalent. En contrepartie, ces outils peuvent occasionner de nombreux problèmes qui aboutissent parfois à des codes parallèles plus lents que leurs équivalents séquentiels. Voici d'autres exemples souvent rencontrés dans le cadre la programmation parallèle.

La suite le mois prochain.

■ Bruno Boucard

boucard.bruno@free.fr

Spécialisé dans les technologies Microsoft,

il anime avec d'autres spécialistes le blog

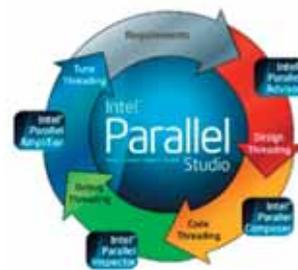
Développement parallèle de Microsoft

<http://blogs.msdn.com/devpara/default.aspx>

Version finale

Intel Parallel Studio débarque

Sur le CD du numéro !



La programmation est l'art de schématiser et simplifier un processus pour l'automatiser. Nous aboutissons donc le plus souvent à un code sériel. Avec l'arrivée de puces multi-core, et bientôt massivement multi-core cette approche atteint ses limites : Il vous faut passer à la programmation parallèle. N'ayez crainte, Intel est là pour vous aider avec un ensemble logiciel aussi convivial que complet : Intel Parallel Studio.

Le parallélisme se conçoit à plusieurs niveaux, mais celui qui nous intéresse ici est le "thread". Intel Parallel Studio est un ensemble de logiciels pour vous aider à introduire du multi-threadisme dans votre logiciel, corriger les éventuels bugs parallèles et enfin améliorer la performance. En pratique, Parallel Studio se présente comme un ajout à Microsoft Visual Studio avec lequel il est fortement intégré.

Introduire du multi-threadisme

La première étape est l'ajout de sections parallèles dans votre code. Choix cornélien tant les technologies proposées sont variées. Intel recommande et fournit une très bonne implémentation d'OpenMP et de la librairie Threading Building Block. Le standard OpenMP est le bon choix pour les programmeurs orientés C qui préfèrent garder un style sériel et minimiser les changements de code. La librairie TBB est, elle, destinée aux adeptes du C++ qui ne sont pas contre une incursion dans la programmation fonctionnelle. Avec ces deux solutions vous devriez pouvoir résoudre une grande partie de vos besoins. Une fois la méthode choisie, paralléliser une partie simple de votre code est étonnamment rapide. Mais il ne s'agit que d'un début.

Chasser les bugs

Avec la programmation parallèle apparaissent des bugs spécifiquement parallèles. Ils sont sournois et complexes à tracer. Il est alors très utile d'avoir un outil comme Parallel Inspector. Cet outil instrumente et analyse de manière extrêmement avancée l'exécution de votre logiciel pour trouver des bugs, et ce même s'ils ne s'exécutent pas dans les conditions particulières du test. Le résultat est un simple tableau très complet pointant vers les bugs dans votre code. Il faut cependant comprendre quelles sont les classes de bugs parallèles pour pouvoir comprendre les indications et corriger. Les bugs parallèles sont certes sournois mais ils ont souvent le même mécanisme et des causes similaires. Les corriger est donc souvent simple une fois que l'outil les a pointés.

La touche finale

Vous avez un logiciel parallèle, bravo, mais en pratique encore faut-il qu'il utilise correctement tous ces merveilleux cœurs. Dans une équipe, ajouter des développeurs ne sert que si la méthode de travail permet un partage efficace du travail. Pour vous aider à comprendre comment fonctionne votre logiciel en parallèle, Intel propose Parallel Amplifier.

Cet outil mesure comment chaque instruction de votre logiciel interagit avec le processeur à une échelle infime lors de l'exécution, puis agglomère les données pour en faire un résumé simple. Vous pouvez alors trouver les régions chronophages en allant jusqu'à l'échelle de la ligne de code, et les régions qui n'utilisent qu'une partie des cœurs. Vous pour-

rez aussi savoir quel temps est pris dans la synchronisation d'une variable partagée.

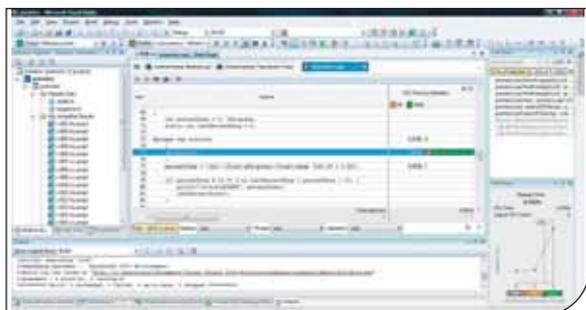
Compilateur et IPP

Intel Parallel Studio a beau mettre en avant ses logiciels de parallélisation, il fournit aussi tout ce qui est nécessaire pour obtenir la meilleure performance en sériel avant même de paralléliser. L'outil clé est le compilateur Intel. Bien intégré dans Visual Studio il est le fruit d'une intense collaboration avec les architectes processeurs. Nul autre compilateur ne sait mieux tirer parti des derniers jeux d'instructions. Apprenez la signification des principaux flags et vous gagnerez à coup sûr, sur le temps d'exécution. Le compilateur Intel ne respecte pas seulement le standard, il le devance. Ainsi vous trouverez le support pour les fonctions lambda et autres innovations syntaxiques dans la dernière version. Il serait triste de consacrer du temps à l'optimisation de votre code et d'utiliser par ailleurs des librairies peu performantes. Aussi Parallel Studio est fourni avec la librairie Intel Performance Primitives. Derrière ce nom se cachent en fait des codecs audio ou vidéo ultra optimisés, des fonctions d'imagerie par pipeline, de cryptographie, mathématiques ... et bien d'autres.

Conclusion

Les concepteurs de Parallel Studio avaient pour ambition de démystifier et simplifier la programmation parallèle. Je pense qu'ils ont réussi. Jugez en par vous-même avec la version d'essai.

■ Paul Guermonprez
Ingénieur logiciel - Intel



OFFRE LIMITÉE

SAVOIR c'est POUVOIR !



11 numéros par an : 49 €*
Economisez 16,45 €*

* au lieu de 65,45€ prix de vente au numéro, Tarif France métropolitaine

EN CADEAU

Accès illimité de 10 jours à la Bibliothèque Numérique des Editions ENI

A choisir parmi les Bouquets :

Consultez : www.eni-bibliotheque.fr

- **Développement** : Plus de 30 livres numériques disponibles, de ADO.Net 3.5 à Flex
- **Microsoft** : Plus de 30 livres numériques disponibles, de Windows Server 2008 à Silverlight
- **Open Source** : Plus de 20 livres numériques disponibles, de Nagios à Java EE

Pour tout abonnement au magazine Programmez un accès de 10 jours à la Bibliothèque Numérique de votre choix vous est offert. Offre limitée aux 1000 premiers souscripteurs.

Choisissez votre formule

Inclus : accès à la Bibliothèque Numérique !

- **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 avec Accès 10 jours à la Bibliothèque Numérique des Editions ENI ! (offre limitée)

- 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*

Choisissez votre Bouquet de la Bibliothèque Numérique : Développement Microsoft Open Source

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 30 juin 2009

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 120

Comprendre les Design Pattern Fabrique et Fabrique Abstraite

Tous les langages à objet disposent d'un opérateur `new`, ou équivalent, pour créer des objets. Fort pratique en apparence, cet opérateur recèle des problèmes dont les Pattern Fabrique et Fabrique Abstraite permettent de s'affranchir.



Comme nous le savons, les Design Pattern, ou modèle de conception, sont des lignes directrices, des bonnes pratiques destinées à rendre plus simple et plus sûre la programmation objet. Ces bonnes pratiques ont été définies par des programmeurs expérimentés en POO et qui se sont aperçus que fondamentalement le programmeur objet doit sans cesse résoudre les mêmes problèmes. Les Design Pattern sont répartis en catégories : comportement, structuration et création d'objets. Nous abordons aujourd'hui un modèle classé dans la catégorie création d'objets en nous appuyant sur le langage Java. Il est indéniable que le programmeur objet est en permanence en situation de créer des objets. Mais en quoi est-ce un problème ? Nous disposons pour cela de l'opérateur `new` (ou équivalent selon le langage). Nous utilisons cet opérateur, les objets sont créés et le tour est joué ! En théorie oui c'est aussi simple que cela. Et dans les cas très simples l'approche convient très bien. Mais dès qu'une application devient ne serait-ce qu'un peu compliquée, l'utilisation de l'opérateur `new` au "coup par coup" conduit à un code spaghetti, difficile à maintenir, ce que théoriquement le paradigme objet devrait éviter et non générer. Tel est le but des Pattern Fabrique et Fabrique Abstraite: rationaliser la création d'objets et donc l'emploi de l'opérateur `new`.

1 SIMULONS UN WARGAME

Pour apprécier les atouts du Pattern Factory, nous reprenons l'idée du wargame fictif qui nous a déjà servi pour les Pattern Observateur et Stratégie dans Programmez! 93 et 94 respectivement. Par wargame nous voulons parler de ce type de jeux en temps réel où le joueur clique sur des unités combattantes pour leur assigner une action. Toutes les interfaces de ce type de jeux présentent une zone sur laquelle on clique pour déclencher la création d'unités combattantes en fonctions des ressources dont on dispose. Vous l'avez compris, créer des unités combattantes, c'est du point de vue de la POO, instancier des classes, ou si l'on préfère, fabriquer des objets. Une occasion toute trouvée de mettre en oeuvre notre Pattern

2 UN PREMIER JET

Mais avant cela, faisons comme si nous ne connaissions pas l'existence des Design Pattern et codons à l'instinct, sans trop de réflexion préalable. Nous avons des unités combattantes avec des fonctionnalités communes, nous faisons donc dériver celles-ci d'une classe commune baptisée Entité. (Et encore, déjà il faut se méfier de

cette approche comme vous le savez. Ou si vous l'avez oublié, peut être apprécierez vous de relire Programmez! n° 94 :) Nous déclarons seulement deux classes, *Soldat* et *Tank*, c'est amplement suffisant pour notre propos. Ensuite nous avons un panneau de jeu, sur lequel se déroule la bataille qui présente l'interface de génération des unités. Faisons simple et admettons que lorsque le joueur clique sur cette interface, la méthode *creerEntite* du panneau est invoquée. Enfin nous avons la classe *JeuBasic*, qui représente le jeu lui-même. Voici l'implémentation de notre mini cahier des charges. Un wargame totalement fonctionnel :)

```
package sanspattern;

import java.util.Vector;

enum TypeEntite {
    TYPESOLDAT,
    TYPETANK,
}

class Entite {
    protected String categorie;
    protected int pv; // Points de vie

    Entite() {
        init();
    }

    public void init() {
        categorie = "Entite";
        pv = 0;
    }

    public void display() {
        System.out.println(categorie + ", pv: " + pv);
    }

    public void action() {}
}

class Soldat extends Entite {
    //voir sources sur notre site
}

class Tank extend Entite {
    //voir sources sur notre site
}
```

```

class PanneauJeu {
    Vector<Entite> pieces = new Vector<Entite>();

    public void creerEntite(TypeEntite type) {
        switch(type) {
            case TYPESOLDAT:
                pieces.add(new Soldat());
                break;
            case TYPETANK:
                pieces.add(new Tank());
                break;
            default:
                break;
        }
    }

    public void jouer() {
        for(Entite e : pieces) {
            e.display();
            e.action();
        }
    }
}

public class JeuBasic {

    public static void main(String[] args) {
        PanneauJeu pj = new PanneauJeu();
        // Ici le joueur clique sur l'interface
        // et crée des entités
        pj.creerEntite(TypeEntite.TYPESOLDAT);
        pj.creerEntite(TypeEntite.TYPETANK);
        // puis le jeu se déroule
        pj.jouer();
        // et ainsi de suite
    }
}

```

3 ANALYSE DU CODE ET INVENTAIRE DES PROBLÈMES

Commentons notre code, qui s'il est écrit d'une manière naturelle, est très mauvais :) Commençons par les classes des unités combattantes. Chacune comporte trois méthodes en plus du constructeur. La méthode *init* comme son nom l'indique initialise les membres de la classe, la méthode *display* s'occupe de l'affiche de l'unité et la méthode *action* s'occupe de l'action de l'unité au plus fort de la bataille. Déjà nous remarquons du code redondant. Le corps de la méthode *display* est trois fois le même. Ne riez pas, pareille horreur s'est déjà vue en situation réelle. Et si ceci ne concerne pas directement notre Pattern Fabrique, nous constaterons que la mise en oeuvre de celui-ci contribue à éradiquer ce défaut. Ensuite, la méthode *init*. A chaque fois son corps contient ce qui est nécessaire à l'initialisation de l'objet. C'est logique, direz-vous, qu'un objet sache s'initialiser. Oui, c'est logique. Mais c'est une logique. Il n'est pas obligatoirement requis qu'il en soit ainsi. Enfin la méthode *action*, dans la classe Entité, n'est qu'une coquille vide. C'est peu élégant. Regardons maintenant la classe PanneauJeu. Sa méthode *jouer* n'appelle pas de commentaire. Sa méthode *creerEntite* est une suite de tests

sur le type de l'unité à créer. Lorsque le type est trouvé, l'unité est créée via l'opérateur new.

Ce code semble tout à fait convenable, et pour notre exemple très simple il l'est. D'ailleurs pour les cas très simples à peu près tout convient. Mais les choses se gâtent sérieusement pour les situations non triviales, ce qui en informatique pratique sont les seules situations que l'on rencontre. Ensuite il y a les deux ennemis jurés du développeur: le chef de projet et le client :) Ennemis parce qu'à coup sûr ils vont demander une modification du code. De mémoire de codeur le contraire ne s'est jamais vu. "Le jeu serait enrichi si on avait des entités camion" dit le chef de projet. Pour faire face à cette modification il suffit d'ajouter un test sur le type. On s'en tire encore facilement. Mais voilà que le client dit : "Je voudrais avoir deux zones, sur lesquelles cliquer pour créer des unités soit terrestres soit maritimes et je voudrais avoir aussi trois niveau de jeux, des panneaux de jeu différent autrement dit, utilisant a priori leur ensemble propre d'unités ". Ca y est, cette fois nous sommes fichus ! Nous nous retrouvons potentiellement avec 6 méthodes *creerUnite* à gérer. Et le chef de projet revient: "Nous avons oublié les sous-marins!". Modifier le code de 6 méthodes qui font sur le fond la même chose, créer des entités, est pénalisant. Beaucoup de travail, du temps perdu, sans doute du code à la fois redondant et difficile à maintenir, et une forte probabilité de bugs sont les ennuis qui nous guettent. Tout cela, fondamentalement, parce que la classe PanneauJeu n'est pas découplée des classes Entité. Il est maintenant clair que nous devons rationaliser nos créations d'entités, et nous le faisons au moyen du Pattern Fabrique simple.

4 LE PATTERN FABRIQUE

Son but est de déléguer la création d'objet à une ou plusieurs classes dont ce sera le seul rôle. Lorsqu'il y a plusieurs classes celles-ci seront utilisées selon le contexte ainsi que nous le verrons plus loin. Voyons d'abord le bien fondé d'une classe unique, ce qui correspond au Pattern Fabrique Simple. Nous commençons par assainir les unités qui dériveront d'une classe abstraite, et dans cette classe nous factorisons le code redondant de notre premier exemple. Ensuite nous créons notre fabrique qui dans l'exemple ci-dessous est la classe... SimpleFabrique.

```

package demofactorypattern;
import java.util.Vector;

enum TypeEntite {
    TYPESOLDAT,
    TYPETANK,
}

abstract class Entite {
    protected String categorie;
    protected int pv; // Points de vie

    public void init(String categorie, int pv) {
        this.categorie = categorie;
        this.pv = pv;
    }

    public void display() {

```

```

System.out.println(categorie + ", pv: " + pv)
}

abstract public void action();
}

class Soldat extends Entite {

@Override
public void action() {
    System.out.println("Action soldatesque");
}
}

class Tank extends Entite {
// voir sources sur notre site
}

class SimpleFabrique {

Entite creerEntite(TypeEntite type) {
    Entite e;
    switch (type) {
        case TYPESOLDAT:
        default:
            e = new Soldat();
            e.init("Soldat", 10);
            break;
        case TYPETANK:
            e = new Tank();
            e.init("Tank", 500);
            break;
    }
    return e;
}
}

class PanneauJeu {
    Vector<Entite> pieces = new Vector<Entite>();
    SimpleFabrique fabrique = new SimpleFabrique();

    public void creerEntite(TypeEntite type) {
        pieces.add(fabrique.creerEntite(type));
    }

    public void jouer() {
        for(Entite e : pieces) {
            e.display();
            e.action();
        }
    }
}

public class JeuSimpleFabrique {

    public static void main(String[] args) {
        System.out.println("Demo Simple Factory");
        PanneauJeu pj = new PanneauJeu();
        // Ici le joueur clique sur l'interface

```

```

// et crée des entités
pj.creerEntite(TypeEntite.TYPESOLDAT);
pj.creerEntite(TypeEntite.TYPETANK);
// puis le jeu se déroule
pj.jouer();
// et ainsi de suite
}
}

```

Maintenant le panneau de jeu et les entités sont découplés. Un même panneauinstanciera aussi bien une horde de barbares que des soldats du futur, selon la fabrique à laquelle il délèguera les instanciations, fabrique qui peut très bien être passée en argument au constructeur du panneau. Un autre atout de cette création centralisée apparaît maintenant avec la méthode *init*. En effet si celle-ci n'avait pas grand sens dans le premier exemple, on voit maintenant que les classes entités elles-mêmes n'ont plus besoin de connaître les attributs. Notre fabrique pourrait bien à son initialisation lire une seule fois un fichier de configuration contenant toutes les données, ce qui est plus rationnel que de voir le constructeur de chaque classe Entite faire ce travail, à chaque instanciation. Enfin si nous devons ajouter un type d'entité, le seul code à modifier est celui de la méthode *creerEntite* de notre fabrique.

5 UNE VARIANTE AU PATTERN FABRIQUE

Dans notre exemple nous utilisons une seule fabrique, alors plutôt que d'instancier celle-ci, pourquoi ne pas déclarer statique sa méthode *creerEntite* ? Cela existe. Les Pattern sont une ligne directrice qu'il faut savoir adapter au cas particulier. La classe BorderFactory de la JDK est un exemple de fabrique à méthode statique. Le revers de la médaille est que le polymorphisme est cassé. C'est un choix.

6 LE PATTERN FABRIQUE ABSTRAITE

Nous arrivons à la définition complète du Pattern qui est: offrir une interface pour la création d'objets, mais en laissant aux sous-classes le choix des classes à instancier. Revenons à notre wargame. Notre chef de projet soucieux de donner du réalisme à notre jeu, s'écrie tout à coup: *"personne ne fait la guerre sans ressources. Notre jeu doit comporter un panneau civil dont les acteurs produiront les ressources nécessaire à la guerre!"*. Et il ajoute: *"En outre, les interfaces utilisateur des panneaux civils et militaire sont similaires, nous devrions nous en tirer sans rien coder de plus à ce niveau"*. En effet, nous le pouvons... parce que notre code est basé sur la Pattern Fabrique. Le rôle de celui-ci est d'offrir une interface. Le terme est à prendre au sens de la programmation objet. Dans notre exemple cette interface se réduit à une méthode: *creerEntite*. Les classes fabriques implémenteront cette interface ou selon l'usage habituel dériveront d'une classe abstraite comportant cette méthode. Le diagramme ci-dessous est notre Pattern Fabrique Abstraite vu par UML. Pour ce qui est de notre jeu, c'est tout simple. Nous créons des unités civiles, une fabrique d'unité civile et... c'est tout :) Voici ce que cela donne (code complet sur notre site) :

```

abstract class JeuFabrique {
    abstract Entite creerEntite(TypeEntite type);
}

```

```

class FabriqueMilitaire extends JeuFabrique{
//etc
}

class FabriqueCivile extends JeuFabrique{

Entite creerEntite(TypeEntite type) {
Entite e;
switch (type) {
case TYPEBOULANGER:
default:
e = new Boulanger();
e.init("Boulangier", 30);
break;
case TYPEMECANICIEN:
e = new Mecanicien();
e.init("mecanicien", 40);
break;
}
return e;
}
}

class PanneauJeu {
Vector<Entite> pieces = new Vector<Entite>();
JeuFabrique fabrique;

PanneauJeu(JeuFabrique fabrique) {
this.fabrique = fabrique;
}
}

```

```

}

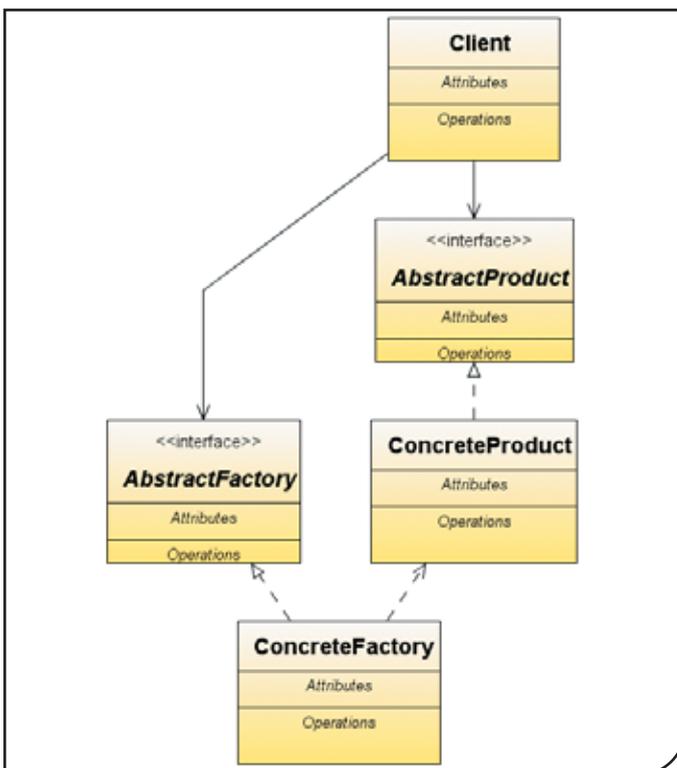
public void creerEntite(TypeEntite type) {
pieces.add(fabrique.creerEntite(type));
}

public void jouer() {
for(Entite e : pieces) {
e.display();
e.action();
}
}
}

public class JeuFabriqueAbstraite {
public static void main(String[] args) {
System.out.println("Demo Simple Factory");
PanneauJeu militaire = new PanneauJeu(new FabriqueMilitaire
());
PanneauJeu civil = new PanneauJeu(new FabriqueCivile());
// Ici le joueur clique sur les interfaceq
// et crée des entités
militaire.creerEntite(TypeEntite.TYPESOLDAT);
militaire.creerEntite(TypeEntite.TYPETANK);
civil.creerEntite(TypeEntite.TYPEBOULANGER);
civil.creerEntite(TypeEntite.TYPEMECANICIEN);

// puis le jeu se déroule
militaire.jouer();
civil.jouer();
// et ainsi de suite
}
}

```



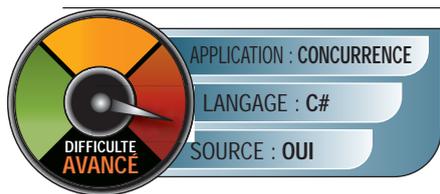
Ce qui est remarquable avec cet exemple c'est que la classe PanneauJeu est unique. Nous l'instancions deux fois et nous lui passons la fabrique en adéquation avec son contexte d'utilisation. Le code est unique tout simplement parce qu'il ne travaille avec aucune classe, mais seulement avec une interface. Le boulanger et le mécanicien sont des citoyens. Voulons-nous un panneau de jeu pour des entités campagnardes ? La classe PanneauJeu restera la même. Simple-ment nous lui passerons une autre fabrique, adaptée à ce contexte. On peut se poser la question de savoir si, pour une application plus réelle, on pourrait vraiment utiliser un même code de panneau. Après tout le code l'affichage serait différent dans chaque cas et il faudrait bien s'en occuper. La réponse est pourtant oui. On réglerait ce problème avec le Pattern Stratégie vu dans Programmez! 94 :) Les Pattern collaborent volontiers ensemble. Simple à comprendre, simple à mettre en oeuvre, le Pattern Fabrique Abstraite est très puissant. Dans les applications complexes il est impossible de s'en passer. Attention! Voici notre chef de projet qui revient avec une nouvelle idée géniale en tête. Heureusement nous avons quelques Pattern en réserve... pour une prochaine fois :)

■ Frédéric Mazué - fmazue@programmez.com

Tous les jours : l'actu et le téléchargement
www.programmez.com

CCR, une API de programmation concurrente pour .Net

La programmation concurrente et le cortège de deadlocks qui l'accompagne sont plus que jamais d'actualité. CCR est une API concurrente proposée par Microsoft pour .Net, dont l'originalité est de ne pas utiliser de verrous. Partons à sa découverte.



Tout programmeur qui s'est essayé à écrire un petit peu de code concurrent a subi les turpitudes de l'ennemi public n°1 en la matière: le deadlock. Cette

condition de verrou mortel se produit quand deux flux d'exécution attendent chacun que l'autre libère une ressource telle que verrou, section critique, sémaphore ou autre. Et s'il était possible de faire de la programmation concurrente sans verrou ? Tout serait beaucoup plus simple. C'est l'idée qui a présidé au développement de l'API CCR, qui fournit un modèle de programmation minimisant notablement les risques et facilitant la conception des applications. Si CCR n'éradique pas le deadlock définitivement (est-ce seulement possible ?), il est un outil à découvrir absolument. CCR est la fondation de Microsoft Robotics Developer Studio, mais elle peut parfaitement être utilisée seule, avec n'importe quel langage .Net, car ce n'est concrètement qu'un jeu d'assemblées. CCR est d'ailleurs également à la base de DSS, une API pour le développement d'applications concurrentes distribuées. Nous allons quant à nous découvrir CCR en écrivant un petit peu de code concurrent classique. CCR n'existe pas en distribution autonome gratuite, mais il est au coeur de Robotics Developer Studio en Express Edition, donc gratuite.

1 VUE D'ENSEMBLE

Fondamentalement CCR est un mécanisme à passage de messages. Il en existe de nombreux pour la programmation concurrente. Les implémentations de MPI par exemple. Ou bien les applications Erlang. Dans ces deux cas, c'est le programmeur qui expédie explicitement les messages, depuis des threads ou des processus respectivement. L'originalité de CCR réside dans une plus grande subtilité. Des classes de haut niveau surveillent des changements de conditions dans des ports et y réagissent en démarrant des threads. Avec ce mécanisme, le programmeur ne manipule pas les messages directement. Une application CCR met en oeuvre 5 classes au minimum. La première est le port dont nous venons de parler et autour duquel tout s'articule. Concrètement, un port peut être vu comme une pile FIFO de données typées. Vient ensuite le Dispatcher qui détient des files de tâches potentielles et qui, comme son nom l'indique, diffusera les messages et démarrera les tâches. Ces files de tâches sont encapsulées dans une classe du nom de DispatcherQueue. Vient ensuite le Receiver, qui encapsule la notion de fonction de rappel (ou callback). Cette classe implémente une interface `ITask`, et, dans les instances de ces classes, un port est associé avec du code à exécuter en parallèle. Vient enfin l'Arbiter dont une partie des méthodes statiques sont invoquées pour obtenir l'asso-

ciation dont nous venons de parler, et dont la méthode `Activate` associe des objets implémentant l'interface `ITask` (tel un `Receiver`) avec une `DispatcherQueue`. Tout ceci peut sembler assez complexe, et dans un sens ça l'est. D'autant plus que nous n'avons pas décrit les classes dans le détail. Mais avec un peu de pratique, tout s'éclaircit et devient, finalement, plutôt facile.

2 PREMIER EXEMPLE RUDIMENTAIRE

Nous ne pouvons pas faire plus simple, c'est le "Hello World!" de la programmation concurrente à la sauce CCR. Nous créons un port destiné à recevoir des entiers. Les entiers postés dans ce port sont imprimés en écho sur la console par du code parallélisé :

```
using System;
using Microsoft.Ccr.Core;

namespace SimplePort
{
    class Program
    {
        Port<int> lePort;
        Dispatcher leDispatcher;
        DispatcherQueue leDispatcherQueue;
        Receiver<int> leReceiver;

        Program()
        {
            lePort = new Port<int>();
            leDispatcher = new Dispatcher(1, "Mon Dispatcher");
            leDispatcherQueue = new DispatcherQueue(
                "Mon DispatcherQueue", leDispatcher);
        }

        void Go()
        {
            leReceiver = Arbiter.Receive<int>(
                true,
                lePort,
                Rappel);

            Arbiter.Activate(leDispatcherQueue, leReceiver);

            for (int i = 0; i < 10; i++)
            {
                lePort.Post(i);
            }
        }
    }
}
```

4 NOMBRE DE THREADS ET COEURS DE PROCESSEUR

Il est important de bien noter que les threads lancés par CCR sont des threads systèmes. Ils ne sont pas gérés et planifiés par le runtime comme c'est le cas par exemple en Erlang. Que l'on ait des threads systèmes implique une limitation: Windows ne supporte qu'un maximum de 64 threads pour une application. Lancer un grand nombre de threads ne présente d'ailleurs que peu d'intérêt en général. Il est plus intéressant de répartir le code parallélisé sur les coeurs d'un processeur. Voici un code qui demande explicitement à exploiter de la machine hôte. Le code parallélisé n'a pour rôle que de dévorer du temps d'exécution glouonnement :

```

Console.ReadLine();
}

void Rappel(int valeur)
{
    Console.WriteLine(valeur);
}

static void Main(string[] args)
{
    Program p = new Program();
    p.Go();
}
}

```

Il y a plusieurs façons de passer le code parallélisé à `Arbiter.Receive`. Si l'on préfère, on peut passer une fonction anonyme :

```

leReceiver = Arbiter.Receive<int>(
    true,
    lePort,
    delegate(int valeur) { Console.WriteLine(valeur); });

```

Et même saupoudrer le tout d'un peu de sucre syntaxique :)

```

leReceiver = Arbiter.Receive<int>(
    true,
    lePort,
    valeur => Console.WriteLine(valeur));

```

Dans tous les cas, bien remarquer le booléen positionné à `true`. Ce booléen assure la persistance de la connexion entre le port et le code parallélisé. Avec le booléen à `false`, seul le premier entier serait affiché. Ce code est finalement fort simple pour de la programmation parallèle. Une fois que tout est initialisé, on poste 10 entiers dans le port, sans se préoccuper de rien. Si vous essayez ce code vous verrez que le callback, du nom de `Rappel`, est bien invoqué par le runtime, comme on s'y attend, et les 10 entiers sont affichés sur la console, dans l'ordre dans lequel ils ont été postés.

3 ASYNCHRONE OU CONCURRENT ?

Les nombres affichés dans l'ordre dans lequel ils ont été postés ? Voilà qui est surprenant. On dirait avoir plus affaire à une API de programmation asynchrone que concurrente. Notre exemple est effectivement asynchrone et ressemble au mécanisme des messages de fenêtre en programmation Win32. L'explication se cache dans l'instanciation du `Dispatcher` :

```

leDispatcher = new Dispatcher(1, "Mon Dispatcher");

```

La valeur 1 représente, nous dit la documentation de CCR, le nombre de threads systèmes que peut utiliser le runtime CCR pour traiter les tâches en attente dans la ou les queues qu'il gère. Avec un seul thread permis, nous avons un comportement asynchrone à la Win32. Remplacez 1 par 10 et là vous verrez que les 10 entiers ne s'affichent plus dans l'ordre. Cette fois le runtime a bien démarré plusieurs threads simultanément, et nous avons le comportement caractéristique des applications concurrentes: les résultats dans le désordre, de façon imprévisible, ou du moins non garantie par l'API.

```

using System;
using System.Threading;
using Microsoft.Ccr.Core;

namespace PostCore
{
    class Program
    {
        Port<int> lePort;
        Dispatcher leDispatcher;
        DispatcherQueue leDispatcherQueue;
        Receiver<int> leReceiver;

        Program()
        {
            lePort = new Port<int>();

            leDispatcher = new Dispatcher(
                10, ThreadPriority.Normal,
                DispatcherOptions.UseProcessorAffinity,
                "Mon Dispatcher");

            leDispatcherQueue = new DispatcherQueue(
                "Mon DispatcherQueue", leDispatcher);
        }

        void Go()
        {
            leReceiver = Arbiter.Receive<int>(
                true,
                lePort,
                Rappel);

            Arbiter.Activate(leDispatcherQueue, leReceiver);

            for (int i = 0; i < 10; i++)
            {
                lePort.Post(i);
            }
            Console.ReadLine();
        }

        void Rappel(int valeur)
        {

```

```

Console.WriteLine(
    "Entrée dans le thread {0}", valeur);
int n = 250;
for (int i = 0; i < n; i++)
    for (int j = 0; j < n; j++)
        for (int k = 0; k < n; k++)
        {
            DateTime t = DateTime.Now;
        }
    Console.WriteLine("Sortie du thread {0}", valeur);
}

static void Main(string[] args)
{
    Program p = new Program();
    p.Go();
}
}

```

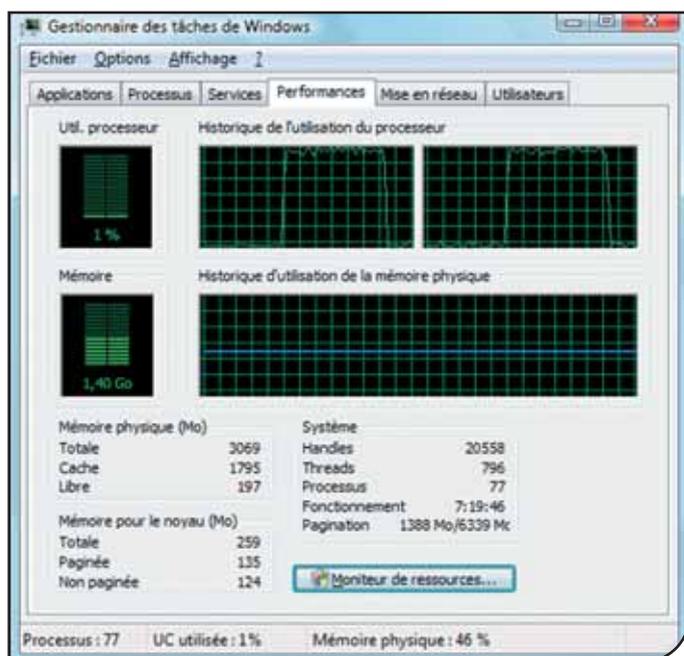
Examinons attentivement comment le Dispatcher est instancié cette fois :

```

leDispatcher = new Dispatcher(10,
    ThreadPriority.Normal,
    DispatcherOptions.UseProcessorAffinity,
    "Mon Dispatcher");

```

Nous lui demandons de gérer les threads par paquets de 10. Nous donnons aux threads une priorité normale. Cette notion de priorité étant à prendre au sens "programmation Win32" du terme. Enfin nous demandons explicitement au runtime de répartir les threads sur les coeurs du processeur de la machine. Quoique selon les observations faites pas votre serveur, ce comportement est de toute façon le comportement par défaut. Essayons notre code sur un Dual-Core. La capture ci-dessous montre nettement le pic de charge: les deux coeurs travaillent au maximum. Le programme émet cette sortie sur la console



CCR répartit bien la charge de travail sur les coeurs de notre processeur Dual-Core

```

Entrée dans le thread 0
Entrée dans le thread 2
Entrée dans le thread 4
Entrée dans le thread 1
Entrée dans le thread 7
Entrée dans le thread 6
Entrée dans le thread 5
Entrée dans le thread 8
Entrée dans le thread 9
Sortie du thread 1
Entrée dans le thread 3
Sortie du thread 8
Sortie du thread 5
Sortie du thread 2
Sortie du thread 6
Sortie du thread 7
Sortie du thread 9
Sortie du thread 0
Sortie du thread 3
Sortie du thread 4

```

Pour autant que cette sortie sur console décrive fidèlement les opérations, ce qui n'est pas forcément certain en environnement concurrent, le résultat des courses est assez surprenant et donne à réfléchir. D'abord les threads sont lancés dans un beau désordre. Mais après tout comme rien n'est garanti à ce niveau, pourquoi pas ? Preuve est faite que l'ordre des postages dans un port n'influence par l'ordre de lancement des threads. Quand même... cet ordre est étonnant :) L'ordre de fin d'exécution, sans rapport avec celui de lancement ne l'est pas moins, mais bon, concurrence oblige.... Mais la vraie surprise, si on considère que chaque thread a un temps d'exécution très long, c'est de voir que le dernier est lancé après qu'un se soit déjà terminé. Tout semble se passer comme si en interne, c'était un thread qui se charge de lancer les autres. Dans notre exemple, les coeurs des CPU étant vampirisés, le supposé thread de lancement doit attendre qu'un peu de ressource soit libérée pour faire son travail. On est tenté de conclure qu'avec CCR il convient d'éviter les situations avec un système très chargé sous peine d'obtenir un fonctionnement très besogneux, problème qui ne se rencontre pas avec Erlang que nous avons déjà cité plus haut. Il est vrai que notre exemple est un cas extrême de mauvais usage de code parallélisé :)

5 DES PORTS MULTI-TYPES

CCR recèle encore beaucoup de choses à découvrir. Ainsi le PortSet qui est une classe de port pouvant recevoir plusieurs types différents, jusqu'à dix neuf. Pour chaque type, on établit une connexion du port avec le code parallélisé adapté au cas. Cette connexion est établie via l'Arbiter, comme dans le premier exemple. Ceci installé, c'est tel ou tel thread qui sera lancé, selon le type de la valeur posée dans le port. Une fonctionnalité fort intéressante, agrémentée, là aussi, d'une petite surprise. Vous trouverez sur notre site le code DemoPortSet qui est un exemple mettant en oeuvre un port supportant trois types : entier, double et String.

Remarquons que l'instanciation du Dispatcher ne permet qu'un thread actif à la fois, comme dans le tout premier exemple de cet article. Dans le port nous postons successivement un entier, un

double, une chaîne et un flottant (Tiens le flottant est traité comme un double. C# n'est pas Haskell en ce qui concerne la rigueur sur les types :). Puisque nous ne pouvons avoir qu'un seul thread à la fois, nous nous attendons à obtenir une sortie dans le même ordre. La surprise est que nous obtenons :

```

C:\Windows\system32\cmd.exe
Reçu entier: 1
Reçu double: 1
Reçu double: 2
Reçu chaîne: Programmez!

```

Une sortie de console dans un ordre inattendu...

```

Reçu entier: 1
Reçu double: 1
Reçu double: 2
Reçu chaîne: Programmez!

```

Nous avons obtenu ce comportement sur toutes nos machines, aussi bien Dual-Core que Quad-Core. Ce qui vient immédiatement à l'esprit est que la constitution de la chaîne demande plus de temps et que son thread est doublé, c'est le cas de le dire, par le dernier thread traitant un double. Mais en fait, cela n'est pas normalement possible puisque nous ne pouvons avoir qu'un thread actif à la fois. Alors que se passe-t-il ? Il semble, sous toutes réserves, que le runtime vide les piles FIFO attachées à chaque type l'un après l'autre, dans l'ordre de leur déclaration et/ou dans l'ordre des gestionnaires spécifiés dans l'appel à `Arbiter.Activate`.

6 DE LA PROGRAMMATION CONCURRENTE SÉQUENTIELLE

Comment ? Le code est soit parallèle soit concurrent! Non, plus avec CCR :) Celui-ci propose un usage original des itérateurs C#. Au lieu d'avoir une fonction de code parallélisé ne retournant rien, on retourne un itérateur de tâches par une instruction `yield return`. Voici une fonction parallélisée extraite de l'exemple `Demolterator` que vous trouvez complet sur notre site :

```

IEnumerator<ITask> Rappel(int valeur)
{
    while (lePort.ItemCount != 0)
    {
        yield return lePort.Receive();
        result += lePort;
        Console.WriteLine(result);
    }
    Console.WriteLine("Résultat final {0}", result);
}

```

La tâche est retournée au planificateur du runtime, qui peut alors l'activer. L'instruction `yield return` devient une sorte de point de synchronisation. Dans notre exemple la partie de code qui poste dans le port ne fait pas partie de la section parallèle. La documentation de CCR va plus loin et propose un exemple où c'est le code parallélisé lui-même qui poste dans le port. Voici ce code extrait de `DemolteratorSequential`.

```

IEnumerator<ITask> Rappel(int valeur)
{
    for (int i = 1; i < 10; i++)
    {
        lePort.Post(i);
        yield return lePort.Receive();
        result += lePort; // <- sucre syntaxique
        Console.WriteLine(result);
    }
    Console.WriteLine("Résultat final {0}", result);
}

```

Et nous arrivons à un code parallélisé à l'exécution parfaitement séquentielle :)

7 UNE GESTION DES ERREURS ORIGINALE

Gérer les erreurs en environnement concurrent n'est pas une mince affaire. Pour traiter le problème, CCR dispose d'un système qui capture les exceptions éventuellement levées par les sections parallélisées, capture qui aboutit à l'exécution d'une section parallélisée dédiée. Nous avons en fait une sorte d'extension du mécanisme d'exception classique, implémenté pour les threads. Ce mécanisme assez complexe, qui permet d'avoir des gestionnaires d'erreurs imbriqués, ne sera pas décrit ici. Le lecteur trouvera toutefois un exemple rudimentaire sur notre site, du nom de `DemoCausality`.

8 CONCLUSION

Nous sommes très loin d'avoir fait le tour de CCR. Ainsi n'avons-nous, par exemple, pas parlé des fonctionnalités de synchronisation de threads (autre que le mécanisme de l'itérateur) Mais nous avons vu qu'il s'agit d'une API originale. Son comportement réserve son lot de surprises, mais c'est un peu la loi du genre.

Nous pensons que CCR, s'il obtient la popularité qu'il mérite, est de taille à révolutionner l'approche de la programmation concurrente sous .Net. Pour preuve de son potentiel, il est à la base de DSS (Decentralized Software Services), une API pour l'écriture d'applications concurrentes, distribuées et tolérantes aux fautes, ce qui n'est pas rien. Dans des articles à venir, nous approfondirons notre connaissance du maniement de CCR et nous découvrirons DDS, autre univers passionnant.

■ Frédéric Mazué
fmazue@programmez.com

Introduction à la construction d'un DSL sous Eclipse

Créer un langage spécifique à un domaine permet de proposer à vos utilisateurs un environnement de travail adapté à ce domaine, c'est-à-dire manipulant directement les concepts de celui-ci. Nous verrons aujourd'hui comment l'Ingénierie Dirigée par les Modèles (IDM) va nous aider à construire un tel langage et son environnement.



Avec les outils disponibles aujourd'hui dans Eclipse, il est possible de choisir entre réutiliser et adapter un langage existant générique comme UML, ou bien directement créer un langage dédié (ou Domain Specific Language). En choisissant cette seconde solution, l'un des avantages sera pour l'utilisateur final d'être naturellement guidé dans l'utilisation de ses modèles. Pour construire un DSL, nous vous proposons de suivre un processus qui permet une boucle de prototypage entre chaque étape de construction. Cela permet donc d'expérimenter facilement le langage avant de l'outiller complètement. Suivant notre expérience, cela assure une meilleure progression et permet d'en améliorer la fiabilité.

PRÉSENTATION DE L'EXEMPLE

Pour illustrer nos propos, nous nous appuyerons sur l'exemple du langage de la tortue **Logo** (que certains d'entre nous ont peut-être déjà connu sur leur MO5). Logo est un petit langage permettant de diriger une tortue équipée d'un crayon pour lui faire dessiner une figure là où elle est passée. Ce langage a d'abord été destiné à initier les enfants aux concepts de la programmation. Nous allons décrire comment l'IDM va nous aider à construire un environnement de programmation pour Logo, et au final l'appliquer concrètement pour effectivement piloter un petit robot construit en Lego Mindstorm.

UN PEU DE THÉORIE

Traditionnellement, quand on pense à concevoir un langage, on est tenté de réfléchir d'abord à la syntaxe concrète qu'on va donner à celui-ci, et donc en terme de grammaire et de parser. En IDM, la syntaxe concrète n'est qu'un aspect secondaire. C'est d'abord sur

les concepts du cœur du langage que l'on se base (la syntaxe abstraite). Ces concepts de base servent de fondations sur lesquelles nous allons bâtir autant de vues que nécessaire.

Nous allons profiter ici des facilités offertes par un outil Eclipse comme Kermeta. Tout d'abord parce qu'étant un Langage Orienté Modèle, Kermeta offre nativement des fonctions qui en facilitent la manipulation, mais surtout parce qu'il nous permet de tisser et d'assembler les éléments nécessaires sans modifier le cœur du langage que l'on souhaite construire. Nous verrons par la suite que cette fonctionnalité est intéressante car elle nous permet de viser plusieurs usages et outils autour du langage. Cela nous permettrait même d'envisager plusieurs variantes de sémantiques si le cœur nous en disait !

DÉFINITION DES CONCEPTS DU LANGAGE

Ainsi, dans le monde IDM, les DSL sont communément représentés avec ce que l'on appelle un méta-modèle. En fait, c'est un diagramme de classe qui définit les concepts de notre langage : dans Eclipse, cela se fait avec un modèle Ecore.

Avec notre exemple Logo, les concepts seront typiquement les instructions que l'on voudra faire exécuter à notre tortue. Par exemple, lever le crayon, avancer d'un certain nombre (entier) de pas ou tourner d'un certain angle (exprimé, pour simplifier, comme un nombre entier de degrés). Nous allons donc créer et relier les concepts correspondants : une classe PenUp, une classe Forward, et une classe Right. Les instances de ces classes représenteront les instructions du programme Logo de l'utilisateur, par exemple "FORWARD 10" ou "RIGHT 3*30". Les classes Forward, et Right étant paramétrées par une expression arithmétique, elles sont reliées à une classe "Expression", qui peut être par exemple soit

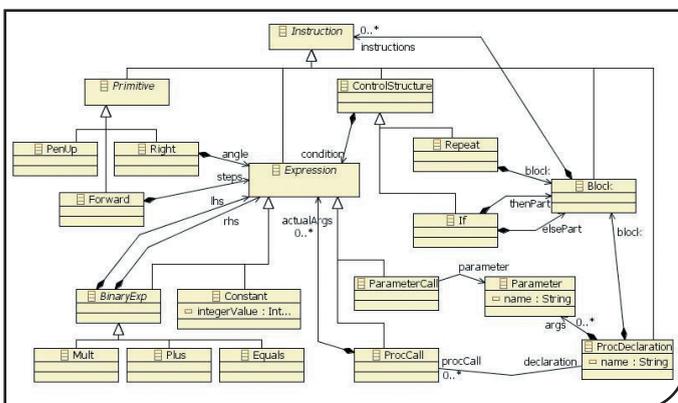


Figure 1. Version minimaliste du méta-modèle du langage Logo

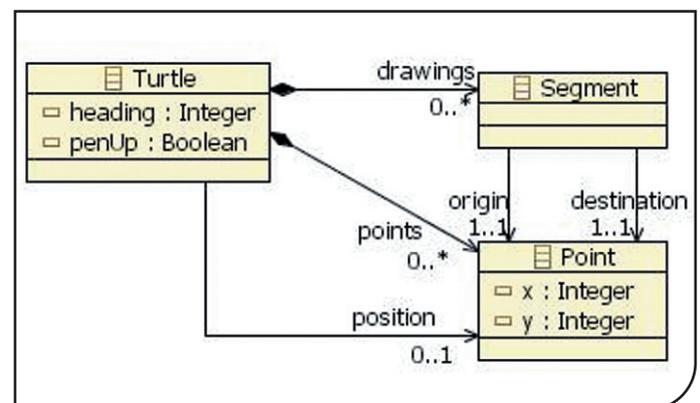


Figure 2. Machine Virtuelle pour le simulateur Logo

une expression binaire (ie. qui compose deux expressions par un opérateur de type +, *, etc.), soit une constante entière, ce qui est représenté par de l'héritage. L'éditeur arborescent de base fourni avec Ecore est certes fonctionnel, mais n'hésitez pas à lui adjoindre un diagramme de classe (fichier avec l'extension ecorediag) grâce à l'éditeur fourni par le projet Eclipse Ecore Tools.

Afin de profiter des avantages de génération des outils basés sur EMF, il faut penser à ajouter une notion de conteneur sur certains liens pour créer une hiérarchie. En effet, même si les concepts de notre langage sont bien là et forment un modèle valide, la plupart des générateurs s'appuient sur cette notion pour automatiser l'affichage de certaines vues. C'est le cas par exemple de la vue "outline". Pour rendre la tortue plus intéressante à piloter, nous compléterons ces instructions avec quelques structures de contrôle telles que *Block*, *If* ou *Repeat*... [Figure 4] Celles-ci permettent de structurer et d'ordonner les séquences d'instructions. De même, l'instruction *ProcCall* donnera la possibilité d'appeler des blocks définis dans *ProcDeclaration* avec des paramètres.

CRÉATION DES PREMIERS MODÈLES LOGO AVEC L'ÉDITEUR RÉFLEXIF

Pour débiter avec votre nouveau langage, le plus simple maintenant est d'utiliser l'éditeur arborescent réflexif. Depuis votre méta-modèle *logo.ecore*, il vous suffit de sélectionner l'élément racine de votre langage et de créer une nouvelle instance. Vous pouvez maintenant ajouter des instructions pour créer vos premiers programmes Logo.

TISSER DES CONTRAINTES COMPLÉMENTAIRES

En expérimentant un peu votre langage, vous vous apercevrez qu'Ecore ne vous permet pas d'exprimer certaines contraintes, par exemple que les paramètres formels d'une procédure doivent avoir des noms différents. C'est normal puisqu'il n'en définit que la structure. Comme l'objectif d'un DSL est d'aider les utilisateurs du futur langage, nous allons ajouter des contraintes qui vont les aider à ne pas faire ces erreurs. Le langage Kermeta nous donne ici la possibi-

lité de rouvrir les définitions Ecore pour ajouter des éléments nécessaires au besoin courant. Par exemple, pour vérifier que les arguments des procédures Logo ont des noms uniques, il suffit d'ajouter l'invariant suivant :

```
package kmLogo::ASM;
require kermeta
// importe les définitions du fichier logo.ecore
require "http://www.kermeta.org/kmLogo"

// réouvre la classe ProcDeclaration du méta-modèle Logo pour
lui ajouter un invariant
aspect class ProcDeclaration{
  inv unique_names_for_formal_arguments is do
    args.forAll{ a1 | args.forAll{ a2 |
      a1.name.equals(a2.name).implies(a1.equals(a2)) }
    }
  end
}
```

Obtenir un vérificateur de modèle revient alors à charger un modèle Logo et appeler la méthode *checkAllInvariants* sur les éléments à la racine du modèle. Cet invariant aurait aussi pu être écrit en OCL, (le langage officiel de l'OMG) et importé de la même manière, d'ailleurs Kermeta en reprend les facilités de navigation dans les modèles existant en OCL. Néanmoins, Kermeta nous sert surtout de système d'assemblage et lui ajoute d'autres fonctions qui seront utiles pour les besoins des autres activités d'ingénierie des méta-modèles.

EXTENSION DES CONCEPTS POUR FOURNIR UNE SIMULATION (SÉMANTIQUE OPÉRATIONNELLE)

Maintenant nous souhaitons rendre ce modèle plus "vivant" et voir notre tortue bouger. Pour en définir le comportement, le plus simple et le plus rapide est de fournir un simulateur. La première étape consiste à définir une représentation du domaine d'application (une sorte de Machine Virtuelle) sur laquelle le langage va s'appliquer. Pour notre exemple, ce sera tout simplement une ... tortue et les

traits qu'elle trace sur son terrain de jeu. Cela peut ici encore être représenté avec un diagramme Ecore. Ensuite, nous allons donner un peu de comportement à notre tortue en lui fournissant des primitives telles que *move* ou *rotate* qui sont spécifiques à cette représentation. Pour cela, il suffit d'indiquer que l'on a besoin d'étendre les classes du domaine d'application et de leur ajouter directement les

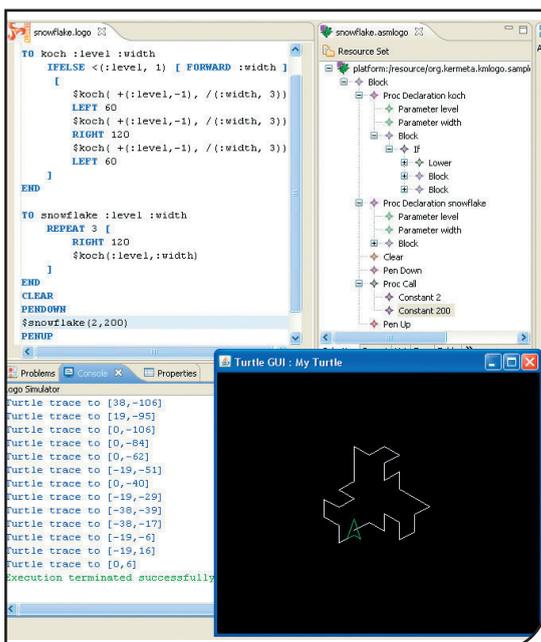


Figure 3. Exemple d'exécution d'un programme Logo avec le simulateur

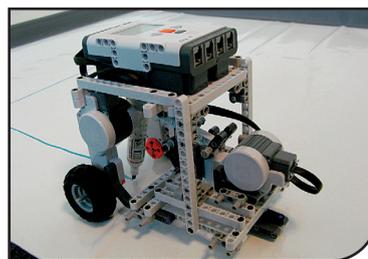


Figure 5 : Tortue robot en Lego

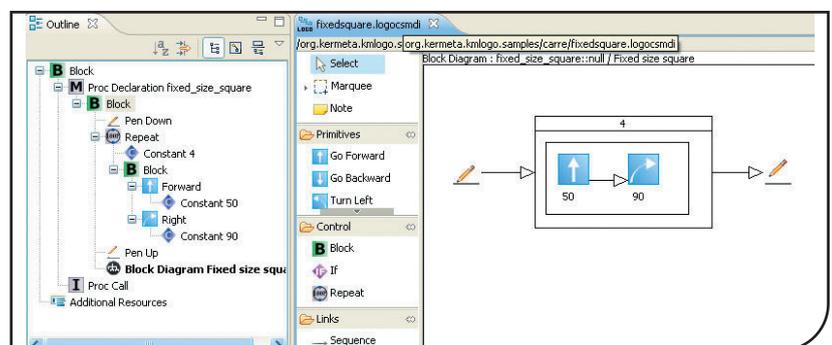


Figure 4. Exemple de modèle généré avec Topcased

actions que l'on souhaite sous forme d'opérations. Enfin, créer le simulateur pour le langage Logo revient donc à ajouter des méthodes `evaluate(context : Context)` sur chacune des instructions du langage. Cette méthode fait appel au contexte pour avoir des interactions avec le domaine d'application et enchaîner sur l'évaluation des instructions suivantes. En fait, c'est tout simplement une variation du patron de conception "Visiteur" qui a été simplifiée grâce à l'utilisation du tissage d'aspect de Kermeta.

```
package kmLogo::ASM;
require kermeta
require "../1.MetaModel/ASMLogo.ecore"
require "../4.VirtualMachine/LogoVMSemantics.kmt"
...
aspect class Block
{
  method eval(context : Context) : Integer is do
    instructions.each{instruction | result := instruction.
eval(context)}
  end
}
aspect class Forward
{
  method eval(context : Context) : Integer is do
    context.turtle.forward(steps.eval(context))
    result := void
  end
}
...
```

Pour lancer une simulation, il suffit de charger un modèle Logo et de lancer l'évaluation de la première instruction.

AMÉLIORATION DU SIMULATEUR

Le langage Kermeta étant principalement dédié à la manipulation de modèles, il ne propose pas par défaut de fonctions graphiques qui seraient trop spécifiques à un domaine et nuiraient à sa compacité. Néanmoins, il offre la possibilité d'accéder à du code java. C'est ainsi que nous pouvons réaliser une petite interface par exemple en AWT pour compléter les sorties textes du simulateur logo.

Parfois, vous pouvez vouloir augmenter les performances de votre simulateur. Dans ce cas, plutôt que d'utiliser la version dynamique de Kermeta (qui interprète le code Kermeta), vous pouvez utiliser son compilateur pour obtenir le code java EMF de votre simulateur. Il pourra ainsi être déployé directement sans Kermeta lui-même dans l'environnement de l'utilisateur final.

A ce stade de la mise au point de la sémantique du langage, le coût des évolutions reste raisonnable. Ceci ne sera probablement plus vrai une fois que vous aurez avancé dans les étapes suivantes et en particulier si vous avez construit un compilateur car leur coût de maintenance est généralement plus élevé.

VOUS AVEZ DIT SYNTAXE CONCRÈTE ?

Maintenant que les concepts de notre langage sont stables, nous pouvons commencer à capitaliser sur son usage. En particulier pour le rendre plus agréable à utiliser, nous souhaitons en améliorer l'IHM, rien de mieux alors que d'en définir une syntaxe dédiée par exemple textuelle ou graphique.

Gérer des syntaxes implique d'utiliser une ou plusieurs techniques, chacune d'entre elles pouvant nécessiter de longues explications. Nous nous contenterons ici d'évoquer les pistes utilisant des outils de l'IDM visant à vous faciliter cette tâche :

- Générer et customiser l'éditeur arborescent par défaut. Cela se fait grâce au genmodel d'EMF. En général, on commence par là, cela permet très facilement d'avoir une extension de fichier qui est propre à notre modèle, d'utiliser des libellés et des icônes plus parlantes. C'est d'autant plus intéressant que ces informations seront reprises par d'autres éditeurs plus évolués (typiquement dans la vue "outline" qui leur sera associée)
- Générer un éditeur graphique. Cela se fait grâce à un modèle permettant de faire le lien entre le langage et l'éditeur graphique. L'outil, (GMF ou Topcased) générera alors la plupart du code pour fonctionner dans Eclipse.
- Obtenir un éditeur textuel. Traditionnellement cela consiste à fournir un parser et un pretty printer pour notre langage (typiquement avec des outils comme sableCC ou antlr). Plus récemment, des outils suivant la même philosophie que pour les éditeurs graphiques ont commencé à apparaître pour obtenir des éditeurs plus intelligents à moindre coût, mais ceci est hors du spectre de cet article.

CONNECTONS-LE AU MONDE RÉEL

Nous avons choisi un robot construit avec Lego Mindstrom. Il est équipé de trois moteurs indépendants : deux pour piloter les roues, un pour actionner le stylo. Il faudra que nous transformions nos modèles de programme Logo en du code compréhensible par la machine. Pour cette tâche, il existe de nombreuses techniques et langages de transformations de modèle qui ont leurs avantages et inconvénients. Le choix dépend généralement de la complexité et du niveau de maintenabilité voulus pour la transformation à développer. Notre robot peut être programmé en NXC (Not eXactly C). Cette fois encore, nous utilisons ici les capacités de tissage pour compléter notre méta-modèle de langage avec un compilateur. En effet, pour une compilation simple comme celle dont nous avons besoin ici, il suffit de traverser ce modèle pour générer directement le code NXC correspondant. Si la transformation avait été plus complexe nous serions passés par un modèle intermédiaire spécifique au NXC avant de générer le code. Dans tous les cas, la structure de base du langage nous sert de trame directrice.

A notre avis, la construction d'un compilateur doit être faite après avoir fait un simulateur. Il y a bien évidemment des exceptions mais la mise au point et la maintenance d'un compilateur est sensiblement plus coûteuse que celle d'un simulateur et doit donc être faite seulement quand le langage est suffisamment stable. De plus, votre simulateur vous servira de version de référence pour tester les différentes implémentations concrètes. Dans notre exemple, nous visons une plate-forme Lego Mindstorm, mais déjà sans même parler d'un autre robot, nous pouvons envisager plusieurs dialectes et bibliothèques de programmation pour celui-ci !

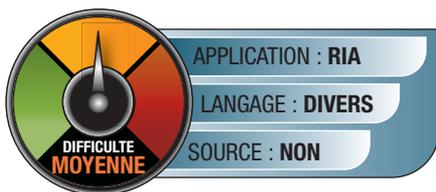
Ainsi il est possible de mettre au point des programmes dans notre nouveau langage tortue Logo sans avoir à dépendre de la lenteur du robot réel ou de la santé des batteries qui le meuvent. On voit ici vraiment tout l'intérêt des modèles qui donnent une abstraction de la réalité.



■ Didier Vojtisek - Ingénieur de recherche INRIA
www.irisa.fr/triskell - www.kermeta.org

Eclipse Tools for Silverlight : Silverlight sur Eclipse, c'est possible !

Silverlight est un plugin gratuit s'exécutant sur diverses plates-formes et dans différents navigateurs Web. Avec eclipse4SL, il devient désormais " multi-development environment " ! Eclipse4SL permet en effet de développer des applications Silverlight 2 directement dans l'environnement Eclipse. La communauté peut alors profiter de toute la richesse de Silverlight 2. [Fig.1]



Soyatec est une société française experte dans le développement de plugins pour Eclipse. Parmi ses produits on remarquera particulièrement eFace, un plugin compatible avec le langage déclaratif XAML pour décrire des interfaces graphiques Java. Cette particularité, Stève Sfartz (architecte Microsoft France) va la repérer. La suite logique, on la connaît : un partenariat est créé entre Microsoft et Soyatec, et le tout premier plugin Silverlight pour Eclipse est né : eclipse4SL. 6 mois après son lancement, faisons un état des lieux.

LES FONCTIONS

Visual Studio est certes un environnement très complet, le plugin eclipse4SL est néanmoins très bien équipé et en osmose avec les outils Microsoft. Les projets créés avec eclipse4SL sont complètement compatibles MSBUILD et donc avec

Visual Studio 2008 ou encore Expression Blend 2. Les projets eclipse4SL sont éditables aussi bien par Visual Studio 2008 que par la suite Expression Studio 2 et vice versa ! [Fig.2]

De ce fait, eclipse4SL peut importer des projets Silverlight dans l'environnement Eclipse. Il vous est proposé de créer une application Silverlight avec ou sans projet Web attaché, les projets sont exécutés dans un environnement Web via Chiron, un mini-serveur web embarqué dans le SDK de Silverlight. [Fig.3]

L'expérience de l'éditeur XAML est très intéressante puisqu'on est aidé par l'auto-complétion à la façon " IntelliSense ", on note la présence d'une palette qui permet de glisser-déposer des composants Silverlight (près d'une cinquantaine). On pourra générer aisément le corps des méthodes dès lors qu'on s'abonne à des événements côté XAML. L'éditeur de code C# est quant à lui

bien loin d'être complet, outre la coloration syntaxique et quelques snippets, il ne bénéficie pas d'assistant d'auto-complétion de code. Les erreurs sont remontées dans la console et dans la vue Problems. [Fig.4] [Fig.5]

Autre point ergonomique, l'explorateur de projet est similaire à celui de Visual Studio 2008, on retrouve donc la notion de " code-behind ", les différentes assemblies du projet sont concaténées dans un onglet References. On a la possibilité d'ajouter d'autres assemblies, de nouvelles classes C#, des User Controls, du



Fig.1



Fig.2

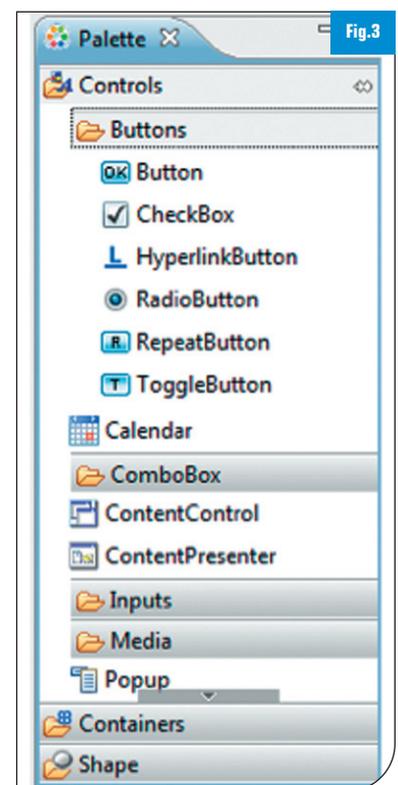


Fig.3

JavaScript, etc. L'éditeur XAML se dote d'un rendu instantané. Comme je vous le disais précédemment, la solution est compatible avec Visual Studio 2008 et la suite Expression Studio 2, il est même possible d'éditer le projet dans l'un de ces 2 environnements à l'aide du menu contextuel (si toutefois, ils sont installés sur votre machine). [Fig.6]

Le débogage est une feature prévue pour plus tard. Le projet peut être compilé et exécuté en s'appuyant sur MSBUILD et Chiron, la compilation packagera l'application pour pouvoir être utilisée dans n'importe quel projet Web.

L'INTEROPÉRABILITÉ

Non content d'avoir intégré Silverlight dans un environnement Java, l'équipe d'eclipse4SL pousse l'interopérabilité en proposant des scénarios en mode SOAP ou REST entre du métier Java et une couche d'exposition Silverlight. Celui-ci supporte en effet SOAP V1.1 et embarque un client HTTP qui lui permet d'interroger un service REST. La mise en place de ces différents scénarios est disponible sur leur site. Dans l'ensemble, le plugin possède de bonnes bases pour démarrer des projets Silverlight. Le point pénalisant à mon sens est le support limité de l'édition de code C# et aucun débogage de code possible, il reste néanmoins très intuitif dans son utilisation et se

rapproche beaucoup de l'environnement Visual Studio. Un point important à souligner : le plugin est Open Source sous licence EPL et disponible sur SourceForge :

<http://sourceforge.net/projects/eclipse4sl/>

SILVERLIGHT : UN PLUGIN

" cross-browser, cross-platform, cross..."

L'histoire ne s'arrête pas là, lors du Mix 09 qui a eu lieu à Las Vegas, Vijay Rajagopalan, architecte dans la division Stratégie Interopérabilité de Microsoft, faisait la démonstration du plugin, mais cette fois-ci sur Mac OS X ! Le développement d'applications Silverlight est désormais rendu possible sur plusieurs OS. On parle vraiment là d'un plugin Cross-Platform, voire Cross-Development Environment. [Fig.7]

Lors de cette même session, Vijay R. faisait la démonstration d'un POC (proof of concept) permettant de développer des applications PHP pour Windows Azure et tout cela dans Eclipse ! En bonus, on trouve même un Azure Storage Explorer. Aucune annonce officielle de la sortie de ce plugin n'a été communiquée.

CONCLUSION

Six mois après son lancement, eclipse4SL est déjà classé parmi les plugins les mieux notés de la

communauté. Beaucoup de chemin reste à parcourir avant de faire de l'ombre au mastodonte Visual Studio, ce n'est certainement pas l'objectif de ce plugin. Intégrer le développement d'applications Silverlight dans Eclipse c'est pouvoir profiter de la richesse de la plate-forme et de toute la communauté autour. Finalement, avec eclipse4SL et ce fameux POC sur Azure, Microsoft montre la volonté de s'ouvrir à d'autres univers et d'être de plus en plus interopérable. Affaire à suivre...

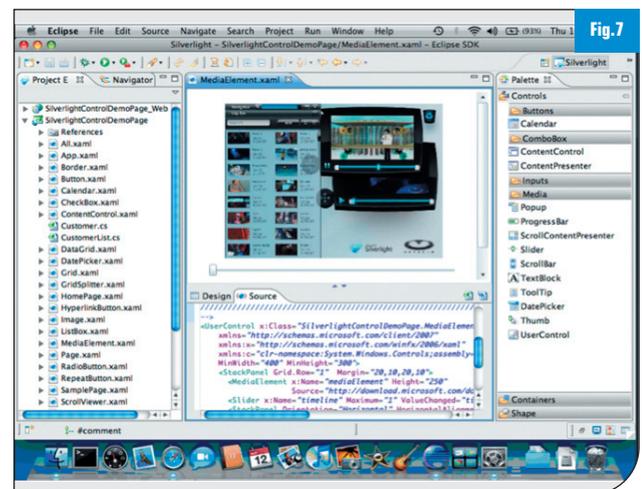
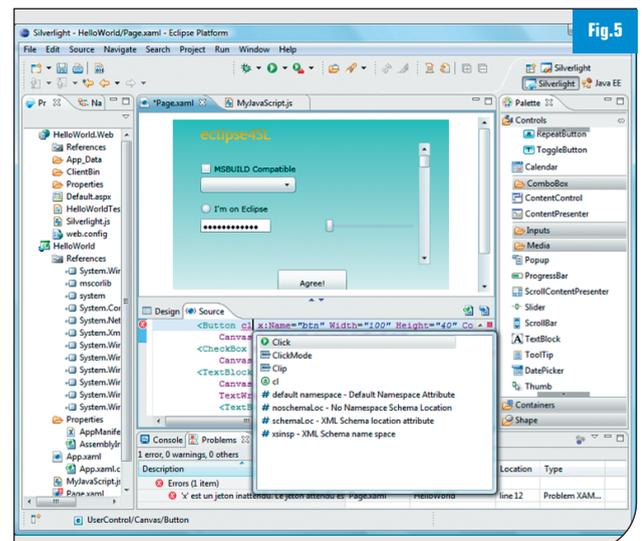
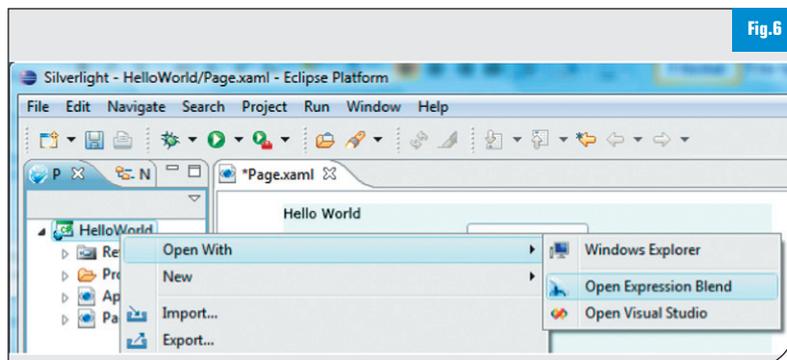
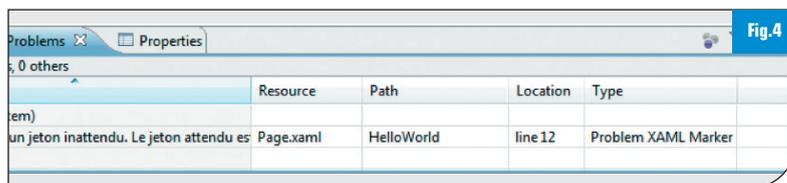
Ressources :

- Eclipse4SL : <http://www.eclipse4sl.org>
- Session de Vijay Rajagopalan au Mix 09 : <http://videos.visitmix.com/MIX09/T58F>

■ Ronny Kwon

Etudiant CSII3 à l'EPISI de Montpellier

<http://blogs.developpeur.org/ronnyk>



Utilisation de LINQ avec différentes bases de données

LINQ (Language Integrated Query) est un composant du framework.NET 3.5 qui permet d'interroger des données. Bien qu'on puisse effectuer des requêtes sur n'importe source de données, il exige que les données soient encapsulées dans des objets. Les requêtes LINQ sont soit exécutées directement par le moteur de traitement de LINQ, soit par un mécanisme d'extension géré par les providers LINQ.



Ces providers, implémentent un moteur de traitement des requêtes permettant de convertir les données dans un format différent afin de les exécuter sur divers stockages de données. Différents providers natifs pour LINQ sont disponibles mais deux technologies rivales de Microsoft vous permettent aujourd'hui de réaliser un développement de bases de données orientées objet. Ces technologies sont LinqToSql et ADO.NET Entity Framework.

LINQTOSQL

Ce provider a pour cible les bases SQL Server et SQL Server Compact uniquement. Puisque les données SQL Server se trouvent sur un serveur distant qui possède son propre moteur de requêtes, LinqToSql ne fait pas appel au moteur de requêtes de LINQ mais se contente de convertir la requête LINQ en requête SQL qui est ensuite envoyée à SQL Server pour être traitée. SQL Server stocke les données au format relationnel alors que LINQ interroge les données encapsulées dans des objets. Ces deux représentations doivent être " mappées " l'une à l'autre.

ADO.NET ENTITY FRAMEWORK

ADO.NET Entity Framework est un ORM (object-relational mapper) qui est inclus dans le Framework.NET 3.5 Service Pack depuis août 2008. Il permet également d'exécuter les requêtes LINQ par rapport aux entités du framework ADO.NET Entity Framework. Il est destiné à être ouvert à d'autres moteurs de bases de données comme l'annonce Microsoft.

ADO.NET ENTITY FRAMEWORK OU LINQTOSQL

Au premier coup d'œil ces deux technologies proposent des fonctionnalités similaires. Les aspects suivants vous permettront de choisir une des ces technologies pour votre développement :

Les avantages de LinqToSql :

- La prise en main est plus facile, et mieux adaptée pour un développement RAD.
- Possède une meilleure intégration des fonctionnalités orientées serveur, comme par exemple le type BLOB.
- Est moins compliqué qu'ADO.NET Entity Framework.

- performances supérieures à celles d'ADO.NET Entity Framework.
- Aucune restriction n'est à appliquer à vos objets afin qu'ils soient compatibles avec LinqToSql. Vous devez uniquement implémenter quelques interfaces.

Les avantages d'ADO.NET Entity Framework :

- Entity Framework est une approche conceptuelle du développement avec les bases de données. Cependant il nécessite plus de connaissances pour la prise en main et le maintien les développements.
- Est davantage portable sur différents serveur de bases de données car sa conception est plus abstraite que celle de LinqToSql.
- Intégration avec WCF (Windows Communication Foundation).
- Intégration avec ADO.NET Data Services.
- Intégration avec quelques services de Microsoft comme par exemple Reporting et Analysis Services.

Cependant aucune des deux solutions ne propose en natif le développement avec d'autres bases de données que SQL Server. La question suivante se pose alors :

Peut-on utiliser LINQ avec d'autres bases de données que SQL Server ?

Si vous utilisez une autre base de données que SQL Server, il est préférable de savoir si vous pouvez avoir recours à ADO.NET Entity Framework ou LinqToSql. Le tableau ci-dessous, répertorie les providers disponibles (la liste est non exhaustive) :

	LinqToSql (natif)	LinqToSql (tiers)	Entity Framework (natif)	Entity Framework (tiers)
SQL Server	Oui	-	Oui	-
Oracle	Non	DBLinq, LinqToOracle, LightSpeed, dotConnect	Oui	dotConnect, EFOracle
DB2	Non	-	Oui	-
MySQL	Non	DBLinq, LightSpeed	Non	dotConnect
PostgreSQL	Non	Npgsql, DBLinq, LightSpeed, dotConnect	Non	Npgsql, dotConnect

Dans le cadre de notre article, nous allons réaliser une application C# qui n'a pas une réelle utilité, à part la démonstration de l'utilisation de LINQ avec la base de données MySQL (version 5.1.30). Afin

de simplifier la démarche, nous allons créer une application de type console (sans le code GUI) qui permet de nous concentrer sur un cas concret d'utilisation. Première question à se poser : quel provider choisir pour réaliser le développement avec la base MySQL ? Dans le cas de MySQL, ni LinqToSql ni Entity Framework ne fournissent aucun provider en natif. MySQL AB est en train de développer son provider " Connector/.NET 6.0 " pour Entity Framework (actuellement il est disponible en version alpha), mais aucune date de sortie officielle n'est donnée au moment de l'écriture de cet article. Il est donc difficile de tester MySQL sur ADO.NET Entity Framework pour le moment, à moins d'avoir recours aux providers commerciaux tels que par exemple dotConnect de Devart ou LightSpeed de Mindscape.

Quant à LinqToSql nous pouvons également nous tourner vers les providers commerciaux (voir le tableau) ou utiliser une solution alternative telle que DBLinq qui aujourd'hui gère **MySQL, Oracle et PostgreSQL** en s'appuyant sur LinqToSql. Ce projet OpenSource a débuté en 2007, donc est relativement jeune, mais permet de faire les premiers tests avec MySQL. Dans le cadre de notre application, nous allons utiliser DBLinq v.O.18.

EXEMPLE : " GESTION COMMERCIALE ".

Dans notre exemple, nous allons développer une simple application appelée " Gestion Commerciale ". Pour pouvoir commencer notre projet nous allons avoir besoin des pré-requis suivants :

- Le serveur MySQL que vous pourrez télécharger à l'adresse suivante <http://dev.mysql.com/downloads/mysql/5.1.htm>.

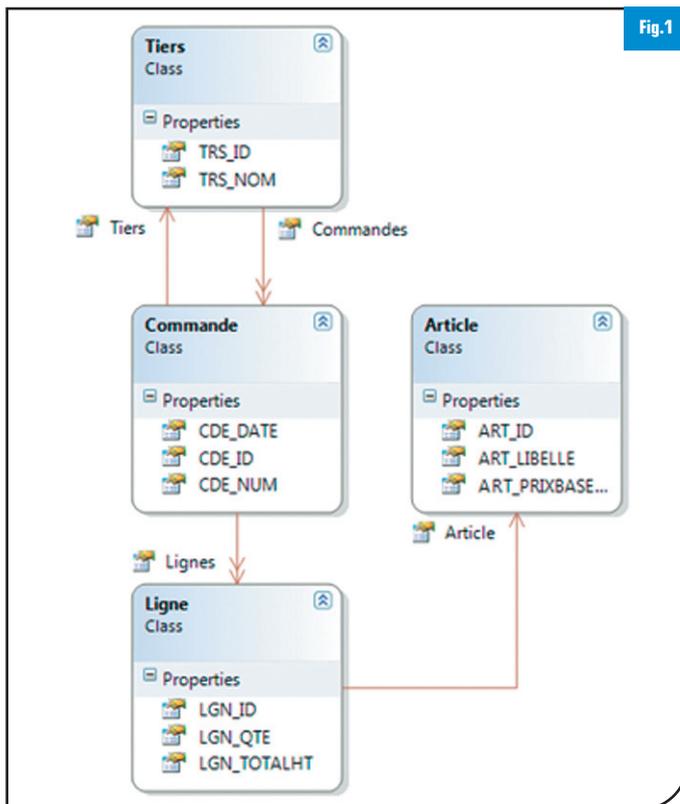


Fig.2

```

DbLinq Database mapping generator 2008 version 0.18.0.0
for Microsoft (R) .NET Framework version 3.5
Distributed under the MIT licence (http://linq.to/db/license)
>>> Reading schema from MySQL database
<<< writing C# classes in file 'GestComm.cs'
  
```

- La bibliothèque DbLinq v.O.18 <http://code.google.com/p/dblinq2007/>.

Modèle métier

Avant de nous lancer dans le développement de notre application, nous allons créer notre modèle métier que nous pouvons illustrer de manière suivante : [Fig.1]

Ce modèle est juste une partie simplifiée d'un système de prise de commandes. Il n'est pas complet mais cela est bien suffisant pour illustrer l'utilisation de LINQ avec une base de données MySQL.

DBMetal

Un fois notre base de données créée, nous devons générer nos classes. Il n'y a malheureusement pas de support graphique pour les modèles, à l'instar de LinqToSql. Le DataContext doit donc être généré via une adaptation de SQLMetal appelé DBMetal, comme aux premières heures de LINQ. DBMetal est un projet permettant de générer du code source en s'appuyant sur des schémas de bases de données. Vous trouverez le générateur DBMetal.exe dans l'archive zip de la bibliothèque DbLinq que vous avez téléchargée. Nous allons donc générer pour le besoin de notre application le DataContext appelé GestComm, il sera contenu dans le fichier du même nom avec l'extension .cs (C#). Afin de générer notre fichier nous devons taper la commande suivante dans la console Windows :

```

bin\DbMetal.exe -provider=MySQL -database:GestComm -server
:localhost -user:LinqUser -password:linq2 -namespace:GestComm
-code:GestComm.cs -sprocs
  
```

La ligne d'exécution est assez facile à comprendre et ne nécessite pas d'explications supplémentaires. Afin que la génération réussisse, faites bien attention que l'utilisateur " LinqUser " ait les droits en sélection à la table système mysql.proc qui est nécessaire pour la génération du code source. A la fin de la génération vous devriez avoir un message similaire à celui sur l'image : [Fig.2]

Ce fichier contient le mapping entre la base de données MySQL et le code source. Une fois le fichier généré, nous pouvons le joindre à notre application.

Code source de l'application.

Nous pouvons à présent passer à l'écriture de notre application. Il faut commencer d'abord par ajouter les bonnes références.

Références

Afin de pouvoir compiler votre application vous devez ajouter les références à votre projet vers les bibliothèques suivantes :

- DbLinq.dll
- DbLinq.MySql.dll
- MySql.Data.dll
- System.Data.Linq.dll

Ensuite en en-tête de votre code, ajoutez les références vers les espaces de noms suivants :

```

using System.Linq;
using DbLinq.Factory;
using DbLinq.Logging;
using MySql.Data.MySqlClient;
  
```

Se connecter à la base MySql

La connexion à la base de données s'effectue par le biais de notre DataContext (mapping de données) généré dans le fichier GestComm.cs. Il suffit d'instancier le DataContext en lui passant en paramètre la chaîne de connexion. Le code suivant illustre cette étape :

```
string dbServer = Environment.GetEnvironmentVariable("DbLinq
Server") ?? "nodeovmware";

string connStr = String.Format("server={0};user id={1}; password
={2}; database={3}", dbServer, "LinqUser", "linq2", "GestComm");

GestComm db = new GestComm(new MySqlConnection(connStr));
```

Pour faciliter le débogage de l'application, vous pouvez avoir recours au logger se trouvant dans l'espace de nom **DbLinq.Factory**. Vous pouvez le récupérer avec la ligne de code suivante :

```
var logger = ObjectFactory.Get<ILogger>();
```

Ensuite, son utilisation reste très simple. Il suffit de lui passer en paramètre la requête linq que vous voulez déboguer :

```
logger.Write(Level.Debug, "from p in db.article orderby p.ART
_PRIXBASEHT select p;");
```

Sélections

Dans un premier temps à titre d'exemple nous pouvons afficher la liste d'articles triée par le prix :

```
var q2 = from p in db.Article orderby p.ArtPriXBaseHT select p;
foreach (var v in q2)
Console.WriteLine(v.ArtID + " - " + v.ArtLiBelle + " : " + v.Art
PRIXBaseHT);
```

Vous pouvez le constater, l'opération est très simple, comme s'il s'agissait de LinqToSql avec la base SQL Server.

Nous pouvons aussi effectuer une récupération partielle des propriétés en utilisant une projection simple comme celle-ci :

```
var articlePartiel = from c in db.Article select new Article
{ ArtID = c.ArtID, ArtLiBelle = c.ArtLiBelle };
```

Cela permet de réduire le trafic réseau car vous sélectionnez et mettez à jour les propriétés dont vous avez vraiment besoin. Si vous essayez de faire la même chose dans LinqToSql vous obtiendrez l'erreur suivante **"Explicit construction of entity type 'Article' in query is not allowed."**

Les requêtes un peu plus compliquées s'exécutent également sans problèmes. La requête Linq ci-dessous affiche la liste des commandes dont l'article est la console "XBOX360" :

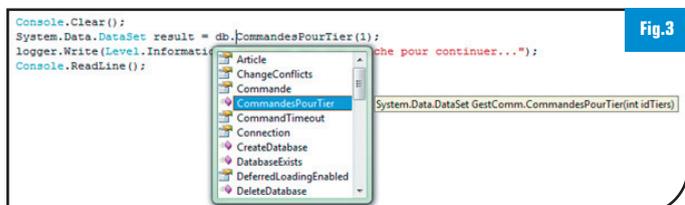


Fig.3

```
var q6 = from c in db.Commande from l in c.Lignes where l.Article.
ArtLiBelle == "XBOX360" select new { c, a = l.Article, l };
```

Procédures stockées

Quand vous générez votre fichier de mapping GestComm.cs (DataContext), les procédures stockées sont implémentées en forme des méthodes. [Fig.3]

Vous pouvez facilement les invoquer afin de les exécuter. Dans notre exemple, nous avons créé une procédure stockée **"Commande-PourTiers(idTiers int) "** qui prend en paramètre l'identifiant d'un tiers et retourne toutes ses commandes. Le résultat retourné est du type DataSet. L'extrait du code ci-dessous illustre son utilisation :

```
System.Data.DataSet result = db.CommandesPourTier(1);
```

Insertions

Il est également très simple d'insérer des données. Dans l'exemple ci-dessous nous allons insérer un tiers dans la base MySQL. Voici le code qui permet de le réaliser :

```
db.Tiers.InsertOnSubmit(new Tiers { TRSID = 4, TRSNOM = "Martin" });
db.SubmitChanges();
```

Suppressions

De même, nous pouvons aussi facilement supprimer le dernier tiers ajouté dans l'exemple ci-dessus. Voici l'extrait du code qui permet de réaliser cette opération :

```
db.Tiers.DeleteOnSubmit(db.Tiers.Last());
db.SubmitChanges();
```

CONCLUSION

Dans cet article, nous avons pris comme exemple la base de données MySQL. Cependant le projet DbLinq étant compatible avec d'autres plates-formes, nous pouvons choisir un autre serveur de données tel que par exemple Oracle, PostgreSQL, Ingres, SQLite, Firebird ou même SQL Server. Comme vous pouvez le constater nous avons réussi à réaliser notre petite application d'essai bien que DbLinq comporte encore des lacunes fonctionnelles. D'après ce que nous pouvons lire sur le site web officiel du projet, les fonctionnalités majeures manquantes sont :

- Manque de la gestion des transactions.
- Non support des sous-requêtes.
- Non intégration de LinqToEntities.

Nous pouvons également constater que le développeur ne dispose pas d'une interface d'intégration graphique comme LinqToEntities. De plus, un manque quasi total de documentation est un point handicapant. Il faut également se poser la question de la pérennité d'une telle solution. Cependant, si vous ne voulez pas investir dans des composants tiers souvent onéreux, vous pouvez vous tourner vers DbLinq, car son utilisation ainsi que sa prise en main est très facile, comme vous avez pu le constater à la lecture de cet article.

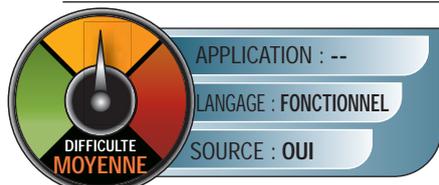
Références

Devart dotConnect : <http://www.devart.com/dotconnect/linq.html>
 DBLinq : http://code2code.net/DB_Linq/
 LightSpeed : <http://www.mindscape.co.nz/products/lightspeed/>

■ Thomas **JASKULA** - Chef de Projet .NET

F# : l'avenir de .Net et du développement Windows ?

F# est un langage fonctionnel qui possède un noyau compatible avec OCaml, il s'inspire également de C# et Haskell. Aujourd'hui, F# est un projet du laboratoire Microsoft Research. Il pourrait être intégré au prochain Visual Studio (VS 2010) qui devrait voir le jour fin 2009, début 2010.



Les langages fonctionnels sont particulièrement bien adaptés pour l'écriture d'applications scientifiques ou d'intelligence artificielle. A

ce jour, Microsoft n'édite pas un tel langage pour la plate-forme .net, F# vient donc combler ce manque. F# est un langage .net, cela signifie que comme C#, VB.net, Boo ou d'autres langages, il repose sur un socle commun : la CLS (*Common Language Specification*). Ainsi, F# a accès à l'ensemble des bibliothèques .net, c'est-à-dire qu'il est utilisable pour écrire une application WPF ou ASP.net, ou simplement une assembly utilisée par une application C#. Concrètement, cela veut dire que le compilateur génère une assembly .net classique qui sera exécutée par la machine virtuelle .net ; F# a donc plus ou moins les mêmes performances que C#. F# et OCaml partagent les mêmes syntaxes et fonctions de base, certaines librairies faites en OCaml peuvent être compilées directement via F#, sans changement ou seulement quelques modifications mineures. Contrairement à C#, F# ne définit pas seulement une grammaire, il apporte également un ensemble de fonctions prédéfinies.

LES DIFFÉRENTS PARADIGMES

Il existe plusieurs catégories de langages répartis selon des concepts différents ; on parle de paradigme pour désigner un ensemble de concepts particuliers. Les langages de programmation impératifs représentent le premier paradigme de programmation: le langage C en est probablement le meilleur représentant. Le paradigme orienté objet a connu le succès avec le C++ et, au début du XXI^{ème} siècle grâce aux langages Java puis C#. Les autres paradigmes, comme la programmation fonctionnelle, sont moins utilisés, ce dernier comprend LISP, Scheme et F# entre autres. Arrêtons-nous sur la programmation fonctionnelle, ce paradigme consiste à voir un programme comme un ensemble de fonctions mathématiques. Ces différentes fonctions peuvent s'emboîter, on les voit en fin de compte comme des boîtes noires qui prennent des paramètres en entrée et les transforment en une sortie. A l'inverse de la programmation impérative, la programmation fonctionnelle rejette le changement d'état ainsi que la mutation des données ; par défaut, toute opération d'assignation est interdite. [Fig.1]

LE LANGAGE F#

Avant de commencer avec une série d'exemples, il est nécessaire de maîtriser quelques concepts de base. Tout d'abord, il existe en F# un mot-clé permettant la déclaration de liste, variable et fonction ; c'est le mot-clé *let*. Ensuite, F# est un langage fortement typé, pour cela il utilise massivement ce qui est appelé l'inférence de type (*type inference*). Ce concept permet de ne pas spécifier explicitement le type

Fonctionnel	Objets	.net	Outils
<ul style="list-style-type: none"> Fortement typé Peu verbeux Inférence (type) Data Types et Patterns Meta-Programming 	<ul style="list-style-type: none"> Modèle objet de .net Interopérabilité 	<ul style="list-style-type: none"> Visual Studio Librairies Outils LINQ 	<ul style="list-style-type: none"> Compilateur F# F# Interactive Intégration dans Visual Studio

de l'élément déclaré. Le compilateur va analyser le contexte actuel puis va déduire le type automatiquement.

Le reste du dossier présente quelques fonctionnalités et concepts qu'il nous a semblé intéressant d'aborder.

FONCTIONNALITÉS ET CONCEPTS

Les bouts de codes ci-dessous ont été réalisés et compilés avec Visual Studio 2008 en utilisant la CTP de septembre 2008 (1.9.6.2) de F#.

Hello World

Le classique Hello World ci-dessous présente une manière d'afficher une chaîne de caractères sur la console. Nous utilisons ici la méthode F# *printfn* mais la méthode *System.Console.WriteLine* du framework .net aurait également pu faire l'affaire.

```
#light

(* Hello World
Une fonction sayHello qui prend en paramètre hello *)
let sayHello hello = printfn "Hello %s" hello
printfn sayHello "World" // Appelle sayHello avec World comme paramètre
```

En exécutant ce code, le résultat suivant est obtenu :

```
> Hello World
```

Etant donné que nous travaillons en F#, il est plus logique d'utiliser la fonction *printfn* car elle se combine parfaitement avec le système de types de F#. Actuellement, tout code doit commencer avec l'instruction *#light* ou *#light "off"*. Dans les futures releases, *#light* devrait être ajouté par défaut si rien n'est spécifié. *#light* a pour but de simplifier le code en rendant les espaces significatifs. De cette manière, certains mot-clés tels que *in*, *begin* et *end* peuvent être omis. La deuxième variante est à utiliser pour conserver une compatibilité avec le langage OCaml. Ce premier exemple montre également l'utilisation des commentaires. Deux barres obliques (slash) permettent de faire un commentaire sur une seule ligne alors que (* et *) permettent d'ouvrir, respectivement fermer, un bloc de commentaire.

Les fonctions d'ordre supérieur

Une fonction d'ordre supérieur est une fonction qui prend ou retourne une autre fonction en argument. Cela permet d'augmenter radicalement la généricité du code. Ce concept n'est aucunement lié à la programmation fonctionnelle, mais son implémentation est, une fois de plus, plus concise que dans d'autres langages de programmation comme nous allons le voir avec un exemple.

Supposons que nous souhaitons faire des fonctions de comparaison qui s'appliquent sur une liste. Nous aimerions pouvoir trier cette liste par ordre croissant, décroissant, par ordre de valeur absolue, ou n'importe quel autre ordre. Une possibilité serait de définir une méthode pour chacune des fonctions dont nous avons besoin. Cette approche n'est évidemment pas générique et nous aurions besoin de beaucoup de code similaire.

Avec les fonctions d'ordre supérieur, nous définissons une fonction (disons *f*) qui va prendre en argument notre fonction (disons *t*) ainsi qu'une liste d'éléments à trier (disons *e*). De cette manière, nous pouvons envoyer n'importe quelle fonction de tri *t* à *f* pour ordonner *e*. Voici la manière de procéder en F# pour faire une fonction qui trie les éléments par ordre de valeur absolue. Pour commencer, on définit *t*, notre fonction de tri. Cette fonction prend en paramètre deux entiers à comparer (appelés ici *x* et *y*) et renvoie 1, 0 ou -1 selon le résultat de cette comparaison.

```
let t x y =
    if abs x > abs y then -1
    elif abs x = abs y then 0
    else 1
```

Maintenant que notre fonction *t* est définie, il nous faut une fonction *f* qui prend une fonction de tri en paramètre et une liste à trier. Cette fonction est déjà présente au sein de F#, *List.sort*, nous ne la redéfinissons donc pas. L'appel se fait comme ceci

```
let l = List.sort t [-2; 5; -10; 7; 3]
printf "%A" l

> [-10; 7; 5; 3; -2]
```

Nous pouvons créer cette fonction d'ordre supérieur *f*. Par exemple, pour en faire une qui retourne la valeur absolue la plus élevée entre deux entiers, nous faisons :

```
let f t x y =
    if t x y < 0 then x
    else y
```

La fonction *f* est donc une fonction d'ordre supérieur car elle prend *t*, une autre fonction, qui retourne une valeur de 1, 0 ou -1 selon que le premier argument est soit plus petit, égal, ou plus grand que le deuxième. L'appel se fait donc comme ceci :

```
let a = f t 4 -5
printf "%A" a
```

Et la fonction nous retourne -5 car c'est la valeur absolue la plus grande entre 4 et -5. Cette notion de passage de fonction n'est pas spécifique à F# ni même aux langages de programmation fonctionnels, comme nous l'avons déjà dit ci-dessus.

A titre de comparaison, voici un code similaire fait en C# qui, lui, n'est pas un langage fonctionnel (mais objet uniquement). Premièrement, une fonction qui compare deux entiers comme dans le code F# et qui retourne le plus grand en valeur absolue

```
var t = new Comparison<int>((x, y) =>
{
    if (Math.Abs(x) > Math.Abs(y)) return -1;
    else if (Math.Abs(x) < Math.Abs(y)) return 1;
    else return 0;
});
```

Pour trier la liste, nous appliquons notre fonction sur la classe *List*, comme en F#, et les éléments sont triés :

```
List<int> l = new List<int> { -2, 5, -10, 7, 3 };
l.Sort(t);

> [-10; 7; 5; 3; -2]
```

Tout comme en F#, nous pouvons créer une fonction d'ordre supérieur *f*, qui va donc prendre une fonction de tri *t* en paramètre. Voici comment procéder :

```
int f(Comparison<int> t, int x, int y)
{
    return t(x, y) < 0 ? x : y;
}
```

En testant avec :

```
f(t, 4, -5);
```

Nous obtenons comme dans le code F#, la valeur -5. Grâce à cet exemple, nous constatons que le passage de fonction n'est pas réservé aux langages fonctionnels (d'autres langages, comme C++, possèdent également les pointeurs sur fonctions). L'implémentation en F# a cependant l'avantage d'être plus concise.

Les listes

Les listes dans un langage fonctionnel sont omniprésentes et F# ne déroge pas à cette règle. Afin de créer une liste, le plus simple est d'utiliser les crochets et séparer les valeurs par des points-virgules :

```
let listA = [ 1; 2; 3 ]
```

Il est également possible d'utiliser la méthode *create* de l'objet *Array*. L'exemple ci-dessous est équivalent à l'exemple précédent :

```
let arr = Array.create 3 1
arr.[1] <- 2
arr.[2] <- 3
```

F# donne aussi la possibilité de définir une liste à partir de ces bornes inférieures et supérieures : on parle alors de "range list"

```
let listB = [1..10]
```

Mieux encore, on peut faire des listes générées (generated list), c'est-à-dire que la liste sera créée à partir d'une fonction. L'exemple ci-dessous crée une liste contenant le carré des nombres 1 à 10 :

```
let listC = [ for x in 1..10 -> x*x ]
```

On peut obtenir une liste en extrayant une sous-liste d'une autre liste :

```
let listD = listC.[0..2]
```

Enfin, on peut concaténer 2 listes via l'opérateur *@* :

```
let listE = listA @ listB
```

Fonction map

Puisque les langages fonctionnels utilisent très souvent des listes, il est donc intéressant de présenter la méthode `map`. Cette dernière applique une certaine fonction sur tous les éléments d'une liste. Prenons comme exemple le code ci-dessous

```
let l = List.map (fun x -> x * x) [1 .. 10]
printf "%A" l
```

La méthode `map` prend deux paramètres : la fonction à appliquer sur chaque élément de la liste d'une part ainsi que la liste en question d'autre part. Dans notre cas, on passe une fonction qui élève la valeur passée en paramètre au carré (premier paramètre) et une liste allant de 1 à 10 (deuxième paramètre). La méthode `map` nous retourne une liste contenant les éléments de la liste passés en deuxième paramètre élevé au carré. Le premier argument qui se charge d'élever un élément au carré ne porte pas de nom et est donc appelé fonction anonyme. La fonction anonyme est construite grâce au mot-clé `fun`. Le résultat obtenu après exécution est le suivant

```
> [1; 4; 9; 16; 25; 36; 49; 64; 81; 100]
```

Les tuples

Une des fonctionnalités pratique mais pas implémentée à ce jour en C# est la possibilité de faire retourner plusieurs variables à une méthode. F# le permet grâce à l'utilisation des *tuples*, une fonction peut ainsi retourner plusieurs résultats. Par exemple

```
let rec f = fun x -> x, x*x
let a, b = f 2

printf "a = %i\nb = %i" a b
> a = 2
> b = 4
```

Le mot-clé mutable

En F#, lorsque nous assignons une valeur à une variable à l'aide de l'opérateur d'assignation, il n'est plus possible d'en modifier la valeur. Cependant, dans certains cas, nous aimerions passer outre ce comportement. C'est à ce moment qu'entre en jeu le mot-clé `mutable` qui permet de surpasser cette restriction.

```
let mutable mut = "First value"
printfn "Current value '%s'" mut

(* Sans le mot-clé mutable lors de la définition,
cette affectation va créer une erreur de compilation *)
mut <- "Second value" // On utilise pas = mais <-
printfn "Current value '%s'" mut

> Current value 'First value'
> Current value 'Second value'
```

Puisque notre variable a été déclarée avec le mot-clé `mutable`, nous pouvons lui assigner une valeur après l'initialisation, mais en utilisant le signe `<-` au lieu du traditionnel `=`.

Les références

Une autre solution pour modifier la valeur d'une variable consiste à utiliser les références. Ce procédé peut être comparé aux pointeurs : nous stockons l'adresse d'une valeur dans une variable et en changeant l'adresse de celle-ci, nous la faisons pointer sur une autre

valeur. Pour déclarer une variable en tant que pointeur, il faut utiliser le mot-clé `ref` ; le compilateur s'occupera automatiquement de trouver de quel type il s'agit et ce type ne pourra plus être modifié par la suite. Par exemple, en déclarant une référence sur une valeur de type `int`, il faudra toujours lui passer des adresses qui pointent sur des valeurs du même type sinon une erreur sera levée.

```
// Définition de la référence sur une valeur de type integer
let i = ref 1
printfn "i: %d" !i // Affiche 1
// Déréférencement avec l'opérateur !
i := !i + 1
printfn "i: %d" !i // Affiche 2

> i: 1
> i: 2
```

Le code ci-dessus nous montre que l'assignation d'une nouvelle valeur à une variable de type référence se fait avec l'opérateur `:=`. Le déréférencement de la variable se fait quant à lui avec l'opérateur `unaire !`

Pipeline

F# permet l'utilisation de pipeline grâce à l'opérateur `|>`. Un pipeline permet d'envoyer le résultat d'une expression dans l'entrée de l'expression suivante. L'exemple utilisé dans le paragraphe sur la fonction `map` peut alors s'écrire comme suit

```
let l = [1..10]
        |> List.map (fun x -> x*x)
printf "%A" l

> [1; 4; 9; 16; 25; 36; 49; 64; 81; 100]
```

Cette fonctionnalité permet de facilement chaîner les méthodes et rend la lecture du code plus aisée. Voici un autre exemple illustrant davantage l'utilité des pipelines

```
let l = [1..10]
        |> List.filter (fun x -> x % 2 = 0)
        |> List.map (fun x -> x*x)
        |> List.rev

printf "%A" l

> [100; 64; 36; 16; 4]
```

Sans les pipelines, nous aurions dû écrire :

```
let l =
    List.rev (
        List.map(fun x -> x*x) (
            List.filter (fun x -> x % 2 = 0) [1..10]
        )
    )
```

Nous poursuivrons la découverte de F# le mois prochain. Merci à Damien DOLIGEZ de l'INRIA pour sa relecture et ses conseils



■ Nicolas Biedermann - MVP C#
<http://www.nicolasbiedermann.ch>



■ Cyril Durand - MVP ASP.net
Consultant indépendant
<http://blogs.developpeur.org/Cyril>

En direct des labos !

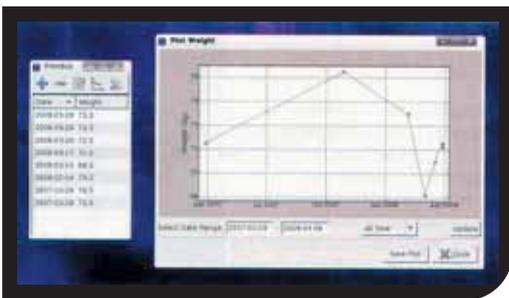
Dans les laboratoires, écoles, centres de recherche, on découvre parfois de petites merveilles. Voici quelques exemples que nous avons retenus ! Bonne découverte.

Pondus

(www.ephys.de/software/pondus)

Voici un gestionnaire personnel de poids (personal weight manager), écrit en Python et Gtk+2, placé sous la licence GPL. Son usage est très simple, et il est de surcroît léger et rapide.

On peut y tracer la courbe des données pour obtenir un aperçu rapide de l'historique de son poids, les données étant stockées sous forme de fichiers XML pour un accès instantané et une modification avec d'autres programmes. Il est simple, en effet, et l'installation n'est pas non plus trop complexe. Pondus vous permet de suivre pas-à-pas votre poids sur une longue période et il affiche vos progrès à l'aide d'un graphe. Il peut également basculer entre des mesures métriques (" en mètres ") et des mesures impériales (" Système Impérial Britannique de mesure de poids ").



L'adorable minimalisme de Pondus et un exemple de diagramme pondéral.

Installation. En termes de dépendances, il vous faudra disposer de l'installation de quelques bibliothèques Python avant de commencer. A chaque fois que vous compilez quelque chose, l'installateur exigera systématiquement les fichiers de développement. Alors, assurez-vous que vous installez les fichiers python-dev en premier. Si vous rencontrez toujours des problèmes, une recherche sur Google permettra de trouver certains packages dont dépend Pondus :

- python 2.4.4-6
- python-gobject 2.14.1-1
- python-gtk2 2.12.1-1
- python-matplotlib 0.90.1-2
- python-support 0.7.6

Une fois le problème des dépendances résolu, téléchargez le code source du site web du projet, extrayez-en le contenu et ouvrez un terminal dans le nouveau répertoire.

Comme *racine*, entrez la commande :

```
# python setup.py install
```

Usage : Pondus est très minimaliste, mais ce n'est pas forcément une mauvaise chose. En tapant Pondus, vous verrez une petite fenêtre munie de cinq boutons. Le premier ajoute une ligne de données – celui de votre poids et la date d'entrée.

Le second efface une ligne et le troisième affiche une ligne de données. Une fois que vous avez entré des poids et des dates, vous pouvez alors les afficher sous la forme d'un graphe en cliquant sur le quatrième bouton.

Si vous voulez passer de livres (" pounds ") à kilogrammes, le cinquième bouton ouvre la fenêtre des " settings " et vous permet de changer cela (c'est l'une des deux seules options, l'autre permet de se rappeler de la taille de la fenêtre).

Garder trace de vos progrès est central dans Pondus. Nous arrivons maintenant à la section graphique – le quatrième bouton, ou *dessiner les données*, **Plot data**. Cliquez sur ce bouton, et une fenêtre intitulée **Plot Weight** apparaît avec un graphique à ligne nette représentant votre poids au cours d'une période. Si vous regardez tout en bas à droite, il y a une boîte " glisser-déplacer " (drop-down box) avec **All Time** écrit à l'intérieur. Ceci vous permet de filtrer le reste des informations par rapport à vos résultats de l'année précédente, ou juste le mois dernier. Si vous voulez filtrer votre période par rapport à quelque chose de plus spécifique, il y a deux champs tout en bas à gauche appelé **Select Date Range**.

Tapez la date de début que vous voulez voir apparaître dans le premier champ et la date de fin dans le second champ, cliquez sur **Update** à l'extrême droite, et le graphique fera la mise à jour avec les informations sélectionnées.

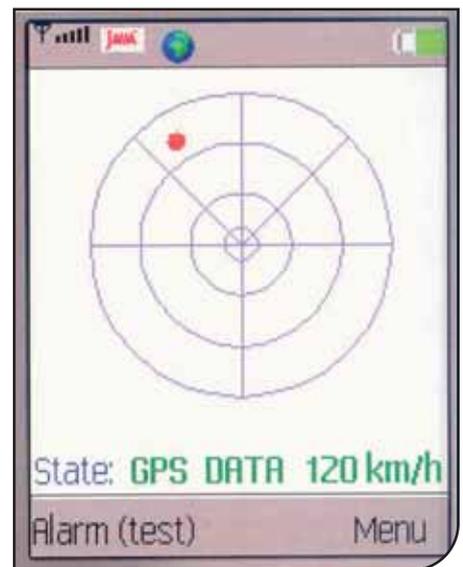
Pour ceux d'entre vous qui veulent enregistrer une copie de leurs progrès, en cliquant

sur le bouton **Save Plot** tout en bas, vous pourrez sauvegarder votre graphique dans un fichier *.png*.

FoxyTag

(www.foxytag.com)

Voici un projet dont j'ai hâte de voir le résultat : un système gratuit d'avertissement anti-radar conçu pour fonctionner sur une grande variété de téléphones portables et de GPS. FoxyTag est un système collaboratif conçu pour inciter les utilisateurs à partager des données radar – plus il y a d'utilisateurs et de feedback, plus le système devient fiable.



FoxyTag, un système populaire anti-radar

Le système ne suppose pas simplement qu'un radar se trouve à un endroit particulier. Les utilisateurs ont la possibilité de signaler la présence permanente d'un radar, de son installation ou de sa suppression. Tout téléphone portable JAVA équipé de MIDP 2.0, CLDC 1.1 et Bluetooth devrait être compatible. Pour les systèmes GPS, tout GPS Bluetooth devrait être compatible (y compris les modules GPS de certains systèmes de navigation),

■ Dr. Rodrigue Sabin Mompelat
Enseignant-Chercheur
Ingénieur Logiciel - Copenhague - Danemark

PARALLELISME

Programmation concurrente en Java

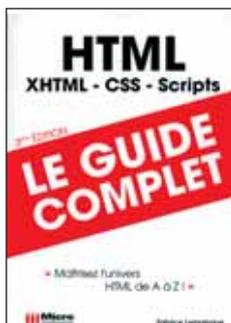


Difficulté : ****
 Editeur : Pearson
 Auteur : Brian Goetz
 Prix : 39 €

La programmation concurrente est un des défis majeurs des développeurs pour paralléliser leurs codes. Plus que jamais, le développeur doit connaître et maîtriser l'ensemble des contraintes du parallélisme et la concurrence fait partie de ces problèmes. Il s'agira tout d'abord de définir le sujet et les contours de la concurrence. Puis on rentrera dans le vif avec les éléments basiques à maîtriser absolument : thread safety, partage des objets, la composition de ceux-ci... Ensuite, on attaque la concurrence : exécution des tâches, leurs annulations et arrêts, le pool de thread... Au-delà de cela, se pose la difficulté de stabilité de la concurrence dans le code et la nécessité d'avoir un code optimisé et robuste, d'où l'importance des tests. Et pour ceux qui poursuivront leur lecture jusqu'au bout, l'auteur propose quelques morceaux de choix : verrous explicites, synchronisation non bloquante. Parfois ardu, ce livre nécessite une bonne connaissance de Java mais les principes de la concurrence peuvent s'appliquer à d'autres langages. De nombreux codes, des conseils émaillent les pages ! Incontournable !

WEB

HTML, le guide complet 3e édition



Difficulté : **
 Editeur : Micro Application
 Auteur : Fabrice Lemainque
 Prix : 22 €

HTML et cie ne sont pas morts. La preuve, avec la 3e édition de ce guide complet. Le but ici est d'expliquer le langage et de rapidement pouvoir programmer ses propres sites. L'auteur y présente les bases à connaître : les fonctions, les balises, le fonctionnement de HTML, XHTML, etc., la sécurité, les données, etc.

Mais dynamisme oblige, les CSS et scripts ne sont pas oubliés. On regrettera tout de même l'absence de HTML 5.



WEB

Du web 2.0 au web 3.0

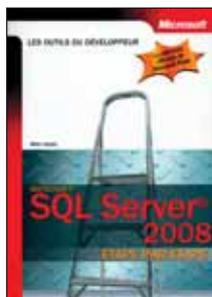
Difficulté : *
 Editeur : Eni éditions
 Auteur : Jean-Noël Anderruthy
 Prix : 20,95 €

Eh oui, sans le savoir nous sommes déjà passés au web 3.0, et on parle déjà du web 4.0 (100 % cloud). Le titre est trompeur car en réalité, l'ouvrage aborde ce qui est déjà là, tout un ensemble de services, plus ou moins " nouveaux " mais qui sont dans la mouvance du web 2.

On parle de flux RSS, de widgets, de mobilité du web. Quelques éléments pourraient être reliés au " web 3 " même si dès maintenant, on peut les mettre en œuvre, par exemple la nouvelle génération de moteur de recherche (tourné vers le côté sémantique) ou encore HTML 5, les web slices... Une grosse partie est aussi consacrée aux applications et documents en ligne, à la collaboration, la cartographie. Le développeur web ou toute personne voulant se tenir au courant, trouvera là un condensé des possibilités actuelles du web.

SGBD

SQL Server 2008



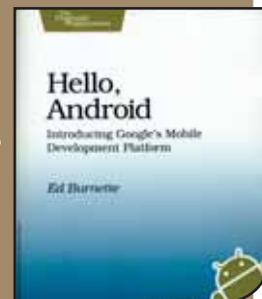
Difficulté : **
 Editeur : Microsoft Press
 Auteur : Mike Hotek
 Prix : 39 €

Apprenez pas à pas SQL Server 2008 avec un guide pratique qui vous explique comment mettre en œuvre des bases de données pour répondre à des besoins professionnels réels. Tout y passe : installation, configuration, création de table, cluster, sécurisation, optimisation et performance. Une centaine de pages complémentaires sur la Business Intelligence et de nombreux exemples de code sont disponibles dans la rubrique Suppléments en ligne associée à cet ouvrage sur le site dunod.com.

LIVRE DU MOIS

Hello, Android

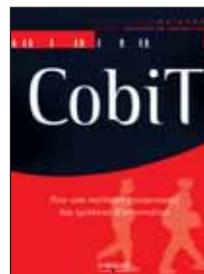
Difficulté : ** / ***
 Editeur : O'Reilly
 Auteur : Ed Burnette
 Prix : 26 €



Vous rêvez de développer pour Android, le nouveau système en vogue pour téléphone ? Pourquoi ne pas se lancer dans sa découverte avec Hello, Android. L'auteur plonge au cœur du kit de développement et des API. On démarre par l'installation des outils et du SDK, puis on fait le tour du propriétaire et notamment de l'émulateur. L'auteur aborde ensuite les briques fondamentales du système avec des fonctions basiques comme l'interface (et le design à adopter, le multimédia, les données locales, etc.). Bien entendu, pour le développeur qui voudrait aller plus loin, il existe de nombreuses fonctions avancées que l'on retrouve dans ce livre telles que OpenGL, la localisation. Détail intéressant, on retrouve les sous-ensembles du langage. Plaisant à lire et instructif, un ouvrage incontournable sur Android. En anglais.

ORGANISATION

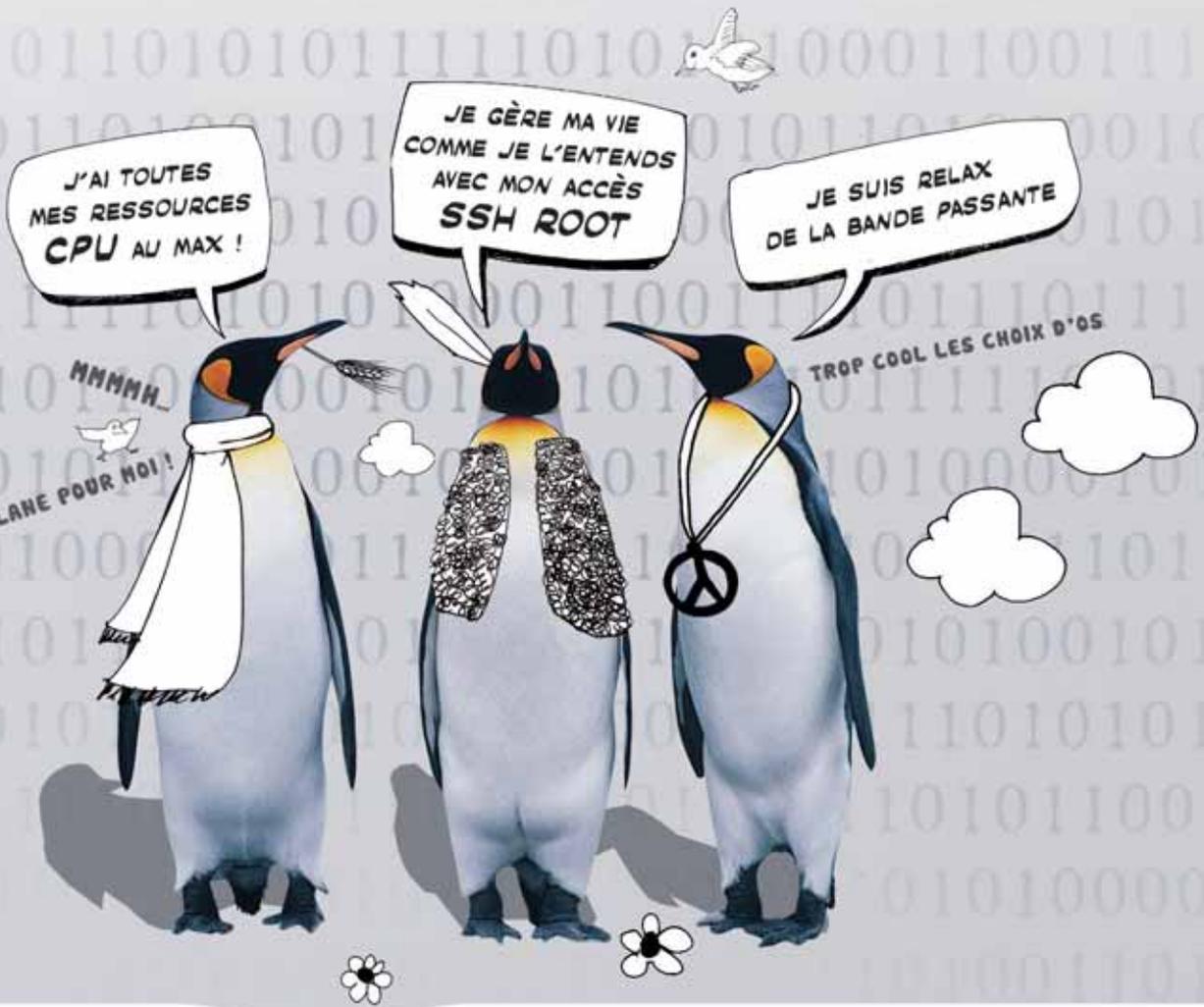
CobIT



Difficulté : ** / ***
 Editeur : Eyrolles
 Auteur : collectif
 Prix : 39 €

Référence incontournable au sein de la communauté des auditeurs informatiques depuis plus de dix ans, CobIT (Control Objectives for Information and related Technology) est devenu un standard de la gouvernance des systèmes d'information. Les auteurs s'appuient sur la version 4.1 de la méthode. La première partie dresse un panorama des différents référentiels existants, en décrivant leurs champs d'action et leur positionnement vis-à-vis de CobIT. Dans la deuxième partie sont détaillés les 34 processus de CobIT selon un plan standard, avec mise en lumière de leurs forces et faiblesses. Enfin, la troisième partie expose des cas pratiques d'utilisation et de déploiement, correspondant à un véritable mode d'emploi du référentiel.

Nouveau VDS+ d'Amen : le bonheur est dans le serveur !



REFLEXION/FAITE Conditions générales de vente sur le site www.amen.fr AMEN RCS PARIS : B 421 527 797.

À partir de
5€ HT/mois
soit 5,98€ TTC/mois*

**SERVEURS PRIVÉS AMEN :
BÉNÉFICIEZ DE
RESSOURCES GARANTIES
QUI VOUS SONT PROPRES
(PROCESSEUR,
MÉMOIRE, DISQUE DUR...)
TOUT EN PROFITANT
D'UNE PLATEFORME
INFOGÉRÉE 24H/24 - 7J/7.**

- Hébergement multi-sites/multi-domaines
- Interface d'administration : Plesk 8.6
- Systèmes d'exploitation : Fedora Core 8, Suse 10.3, Debian 4.0, Ubuntu 8.04 ou CentOS 5
- Part CPU minimum : de 1 à 6
- Mémoire garantie : de 256 Mo à 1 Go
- Espace disque : de 5 Go à 30 Go
- Bases de données : illimitées
- 1 adresse IP fixe
- Accès Root

Amen et Dada : 1,4 million de domaines gérés et plus de 500 000 sites hébergés.



0 892 55 66 77 (0.34€/mn) - www.amen.fr

NOMS DE DOMAINE - EMAIL - HÉBERGEMENT - CRÉATION DE SITE - E-COMMERCE - RÉFÉRENCIEMENT

* Pour un engagement annuel.

