



www.programmez.com

grammmez!

LE MAGAZINE DU DÉVELOPPEMENT

100% Pratique ! Spécial .Net 3.0

*Windows Presentation Foundation, Windows Workflow Foundation, Windows Communication Foundation
Visual Studio Orcas, Ajax*

Web 2.0

- Réalisez votre player MP3 avec WPF / E
- Maîtrisez la 3D dans WPF
- ASP.Net Ajax : Ajax selon Microsoft !

Workflow Foundation

Intégrez des Workflows dans SharePoint Services

Office 2007

Générer des documents OpenXML

Vista

- ✓ ORCAS : découvrez le prochain Visual Studio !
- ✓ Migrer les applications

Communication Foundation

- ✓ La sécurité dans WCF
- ✓ Gérer les exceptions

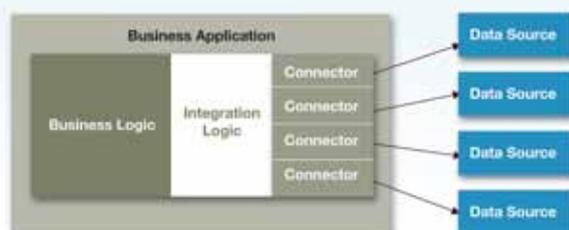
M 02104 - 7 H - F : 5,95 € - RD



Vous voulez accéder facilement
aux données et aux services
où et quelque'ils soient
dans votre entreprise.



Integration Traditionnelle



Intermediation Xcalia



Xcalia fournit un logiciel d'intégration dynamique pour les entreprises agiles qui permet de créer facilement des applications composites transactionnelles. Notre approche unique d'Intermédiation vous donne un accès unifié en temps réel aux données et aux services hétérogènes essentiels à SOA. Si votre société a des applications .NET ou Java demandant beaucoup de performance et/ou utilisant des ressources "legacy" et "mainframe", nous pouvons vous aider à répondre plus rapidement, plus efficacement et à moindre coût, aux

demandes métier et à leurs évolutions. **Xcalia Intermédiation** vous permet de remplacer un accès direct aux données par des services et ce de manière progressive et incrémentale. Migrez à votre rythme et sans risque vers SOA, sans pour autant compromettre ni les capacités, ni la performance de votre environnement.

Accès aux données et services: Performance, Efficacité, Fluidité
Xcalia vous facilite la vie!

Contactez Xcalia dès aujourd'hui pour découvrir comment l'Intermédiation peut améliorer l'efficacité de votre entreprise.

Téléphone +33-1-56-56-12-50 - Site web www.xcalia.com

> Outils

Soyez outillé pour .Net 3.0 6
 .Net 3.0 : au-dessus de .Net 2.0 7

> Interview

Le bon accueil de .Net 3.0 8

> Vista

Visual Studio ORCAS, une simple évolution de Visual Studio 2005 ? ...10
 Migration d'une application vers Windows Vista 13

> Bureautique

Génération de documents Word avec OpenXML et .Net 3 19

> Communication

Sécurité WCF : authentification par certificats mutuels 23
 Gestion des exceptions en WCF 27

> Workflow

Intégrer des workflows dans Windows Sharepoint Services 30
 Une machine à états en guise d'assistant 33

> WPF

WPF et la 3D 36
 Un lecteur MP3 avec WPF/E 40

> Développement Web

AJAX Extensions alias ATLAS 44
 Open Search: affiner les recherches 48



.Net 3.0 : toujours plus !

Il y a deux ans, .Net 2.0 avait hissé le framework Microsoft à sa maturité, tant au niveau des langages, de la stabilité, que dans son environnement de développement, Visual Studio 2005, l'un des meilleurs outils de codage conçu par l'éditeur.

L'étape suivante a été de dévoiler .Net 3.0. Il s'agissait de rajouter des fonctions à .Net 2.0 sans rompre avec celui-ci. .Net 3.0 n'est pas une rupture avec le passé. Le modèle de développement demeure .Net 2.0. Connu sous le nom de code WinFX, .Net 3.0 se pose en futur des applications Windows et Internet. Et ce nouveau framework se donne les moyens de ses ambitions. Windows Presentation Foundation (WPF/E) en constitue la colonne vertébrale grâce à des capacités visuelles et 3D décuplées. Plus souple que les interfaces précédentes, il est doté d'un langage de description d'interface, XAML. Et quand on sait que WPF se décline sur d'autres navigateurs que Internet Explorer et est même présent sur MacOS X grâce à une version " everywhere " (partout), WPF / E (nom de code du projet). Plus intéressant encore, WPF / E sera aussi décliné sur les terminaux mobiles. Et cerise sur le gâteau, le Micro Framework .Net possède lui aussi une pile WPF... Bref, WPF et WPF / E veulent devenir le standard de l'interface Windows et Internet dans les mois et années à venir !

Sur Internet, WPF et WPF / E prennent le train du Web 2.0 à bras le corps avec des interfaces bien plus visuelles, interactives, riches. À cela, se rajoute ASP.Net Ajax (alias Atlas) qui est l'implémentation Ajax de Microsoft !

Après avoir exploré les arcanes de .Net 2 dans notre précédent hors série, nous vous proposons de vous plonger dans .Net 3 ! Vous allez découvrir comment créer un lecteur vidéo ou MP3 en WPF, vous saurez comment faire de la 3D avec la nouvelle librairie .Net. Vous découvrirez aussi l'Ajax de Microsoft ! Comme .Net 3 ne se limite pas à l'interface, nous verrons aussi quelques fonctions avancées de Workflow Foundation et de Communication Foundation.

Windows Vista n'est pas pour autant oublié ! Nous verrons comment migrer une application XP vers Vista : les précautions à prendre, les bonnes pratiques, etc. Nous lèverons aussi le voile sur le prochain Visual Studio, connu sous le nom de code : Orcas. Digne successeur de Visual Studio 2005, Orcas saura tirer parti des nouveautés Windows Vista et de .Net 3.0. Il apportera avec lui de nouvelles versions des langages VB et C#.

Bon codage !

■ François Tonic

Programmez!

LE MAGAZINE DU DÉVELOPPEMENT

Rédaction : redaction@programmez.com

Directeur de la Rédaction : Jean Kaminsky

Rédacteur en Chef : François Tonic

Ont collaboré : D. Tizon, G. Renard, J. Chable, J. Massa, P. Lothsavan, P. Manach, A. Guerot, P. Recchia, X. Vanneste, G. André, C. Durand, F. Milton.

Photo de couverture : ©Templer/zefa/Corbis

Maquette : AJE Conseils

Publicité : Pour la publicité uniquement : Tél. : 01 41 77 16 03
 publicite@programmez.com

Éditeur : Go-02 sarl, 6 rue Bezout - 75014 Paris Coordination@programmez.com - Dépôt légal : à parution - Commission paritaire : 0707K78366 - ISSN : 1627-0908 - Imprimeur : ETC - 76198 Yvetot
 Directeur de la publication : Jean-Claude Vaudecrane
 Le numéro comporte un CD Rom.

Abonnement : Programmez 22, rue René Boulanger, 75472 Paris Cedex 10 - abonnements.programmez@groupe-gli.com Tél. : 01 55 56 70 55 - Fax : 01 55 56 70 20 - du lundi au jeudi de 9h30 à 12h30 et de 13h30 à 17h00, le vendredi de 9h00 à 12h00 et de 14h00 à 16h30.

Tarifs abonnement (magazine seul) : 1 an - 11 numéros France métropolitaine : 45 € - Etudiant : 39 € - CEE et Suisse : 51,83 € Algérie, Maroc, Tunisie : 55,95 € - Canada : 64,33 € Tom : 79,61 € - Dom : 62,84 € - Autres pays : nous consulter. **PDF : 35 € (Monde Entier) souscription en ligne.**

Framework .Net 3.0

Dernière version du package redistribuable, édition 32 bits

Visual Basic 2005 Express

Version gratuite de Visual Studio 2005

Service Pack 1

Service pack : la gamme Visual Studio 2005 et Express

Mise à jour du Service Pack 1

Mise à jour du service pack 1 pour utiliser Visual Studio sur Windows Vista

Windows Workflow Foundation

Extensions Visual Studio 2005 pour le framework .net 3.0

WPF/E CTP Février

Dernière version de Windows Presentation Foundation Everywhere

WPF/E SDK CTP Février

Dernière version du SDK de WPF / E

ASP.NET 2.0 AJAX Extensions 1.0

L'Ajax de Microsoft pour ASP.Net 2.0

Documentation :

ASP.NET 2.0 AJAX Extensions 1.0

ASP.NET AJAX Control Toolkit

Ensemble d'outils : développement Ajax sous ASP.Net

Microsoft Ajax Library

Microsoft AJAX Library est une librairie Javascript, côté client (navigateur), qui simplifie la création et la réutilisation de composants Javascript et de comportement "riche" côté client (html dynamique notamment).

Visual Studio 2005 Extensions

for .NET Framework 3.0 (Windows Workflow Foundation) Librairie d'extensions pour Visual Studio 2005 pour utiliser Windows Workflow Foundation.

Microsoft Application Compatibility Toolkit 5.0

Vérifier la compatibilité des applications avec Windows Vista.

Windows Upgrade Advisor

Outil de diagnostic pour la compatibilité de votre PC avec Vista.



NetAdvantage for WPF 2007 bêta 1

Enrichissez vos développements WPF avec de nouveau/x composants ! *Infragistics*

CodeFluent

Générer votre code avec une usine à logiciel orientée modèle.

1 1 an ECO

Recevez le magazine chaque mois et économisez 20 €

11 Numéros
Prix au numéro : 65,45 €

45 €

(Prix France métropolitaine)

-30%



Abonnez-vous!

www.programmez.com

4 2 ans

Abonnez-vous pour 2 ans et recevez le livre « Windows Vista », Guide de l'Administrateur Microsoft Press William R. Stanek

22 Numéros
Prix au numéro : 130,90 € + livre 39 €

90 €

(offre réservée France métropolitaine dans la limite des stocks disponibles)



ECONOMISEZ 80€

2 1 an PDF

35 € Tarif monde entier

Inscription : www.programmez.com

3 1 an ETUDIANT 39 €

Offre limitée France métropolitaine
Vous devez justifier de votre statut d'étudiant.

- ABONNEMENT 1 an ECO** au prix de 45 € TTC. Tarif France métropolitaine.
Tarifs hors France métropolitaine : CEE et Suisse : 51,83 € - Algérie, Maroc, Tunisie : 55,95 € - Canada : 64,33 € - Tom : 79,61 € - Dom : 62,84 € - Autres : nous consulter
- ABONNEMENT 1 an ETUDIANT** (11 numéros) au prix de 39 € TTC. Offre limitée à la France métropolitaine.
Photocopie de la carte d'étudiant obligatoire
- ABONNEMENT 2 ans + livre «Windows Vista»** (22 numéros) au prix de 90 € TTC. Offre limitée à la France métropolitaine.

M. Mme Mlle Entreprise : Fonction :

Nom : Prénom :

Adresse :

Code postal : Ville :

Tél : E-mail :

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

A remplir et retourner sous enveloppe affranchie à :
Programmez ! - Service Abonnements - 22 rue René Boulanger - 75010 Paris.
abonnements.programmez@groupe-gli.com



Offre limitée, valable jusqu'au 30 juin 2007

Le renvoi du présent bulletin implique pour le souscripteur l'acceptation pleine et entière de toutes les conditions de vente de cette offre.

Conformément à la loi Informatique et Libertés du 05/01/78, vous disposez d'un droit d'accès et de rectification aux données vous concernant.

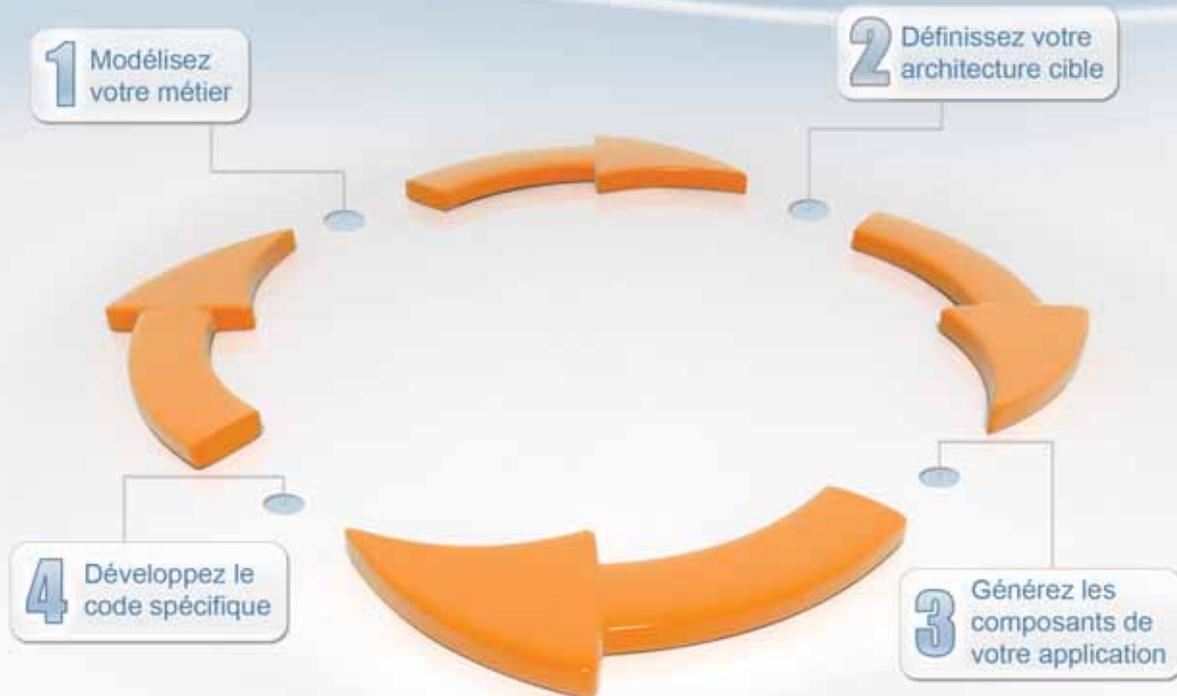
Par notre intermédiaire, vous pouvez être amené à recevoir des propositions d'autres sociétés ou associations.

Si vous ne le souhaitez pas, il nous suffit de nous écrire en nous précisant toutes vos coordonnées.

Code Fluent

Accélérez vos développements .NET

Réalisez votre application métier à l'aide de la première usine de composants à la demande.



CodeFluent.com

La fabrique en ligne est totalement gratuite pour les modèles métier comprenant jusque 30 entités.



Inscrivez-vous dès aujourd'hui sur :
<http://www.codefluent.com>

0€

CodeFluent

Version complète Développeur donnant accès aux codes sources des composants générés.



Prix public : ~~2490€ttc~~

Prix spécial lecteurs de "Programmez" : **1990€ttc**

CodeFluent est une usine logicielle qui automatise la création des composants de votre application métier .NET selon les meilleures pratiques des experts de la plate-forme Microsoft.



SoftFluent est une société spécialisée dans le développement sous plate-forme Microsoft. Elle compte déjà de nombreux clients prestigieux tels qu'ILOG, VCS Timeless, TF1 ou Microsoft France et Europe. Contactez-nous sur info@softfluent.com

SoftFluent 46, rue Auguste Blanqui 94250 GENTILLY - 01 46 16 05 02

Microsoft | PROGRAMME IDÉES
L'INNOVATION COMMENCE ICI

Soyez outillé pour .Net 3.0

Avoir un nouveau framework .Net, c'est bien, mais avoir des outils pour en profiter c'est beaucoup mieux ! Si le choix n'est pas encore immense, l'essentiel est déjà là.

Bien entendu, Microsoft propose des outils adaptés à Vista et .Net 3.0. Nous verrons plus loin que plusieurs éditeurs planchent sur leur prise en compte. Mais comme pour .Net 2.0, il faut le temps que les éditeurs mettent à jour leurs gammes, ce qui prendra quelques mois.

Chez Microsoft

La gamme Visual Studio

L'outillage idéal pour développer avec .Net 3 et sur Vista reste la gamme Visual Studio de Microsoft même s'il faudra attendre la prochaine version, Orcas, pour tirer pleinement parti des nouveautés. Orcas intégrera par défaut les éditeurs XAML, WPF, Workflow, ASP.Net Ajax, etc. En attendant, la gamme actuelle Visual Studio fonctionne sur Vista, à condition de mettre à jour avec le service pack 1 et l'update. Ces patches corrigent un certain nombre de dysfonctionnements. Il ne faudra pas oublier d'installer les extensions nécessaires pour WPF, WCF et WF, ainsi

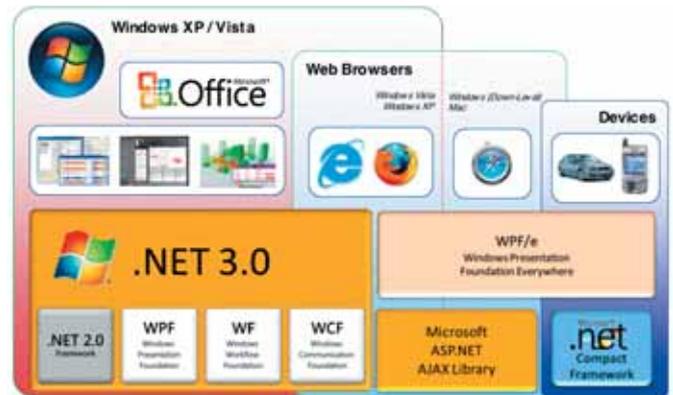
que pour ASP.Net Ajax. Concernant Team System, il faudra là aussi mettre à jour son environnement pour que cela puisse fonctionner sur Vista. Sur le Team Explorer, la mise à jour s'impose. Notez que le Team System server fonctionne sur du Server 2003 et qu'il n'est pas fait pour tourner sous Vista.

La gamme Express

Sur les IDE gratuits Express, la situation se complique quelque peu. Ce qu'il faut retenir de la compatibilité Express / .Net 3 :

- Windows Presentation Foundation : *fonctionne*, excepté le designer visuel WPF, par contre tout ce qui est XAML fonctionne...
- Windows Communication Foundation : *fonctionne*
- Windows Workflow Foundation : *ne fonctionne pas*.
- ASP.Net Ajax : *fonctionne* (attention : avec Web Developer Express uniquement)

Les choses devraient s'améliorer avec la prochaine génération des éditions Express basées sur Visual



Studio Orcas. Le SDK Vista peut être utilisé avec une édition Express.

Les autres éditeurs

• Infragistics

L'éditeur des composants .Net se prépare activement à une édition WPF de sa suite NetAdvantage. Actuellement, en pré-version, cette édition complète les objets d'interface pour son application. Les composants sont écrits en C# et XAML. L'édition " normale " NetAdvantage prend déjà en compte les nouvelles interfaces comme Office 2007.

• Les autres éditeurs de composants

À l'instar d'Infragistics, les éditeurs de composants sortent ou sortiront rapidement les outils pour .Net 3 et Vista. L'éditeur **Xceed Software** propose ainsi

une version WPF de son DataGrid (téléchargement gratuit) mais les différentes solutions ne sont pas encore estampillées Vista ou .Net 3. **ComponentOne** n'a pas encore de version .Net 3 mais cela ne devrait pas tarder. **Netcharting** supporte dès maintenant XAML mais pas le reste.

• CodeGear

La filiale de Borland a rendu disponible une nouvelle version de son outil phare Delphi : Delphi pour Win32. Cette édition se destine à Windows Vista (ainsi que 2000 et XP). Cependant, il ne s'appuie pas sur .Net 3.0.

Pour WPF

Windows Presentation Foundation et XAML suscitent depuis leur disponibilité de nombreux développements pour aider le développeurs. On peut citer les outils suivants :

- **XamlPadX** : petit outil pour tester et déboguer le code XAML
- **Snoop** : débogueur visuel pour les applications WPF
- **ZAM 3D** : outil 3D pour les applications XAML. Version communautaire disponible
- **XAML Converters** : utilitaire de conversion pour 3DS, DXF, Blender, Fireworks, etc.
- **Aurora** : outil graphique destiné aux designers XAML. Il est flexible et s'intègre à l'environnement de développement.

■ François Tonic

Se former à .Net 3.0

Pour maîtriser .Net 3, vous pouvez lire **Programmez !**, vous autoformer, ou opter pour des formations spécifiques. Les centres de formation proposent différentes formules. Les éditeurs de livres sortent depuis quelques mois des ouvrages spécifiques Vista et .Net 3.0, beaucoup sont encore en Anglais, mais ils devraient rapidement arriver en Français.

Centre de développement .Net 3 (MSDN)

Le point d'entrée de tous les développeurs .Net, MSDN propose de nombreuses ressources (documentations, téléchargement, webcast, exemples) autour de Vista et .Net 3. Malheureusement, l'information manque parfois de clarté...

Winwise

Formateurs bien connus du monde Windows et .Net, ils proposent des sessions autour du framework 3.0 (approche globale et par technologies), Live.com et Windows Vista. Les formations durent entre 1 et 5 jours.

Wygwam

Société experte en .Net 2 / 3, Wygwam ne propose pas de catalogue de formation mais peut répondre à la demande des clients et selon les objectifs voulus. Il faut compter en moyenne entre 3 et 5 jours.

SQLi

SQLi formation propose un cours centré sur framework 3 (aperçu de l'ensemble, puis travail sur les différents aspects), sur une durée de 4 jours.

.Net 3.0 : au-dessus de .Net 2.0

.Net 3.0 a souvent été perçu comme le remplaçant de .Net 2.0. Cette version rajoute de nouveaux éléments mais ne modifie pas le modèle de développement.

.Net 3 se compose de Windows Communication Foundation, Windows Presentation Foundation, Windows Workflow Foundation et Windows CardSpace. Il est désormais le modèle de développement de Windows et est disponible sur XP et Server 2003.

Windows Presentation Foundation

Sans doute la plus connue des nouvelles fondations. WPF doit refondre l'interface utilisateur en proposant quelque chose de plus riche visuellement, avec une expérience utilisateur.

Il doit remplacer l'actuel Windows Form. WPF doit offrir tous les éléments pour créer ses interfaces, qu'elles soient pour le desktop, le web, le mobile. Pour ces deux dernières cibles, on dispose de WPF/E, une version capable de s'exécuter sur Windows et MacOS X sur différents navigateurs (Safari, Firefox, IE).

Il s'agit d'un sous-ensemble de WPF privé de quelques fonctions telles que la 3D. L'édition mobile arrivera dans quelques mois.

Pour être plus flexible, plus adaptable, il faut que la description de l'interface se fasse rapidement et qu'il soit possible de la générer à la volée.

C'est pour cela que l'on dispose d'un nouveau langage : XAML. Il permet de créer des interfaces interactives et dynamiques pour les applications et apparaît comme le langage idéal pour les sites Web 2.0 ! Au-delà de l'interface, WPF sert de fondation au nouveau format documentaire de Microsoft : XPS, qui se veut le concurrent d'Adobe PDF.



Windows CardSpace

La gestion des identités demeure un des problèmes cruciaux. Il s'agit de pouvoir naviguer entre différentes applications et sites sans devoir ressaisir son identifiant et mot de passe. Il faut pour cela disposer d'une infrastructure sécurisée et implémentée. Une identité se présente sous la forme d'une carte d'information. Lorsque l'on se connecte à un site "ami" acceptant CardSpace, il suffira d'indiquer sa carte. Dans une perspective SOA, multi appli-

cation, intranet, etc. CardSpace prend tout son sens !

Windows Communication Foundation

Parfois passé en arrière plan, WCF constitue pourtant un des piliers les plus importants de .Net 3 et de la plate-forme .Net en général. Car il s'agit tout bonnement d'une des briques fondamentales pour assurer la communication inter application via différents protocoles et sert à concevoir des architectures SOA

(architecture orientée service). Il s'appuie sur SOAP et permet d'assurer l'interopérabilité entre web services. Surtout, pour le développeur, WCF simplifie le travail d'implémentation et de codage.

Windows Workflow Foundation

Le workflow se définit comme une série d'étapes effectuées dans un ordre donné. Il s'agit de posséder une logique de processus. Jusqu'à présent, Windows et .Net possédaient un unique moteur de workflow, posant des problèmes d'interopérabilité, sauf à concevoir une plomberie. Désormais, WF offre une approche unique pour le développeur et les applications. On dispose de deux types de workflow : humain et système. Il devient " facile " de concevoir un workflow et de le réutiliser dans plusieurs applications, facilitant ainsi les échanges.

ASP.Net Ajax

Un peu à part, ASP.Net Ajax (nom de code Atlas) est l'Ajax sauce Microsoft. Basiquement, il permet de créer des applications Ajax. On peut créer des interfaces avec des composants Ajax, améliorer les pages actuelles avec des contrôles. On continue à développer avec Visual Studio 2005 et ASP.Net 2.0. On peut aussi accéder à distance à des services et aux données. Orcas fournit en standard ASP.Net Ajax, avec notamment l'auto complétion du code...

Pour aller plus loin

http://www.microsoft.com/france/msdn/netframework/3/introduction.aspx#intronetfx30_topic1

■ François Tonic



William POTTIER

(étudiant en informatique/développeur indépendant)

" .Net 3 à l'avantage de présenter une architecture revue au niveau de l'organisation des modules standard. Cette modification permet, par exemple, un regroupement des outils d'interface utilisateur, de communication dans des structures logiques. Les points les plus intéressants pour ma part résident au niveau de l'interface graphique ainsi que de la communication (web services, client/serveur). Pour une utilisation personnelle et bureautique, beaucoup de clients apprécient l'aspect des interfaces WPF. Pour les outils professionnels par contre, beaucoup d'utilisateurs regrettent le manque de sobriété et de clarté dans le cadre d'une utilisation quasi intensive des applications. Pour synthétiser je dirais que .NET3 apporte beaucoup de nouveautés et d'améliorations, mais comme les précédentes version il faudra un peu de temps pour l'accepter au sein des processus de développement et d'utilisation. "

Le bon accueil de .Net 3.0

Disponible depuis environ un an, en pré-version puis en version finale, .Net 3 a rencontré un engouement certain. Nous avons voulu en savoir plus en posant quelques questions à Jean-Christophe Cimetière (Chef de produit Plate-forme & .NET, Microsoft France).



Programmez ! : Quel a été l'accueil de .Net 3 ?

Jean-Christophe Cimetière : Difficile d'être objectif. L'accueil fut bon et même enthousiaste sur les nouvelles fonctions. On est arrivé à un stade de maturité. Les gens comprennent ce qu'il y a dans le framework 3.0. et les utilisateurs de .Net 2, pour leur part, saisissent qu'il ne s'agit pas d'une migration.

P ! : Quels enjeux concrets pour ce nouveau framework ?

JCC : Il y a quelques enjeux simples. Le modèle de développement par défaut de Windows Vista est .Net 3, même si on peut toujours faire du natif. L'autre aspect est le côté graphique avec Windows Presentation Foundation. C'est une nouvelle interface utilisateur, le nouveau modèle. Nous avons aussi WPF / E, ASP.Net Ajax. C'est beaucoup plus riche qu'auparavant. Troisièmement, avec .Net 3, nous démocratisons l'architecture SOA. Le framework fournit les briques essentielles pour en faire, avec Windows Communication Foundation, CardSpace, et on peut aussi y ajouter Windows Workflow Foundation. On rend accessible à tout développeur, les bases de la SOA ! Sur WPF nous avons un an de recul.

Nous sommes maintenant capables de bien expliquer. On adresse des développeurs que l'on n'adressait pas avant, ils n'étaient pas habitués à Microsoft et à .net.

P ! : On parle beaucoup de WPF/E, la version allégée et multi plate forme de WPF, quand sortira la version finale de cette librairie ? Et où en est l'édition mobile ?

JCC : La version 1.0 sera là pour l'été prochain, elle fonctionnera sur Windows et MacOS X. La version mobile de WPF / E constitue un objectif important. On est là sur une sortie courant 2e semestre 2007.

P ! : Côté outil, on a reproché à Microsoft de ne pas proposer un Visual Studio natif Vista. Qu'en est-il exactement ?

JCC : Nous avons le service pack 1 de Visual Studio 2005 depuis fin 2006 qui constitue une première couche de corrections. Dans un 2e temps, nous avons livré un update pour utiliser Visual Studio sur Vista et régler des problèmes très précis comme avec l'UAC. D'autre part, le framework 3 est intégré à Vista. Notons que pour la première fois avec .net 3, nous avons sorti une version de .net sans mettre à jour en même temps Visual Studio. On a dé-corellé les deux.

P ! : Microsoft propose depuis quelques mois les pré-versions

d'Orcas, le prochain Visual Studio. Où en sommes-nous aujourd'hui ?

JCC : Nous avons largement commencé nos explications avec des présentations durant les derniers Techdays à Paris. Nous avons eu beaucoup de monde et les webcast sont disponibles en ligne. On sent un (grand) intérêt. Avec Orcas, sortira le framework 3.5, qui ne sera pas une version majeure du framework.

Nous bénéficierons d'un certain nombre d'innovations dans l'outil tel qu'un designer de workflow pour Workflow Foundation, facilitant la prise en main de ces éléments.

■ François Tonic

D'Orcas à Rosario, en passant par Seven

Microsoft travaille activement aux prochaines évolutions de Visual Studio (alias Orcas) et Team System (alias Rosario). Cette version doit améliorer la gestion des processus et le travail en équipe. L'éditeur met en avant le travail accompli avec les utilisateurs grâce aux nombreux feedbacks. On devrait aussi avoir une plus grande intégration entre Team System et Visual Studio. D'autre part, les parties tests, analyses seront elles aussi revues pour mieux couvrir les besoins. UML ne sera toujours pas implémenté, même si "UML est utile" précise Ian J. Know (group product manager VSTS). Microsoft mise sur les langages de domaines spécifiques (DSL) pour améliorer la disponibilité des projets. L'usine à logiciels continue d'être un objectif en améliorant les processus, la modélisation, les tests, la qualité. Linq (présent dans Orcas) constitue une évolution importante afin de réduire les lignes de code (un designer sera disponible) ! Côté Orcas (voir article

de ce numéro), l'éditeur pour XAML, Cider, sera bien présent. Notons au passage que WPF est considéré comme la nouvelle génération de Windows Form. Par contre, Cider ne sera pas inclus dans la gamme Expression (disponibilité : juin), ni dans la gamme Express de Visual Studio. Sur Express, il y aura bien de nouvelles versions "Express Orcas", la disponibilité des bêtas ne devrait pas tarder. Microsoft prépare déjà la relève de Vista avec Windows Seven (pas avant 2010 ?). Ce futur Windows, dont nous avons très peu d'informations, doit apporter plus de facilité dans la recherche, avoir une meilleure sécurité et assurer une connectivité accrue. En attendant, une mise à jour devrait intervenir pour Vista dans les prochains mois (ou en 2008), puis ce sera peut être la sortie de Windows Vienna. En attendant, cette année, Longhorn Server arrivera et il promet beaucoup de changements.

■ F. T.



“ Windows Presentation Foundation revolutionizes the user experience. It helps me transform business challenges into software solutions. ”

-Software Architect



Première mondiale!
09 avril 2007

xamDataCarousel



xamDataGrid

NetAdvantage® for WPF 2007 Vol. 1

Etablissez les fondations de la prochaine génération de l'expérience Utilisateur

- **Créez des expériences Utilisateur révolutionnaires** – des outils augmentant la puissance de la plateforme WPF afin de maximiser le potentiel de l'interface utilisateur de votre application
- **Délivrez des styles phénoménaux** – des outils permettant d'obtenir un style haute fidélité grâce à des thèmes prédéfinis y compris Windows Vista™ Aero™ et Office 2007™, tous créés par le groupe Visual Design d'Infragistics
- **Faciliter la collaboration entre designer et développeur** - des outils conçus pour dynamiser l'interaction entre designers et développeurs
- **Elargissez la plateforme des données visuelles** – Infragistics xamDataGrid™ et xamDataCarousel™ - redéfinissant la technologie de la présentation de l'information

Pour de plus amples informations:
infragistics.com/wpf/hifi/ui

sales-europe@infragistics.com

 0800 667 307

Infragistics®
Powering The Presentation Layer

WINDOWS FORMS

ASP.NET

WPF

JSF

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

Visual Studio ORCAS

Une simple évolution de Visual Studio 2005 ?

Visual Studio ORCAS est le nom de code interne que donne Microsoft à la prochaine version de sa suite de développement logiciel, Visual Studio.

Cette nouvelle version est en cours de développement. Le nom définitif qui pourrait être Visual Studio 2007 ou 2008, n'est pas encore connu, car la date de sortie du produit prévue pour la fin de l'année 2007 pourrait glisser sur 2008. Pour l'instant, un certain nombre de spécifications sont rendues publiques et Microsoft rend disponibles gratuitement en téléchargement des builds intermédiaires appelées CTP (Community Technology Preview). Les CTP sont des versions très peu testées par rapport à des versions Bêta, ce qui facilite leur mise à disposition plus régulière. Elles permettent aux développeurs les plus pressés de bénéficier des derniers avancements du produit très tôt dans le cycle de développement et à l'éditeur d'obtenir un feedback plus rapide.

Que sait-on aujourd'hui sur ORCAS, si ce n'est qu'il s'agit aussi du nom d'une petite île située à côté de Seattle où il fait bon vivre, c'est ce que nous allons voir dans cet article.

Un environnement

Multi-Framework/Compilateurs

Après Visual Studio 2002, 2003 et 2005, ORCAS sera donc la 4e édition majeure de l'outil de développement d'applications .NET et natives C++.

Bien entendu, comme les versions précédentes, ORCAS permet de développer des applications natives C++ tout en bénéficiant des dernières technologies. Grâce au nouveau pont C++/CLR, il sera possible d'utiliser Windows Presentation Foundation (WPF) ou des contrôles Windows Forms depuis une application C++. La Windows Template Library (WTL), simple extension d'ATL permettra d'écrire des applications utilisant les spécificités de VISTA. Un des reproches souvent fait à Visual Studio, est son incapacité à pouvoir compiler des applications pour des versions différentes du Framework .NET.

Par exemple, un développeur qui a en charge des applications .NET 1.1 et 2.0 doit respecti-

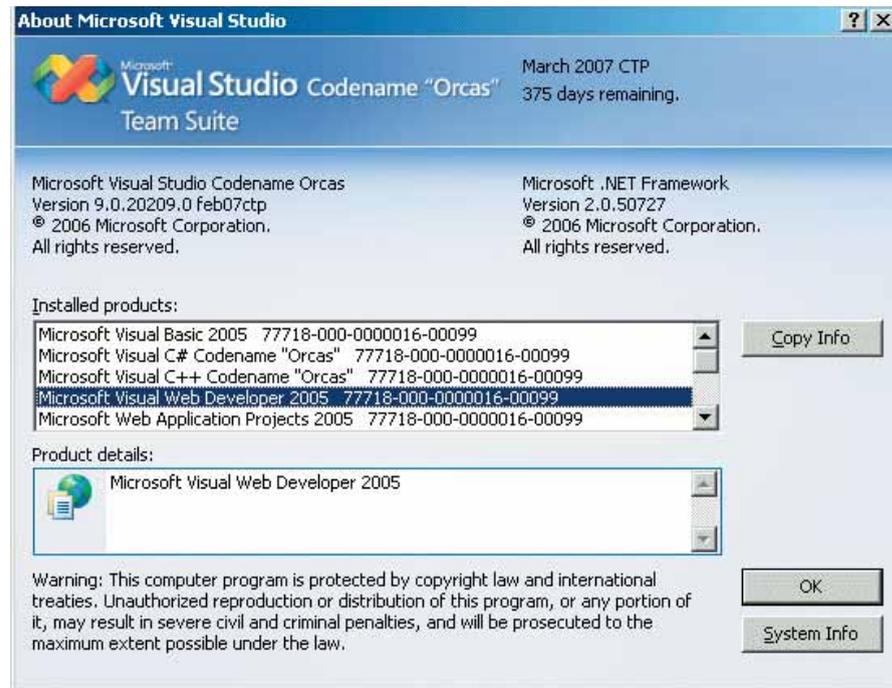


Fig. 1 : Visual Studio 2005

vement utiliser VS2003 et VS2005 pour maintenir ses développements. Avec ORCAS, les développeurs pourront compiler des applications pour les Framework .NET 2.0, 3.0 et 3.5, par simple choix dans la fenêtre de propriété des projets.

Cette bonne nouvelle laisse supposer qu'ORCAS remplacera totalement VS2005, sans que l'on soit obligé de migrer sur le nouveau Framework (Fig. 2).

Les fonctionnalités de refactoring apparues avec VS2005 qui n'étaient disponibles que pour les développeurs C# seront maintenant disponibles pour les développeurs Visual Basic, XML et XAML.

Des nouveautés sur les compilateurs et Frameworks .NET

Visual Studio 2005 permettait jusqu'à la sortie de Windows VISTA de compiler des applications pour le Framework .NET 2.0. ORCAS sera quant à lui, nativement multi-Framework.

Support du Framework .NET 3.0

En même temps que Windows VISTA, Microsoft a sorti le Framework .NET 3.0. Fourni nativement sur VISTA, mais pouvant être utilisé sur XP ou Windows 2003. Le Framework .NET 3.0 inclut: Windows Communication Foundation (WCF), Windows Workflow Foundation (WF), Windows Presentation Foundation (WPF) et CardSpace.

Au niveau de Visual Studio, ou lors de son déploiement, le Framework.NET 3.0 n'est rien d'autre qu'un ensemble de bibliothèques de classes supplémentaires qui s'ajoutent à celles du Framework .NET 2.0, les compilateurs de langages restant inchangés.

Si Visual Studio 2005 autorise aujourd'hui le développement d'applications utilisant le Framework .NET 3.0, cela demande d'ajouter des Add-In à Visual Studio, aujourd'hui en bêta et qui seront fournis nativement avec Visual Studio ORCAS. Le designer WPF de Visual Studio ORCAS, permettra notamment de faire de la

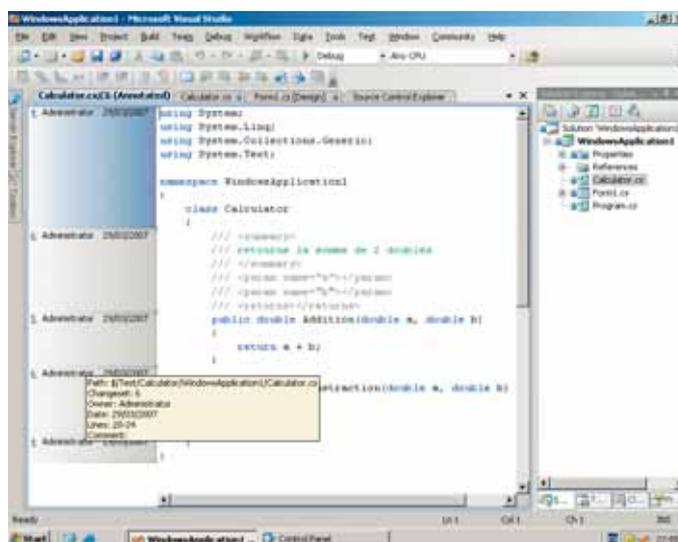
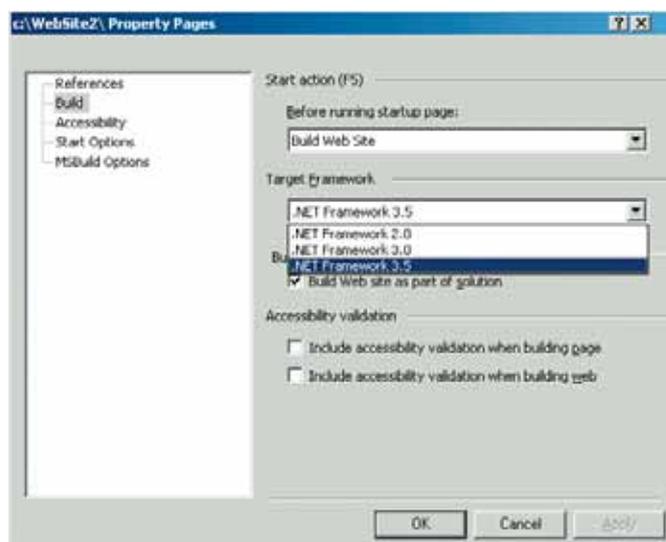


Fig. 2

3D vectorielle et de définir graphiquement les écrans, même s'il n'ira pas aussi loin que le très spécialisé : " Expression Blend " .

Support du Framework .NET 3.5

De nouveaux compilateurs de langages et un nouveau Framework .NET verront le jour en même temps que la sortie de Visual Studio ORCAS. La prochaine itération du Framework devrait s'appeler Framework .NET 3.5, mais ce n'est pas encore certain.

Les compilateurs VB9.0 et C#3.0 se verront dotés d'extensions du langage pour supporter nativement LINQ plus quelques autres nouveautés, telles que les expressions LAMBDA. Language Integrated Query (LINQ) est une syntaxe de requêtage sur des collections d'objets, des bases de données ou enfin sur du XML, directement supportée par les compilateurs C# ou VB.

Pour l'instant, les versions CTP de LINQ fonctionnent sur le Framework 2.0, mais la version finale nécessitera le Framework 3.5, et donc ORCAS. C#2.0 avait introduit les méthodes anonymes, qui sont un moyen de déclarer le code d'une méthode en ligne plutôt que de passer par une méthode déléguée. Les expressions Lambda qui seront supportées par VB9.0 et C#3.0 en sont une évolution dont l'objectif est d'obtenir une syntaxe encore plus concise pour atteindre le même but.

Des nouveautés pour les développeurs Web

Comme à l'accoutumée, les développeurs Web, ne sont pas oubliés par le lot de nouveautés qu'ORCAS mettra directement à leur disposition.

Le retour des Web Application Projects

Visual Studio 2005 avait introduit les " Web Site ", qui n'utilisent pas de fichier de projet et bénéficient de la compilation automatique. Devant les problématiques de migration d'applications ASP.NET 1.1 vers 2.0 et les problèmes de performance de cette solution, le Service Pack 1 de Visual Studio 2005 corrige le tir en ajoutant les "Web Application Projects" pour revenir sur un mode de compilation semblable à celui de VS2003. Visual Studio ORCAS contiendra nativement les Web Application Projects en plus des Web Sites qui répondent toujours à certains besoins.

Une nouvelle surface de conception

Si Visual Studio 2005 arborait la même surface de conception que les versions précédentes, Visual Studio ORCAS marque la rupture en s'appuyant sur une surface de conception totalement refondue, basée sur "Expression Web". Cette nouvelle surface de conception autorise une disposition des éléments HTML et des contrôles bien plus précis que par le passé. Son système de positionnement et son rendu graphique tiennent compte des feuilles de style CSS appliquées. Un nouveau mode d'affichage, " SPLIT " sépare l'affichage en deux pour avoir en même temps la vue " Design " et la vue "Source" et ainsi faciliter l'édition.

Support du JavaScript amélioré

C'est une bonne nouvelle, mais surtout un réel besoin avec l'utilisation croissante de Microsoft ASP.NET Ajax. Le développement JavaScript sera nettement facilité grâce au support amélioré d'IntelliSense. Celui-ci sera capable par exemple de déduire le type d'une variable

non typée JavaScript à partir de son contexte. Les noms des fonctions JavaScript issues de fichiers .js seront également résolus et les méthodes JavaScript correctement commentées seront exploitées par IntelliSense pour fournir une aide aux développeurs.

Toute fonction JavaScript ajoutée à la page dynamiquement par des contrôles serveur sera automatiquement résolue par IntelliSense en mode code. Ceci permettra par exemple de profiter de l'IntelliSense sur tout le Framework ASP.NET Ajax en ajoutant simplement un contrôle ScriptManager dans une page.

Support des CSS amélioré

Les feuilles de style CSS sont maintenant correctement exploitées par Visual Studio ORCAS. Il n'y a plus de risques de casser l'interface d'une page si on déplace un objet par glisser/déplacer. Visual Studio ORCAS ne fera que modifier les styles existants, sans tenter d'en créer de nouveaux. Visual Studio ORCAS permettra de déterminer si un attribut de style est créé sur l'élément, surchargé, ou hérité, directement dans la fenêtre de propriété des contrôles. On pourra facilement connaître la liste instances utilisées d'un style.

Contrôles ASP.NET Ajax

Les contrôles ASP.NET Ajax et les extenders des ASP.NET Ajax control Toolkit seront également fournis de base avec Visual Studio ORCAS, ce qui permettra de développer des applications Web sensiblement plus riches et interactives.

Support de IIS 7.0

Visual Studio ORCAS sera conçu pour dialo-

guer nativement avec IIS 7.0, dont le paramétrage pourra dorénavant être défini dans le fichier de configuration de l'application ASP.NET.

Le paramétrage de l'application Web et de IIS pourra donc être délégué au développeur, réduisant ainsi les tâches d'administration.

Un nouveau Team Foundation Server (TFS)

Team Foundation Server, le serveur associé à Visual Studio bénéficiera d'un certain nombre de nouveautés avec ORCAS, dont les plus remarquables sont les suivantes :

Annotate

Annotate est une fonctionnalité du contrôleur de sources d'ORCAS qui indique ligne par ligne dans le code qui a changé quoi, quand, à partir de l'historique. Les utilisateurs de Subversion connaissent cette fonctionnalité sous le nom de " Blame ".

Migration Toolkit

Le toolkit de migration sera capable de convertir et de synchroniser les données entre n'importe quel contrôleur des sources existant ou système de suivi d'éléments de travail (tâches, bugs) et TFS.

Get Latest on Check-out

Cette fonctionnalité permettra d'obtenir optionnellement le comportement de Visual SourceSafe lorsqu'un développeur extrait un fichier.

Une fois l'option activée, lorsqu'un utilisateur fera une extraction sur un fichier, le système copiera la dernière version du fichier dans l'espace de travail local et le marquera comme éditable. Le comportement actuel de TFS ne fait que rendre le fichier déjà dans l'espace de travail comme éditable, il n'obtient pas une version depuis le référentiel de gestion des sources.

Ce fonctionnement de TFS implique donc que dans certains cas, nous sommes obligés de résoudre des conflits au moment de l'archivage, même dans le cas où on n'a pas autorisé l'extraction multiple. Le conflit survient lorsqu'un développeur a travaillé à partir d'une version ancienne d'un fichier, ce qui arrive facilement si on oublie de faire régulièrement un "Get last version".

Avec ORCAS, le développeur aura ainsi le choix de résoudre les problèmes d'intégration au moment de l'archivage d'une nouvelle fonc-

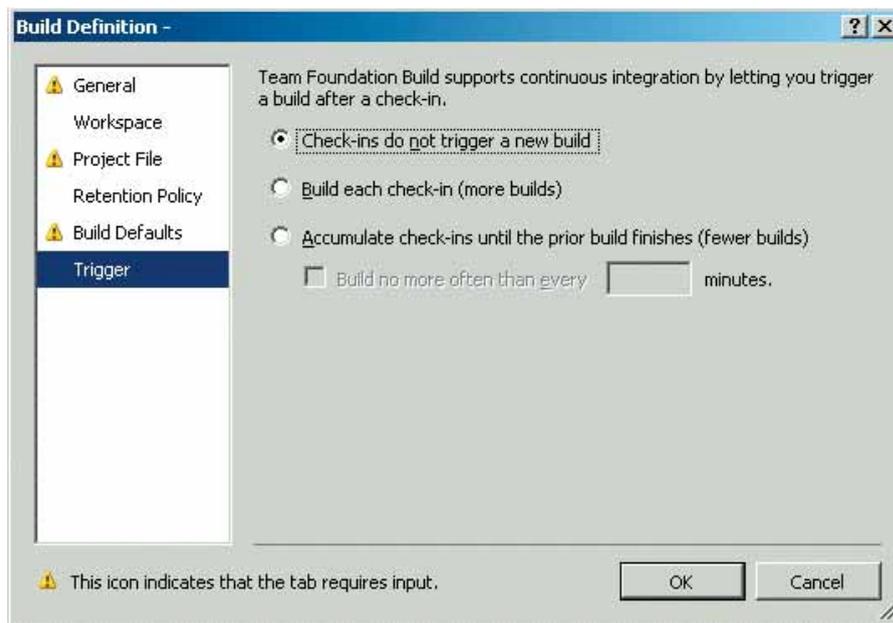


Fig. 3

tionnalité testée unitairement et de façon isolée (fonctionnement actuel), ou dès le début de l'implémentation d'une nouvelle fonctionnalité (fonctionnement de SourceSafe) en prenant le risque que le code ne compile plus à cause d'un get latest version automatique exécuté partiellement et donc non atomique.

Cette fonctionnalité était clairement attendue, en particulier des développeurs habitués de Visual SourceSafe. Le choix de la préférence de l'une ou l'autre des solutions est sujet à débat. L'important est que maintenant ORCAS nous donne le choix.

L'intégration continue

Team Foundation Server ne fournit pas dans sa version VS2005 de solution d'intégration continue native. On peut facilement lancer une compilation automatique à heures fixes, mais s'il s'agit de la déclencher après chaque archivage dans le contrôleur de sources, cela nécessite un développement spécifique.

Team Foundation Server ORCAS sera quant à lui enrichi de fonctionnalités poussées pour faciliter l'intégration continue.

- Des règles de rétention permettront de savoir combien de temps conserver des builds selon leur statut (échouée, stoppée, terminée, terminée - tests échoués, terminée - test passés)
- Des triggers définiront la fréquence des builds (sur chaque check-in, accumuler les check-in tant que la build précédente n'est pas finie, ne pas compiler plus souvent que x minutes), (Fig. 3).
- Des files d'attente seront définies pour les builds, avec une notion de priorité.

- Plusieurs serveurs de build pourront être définis et paramétrés sur le serveur TFS. Le port utilisé pour communiquer avec l'agent de build pourra être choisi, alors que le numéro de port est non modifiable dans la version actuelle de TFS.

Conclusion

Visual Studio ORCAS va inclure nativement tous les derniers Framework et dernières technologies (WCF, WF, WPF, ASP.NET Ajax, LINQ, Expressions Lambda). Il deviendra l'environnement de développement naturel pour les applications compatibles VISTA, tout en prenant en charge les développements réalisés avec VS2005.

La nouvelle version de Team Foundation Server ORCAS apportera, grâce à l'intégration continue côté serveur et autres améliorations côté client, la maturité nécessaire pour convaincre les équipes de développement qui n'auraient pas encore fait le choix de l'industrialisation de leurs développements avec TFS. ORCAS ne va cependant pas arriver tout de suite, il faudra savoir rester patient, mais ce qui est rassurant, c'est que l'investissement fait sur VS2005, le Framework .NET 3.0, ASP.NET Ajax ou Team Foundation Server sera capitalisé.



■ Daniel TIZON - Winwise
Microsoft Regional Director
Email : daniel.tizon@winwise.fr
Blog : <http://blogs.codes-sources.com/daniel/>
Site : <http://www.winwise.fr>

MIGRATION

d'une application vers Windows Vista

Windows Vista est le nouveau système d'exploitation développé par Microsoft. Cette dernière version du système offre un grand nombre de nouveautés et d'améliorations dans des domaines tels que l'interface utilisateur, la sécurité, la connectivité...

Toutes ces nouveautés et améliorations ont un impact sur l'utilisation et le développement des applications existantes. Cet impact peut-être soit visuel, soit de nature à modifier la fonctionnalité de l'application. L'un des premiers impacts que les éditeurs de logiciels devront aborder concerne le contrôle de compte utilisateur (UAC). Comme cela affecte le comportement des applications sous Windows Vista, cet article aborde le contrôle de compte utilisateur et la migration d'applications existantes.

Le contrôle de compte utilisateur (UAC)

Pourquoi le contrôle de compte utilisateur ?

Ces dernières années, un nombre important de virus et de vers a été dirigé contre les plates-formes Windows. Ces attaques ont eu un impact financier pour nos clients aussi bien pour les entreprises que pour les particuliers. La confiance dans les systèmes d'exploitation Windows a été ébranlée. Il était de la responsabilité de Microsoft de répondre aux attentes légitimes de nos clients et de rétablir une informatique digne de confiance (Trustworthy Computing).

Il existe principalement deux types de comptes sous Windows :

1. Le compte administrateur qui possède tous les droits et privilèges pour administrer la plate-forme.
2. Le compte utilisateur standard qui possède des droits et des privilèges limités.

Les administrateurs de l'informatique d'entreprise sont très intéressés par l'utilisation du compte utilisateur standard car il permet d'accroître la sécurité et le contrôle de leur parc machine et réduit le coût total de possession (TCO) (Cf étude du Gartner Security Holes Increase Windows Client). Il est alors légitime de se poser la question : " Pourquoi ne sommes-nous pas tous utilisateurs standard sous Windows XP ? ". La majorité des applications développées requiert des droits et des privilèges excessifs. Il est très difficile, même si cela reste faisable, de ne pas être administrateur de son poste de travail ! Certaines fonctionnalités courantes de Windows nécessitent des droits administrateur pour des fonctions usuelles comme par exemple :

- Changer le fuseau horaire
- Se connecter à un réseau sans fil sécurisé

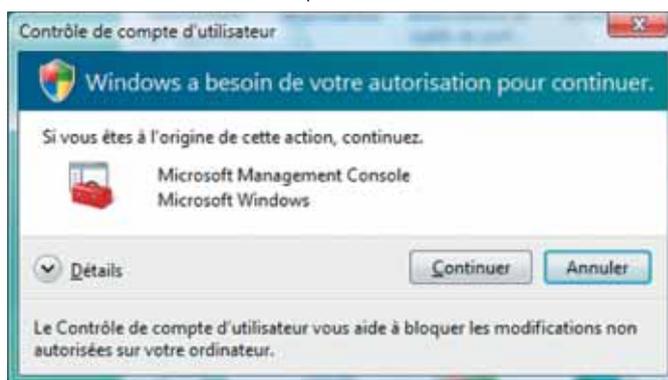
Les droits et les privilèges des fonctions courantes ont été complètement revus sous VISTA. D'autre part, la façon dont les éditeurs de logiciels contribuent à la sécurisation de leurs applications pour la plate-forme Windows est l'un des facteurs clé. Sous Windows VISTA, le contrôle de compte utilisateur a été développé pour les aider dans cette tâche. Le contrôle de compte utilisateur réduit la surface d'attaque contre les logiciels malveillants. Tous les utilisateurs, y compris les administrateurs, s'exécutent avec un compte

limité par défaut. Ils peuvent élever sélectivement leurs privilèges lorsqu'il est nécessaire d'exécuter des tâches d'administration ou d'installation.

L'UAC dans Windows Vista.

Objectifs

Le contrôle de compte utilisateur fournit une méthode pour séparer les tâches et les privilèges de l'utilisateur standard de ceux nécessitant un accès Administrateur. Par défaut, les administrateurs exécutent la plupart des tâches avec un privilège d'utilisateur standard. Lorsqu'ils doivent effectuer une tâche administrative, ils doivent d'abord donner leur consentement dans la fenêtre qui s'affiche alors :



ils n'obtiennent des privilèges élevés que pendant la durée de vie de ce processus. Toutes les autres tâches continueront de s'exécuter en mode utilisateur standard.

Le contrôle de compte utilisateur assure :

- Que tous les utilisateurs s'exécutent avec des privilèges standard par défaut.
- Qu'un consentement explicite est obligatoirement requis pour une élévation de privilège.
- Qu'un haut niveau de compatibilité est assuré pour les applications existantes.

Architecture



Lorsque un administrateur se logue, Windows Vista reconnaît s'il possède des privilèges élevés, et découpe alors le jeton d'accès (Token) en deux jetons :

1. Un jeton contenant les privilèges spécifiques (jeton filtré)
2. Un jeton contenant tous les privilèges (jeton complet)

Le jeton filtré est utilisé par défaut. Lorsqu'une tâche requiert des privilèges d'administrateur, Windows affiche la boîte de dialogue de consentement. L'aspect de cette boîte de dialogue dépendra de la présence ou non d'une signature digitale pour l'exécutable :

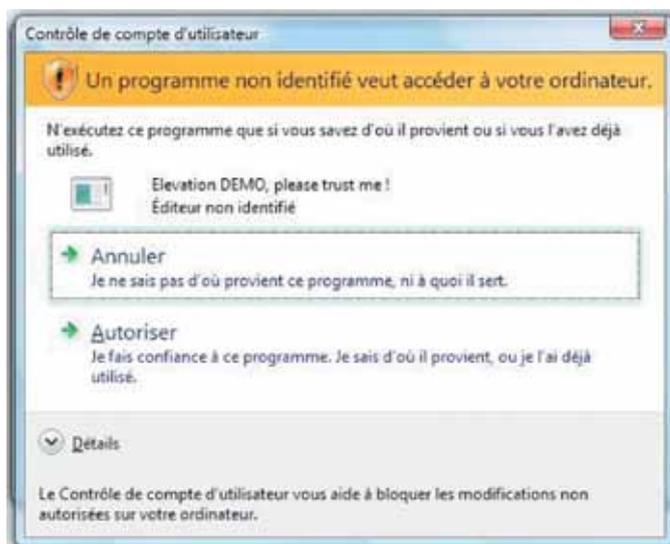


Figure 1 : Exécutable non signé

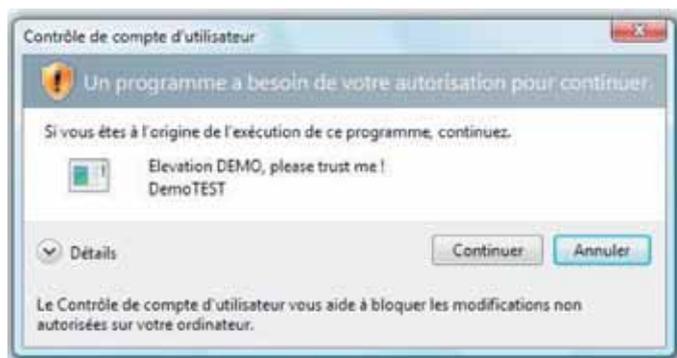


Figure 2 : Exécutable signé

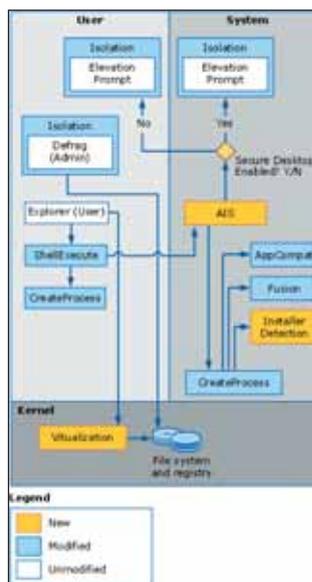
Description d'exécution d'une application pour un utilisateur standard

1. ShellExecute appelle CreateProcess()
2. CreateProcess() vérifie si l'application nécessite ou non une élévation de privilège. Si oui, l'exécutable est alors inspecté pour déterminer son requestedExecutionLevel, qui est fourni par le manifeste de l'application s'il existe.
3. CreateProcess retourne une erreur Win32 : ERROR_ELEVATION_REQUIRED
4. ShellExecute vérifie ce retour d'erreur et passe l'appel vers le service d'information des applications (AIS) qui est en charge de demander l'élévation de privilège.

Description d'élévation de privilège

1. AIS reçoit l'appel de ShellExecute et réévalue le niveau d'exécution requis ainsi que les stratégies de groupes pour déterminer si l'élévation est permise.

2. Si le niveau d'exécution requiert une élévation, AIS affiche la fenêtre de consentement.
3. Dès que l'utilisateur donne son consentement, AIS retrouve le jeton complet associé à l'utilisateur.



4. AIS appelle CreateProcessAsUser avec le jeton complet et crée un processus avec un niveau de privilège élevé.

Le seul moyen d'élever un privilège consiste à créer un nouveau processus avec un jeton complet. Un processus existant ne peut être élevé !

Taxonomie du jeton de l'utilisateur standard

Lorsqu'un utilisateur se logue, Vista examine les privilèges et l'appartenance aux différents groupes (RIDs) auxquels appartient l'utilisateur. Il détermine alors, s'il faut ou non créer un jeton d'accès filtré et un jeton d'accès complet.

Windows créera 2 jetons d'accès pour l'utilisateur si l'une des deux conditions est vérifiée :

1) Si le compte utilisateur contient au moins l'un des RIDs suivants :

- DOMAIN_GROUP_RID_ADMINS
- DOMAIN_GROUP_RID_CONTROLLERS
- DOMAIN_GROUP_RID_CERT_ADMINS
- DOMAIN_GROUP_RID_SCHEMA_ADMINS
- DOMAIN_GROUP_RID_ENTERPRISE_ADMINS
- DOMAIN_GROUP_RID_POLICY_ADMINS
- DOMAIN_ALIAS_RID_ADMINS
- DOMAIN_ALIAS_RID_POWER_USERS
- DOMAIN_ALIAS_RID_ACCOUNT_OPS
- DOMAIN_ALIAS_RID_SYSTEM_OPS
- DOMAIN_ALIAS_RID_PRINT_OPS
- DOMAIN_ALIAS_RID_BACKUP_OPS
- DOMAIN_ALIAS_RID_RAS_SERVERS
- DOMAIN_ALIAS_RID_PREW2KCOMPACCESS
- DOMAIN_ALIAS_RID_NETWORK_CONFIGURATION_OPS
- DOMAIN_ALIAS_RID_CRYPTO_OPERATORS

2) Si le compte utilisateur contient n'importe quel privilège autre que ceux de l'utilisateur standard.

Les privilèges du jeton d'accès filtré dépendront de la condition qui a été vérifiée.

Si la condition 1) a été vérifiée, tous les privilèges Windows sont supprimés à l'exception des privilèges :

- SeChangeNotifyPrivilege
- SeShutdownPrivilege
- SeUndockPrivilege
- SeReserveProcessorPrivilege
- SeTimeZonePrivilege

Si la condition 2) a été vérifiée, uniquement les privilèges suivants sont supprimés :

- SeCreateTokenPrivilege

- SeTcbPrivilege
- SetTakeOwnershipPrivilege
- SeBackupPrivilege
- SeRestorePrivilege
- SeDebugPrivilege
- SeImpersonatePrivilege
- SeRelabelPrivilege

Windows marque tous les RIDs du jeton filtré comme USE_FOR_DENY_ONLY. Le jeton filtré sera utilisé par défaut pour démarrer les applications. Le jeton complet ou non modifié sera utilisé lorsqu'une élévation de privilège sera demandée.

Le compte standard contient seulement les privilèges suivants :

- SeChangeNotifyPrivilege
- SeShutdownPrivilege
- SeUndockPrivilege
- SeIncreaseWorkingSetPrivilege
- SeTimeZonePrivilege

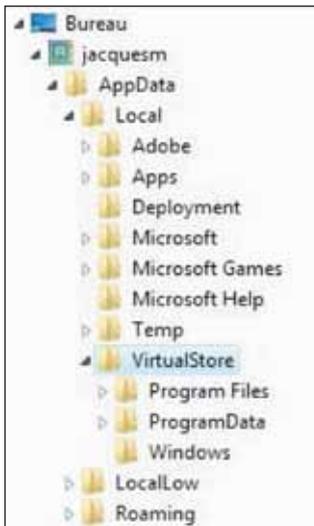
La redirection

VISTA apporte une nouvelle fonctionnalité afin d'améliorer la compatibilité des applications existantes. Il redirige les écritures/lectures vers certains objets systèmes. Cette redirection s'effectue par utilisateur. La redirection ou " virtualisation " comporte deux composantes : redirection de fichiers et redirection de la base de registre.

La redirection de fichiers

Dans le cas où une application est écrite sous un des répertoires systèmes (Windows, Programmes Files\ ...), cela fonctionne correctement si un administrateur avec son jeton complet exécute l'application. Par contre, cela échoue si l'utilisateur est un utilisateur standard ou protégé

par le contrôle de compte utilisateur. Windows redirige l'écriture vers un répertoire propre à l'utilisateur qui exécute l'application.

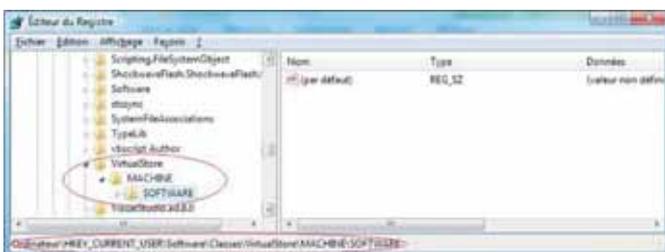


La redirection de la base de registre

La redirection de la base de registre est identique à la redirection des fichiers sauf qu'elle s'applique à la clé HKEY_LOCAL_MACHINE\SOFTWARE.

Les écritures sont alors redirigées vers la clé HKEY_CURRENT_USER\Software\Classes\VirtualStore\MACHINE\SOFTWARE

Les redirections de fichiers et de la



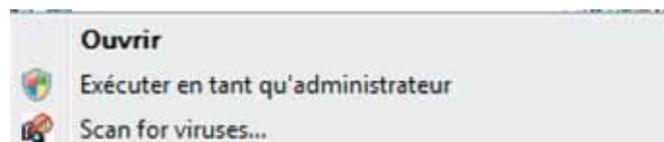
base de registre ont pour seule intention d'assurer la compatibilité des applications existantes. Une application conçue pour VISTA ne devrait pas écrire de données dans ces lieux systèmes.

Comment élever le privilège

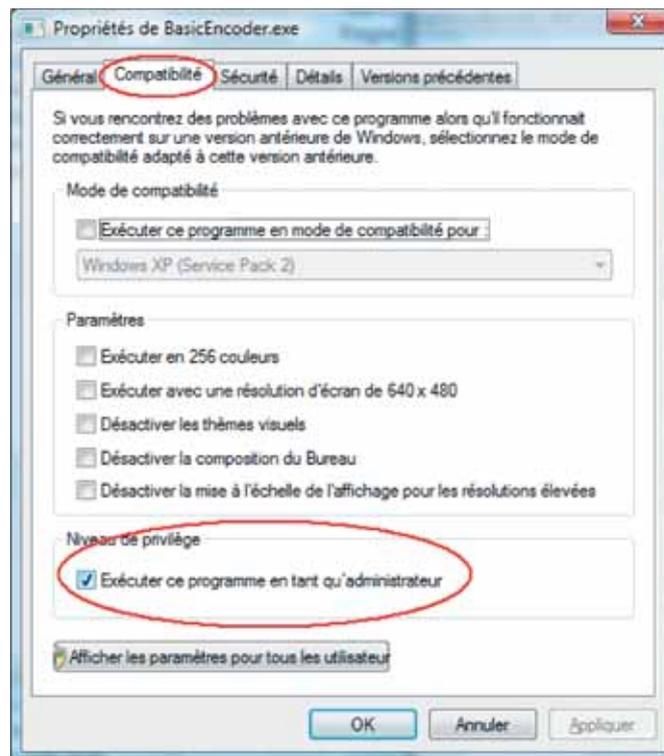
Il nous reste à comprendre comment concrètement l'utilisateur ou Windows peut effectuer une élévation de privilège.

L'utilisateur peut effectuer une élévation de privilège par l'une des opérations suivantes :

- Un clic droit sur le nom de l'exécutable de l'application et le choix Exécuter en tant qu'administrateur :



- En utilisant la propriété de compatibilité Niveau de Privilège:



- L'ajout d'un fichier manifeste contenant les informations :

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
  <assemblyIdentity name="monapplication"
    version="1.0.0.0"
    processorArchitecture="X86"
    type="win32" />
  <description>Mon Application</description>
  <!-- Identify the app's security requirements. -->
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
      <requestedPrivileges>
        <requestedExecutionLevel level="requireAdministrator" />
      </requestedPrivileges>
    </security>
  </trustInfo>
</assembly>
```

```
</requestedPrivileges>
</security>
</trustInfo>
</assembly>
```

Le fichier manifeste peut accompagner l'exécutable ou bien être hébergé dans l'exécutable.

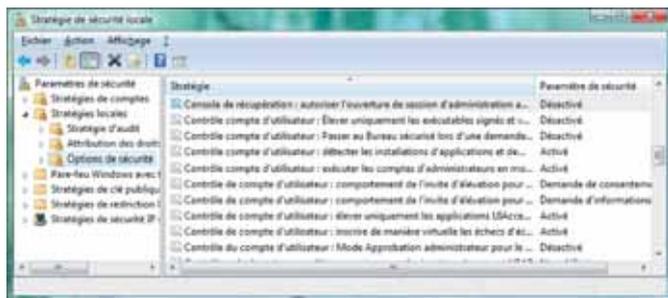
Les valeurs possibles pour requestedExecutionLevel sont :

Valeur	Description	Redirection
asInvoker	L'application s'exécute avec le même jeton d'accès que le parent	Non
highestAvailable	L'application s'exécute avec les privilèges les plus hauts que l'utilisateur puisse obtenir	Non
requireAdministrator	L'application ne peut-être exécutée que par un administrateur et requiert que l'application soit démarrée avec le jeton complet d'un administrateur.	Non

Windows peut élever automatiquement le privilège au démarrage d'une application s'il détecte une application d'installation. L'icône de l'exécutable est alors décorée de la métaphore du bouclier. Chaque fois, que ce bouclier apparaît sous VISTA, cela signifie qu'une élévation de privilège sera demandée.

Stratégies de groupe

L'activation du contrôle de compte utilisateur se gère depuis les stratégies de sécurité locale :



L'impact pour les applications.

L'impact du contrôle de compte utilisateur pour une application existante s'évalue de la façon suivante :

- Si une application s'exécute correctement sous Windows XP en tant qu'utilisateur standard, alors il n'y a aucun impact.
- Si une application nécessite des privilèges administrateur, il faudra ajouter un fichier manifeste avec le niveau requis : Administrateur.
- Si une application est soumise à la redirection et que celle-ci change le comportement de l'application, alors il faudra :
 - Soit modifier le code afin d'écrire les données au bon endroit ! (par exemple le répertoire ProgramData)
 - Soit ajouter un fichier manifeste avec le niveau requis : Administrateur.

Dans certain cas, une application ne requiert que ponctuellement des privilèges administrateur. On dispose alors, des choix suivants :

- Relancer l'application avec un niveau de privilège administrateur.
- Séparer le code nécessitant les privilèges administrateur du reste de l'application et l'exécuter dans un processus séparé.

La séparation de code, implique une revue d'architecture et le choix du processus pour héberger le code administrateur. On dispose de 3 "Design Patterns" :

1. Service Broker : Communication interprocessus à base de Remote Procedure Call (RPC)
2. Side by Side processes : Communication interprocessus à base de RPC ou de SharedMemory
3. Administrator COM Object. : Communication interprocessus à base de COM.

Migration d'une application existante

Cette section propose une méthode pour migrer une application impactée par la redirection et la nécessité d'un privilège élevé. Elle met en œuvre la séparation du code administrateur et la signature digitale de l'application et de l'installation.

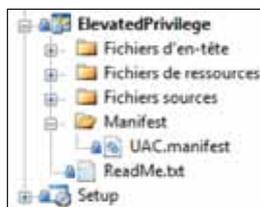
Présentation d'un exemple.

Prenons une application qui écrit dans la base de registre sous la clé : HKLM\Software\UACDEMO et donc implique la redirection.

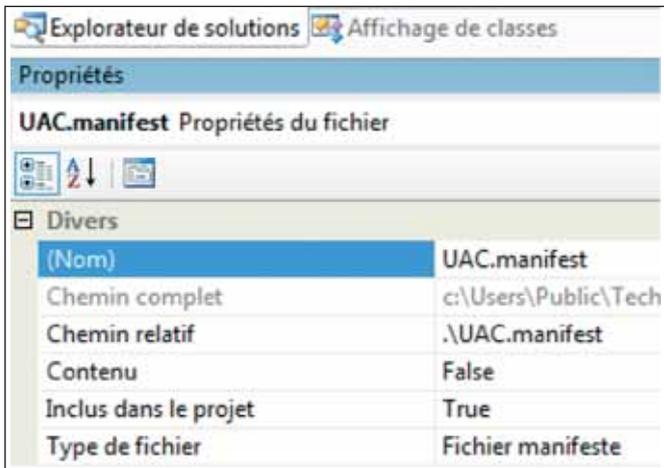
```
// Try to open HKLM\Software in write access
HKEY hkey;
DWORD err = RegCreateKeyEx(HKEY_LOCAL_MACHINE, _T("SOFTWARE\UACDEMO"), 0, 0, 0, KEY_SET_VALUE, 0, &hkey, 0);
if (err == ERROR_SUCCESS)
{
    SYSTEMTIME now;
    ::GetLocalTime(&now);
    DWORD errSet = RegSetValueEx(hkey, _T("LastRun"), 0, REG_BINARY, (BYTE*)&now, sizeof now);
    *perr=errSet;
    RegCloseKey(hkey);
}
else
{
    *perr=err;
}
```

De plus, cette application énumère la liste des tâches planifiées et donc implique des droits administrateur.

```
// Try to enum Schedule Jobs
const int MAX_JOBS = 10; // this sample app shows at most 10 jobs
AT_ENUM* jobInfo = new AT_ENUM[MAX_JOBS];
DWORD numRead, hint;
// This call requires admin privileges!
NET_API_STATUS status = NetScheduleJobEnum(0, (BYTE*)&jobInfo, MAX_PREFERRED_LENGTH, &numRead, &hint, 0);
*perr=status;
if (NERR_Success == status)
{
    NetApiBufferFree(jobInfo);
}
```

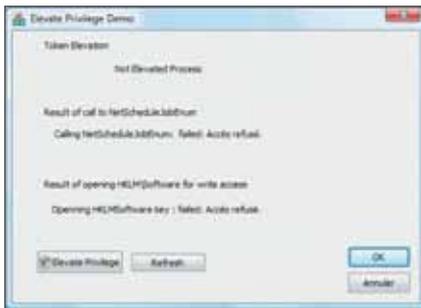


Pour éliminer l'effet de la redirection de la base de registre, on commence d'abord par ajouter un fichier manifeste hébergé dans l'exécutable. Pour cela, il suffit de l'ajouter au projet Visual Studio 2005 qui le reconnaît automatiquement.



```
<?xml version="1.0" encoding="UTF-8"?>
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0">
  <trustInfo xmlns="urn:schemas-microsoft-com:asm.v3">
    <security>
      <requestedPrivileges>
        <requestedExecutionLevel level="asInvoker"/>
      </requestedPrivileges>
    </security>
  </trustInfo>
</assembly>
```

On ne demande que le niveau asInvoker. Par conséquent, l'écriture dans la clé HKLM/Software échouera en mode standard.



On ne souhaite pas que toute l'application s'exécute avec un niveau de privilège élevé. Le code qui accède à la base de registre pour la clé HKLM/Software et le code qui utilise la fonction NetScheduleJob-

num sera séparé et hébergé dans un composant COM DLL.

L'élévation de privilège.

Le simple fait de séparer le code dans un composant COM ne résout en rien le problème car le composant COM DLL sera dans le même processus que l'application. Il doit donc s'exécuter dans un autre processus via le mécanisme, par exemple, de DLLSurrogate.

Pour cela on ajoute un APPID et une clé DLLSurrogate= " " dans le fichier rgs du composant :

```
'%APPID%' = s 'CanElevate'
{
  val DLLSurrogate = s ""
}
```

Mais cela ne suffit toujours pas, car il faut que ce processus s'exécute avec des privilèges élevés. Pour résoudre ce problème, on doit utiliser le

nouveau Moniker fourni avec Windows VISTA : Elevation:Administrator. Au lieu d'utiliser la fonction CoCreateInstance, on utilise alors le code suivant :

```
HRESULT CoCreateInstanceAsAdmin( REFCLSID rclsid, REFIID riid, __out
void ** ppv)
{
  BIND_OPTS3 bo;
  WCHAR wszCLSID[50];
  WCHAR wszMonikerName[300];

  StringFromGUID2(rclsid, wszCLSID, sizeof(wszCLSID)/sizeof(wszCLSID[0]));
  HRESULT hr = ::StringCchPrintf(wszMonikerName, sizeof(wszMonikerName)/sizeof(wszMonikerName[0]), _T("Elevation:Administrator!new:%s"), wszCLSID);
  if (FAILED(hr))
    return hr;
  memset(&bo, 0, sizeof(bo));
  bo.cbStruct = sizeof(bo);
  bo.hwnd = NULL;
  bo.dwClassContext = CLSCTX_LOCAL_SERVER;
  return CoGetObject(wszMonikerName, &bo, riid, ppv);
}
```

Et pour créer l'instance de l'objet, il suffit d'utiliser l'appel:

```
hr::CoCreateInstanceAsAdmin(CLSID_CanElevateWorker,
  __uuidof(pWorker),
  (LPVOID*)&pWorker);
```

Cependant, ce moniker impose quelques règles :

1. Il faut ajouter une clé Elevation dans l'enregistrement de l'objet, cela s'effectue aisément depuis le fichier rgs correspondant à l'objet :

```
Elevation
{
  val Enabled = d 1
}
```

2. Il faut aussi ajouter une clé définissant le message localisé à afficher dans la boîte de dialogue de consentement :

```
val LocalizedString = s '@%MODULE%,-101'
```

La variable 101 est définie dans les ressources du composant COM :

IDS_PROJNAME	100	CanElevate
IDS_ELEVATIONTEXT	101	Elevation DEMO, please trust me!

En utilisant la fonction CoCreateInstanceAsAdmin () la fenêtre de consentement s'affiche (Figure A).

L'objet COM s'exécute dans le processus DLLSurrogate avec des privilèges élevés alors que l'application qui consomme l'objet COM utilise des privilèges limités (Figure B).

Signature de l'application et de l'installation.

Lors de l'élévation de privilège, la fenêtre de consentement ne reconnaît pas l'éditeur. Pour résoudre ce problème et parce que les stratégies de



Figure A

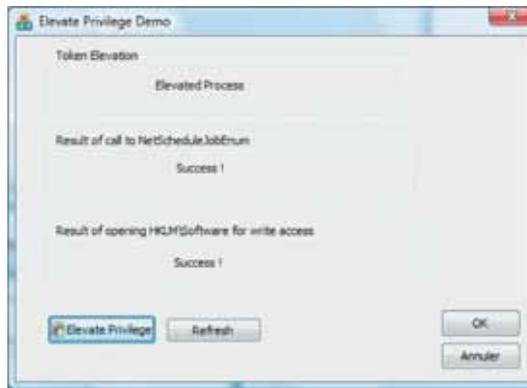


Figure B

De la même façon, on signe aussi l'application (pour le cas où l'utilisateur déciderait de l'exécuter en tant qu'administrateur). Pour l'installation, la signature est nécessaire afin d'identifier l'éditeur et assurer l'origine et l'intégrité des fichiers. Il suffit d'ajouter au projet d'installation un événement après génération pour signer le fichier setup.exe et le fichier msi :

groupe peuvent interdire l'exécution de ce type d'application il faut que le composant COM soit signé. La méthode à utiliser est Authenticode. Un éditeur obtient son certificat et sa clé privée depuis une autorité de certification reconnue comme Verisign ou Thawte. Pour l'exemple, on se contente d'un certificat de test produit à partir des outils de Windows VISTA SDK.

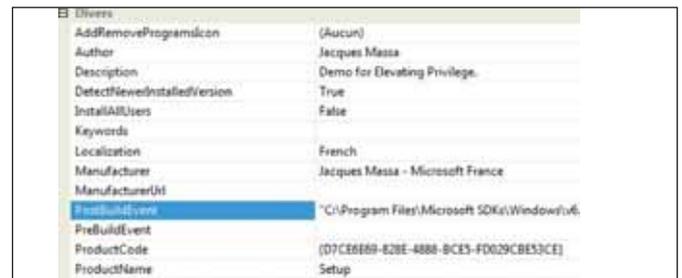
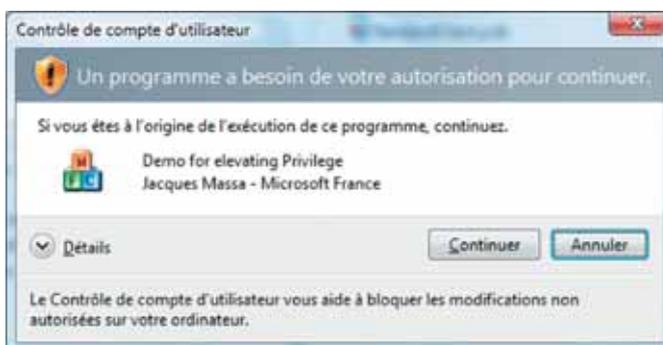
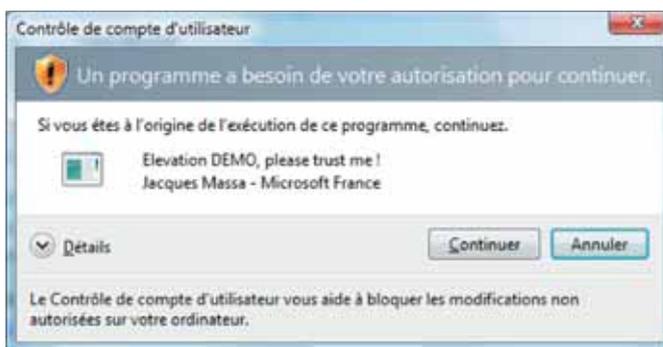
Création d'un certificat de test

```
Makecert -r -pe -sr localMachine -ss Test -n "CN=Jacques Massa - Microsoft France" democert.cer
Ajout du certificat sur la machine à la racine
Certmgr -add democert.cer -s -r localMachine Root
```

Pour signer automatiquement le composant COM depuis Visual Studio il suffit d'ajouter un événement après génération dans les propriétés du projet :

```
Signtool Sign /v /s Test /n "Jacques Massa - Microsoft France" "$(TargetDir)$(TargetFileName)"
```

La fenêtre de consentement devient alors :



```
"C:\Program Files\Microsoft SDKs\Windows\v6.0\Bin\signtool.exe"
Sign /v /s TEST /n "Jacques Massa - Microsoft France" "$(BuiltOutputPath)" "$(ProjectDir)$(Configuration)\setup.exe"
```

En résumé

La méthode exposée permet de séparer le code nécessitant des privilèges administrateur du code de l'application. L'application peut continuer à s'exécuter avec des privilèges standard et ponctuellement demander des privilèges administrateur. Le nouveau mécanisme de moniker COM permet d'élever le processus hébergeant le composant. La surface d'attaque de l'application par les logiciels malveillants reste réduite. On participe ainsi à la sécurisation de Windows.

Conclusion

Avec pour objectif de réduire la surface d'attaque par les logiciels malveillants, Windows VISTA met en œuvre des mécanismes de sécurisation par défaut, à l'exécution et lors du déploiement. Le contrôle de compte utilisateur participe à rendre VISTA digne de confiance. Pour cela, des modifications ont été effectuées au niveau de VISTA et de l'interface utilisateur. Dans certains cas, les éditeurs de logiciels pour Windows VISTA seront appelés à effectuer des changements plus ou moins importants dans leurs applications. La méthode exposée ici, permet de répondre à un scénario d'une application nécessitant ponctuellement des privilèges administrateur. L'application continue toutefois de s'exécuter avec des privilèges limités. Afin d'assurer la compatibilité avec les applications existantes, la redirection de fichiers et de la base de registre est disponible par défaut. Il faut cependant être vigilant car elle réduit les fonctionnalités par utilisateur. Ce mécanisme, pourrait disparaître dans les prochaines versions de Windows. La signature digitale des applications et des fichiers d'installation permet d'identifier l'origine des fichiers et d'assurer leur intégrité pour l'utilisateur et les administrateurs. C'est une pratique que les éditeurs devraient adopter pour leurs applications VISTA.

■ Jacques Massa - Consultant Senior en développement d'application. Microsoft France

Génération de documents Word avec OpenXML et .NET 3

Après avoir vu quelques uns des concepts de base du standard Open XML et notamment de OPC - Open Packaging Convention - dans un précédent article (Programmez! n°93), attaquons nous aujourd'hui à la génération de documents de type traitement de texte, avec le langage `WordProcessingML` du standard ouvert Open XML. Pour ce faire, nous utiliserons les API fournies avec .NET 3 et notamment l'espace de nom `System.IO.Packaging` mis à notre disposition.

Il y a au moins deux façons de générer des documents `WordProcessingML` : la première est de partir d'un document existant déjà formaté (et pourquoi pas enrichi de données métiers), la seconde est de créer un document 'from scratch', autrement dit à partir de rien. C'est cette dernière option que nous allons retenir, l'autre méthode sera utilisée dans de prochains articles plus complexes, dans lesquels vous n'aurez pas besoin de vous occuper de créer intégralement un document. L'exhaustivité des spécifications de ce standard ouvert de l'ECMA facilite grandement l'exploitation des documents Open XML. Vous trouverez toutes les informations relatives au `WordProcessingML` dans la partie 4:2 du standard.

Introduction au document `WordProcessingML`

Un document `WordProcessingML` - un document Office Open XML orienté traitement de texte - est une compilation de deux types d'informations :

- les propriétés : les styles, la définition de la numérotation, ...
- les stories : le document principal, les en-têtes, les pieds de page, les commentaires, ...

Les propriétés représentent les différents éléments de présentation communs aux stories tels que la définition des styles ou de la numérotation des listes

Les stories - terme difficilement traductible en français - représentent les régions uniques dans lesquelles un utilisateur peut ajouter ou modifier le contenu. La story la plus importante, et la seule dont la présence est obligatoire, est celle du document principal qui décrit le contenu général du document. Comme pour les autres types de documents OpenXML (`SpreadsheetML` pour les documents de tableur et `PresentationML` pour les documents de présentation), c'est vers la story principale que vous pointerez lorsque vous demanderez la relation de type <http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument> à votre paquet. Dans la suite de cet article, nous ne traiterons que la story principale. On le désignera aussi sous l'appellation de partie principale, en rapport avec la structure physique de stockage OPC des documents Office Open XML. La partie principale d'un document `WordProcessingML` possède habituellement, bien que

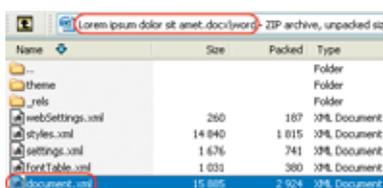


Fig. 1

cela ne soit qu'un usage, l'URI `/word/document.xml` comme le montre l'illustration (Fig. 1).

Le format de fichier Open XML utilise bien évidemment le XML pour décrire

l'intégralité des documents. Concernant la story principale d'un document `WordProcessingML`, son contenu se trouve sous la balise `body` selon la structure suivante :

```
<w:document
  xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
  <w:body>
    <w:p>
      <w:r>
        <w:t>Welcome to Office Open XML world !</w:t>
      </w:r>
    </w:p>
  </w:body>
</w:document>
```

Listing 1

L'exemple du listing 1 décrit un document possédant un seul et unique paragraphe. Voici le détail de cette structure :

- document : c'est l'élément racine du document principal.
- body : cet élément peut contenir plusieurs paragraphes.
- p : représente un paragraphe, et peut contenir un ou plusieurs run et des propriétés dans une balise fille `pPr` communes à tous les runs de ce paragraphe.
- r : représente un run, c'est-à-dire un ensemble contigu de caractères ayant les mêmes propriétés. Tout comme pour les paragraphes, chaque run peut posséder un élément fils `rPr` définissant les propriétés spécifiques de ce run. Nous reviendrons juste après sur ce concept fondamental de Open XML.
- t : le contenu textuel du run parent. Le contenu est principalement du texte non formaté. Le formatage de ce texte est hérité des propriétés du run et des propriétés du paragraphe. Cet élément contient aussi souvent l'attribut `xml:space="preserve"` afin de bien spécifier la prise en compte des espaces dans le texte.

Contrairement à des formats tel que le HTML, le format Open XML n'accepte pas l'intégration de propriétés de formatage directement à l'intérieur du contenu du texte. C'est pourquoi lorsque vous voudrez décrire le texte "Office **Open** XML" dans un document Open XML, il vous faudra créer au moins trois runs dans le paragraphe : un pour 'Office', un autre pour 'Open' avec la propriété 'gras' activé, et un dernier pour 'XML'.

Document `WordProcessingML` minimal

La création d'un document `WordProcessingML` n'a rien de très compliqué, les spécifications (1:11.2) définissent un ensemble minimal pour

Bureautique

le contenu d'un document de ce type. Logiquement, il y a trois parties distinctes qui doivent retenir notre attention : la partie de type de contenu, la partie de relations du paquet et la partie principale du document.

La partie de type de contenu

Cette partie dont l'URI est `/[Content_Types].xml` doit au moins décrire le type de contenu des deux autres parties, celui des relations du paquet et du contenu principal) :

```
<Types xmlns="http://schemas.openxmlformats.org/package/2006/content-types">
  <Default Extension="rels" ContentType="application/vnd.openxmlformats-package.relationships+xml"/>
  <Default Extension="xml" ContentType="application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml"/>
</Types>
```

Listing 2

La partie de relations du paquet

Cette partie possédant l'URI `/_rels/.rels` décrit l'ensemble des relations entre le paquet, c'est-à-dire votre document bureautique, et les différentes parties qui composent le contenu de celui-ci. Etant donné que seule la story principale du document est indispensable, le fichier de relations ne comporte qu'une seule et unique relation vers la partie principale :

```
<Relationships
xmlns="http://schemas.openxmlformats.org/package/2006/relationships">
  <Relationship Id="rId1"
Type="http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument" Target="document.xml"/>
</Relationships>
```

Listing 3

Remarque : bien que la partie principale d'un document `WordProcessingML` possède habituellement l'URI `/word/document.xml`, nous créons cette partie à la racine du paquet pour illustrer la souplesse du mécanisme de relations.

La partie principale

Cette partie décrit le contenu du document, et la structure minimale à observer est relativement simple (un paragraphe vide) :

```
<w:document
xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
  <w:body>
    <w:p/>
  </w:body>
</w:document>
```

Listing 4

Génération avec .NET

Maintenant que nous avons pris connaissance de la structure minimale à réaliser pour créer un document, nous allons nous adonner au développement d'un petit programme qui vous permettra de générer des documents `WordProcessingML` (lisible par Word 2007 et Word 2000/XP/2003 avec le pack de compatibilité à l'heure où cet article est écrit, et par OpenOffice, WordPerfect, ... quand ces suites supporteront pleinement ce format). Ce programme dont vous trouverez les

sources sur le site www.programmez.com va vous permettre d'entrer du texte et de générer simplement un document Open XML (Fig.2). Pour ce

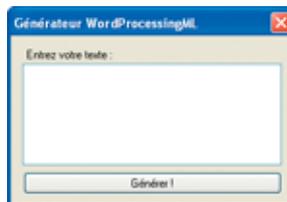


Fig.2.

faire, nous allons utiliser l'API livrée avec le framework .NET 3 et particulièrement l'espace de noms `System.IO.Packaging`. Le seul prérequis nécessaire pour développer un programme avec cette API est donc la présence du framework .NET 3 sur votre machine.

Configuration de l'environnement

Pour pouvoir utiliser l'espace de noms `System.IO.Packaging`, vous devez importer l'assembly `WindowBase.dll` dans votre projet. L'assembly se trouve par défaut dans le répertoire `c:\Program Files\Reference Assembly\Microsoft\Framework\v3.0\`.

Création du package

La création d'un document débute par la création d'un paquet OPC qui contiendra l'ensemble des parties du document, dans notre cas, les parties nécessaires pour créer un document minimal : la partie de type de contenu, la partie de relations du paquet et la partie principale.

L'utilisation des API `System.IO.Packaging` simplifie grandement la manipulation de la structure OPC ; ainsi pour créer un paquet Open XML, seule la ligne suivante est nécessaire :

```
Package pkg = Package.Open("MonDoc.docx", System.IO.FileMode.Create);
```

Listing 5

L'appel à la méthode `Open()` permet d'ouvrir ou de créer un paquet, et il vous faudra utiliser les méthodes `Flush()` et `Close()` pour signifier au paquet d'enregistrer toutes les parties que vous aurez manipulées lorsque vous voudrez fermer/enregistrer votre paquet.

Création de la relation vers la partie principale

Avant de créer la partie principale, nous allons tout d'abord créer le lien vers celle-ci. A noter que l'ordre des manipulations création partie/création lien importe peu.

Cette tâche nécessite tout d'abord la création d'une URI, ici `document.xml` à la racine du paquet, puis d'une relation à l'aide de la méthode `CreateRelationship()` du paquet nouvellement créé. Les paramètres sont simples et correspondent aux attributs d'une relation : l'URI cible, la portée de la liaison – la partie se trouve dans le paquet : `TargetMode.Internal`, ou alors en dehors du paquet : `TargetMode.External`, le type de la relation et, optionnellement, l'identifiant de la relation.

```
// Création de la relation vers la partie principale
Uri docUri = new Uri("/document.xml", UriKind.Relative);
pkg.CreateRelationship(docUri, TargetMode.Internal,
"http://schemas.openxmlformats.org/officeDocument/2006/relationships/officeDocument", "rId1");
```

Listing 6

Remarque : nous n'avons pas besoin d'ajouter la relation au paquet avec une quelconque méthode, en effet l'appel à `CreateRelationship()` se charge d'ajouter automatiquement la relation ainsi créée dans le

paquet. Dans le même esprit, nous ne créerons pas non plus la partie des relations du paquet, au même titre que la partie contenant les types de contenu : c'est toujours l'API en interne qui se charge de ces tâches élémentaires pour nous.

Création de la partie principale et de son contenu

La création de la partie principale est aussi aisée que l'ajout de la relation au paquet, une seule ligne suffit et suit la même logique :

```
// Création de la partie principale du document
PackagePart mainPart = pkg.CreatePart(docUri,
"application/vnd.openxmlformats-officedocument.wordprocessingml.document.main+xml");
```

Listing 7

Maintenant que la partie est créée, nous allons lui ajouter du contenu XML tout en respectant la structure que nous avons évoquée précédemment. Pour pouvoir modifier le contenu d'une partie, vous devez tout d'abord récupérer le flux de celle-ci à l'aide de la méthode `GetStream()` :

```
using (Stream docStream = mainPart.GetStream())
{
    XmlDocument xmlDoc = new XmlDocument();

    // Création de la structure du story principal
    XmlElement documentElem = xmlDoc.CreateElement("w:document",
wNamespace);
    XmlElement bodyElem = xmlDoc.CreateElement("w:body", wNamespace);
    XmlElement paragraphElem = xmlDoc.CreateElement("w:p", wName
space);
    XmlElement runElem = xmlDoc.CreateElement("w:r", wNamespace);
    XmlElement textElem = xmlDoc.CreateElement("w:t", wNamespace);
    XmlText textElemContent = xmlDoc.CreateTextNode(textContent);

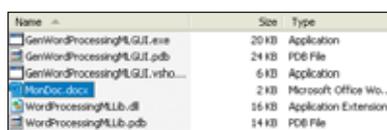
    xmlDoc.AppendChild(documentElem);
    documentElem.AppendChild(bodyElem);
    bodyElem.AppendChild(paragraphElem);
    paragraphElem.AppendChild(runElem);
    runElem.AppendChild(textElem);
    textElem.AppendChild(textElemContent);

    xmlDoc.Save(docStream);

    pkg.Flush();
    pkg.Close();
}
```

Listing 8

Après exécution du programme, vous devriez constater la présence d'un fichier `MonDoc.docx` dans le répertoire de sortie du projet (Fig. 3). Celui-ci contient l'ensemble des parties minimales pour un document Office Open XML



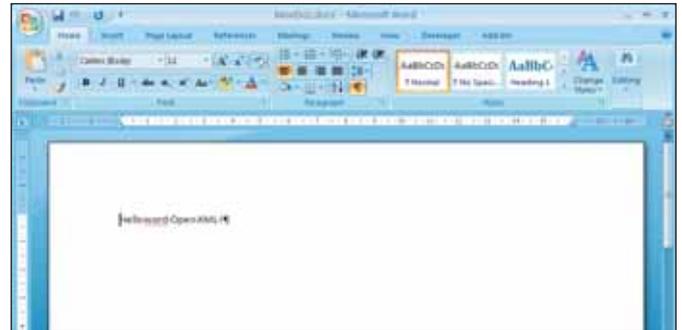
(Fig. 3)

- La partie de type de contenu : `[Content_Types].xml`
- La partie des relations du paquet : `._rels/.rels`
- La partie principale contenant la story principale : `document.xml`

Name	Size	Packed	Type
._rels			Folder
document.xml	211	211	XML Document
[Content_Types].xml	346	346	XML Document

Voici ce que vous observerez en ouvrant le document généré dans Word 2007 (avec le texte 'Hello world

Open XML !' saisi dans le champ de l'interface) :

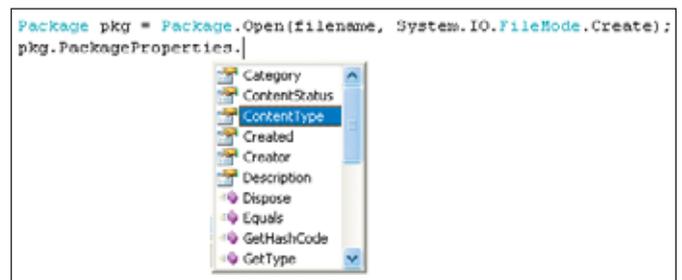


Vous pouvez constater que la génération d'un document basique – même si nous n'avons permis qu'un unique paragraphe homogène – est relativement simple grâce à l'utilisation des API de .NET 3. D'ailleurs, l'exemple que nous avons réalisé n'illustre que très peu des possibilités de cette API.

Néanmoins, vous l'avez sans doute remarqué, nous n'avons pas manipulé le document entièrement à l'aide de cette API. Le contenu même du document a fait l'objet d'une création d'un arbre XML et non d'une manipulation d'objets et d'appels de méthodes. Cela est dû au fait que l'espace de nom `System.IO.Packaging` n'est en réalité qu'une implémentation avancée des spécifications de l'Open Packaging Convention et non des spécifications complètes du format Office Open XML.

Ajouter une partie de propriétés

Après ce petit bémol sur les capacités de l'API de .NET 3 - choix qui néanmoins se justifie en de nombreux points - il serait dommage de ne pas aller un peu plus loin. C'est pourquoi nous allons voir comment changer les propriétés du document : l'auteur, la version, la date de création, ... En réalité rien de plus simple puisque c'est la propriété `PackageProperties` du paquet qui fera le travail à votre place (création de la partie, ajout de la relation et du type de contenu) :



Ainsi, si vous souhaitez ajouter des propriétés ou même les modifier, vous le ferez de la façon suivante :

```
Package pkg = Package.Open(filename, System.IO.FileMode.Open);
pkg.PackageProperties.Modified = DateTime.Now;
pkg.PackageProperties.Creator = createur;
pkg.PackageProperties.Description = description;
pkg.PackageProperties.Title = title;
pkg.Flush();
pkg.Close();
```

Listing 9



Fig. 4

Après l'exécution du listing 9, vous pourrez observer le succès de l'opération avec la boîte des propriétés du document (Fig. 4).

Vous constaterez que dans un souci de permettre aux développeurs d'exploiter rapidement et avec le plus de facilité possible les spécifications, celles-ci sont découpées en 5 parties bien distinctes. En tant que développeur maintenant averti, nous allons utiliser la documentation de 'Référence des schémas' d'Open XML, - soit la partie 4 - qui a elle seule totalise près de 5200 pages. Pour travailler efficacement, je vous encourage à prendre quelques secondes - voire minutes selon votre machine - pour ouvrir le document sous Word 2007 plutôt que d'utiliser sa version PDF, vous gagnerez ainsi du temps précieux à la longue.



Comme exemple d'utilisation de la spécification, nous allons tenter de mettre une couleur de fond à notre document.

Avant tout chose, il est important de chercher dans le sommaire - ou via la carte du document de Word 2007 - la partie réservée à WordprocessingML (§2). Logiquement, vous savez que cette propriété doit se trouver dans la story principal du document, et plus précisément sous la balise document : et c'est effectivement le cas !

Une fois situé dans le paragraphe de la spécification concernant l'élément background (§2.2.1) fils de l'élément document (§2.2.3), vous allez pouvoir trouver une description détaillée et un exemple général de l'utilisation de cette balise. Les autres informations à savoir, les balises

parents et enfants, et les attributs donnent autant de possibilité de pouvoir naviguer directement par hyperliens vers les paragraphes concernés.

Exploiter les spécifications pour développer efficacement

Open XML étant un standard de l'ECMA (ECMA-376 : Office Open XML File Format), vous devrez tout d'abord récupérer les spécifications sur le site officiel (cf. le lien dans les références en

parents et enfants, et les attributs donnent autant de possibilité de pouvoir naviguer directement par hyperliens vers les paragraphes concernés.

Nous voulons spécifier un fond uni, et pour cela l'attribut color va nous être très utile. La spécification définit très précisément cet élément - et son format RRGGBB -, et la façon de l'utiliser avec de surcroît un exemple à la clef : une spécification exhaustive faite pour les développeurs ! Voici le code XML que nous allons rajouter à la suite de l'élément document :

```
<w:background w:color="2C34FF"/>
```

Listing 10

Avant d'exécuter les sources associées à cet article, sachez que pour que cette propriété soit effective, vous devrez spécifier son activation dans la partie de configuration 'settings.xml' avec l'élément 'displayBackgroundShape' (§2.15.1.25).

Conclusion

Cet article a été l'occasion d'une part de montrer un exemple basique, mais concret, de génération 'from scratch' d'un document WordprocessingML ; et d'autre part, de démontrer la facilité d'utilisation de l'API incluse dans .NET 3.

Evidemment, plus le formatage de votre document sera complexe et riche, et plus la structure XML à générer sera complexe. La manipulation des documents Open XML, que ce soit du WordprocessingML, du SpreadsheetML ou du PresentationML, est grandement simplifiée avec l'utilisation de .NET 3. Et même si l'on regrette de ne pas pouvoir manipuler les documents directement de façon orientée objet, .NET 3 et les spécifications du standard ouvert Open XML répondent admirablement aux besoins des développeurs et du marché.

Références

- Lien vers le téléchargement du framework .NET 3 : <http://www.microsoft.com/downloads/details.aspx?FamilyId=19E21845-F5E3-4387-95FF-66788825C1AF&displaylang=en>
- Lien vers le téléchargement du SDK Windows Vista et .NET 3 : <http://www.microsoft.com/downloads/details.aspx?FamilyId=117ECFD3-98AD-4D67-87D2-E95A8407FA86&displaylang=en>
- Lien vers le téléchargement des extensions pour Visual Studio 2005 : <http://www.microsoft.com/downloads/details.aspx?FamilyId=935aabf9-d1d0-4fc9-b443-877d8ea6eab8&DisplayLang=en>
- MSDN France section OpenXML : <http://www.microsoft.com/france/msdn/office/openxml/default.msp>
- Site OpenXMLDeveloper : <http://openxmldeveloper.org/>
- Spécifications Open Packaging Convention (également inclus dans les spécifications de Open XML) : <http://www.microsoft.com/whdc/xps/xpspkg.msp>
- Les spécifications Office Open XML File Format ECMA 376 : <http://www.ecma-international.org/publications/standards/Ecma-376.htm>



Grégory Renard
MS Regional Director,
MVP. CTO de Wygwam
Expertise technique .Net:
conseil, formation, développement et solutions.



Julien Chable
MVP,
Développeur
et Consultant,
Wygwam

Sécurité WCF : Authentification par certificats mutuels

Windows Communication Foundation (WCF), auparavant connu sous le nom de code "Indigo", est un ensemble de nouvelles technologies de communication pour la plate-forme Windows.

Comme dans toutes les architectures orientées services, des questions sur la sécurité telles que l'intégrité, la confidentialité, l'authentification, l'autorisation et l'audit de la communication sont posées. Le modèle de sécurité de WCF apporte une réponse à ces besoins en 3 parties fonctionnelles :

- Sécurité du transfert : responsable de la confidentialité, de l'intégrité, et de l'authentification.
- Autorisation : responsable de l'apport d'une infrastructure pour la gestion des autorisations.
- Audit : responsable de l'enregistrement des événements liés à la sécurité.

Dans cet article, nous nous intéresserons essentiellement à la partie " Sécurité du transfert ", et plus précisément, à la fonction d'authentification qui peut être réalisée à l'aide de plusieurs types d'identifiants, le plus commun étant le couple " nom d'utilisateur et son mot de passe ". Avant d'entrer dans le vif du sujet, voici une petite introduction sur les modes de sécurité ainsi que les types d'identités supportés par WCF. Les concepts de base de WCF, tels qu'un "endpoint" regroupant les notions de " contract, bindings, et address " ne seront pas traités dans cet article.

Mode de sécurité et type d'identité

Voici les différents modes de sécurité supportés par WCF :

Mode	Description
Aucun (None)	Aucune sécurité.
Transport	Utilise un transport sécurisé comme HTTPS pour l'intégrité, la confidentialité et l'authentification mutuelle.
Message	Utilise le standard " WS-Security " pour sécuriser le message SOAP pour l'intégrité, la confidentialité et l'authentification mutuelle.
Mode Mixte	Utilise un transport sécurisé pour l'intégrité, la confidentialité et l'authentification du serveur. Utilise la sécurité message (WS-Security ou autres standard) pour l'authentification du client.

Nous choisirons le mode " Message " dans notre exemple pour une meilleure interopérabilité grâce au standard " WS-Security ". Une fois ce mode déterminé, il nous faut à présent choisir le type d'identité à utiliser. L'objectif de cet article étant de vous guider dans la mise en place d'une authentification par certificats mutuels, nous adopterons évidemment les certificats X.509 comme type d'identité.

Voici les autres types d'identités supportés par le mode de sécurité message :

Mode	Description
Aucun (None)	Autorise le service à interagir avec les clients anonymes.
Windows	Autorise l'échange de messages SOAP sous le contexte d'authentification de Windows. (Kerberos ou NTLM)
Nom d'utilisateur (Username)	Autorise le service à interagir uniquement avec les clients authentifiés à l'aide d'un nom d'utilisateur comme identité.
Certificat (Certificate)	Autorise le service à interagir uniquement avec les clients authentifiés à l'aide d'un certificat.
Windows CardSpace	Autorise le service à interagir uniquement avec les clients authentifiés à l'aide de Windows CardSpace.

Remarque : Il est intéressant de noter au passage que WCF supporte " CardSpace " comme type d'identité avec le mode message, ce qui n'est pas de cas avec le mode transport.

Authentification par certificats mutuels

Nous voici dans le vif du sujet, à savoir quelles sont les étapes à suivre pour la mise en place d'un service et d'un client WCF avec un mode de sécurité " Message " et des certificats mutuels comme identifiants. Pourquoi mutuels me direz-vous? Et bien tout simplement parce que le client et le service sont authentifiés par deux certificats, un pour le client et l'autre pour le service. Ceci permet au client de vérifier de façon sûre et digne de confiance l'identité du service et réciproquement. Voici les caractéristiques du modèle de sécurité que nous allons mettre en œuvre :



Caractéristique	Description
Mode de sécurité	Message
Interopérabilité	Oui, avec les clients et services compatibles avec la norme WS-Security et des certificats X.509 token profile.
Authentification	Authentification mutuelle du service et client.
Intégrité	Oui
Confidentialité	Oui
Transport	http
Binding	WSHttpBinding

Remarque : Dans notre exemple, la négociation d'identité entre le client et le service sera désactivée. Les certificats doivent donc être installés sur le client et le serveur en amont de toute communication. L'attribut qui contrôle cette négociation est " NegotiateServiceCredential ", activé par défaut.

Notre première étape consiste à créer les certificats pour le service et le client WCF. Par la suite, nous définirons les " endpoint " associés.

1 Création des certificats

Nous utilisons les certificats générés par l'outil " Création de certificats " (Makecert.exe) fourni avec le Kit de développement du Framework .Net dans notre environnement de développement. En production, ces certificats devront être obtenus à l'aide d'une autorité de certification. Il est recommandé de créer les scripts afin d'automatiser cette étape et de pouvoir la rejouer dans cet environnement ou dans un autre tel que l'environnement de validation par exemple.

Création du certificat service (server)

Ce certificat permettra au client de s'assurer de l'identité du service. Pour cela, nous allons utiliser l'outil de création " Makecert " avec les options suivantes :

- -sr : Spécifie l'emplacement du magasin du certificat du sujet. En général, défini à " LocalMachine " pour un service.
- -ss : Spécifie le nom du magasin du certificat du sujet qui stocke le certificat de sortie. Cette valeur peut être spécifique à une application ou " My " pour le magasin personnel.
- -sky : Spécifie le type de clé du sujet, qui doit être signature, échange ou un entier qui représente un type de fournisseur. Dans notre cas, nous choisissons " exchange " pour que la clé publique puisse être échangée.
- -pe : Marque la clé privée générée comme étant exportable. Cela permet l'inclusion de la clé privée dans le certificat généré.
- -a : Spécifie l'algorithme de signature. Doit être md5 (valeur par défaut) ou sha1.
- -n : Spécifie le nom du certificat du sujet. La méthode la plus simple consiste à faire précéder le nom de CN= et d'entourer le tout entre guillemets, dans notre cas, nous utilisons le nom de domaine pleinement qualifié (FQDN) du serveur.
- La dernière option spécifie le nom du fichier .cer dans lequel le certificat de test X.509 sera écrit.

La commande suivante permet de créer un certificat auto signé, avec les options définies précédemment :

```
makecert -sr LocalMachine -ss My -sky exchange -pe -a sha1 -n CN=ServerFQDN WCFServerCert.cer
```

Attention : A chaque exécution de cette commande, un certificat va être créé et ajouté au magasin. Sa duplication engendrera un problème lors d'un accès par nom du sujet. Si vous avez besoin d'identifier précisément le certificat à utiliser, utilisez plutôt son " thumb print " qui est unique. Mais soyez averti que cette valeur sera modifiée en cas de renouvellement. Le plus simple reste à supprimer les doublons (Fig.A).

Comme notre service WCF est hébergé dans IIS :

- Le nom présenté par le " CN " du certificat X.509 doit correspondre avec le nom de domaine pleinement qualifié (FQDN) du serveur qui héberge le service WCF.
- Il faut donner la permission de lecture à ce certificat au compte " SERVICE RESEAU " ou " NETWORK SERVICE " sur un serveur Windows 2003 à l'aide du script ci-dessous :

```
echo *****
echo Donne le droit de lecture sur le certificat serveur
```

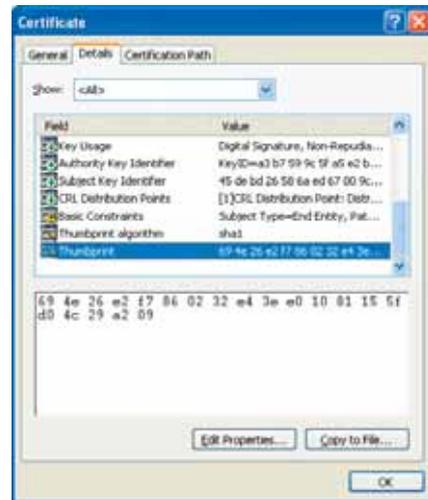


Fig A

```
echo *****
for /F "delims=" %%i in ("C:\FindPrivateKey.exe" My LocalMachine -n
CN^= ServerFQDN -a') do set PRIVATE_KEY_FILE=%%i
echo Y|cacls.exe "%PRIVATE_KEY_FILE%" /E /G "NETWORK SERVICE":R
```

Création du certificat client (poste client)

Ce certificat permettra au service de s'assurer de l'identité du client. Attention à ne pas confondre ici le processus d'authentification qui est différent du processus d'autorisation. Une fois authentifié avec l'identité du client (son certificat dans notre cas), le service peut alors décider d'autoriser ou non l'accès à ses services.

La commande suivante permet de créer un certificat auto signé, avec les options suivantes :

```
makecert -sr CurrentUser -ss My -sky exchange -pe -a sha1 -n CN=WebCenter WCFClientCert.cer
```

Vous avez ici les mêmes options que pour la création du certificat serveur, les seules différences sont :

- -sr : Pour le client, le magasin de stockage est généralement défini à " CurrentUser ".
- -n : Ici, nous utiliserons le nom de l'application " WebCenter ".

Si le certificat n'a pas été créé sur le poste client, ou s'il y a plusieurs postes clients, installer le certificat précédemment créé dans le magasin " Personal " de " Local Computer " à l'aide de ce script.

```
echo off
echo *****
echo Installe le certificat client sur le poste client
echo *****
certmgr.exe -add WCFClientCert.cer -r CurrentUser -s My
```

Remarque : Ce script utilise l'outil " Certificate Manager Tool " (Gestionnaire de certificats) aussi fourni avec le kit de développement du Framework .Net.

Attention : L'outil de création de certificats crée des certificats X.509 uniquement à des fins de tests. Ces certificats ne doivent pas être utilisés dans un environnement de production, ils nous permettent juste de valider le fonctionnement dans des environnements de développe-

ment, de tests, et de validation. De plus, par défaut, cet outil crée les certificats dont l'autorité racine est appelée " Root Agency ". Cette autorité n'étant pas une autorité de confiance certifiée, ces certificats ne sont pas sûrs. Pour se rapprocher un peu plus de l'environnement de production, il est possible de créer en amont des certificats client et serveur un certificat qui jouera ce rôle d'autorité de confiance.

Pour plus d'informations sur l'utilisation de ces outils de création et de gestion de certificat X.509, rendez-vous sur le site Msdn aux adresses suivantes :

- Création de certificats : [http://msdn2.microsoft.com/fr-fr/library/bfskty3\(VS.80\).aspx](http://msdn2.microsoft.com/fr-fr/library/bfskty3(VS.80).aspx)
- Gestionnaire de certificats : [http://msdn2.microsoft.com/fr-fr/library/e78bta0\(VS.80\).aspx](http://msdn2.microsoft.com/fr-fr/library/e78bta0(VS.80).aspx)
- Création d'une autorité de confiance temporaire : <http://msdn2.microsoft.com/en-us/library/ms733813.aspx>

Installation croisée des certificats

Puisque nous n'effectuons pas de négociation d'identité dans notre exemple, le certificat du serveur doit être installé sur le poste client en amont de toute communication et réciproquement.

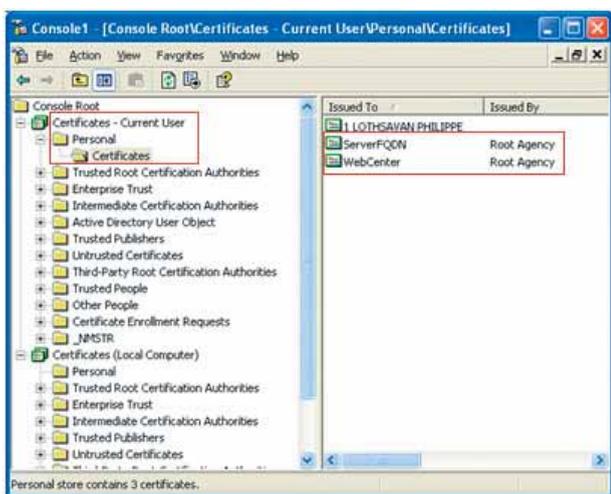
Le script suivant installe le certificat du serveur sur le poste client dans le magasin " Personal " du " Local Computer " :

```
echo off
echo *****
echo Installe le certificat du serveur sur le poste client
echo *****
certmgr.exe -add WCFServerCert.cer -r CurrentUser -s My
```

Le script suivant installe le certificat du client sur le serveur dans le magasin " TrustedPeople " du " Local Computer " :

```
echo off
echo *****
echo Installe le certificat du client sur le serveur
echo *****
certmgr.exe -add WCFClientCert.cer -r LocalMachine -s TrustedPeople
```

Remarque : Vous pouvez vérifier la bonne installation des certificats à l'aide de la console " MMC " .



2 Création du Endpoint service

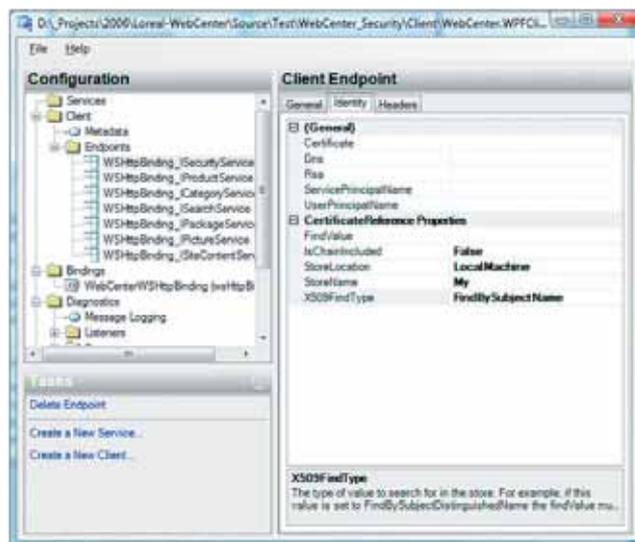
Une fois ces certificats créés et installés, il nous faut maintenant paramétrer le service pour le faire fonctionner avec le mode de sécurité choisi.

Pour créer un " endpoint ", WCF offre deux possibilités :

- Soit programmiquement via les APIs.
- Soit via un fichier de configuration, ce qui à mon avis, est bien plus pratique (pas besoin de compilation en cas de changement).

Pour cela, nous pouvons soit :

- Utiliser l'outil de paramétrage WCF " WCF Service Configuration Editor... " accessible directement dans le menu " Outil " de Visual Studio 2005 :



- Editer directement le fichier de configuration en suivant ces étapes :
- Ouvrir le fichier de configuration " app.config " ou " web.config " dans Visual Studio.
- Créer une section " serviceBehaviors " pour définir un comportement qui utilise notre certificat avec une recherche par nom du sujet (CN) et valide le certificat du client avec le mode " PeerOrChainTrust " :

```
<serviceBehaviors>
  <behavior name="WebCenterServiceBehaviorx509">
    <serviceCredentials>
      <clientCertificate>
        <authentication certificateValidationMode="PeerOrChainTrust" />
      </clientCertificate>
      <serviceCertificate
        findValue="ServerFQDN"
        storeLocation="LocalMachine"
        storeName="My"
        x509FindType="FindBySubjectName" />
      </serviceCredentials>
    </behavior>
  </serviceBehaviors>
```

Remarque : Le mode " PeerOrChainTrust " utilisé valide le certificat s'il le trouve dans le magasin " TrustedPeople ", il est pratique en développement car il ne vérifie pas l'autorité racine du certificat (CA). En production, il est préférable d'utiliser le mode " ChainTrust " .

Communication

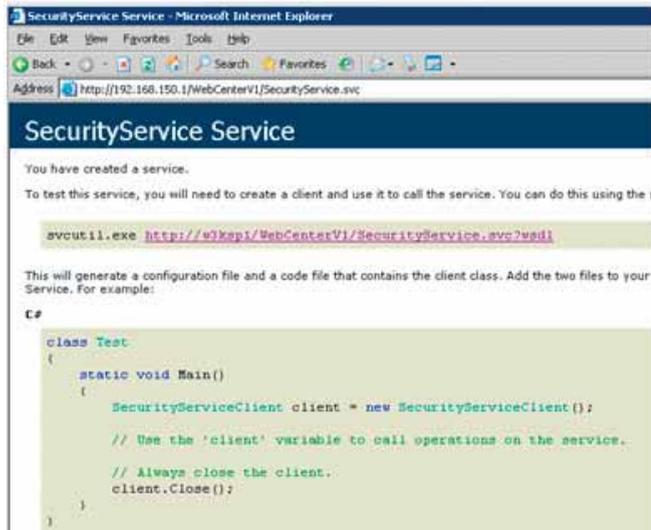
- Ensuite, créer une section " **bindings** " pour définir un mode de sécurité à " message " et un certificat comme identité d'authentification :

```
<bindings>
  <wsHttpBinding>
    <binding name="WebCenterServiceBindingx509">
      <security>
        <message clientCredentialType="Certificate"
        negotiateServiceCredential="false" />
      </security>
    </binding>
  </wsHttpBinding>
</bindings>
```

- Enfin, pour terminer, créer un " **endpoint** " service qui va utiliser le " behavior " et " bindings " précédemment définis :

```
<service name="WebCenterService"
behaviorConfiguration="WebCenterServiceBehaviorx509" >
  <endpoint binding="wsHttpBinding"
  bindingConfiguration=" WebCenterServiceBindingx509"
  contract="WebCenter.Interfaces.IWebCenterService" />
</service>
```

- Vérifier que le service fonctionne correctement sous Internet Explorer avec l'url du service.



3 Création du Endpoint client

Une fois le service opérationnel, il ne nous reste plus qu'à paramétrer le client pour qu'il communique correctement avec le service.

- Ouvrir le fichier de configuration " app.config " ou " web.config " dans Visual Studio.
- Commencer par créer une section " endpointBehaviors " pour définir un comportement qui utilise notre certificat avec une recherche par nom du sujet (CN) et valide le certificat serveur avec le mode " PeerOrChainTrust " :

```
<endpointBehaviors>
  <behavior name="WebCenterClientBehaviorx509">
    <clientCredentials>
      <clientCertificate
      findValue="WebCenter"
      storeLocation="LocalMachine"
      storeName="My"
      x509FindType="FindBySubjectName" />
      <serviceCertificate>
        <authentication certificateValidationMode="PeerOrChainTrust" />
      </serviceCertificate>
    </clientCredentials>
  </behavior>
</endpointBehaviors>
```

- Ensuite, créer une section " **bindings** " pour définir un mode de sécurité à " message " et un certificat comme identité d'authentification :

```
<bindings>
  <wsHttpBinding>
    <binding name="WebCenterClientBindingx509">
      <security>
        <message clientCredentialType="Certificate" negotiateService
        Credential="false" />
      </security>
    </binding>
  </wsHttpBinding>
</bindings>
```

- Enfin, pour terminer, créer notre " **endpoint** " client qui va utiliser le " behavior " et " bindings " précédemment définis :

```
<client>
  <endpoint name="WSHttpBinding_IWebCenterService"
  address=http://192.168.150.1/WebCenterV1/SecurityService.svc
  binding="wsHttpBinding"
  bindingConfiguration="WebCenterClientBindingx509"
  behaviorConfiguration="WebCenterClientBehaviorx509"
  contract="WebCenter.Interfaces.IWebCenterService" />
</client>
```

Conclusion

En choisissant les certificats comme identité d'authentification dans cet exemple, l'objectif était de vous montrer sa simplicité de mise en œuvre avec WCF. Aussi, nous avons choisi l'utilisation d'un seul certificat comme identité d'authentification des clients. Mais il est tout à fait possible et même recommandable de créer un certificat unique par client. En conclusion, WCF offre un choix important dans la sécurisation d'une communication, tant au niveau de son mode de sécurité que du type d'identité supporté. Vous remarquerez aussi que je ne fais aucune référence quand à l'implémentation, puisque avec ce modèle, il est intéressant de constater que la sécurité peut être mise en œuvre après l'implémentation du code métier, et ceci uniquement par le biais de fichiers de configuration.

■ Philippe Lothsavan – Microsoft Consulting Service

La Gestion d'exceptions avec WCF dans une approche SOA

L'objectif de cet article est de présenter les nouveaux mécanismes offerts par Windows Communication Foundation pour la gestion des exceptions dans les services. Il suppose une connaissance de base de l'architecture et du fonctionnement des services WCF.

Avec WCF, la communication entre un service et ses consommateurs (clients) se fait par échange de messages exposant un contrat de service. Ces messages comportent une partie *données* (DataContract) définissant les données échangées et une partie *opération* (OperationContract) définissant les opérations que fournit le service. WCF permet de définir simplement des services, leurs messages et opérations à l'aide d'attributs : ServiceContract, Data Contract/DataMember et OperationContract. La partie données d'un message est ainsi modélisée par une ou plusieurs classe(s) portant l'attribut DataContract. Chaque membre de cette classe devant être sérialisé dans le message, doit porter l'attribut DataMember (c'est un modèle dit " opt-in " : un membre ne portant pas l'attribut DataMember ne sera pas sérialisé dans le message). Exemple :

```
[ServiceContract]
public interface IService1
{
    [OperationContract]
    string MyOperation2(Customer cust);
}

public class service1 : IService1
{
    public string MyOperation2(Customer cust)
    {
        return "Hello: " + cust.FirstName;
    }
}

[DataContract]
public class Customer
{
    string firstName;
    string lastName;

    [DataMember]
    public string FirstName
    {
        get { return firstName; }
        set { firstName = value; }
    }

    [DataMember]
    public string LastName
    {
        get { return lastName; }
    }
}
```

```
set { lastName = value; }
}
}
```

Le service est défini par son interface (IService1) et implémenté dans la classe service1. L'opération MyOperation2 reçoit une instance de Customer (DataContract) en paramètre et renvoie une chaîne. Il existe d'autres façons de définir un service qui sont hors du périmètre de cet article.

Exceptions

L'exécution de code côté service peut lever des exceptions, et il est d'ailleurs recommandé d'utiliser la gestion d'exceptions .NET lors de l'implémentation d'un service. Traditionnellement les exceptions sont classées en :

Exceptions techniques

- Qui n'ont pas vocation à être exposées au consommateur et doivent être remplacées par un message d'erreur non détaillé.
- Ces exceptions peuvent compromettre la stabilité d'un service et doivent donc impérativement être journalisées (log) et doivent générer un évènement alertant l'équipe exploitation (via MOM par exemple).

Exceptions métier

- Ont vocation à être signalées au consommateur du service. Toutefois certaines informations contenues dans l'exception, en particulier la pile d'appel, ne doivent pas être communiquées au consommateur.
- Ne sont pas nécessairement journalisées.
- Peuvent être utilisées côté service pour essayer un autre chemin d'exécution ou côté consommateur pour réessayer un appel au service.

Il est donc de pratique courante de définir, côté service, une hiérarchie de classes d'exceptions dérivant de deux classes de base (par exemple TechnicalException et BusinessException) de façon à capturer l'information pertinente à chaque type d'exception et d'implémenter des comportements par défaut (tels que la journalisation).

Exceptions et SOA

L'approche SOA – qui est la base conceptuelle de WCF – est basée sur quatre règles fondamentales (" tenets ") :

1. Les frontières sont explicites
2. Les services sont autonomes
3. Les services partagent avec leurs consommateurs un schéma et un contrat, pas des classes
4. La compatibilité est exprimée à travers une policy.

Dans une approche SOA, on ne peut donc garantir qu'un consommateur aura accès aux classes spécifiques d'exceptions définies par le service (Règle 3). En outre, la règle 1 (frontières explicites) interdit de propager simplement les exceptions vers le consommateur. Ceci permet d'isoler

Communication

le consommateur des détails d'implémentation du service et évite la fuite d'informations sensibles capturées dans une exception, comme la pile d'appel, qui n'a pas de sens côté consommateur et pourrait par contre se révéler utile pour des attaquants du service.

Un mécanisme est donc nécessaire pour :

- Décrire les erreurs qu'un service (en fait une opération proposée par un service) est susceptible de retourner, qu'elles soient techniques ou métier. Cette description doit être disponible dans un format standard (WSDL) afin de permettre à un consommateur de l'interpréter indépendamment de sa technologie d'implémentation.
 - Traduire, à l'exécution, les exceptions vers le format d'erreur publié.
- WCF propose un tel mécanisme que nous allons décrire ci-après.

Les exceptions en WCF

Par défaut, un service WCF ne propage aucune exception à ses consommateurs. Toute exception non traitée côté service retourne une " `FaultException` " sans aucun détail au consommateur. Afin de déboguer un service, il est possible de modifier ce comportement par défaut en ajoutant le flag " `serviceDebugIncludeExceptionDetailInFaults` " dans la définition du comportement de service :

```
<serviceBehaviors>
<behaviour name="returnFaults">
<serviceDebugIncludeExceptionDetailInFaults="true"/>
</behavior>
</serviceBehaviors>
```

Cette facilité ne doit toutefois être utilisée que pendant la phase de mise au point d'un service, en aucun cas lors de la mise en production.

La notion de Faults

Un service WCF doit utiliser la notion de " `Fault` " pour retourner des erreurs à ses consommateurs.

Définition

Une `Fault` est une classe sérialisable par le `DataContractSerializer` de WCF (donc une classe comportant l'attribut `[DataContract]`) permettant de spécifier de façon structurée l'information sur l'erreur d'exécution. Une `Fault` fait partie du contrat d'opération et permet d'exprimer quels types d'erreurs une opération est susceptible de retourner. On peut d'ailleurs spécifier plusieurs `Faults` pour une opération. Exemple :

```
[DataContract]
public class CustomerFault
{
    string _msg;
    [DataMember]
    public string msg
    {
        get { return _msg; }
        set { _msg = value; }
    }
    public CustomerFault(string message)
    {
        msg = message;
    }
}
```

Déclaration

L'opération susceptible de retourner la `Fault` " `CustomerFault` " doit être décorée de l'attribut `FaultContract` :

```
public interface IService1
{
    [OperationContract]
    [FaultContract(typeof(CustomerFault))]
    string MyOperation2(Customer cust);
}
```

Ceci équivaut à déclarer que l'opération `MyOperation2` accepte un " `Customer` " en paramètre et retourne normalement une chaîne. Toutefois, elle peut aussi retourner une erreur de type " `CustomerFault` ".

Utilisation: `FaultException<>`

Le service peut retourner une `Fault` à l'aide d'un nouveau type d'exception générique proposé par WCF: `FaultException<>`, au constructeur de laquelle est passé l'instance de la `Fault` à retourner à l'appelant de l'opération.

```
throw new FaultException<CustomerFault>(new CustomerFault(" client inconnu "));
```

Côté consommateur, l'exception est interceptée également grâce à `FaultException<>` :

```
catch(FaultException<CustomerFault> e)
{
    // accède aux détail de e (instance de CustomerFault)
}
```

Il existe une version non générique de `FaultException`, proposant une description simplifiée de l'erreur (sans `DataContract` spécifique).

```
throw new FaultException(new FaultReason("message erreur"));

catch(FaultException fe)
{
    // accede aux champs de FaultException, en particulier Reason
}
```

Règles

- Un service ne doit renvoyer que des `Fault` déclarées dans le contrat d'opération
- `FaultException<>` possède plusieurs constructeurs (overloads). Il est de bonne pratique d'utiliser un constructeur acceptant une `FaultReason`. Ceci permet au consommateur de choisir de ne traiter que l'exception non générique `FaultException` tout en obtenant un message suffisamment informatif.

Custom Error Handlers

L'approche décrite plus haut présente toutefois un niveau d'abstraction faible. En effet, la mécanique WCF sous-jacente est exposée au développeur de service (il manipule explicitement des `Fault` et des `FaultException<>`). En outre, il est de la responsabilité du développeur de prendre soin de retourner des `FaultException<>`, ce qui implique une traduction explicite des exceptions standards .Net en `FaultException<FaultContract>`. WCF propose une autre approche, centralisée, à travers la notion de `Custom Error Handler` (CEH).

CEH : Définition

Un CEH reçoit toute exception .NET non gérée au niveau service. C'est donc un point central où l'on peut convertir une exception dans la `Fault`

qui lui correspond. Un CEH est une classe qui implémente l'interface WCF `IErrorHandler` et donc ses deux méthodes :

```
namespace System.ServiceModel.Dispatcher{
    public interface IErrorHandler
    {
        bool HandleError(Exception error);
        void ProvideFault(Exception error, MessageVersion version, ref Message fault);
    }
}
```

Toute exception non gérée par le code du service est interceptée par le CEH juste avant génération du message de réponse du service. Un service peut comporter plusieurs CEH constituant une chaîne et qui sont appelés successivement si le prédécesseur a retourné `Faux` dans `HandleError`. `ProvideFault` est la méthode qui va effectivement convertir l'exception (reçue en paramètre) en un message WCF de `Fault`. `ProvideFault` a la responsabilité de constituer le message WCF qui va être retourné à l'appelant.

Utilisation

Pour appliquer le CEH, il est nécessaire de définir un attribut de service spécifique. C'est une classe dérivant de `System.Attribute` et implémentant `IServiceBehavior`.

```
class ErrorBehavior : Attribute, IServiceBehavior
{
    ...
    public void ApplyDispatchBehavior(...,ServiceHostBase serviceHostBase)
    {
        foreach (ChannelDispatcher d in serviceHostBase.ChannelDispatchers)
        {d.ErrorHandlers.Add(new MyErrorHandler());}
    }
}
```

De fait, cet attribut de service ajoute une instance du Custom Error Handler sur chaque gestionnaire de canal. Enfin, l'activation de ce comportement se fait en annotant la classe d'implémentation du service avec l'attribut de service (c'est l'implémentation qui doit être décorée et non l'interface) :

```
[ErrorBehavior]
public class service1 : IService1
{
    public string MyOperation2(Customer cust)
    {
        return "Hello: " + cust.FirstName;
    }
}
```

Une fois le CEH mis en place, toute exception atteignant la frontière du service sera transformée en message WCF d'erreur par la méthode `ProvideFault` du CEH.

Application : Framework simple de gestion des exceptions

Nous proposons une approche simple, basée sur les principes exposés dans cet article qui permet de définir un framework simple de gestion

centralisée d'exceptions. Le code d'implémentation de ces concepts est disponible sur le site du magazine. Les classes en sont volontairement dépouillées pour conserver un caractère didactique au code. Une implémentation de production serait évidemment plus riche.

Principes :

- On dispose d'une classe de base (abstraite) `ExceptionBase` définissant une méthode abstraite de mapping de l'exception sur un `FaultContract` (`TranslateToFault`), dont l'implémentation doit être fournie par les classes dérivées.

Cette classe fournit une implémentation simple de la création d'un message d'erreur (`DoTranslateToFault`), méthode protégée et statique qui peut être utilisée par les classes dérivées pour fabriquer le message de faute. (Chaque classe dérivée utilise a priori une classe de `Fault` spécifique).

```
public abstract void TranslateToFault(MessageVersion version, ref Message erreur);

protected static void DoTranslateToFault(MessageVersion version, ref Message
erreur, string raison, object instanceErreur)
{
    MessageFault f = MessageFault.CreateFault(new FaultCode("Receiver"),
new FaultReason(raison), instanceErreur);
    erreur = System.ServiceModel.Channels.Message.CreateMessage(version,
f, "failAction");
}
```

Un attribut de service, implémentant l'algorithme suivant :

- Si l'exception à traiter dérive de `ExceptionBase`, il appelle la méthode de transformation de l'exception en `FaultContract`.
- Si l'exception n'en dérive pas (par exemple pour une exception `.NET`), il crée une exception technique (dérivée de `ExceptionBase`) et appelle sa méthode de transformation.

Ce framework permet d'automatiser assez simplement et en un point central la transformation d'une exception en `Fault`. En effet, il suffit :

- De dériver des classes d'exceptions de la classe `ExceptionBase`.
- De définir les contrats de faute et implémenter le mapping `Exception/Contrat` dans la méthode appropriée de chaque classe d'exception.
- Décorer l'implémentation du service de l'attribut [`ExceptionMappingBehavior`], fourni par ce framework.

Dès lors, le développeur de service n'a pas à se préoccuper de manipuler des concepts WCF : il manipule des exceptions applicatives comme il le ferait dans un environnement pré-WCF. Le Framework va prendre en charge la conversion des exceptions vers les contrats de `Fault`. Il reste bien évidemment nécessaire d'annoter les opérations du service avec l'attribut `FaultContract`. Cette approche a toutefois une limite : comme on opère au niveau service (et non opération), il n'y a pas moyen de vérifier que l'exception traitée va renvoyer un `FaultContract` correctement déclaré sur l'opération. Une approche viable semble donc de limiter le nombre de `FaultContract` (on peut généralement se limiter à deux : technique et métier) et s'assurer que toutes les opérations sont déclarées comme pouvant retourner une `fault` technique ou métier. Cela permet de garantir qu'une opération ne retournera jamais une `fault` non déclarée dans son contrat tout en masquant complètement la présence de WCF au développeur de services métier.

■ **François Miton**

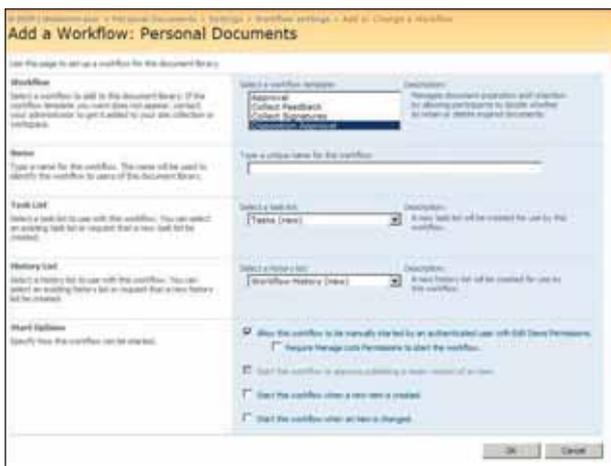
Consultant Principal - Microsoft Services | France

Intégrer des workflows dans Windows SharePoint Services

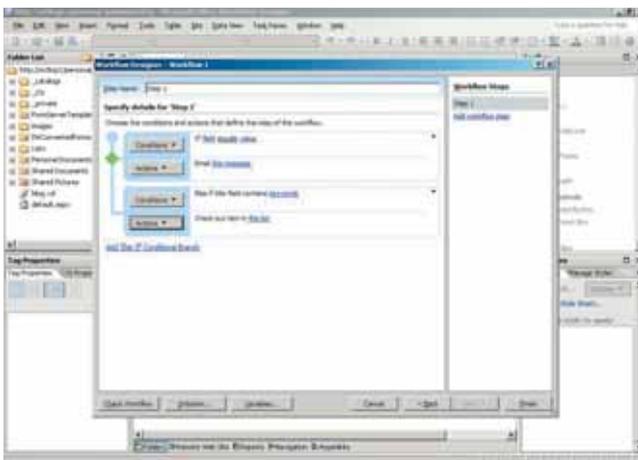
Workflow Foundation constitue aujourd'hui la pierre angulaire des développements autour des workflows chez Microsoft. A cet égard, la nouvelle version de Windows SharePoint Services ne déroge pas à la règle et se présente comme un hôte de choix pour développer et héberger ses workflows.

Utilisation des workflows

Il existe plusieurs manières de consommer des workflows dans SharePoint. La plus simple est d'utiliser ceux fournis en standard. La liste peut varier en fonction de l'édition de SharePoint utilisée, mais le principe reste toujours le même : créer une librairie et modifier ses paramètres pour associer un workflow à la création ou modification d'un item dans la librairie.



Lorsque les workflows fournis en standard ne suffisent pas, il est toujours possible d'en créer de nouveaux avec SharePoint Designer. Cet outil présente l'avantage d'être accessible à un utilisateur expérimenté, n'ayant pas forcément de compétences avancées en informatique :



Bien que cet outil soit extensible, il ne permet pas d'exploiter toutes les fonctionnalités des workflows sous SharePoint (machines à état, modification dynamique du workflow, etc). Pour ce faire, il faut alors utiliser Visual Studio. C'est ce que la suite de cet article va aborder.

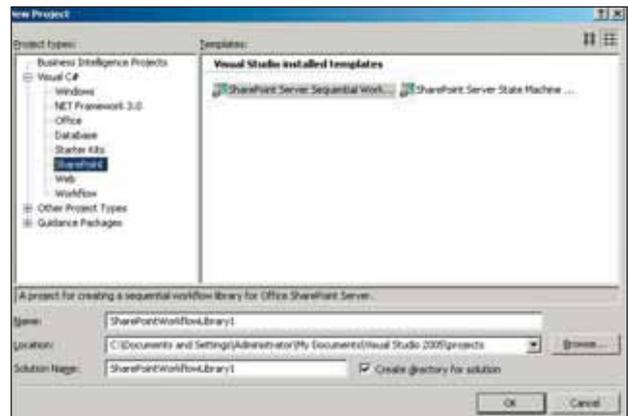
Utilisation de Visual Studio 2005

Avant de pouvoir développer des workflows avec Visual Studio 2005, il faut ajouter plusieurs éléments :

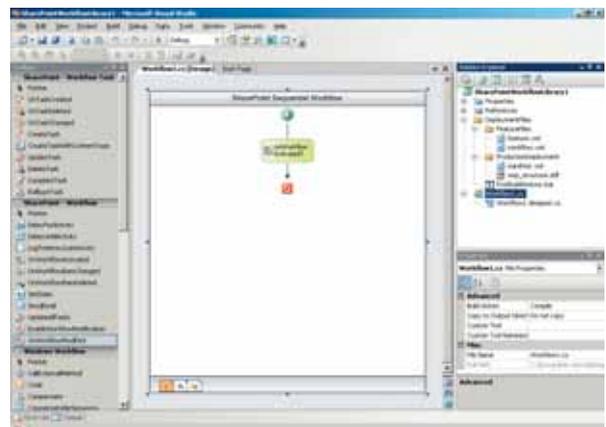
- Le SDK Windows. C'est optionnel, mais il contient toute la documentation, ce qui est toujours appréciable.
- Les extensions du Framework 3.0 pour Visual Studio.
- Le SDK de SharePoint. Il contient les modèles de projets ainsi que de nombreux exemples et la documentation.

Tous les liens sont donnés à la fin de l'article. Une fois ces éléments installés, une nouvelle catégorie de projet devient disponible : les projets SharePoint, de type séquentiel ou machine à état.

CREATION D'UN PROJET



Prenons l'exemple d'un workflow de type séquentiel. Le projet créé contient alors un workflow avec un fichier de code-behind et des fichiers de déploiement :



Commençons par regarder le workflow ainsi créé. La première chose que l'on note est qu'il n'est pas vide et commence par une activité de

type `OnWorkflowActivated`. Cette activité est essentielle au bon fonctionnement du workflow : elle a la charge d'initialiser la corrélation entre l'item ayant provoqué le lancement du workflow et l'instance du workflow. Sans elle, en cas de modification de l'item, le système ne saurait plus quelle instance du workflow réactiver. Elle a aussi la charge de récupérer une copie des propriétés du contexte de lancement (quels utilisateurs, quelles valorisations pour les colonnes de la librairie, etc.). Il faut bien se souvenir qu'il ne s'agit que d'une copie : la modification de ces valeurs par du code n'aura aucun impact sur le contenu réel de l'item. D'autres fichiers de type xml ou commande sont créés. Ils sont utilisés pour le déploiement et seront analysés plus loin.

PRINCIPES DE DEVELOPPEMENT

Le développement d'un workflow suppose l'utilisation d'activités. Toutes les activités fournies en standard avec Workflow Foundation ne sont pas supportées. Il est néanmoins possible d'utiliser au moins :

- L'activité `Code`, pour intégrer du code C# ou Visual Basic.
- L'activité `ConditionedActivityGroup`, pour exécuter des activités selon certaines conditions, tant qu'une condition n'est pas remplie pour l'activité.
- L'activité `Sequence`, pour exécuter séquentiellement une série d'activités.
- L'activité `Replicator`, pour créer de multiples instances d'une activité.

Le projet apporte aussi de nombreuses activités spécifiques à SharePoint, comme la création ou la modification de tâches, l'interaction avec l'item à l'origine du workflow, etc.

Avant de commencer les développements, il est nécessaire de rappeler quelques concepts. Tout d'abord, les workflows que vous créez sont potentiellement destinés à s'exécuter dans une ferme de serveurs. Si vous créez des workflows qui se maintiennent en mémoire (par exemple, une boucle sur un `sleep` pour surveiller le changement d'un item), vous allez limiter votre capacité à monter en charge et/ou à bénéficier de mécanismes de balance de charge. Ensuite, il faut bien faire attention à positionner les propriétés `CorrelationToken` correctement sur toute activité mettant le workflow en attente d'un événement extérieur. Pour reprendre le cas précédent, la bonne méthode est d'utiliser une activité `OnWorkflowItemChanged`. Cette activité va mettre le workflow en attente de toute modification sur l'item associé au workflow. Mais pendant cette attente, le workflow pourra être déshydraté par SharePoint si besoin. Et pour savoir quelle instance de workflow il faut réhydrater en cas de modification d'un item, SharePoint va utiliser son système de corrélation. Si la propriété `CorrelationToken` n'a pas été correctement positionnée sur l'activité, l'instance ne sera jamais réhydratée. Enfin, SharePoint optimise les transactions en ne procédant à la validation (`commit`) que lorsque le workflow se met en attente d'un événement. Par contre, lorsque vous modifiez un item dans votre code, la mise à jour est immédiate. Cela peut entraîner des incohérences entre les états des divers éléments modifiés par le workflow. Pour pallier ce problème, il est nécessaire de compenser via les gestionnaires de faute.

MODIFICATION DE L'ITEM VIA DU CODE

Voyons comment modifier un item depuis une activité de type `Code`. Dans le code-behind associé à cette activité, ajoutez le code suivant :

```
SPWeb myWeb = new SPSite(workflowProperties.SiteId).OpenWeb(
    workflowProperties.WebId);
SPListItem myItem = myWeb.Lists[workflowProperties.ListId].GetItemById(
    workflowProperties.ItemId);
```

```
// modification de l'item
myItem.Update();
```

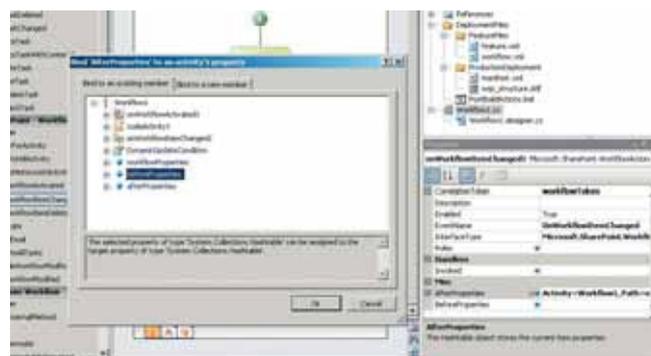
Ce code utilise les propriétés copiées lors de l'initialisation pour récupérer un pointeur sur l'item. Il peut alors modifier les propriétés (via `Properties`, consulter les versions, etc.). Enfin, il met à jour l'item avant de quitter. Sans ce dernier appel, toutes les modifications seraient perdues.

ATTENTE DE LA MISE A JOUR DE L'ITEM

Dans le code-behind, ajoutez deux membres de type `Hashtable`. Par convention, ils seront nommés `beforeProperties` et `afterProperties` :

```
public Hashtable beforeProperties = new Hashtable();
public Hashtable afterProperties = new Hashtable();
```

Dans le designer, ajoutez une activité de type `OnWorkflowItemChanged`. Commencez par positionner la propriété `CorrelationToken`. Par défaut, pour ce type de projet, l'identifiant `workflowToken` est utilisé pour le jeton associé au workflow. Ensuite, utilisez l'assistant de binding pour associer les propriétés `BeforeProperties` et `AfterProperties` aux deux propriétés précédemment créées. Si vous ne les aviez pas créées, vous pouvez aussi les générer en utilisant l'onglet "Bind to new member" de l'assistant. L'assistant se lance en cliquant deux fois sur le symbole bleu à côté du nom de la propriété.



A l'exécution, ces deux éléments contiendront une copie de toutes les propriétés de l'item avant et après sa modification. Pour exécuter un code en réaction à la modification, utilisez l'événement `Invoked` de l'activité. Vous pouvez générer le code en utilisant l'option "Generate Handlers" dans le menu contextuel associé à l'activité.

DEPLOIEMENT DU WORKFLOW

La première étape va consister à modifier le fichier `Feature.xml`. Ce fichier décrit quels éléments vont être déployés sur le serveur. Après ajout de la snippet `Feature`, le fichier contient les éléments suivants :

```
<Feature Id="GUID"
  Title="Default Title"
  Description="This feature is a workflow that ..."
  Version="12.0.0.0"
  Scope="Site"
  ReceiverAssembly="Microsoft.Office.Workflow.Feature, Version
=12.0.0.0, Culture=neutral, PublicKeyToken=71e9bce111e9429c"
  ReceiverClass="Microsoft.Office.Workflow.Feature.WorkflowFeatureReceiver"
  xmlns="http://schemas.microsoft.com/sharepoint/">
  <ElementManifests>
  <ElementManifest Location="workflow.xml" />
```

```
<ElementFile Location="MyForm.xsn"/>
</ElementManifests>
<Properties>
  <Property Key="GloballyAvailable" Value="true" />

  <!-- Value for RegisterForms key indicates the path to the forms relative
to feature file location -->
  <!-- if you don't have forms, use *.xsn -->
  <Property Key="RegisterForms" Value="*.xsn" />
</Properties>
</Feature>
```

Générer un Guid (par exemple avec guidgen.exe du SDK) et remplacer le contenu de l'id. Comme ce workflow n'est pas associé à des formulaires, supprimer la ligne associée aux formulaires et référenciez MyForm.xsn. Modifiez maintenant le fichier workflow.xml avec le snippet Workflow. Le fichier contient alors :

```
<Elements xmlns="http://schemas.microsoft.com/sharepoint/">
  <Workflow
    Name="My Workflow"
    Description="This workflow ..."
    Id="GUID"
    CodeBesideClass="ProjectName.Workflow1"
    CodeBesideAssembly="ProjectName, Version=1.0.0.0, Culture
=neutral, PublicKeyToken=publicKeyToken"
    TaskListContentTypeId="0x01080100C9C9515DE4E240019
05074F980F93160"
    AssociationUrl="_layouts/CstWrkflIP.aspx"
    InstantiationUrl="_layouts/IniWrkflIP.aspx"
    ModificationUrl="_layouts/ModWrkflIP.aspx"
    StatusUrl="_layouts/WrkStat.aspx">

    <Categories/>
    <!-- Tags to specify InfoPath forms for the workflow; delete tags for
forms that you do not have -->
    <MetaData>
      <Association_FormURN>associationFormURN</Association
_FormURN>
      <Instantiation_FormURN>instantiationFormURN</Instantiation
_FormURN>
      <TaskO_FormURN>taskFormURN</TaskO_FormURN>

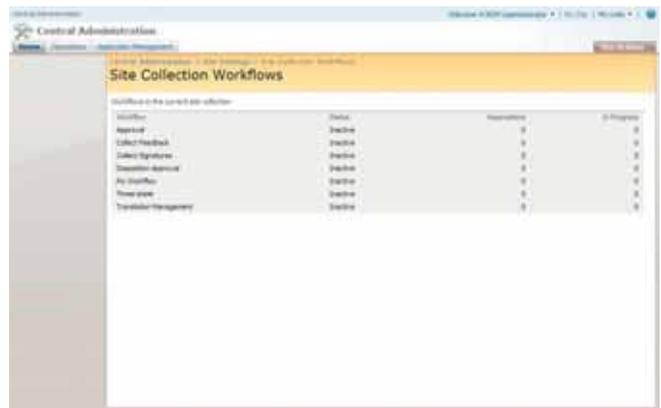
      <Modification_GUID_FormURN>modificationURN</Modification_
GUID_FormURN>
      <Modification_GUID_Name>Name of Modification</Modification
_GUID_Name>

      <AssociateOnActivation>>false</AssociateOnActivation>
    </MetaData>
  </Workflow>
</Elements>
```

Générez un autre Guid et changez la référence à la classe implémentant le workflow. Comme ce workflow n'est pas associé à des formulaires, vous pouvez supprimer toutes les métadonnées, à l'exception d'Asso-

ciateOnActivation. Dans un contexte de déploiement en production, il faudrait aussi modifier les fichiers manifest.xml et wsp_structure.ddf. Ces deux fichiers permettent ensuite la création d'un package de déploiement qui facilite les déploiements sur les fermes de serveurs. Ici, en développement, ces fichiers ne seront pas modifiés.

Enfin, il faut modifier les propriétés du projet. Tout d'abord, il faut que l'assembly soit signée. Dans l'onglet " Signing ", vérifiez que l'option "Sign the assembly" est bien cochée et sélectionnez un fichier de signature existant ou générez-en un si besoin. Ensuite, dans les événements de génération (onglet " Build Events "), changez NODEPLOY en DEPLOY. Générez le projet. La première fois, vous aurez des avertissements pendant la phase de désinstallation. Vous pouvez les ignorer sans souci. S'il n'y a pas eu d'autres erreurs, le workflow est alors déployé et disponible :



Conclusion

La création de workflows avec Visual Studio n'est pas une tâche très compliquée, mais elle demande de la minutie et le respect d'un certain nombre d'étapes. Elle ouvre de nombreuses perspectives comme la création d'une bibliothèque d'activités métiers intégrables par glisser/déplacer. L'utilisation de Visual Studio pourrait être un frein à l'utilisation de ce type de bibliothèque par des utilisateurs fonctionnels. Mais la capacité de designer à être hébergé dans une application Windows Forms ouvre des perspectives sur la création d'outils de modélisation et génération de workflows orientés métier. Il faut alors faire attention à générer un XAML sans code-behind.

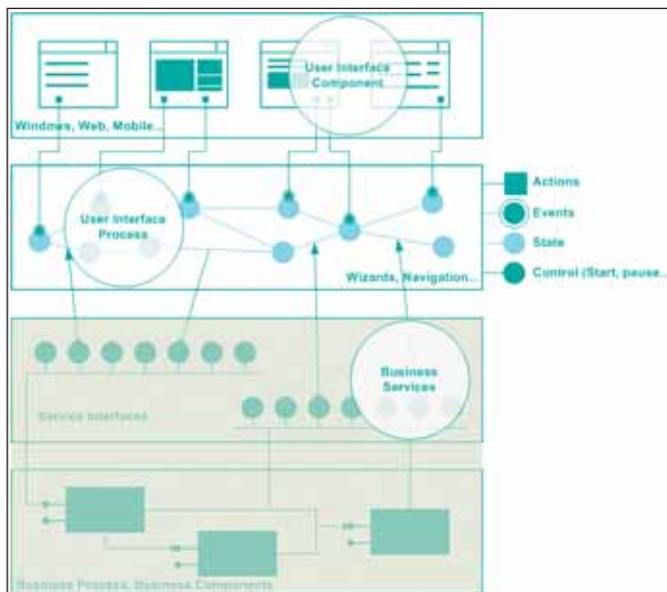
Ressources

Le développement d'application pour le Framework 3.0 nécessite le téléchargement des éléments suivants :

- Le SDK Windows, sur <http://www.microsoft.com/downloads/details.aspx?FamilyId=C2B1E300-F358-4523-B479-F53D234CDCCF&displaylang=en>.
- Les extensions de Visual Studio pour WCF et WPF, sur <http://www.microsoft.com/downloads/details.aspx?FamilyId=F54F5537-CC86-4BF5-AE44-F5A1E805680D&displaylang=en>.
- Les extensions de Visual Studio pour Workflow Foundation, sur <http://www.microsoft.com/downloads/details.aspx?familyid=5D61409E-1FA3-48CF-8023-E8F38E709BA6&displaylang=en>.
- Le SDK SharePoint, sur <http://www.microsoft.com/downloads/details.aspx?familyid=6D94E307-67D9-41AC-B2D6-0074D6286FA9&displaylang=en>.

■ P. Manac'h - Senior Consultant - Microsoft Consulting Services

Workflow



Un schéma d'architecture. Ce dernier positionne notre travail au sein d'une architecture n-tiers. Les couches de cette architecture reprennent ce qui avait été présenté par Microsoft dans son document "application architecture for .net : designing applications and services".

La déclaration de ces actions et donc leur mise à disposition aux éditeurs d'assistants se traduit par le développement d'un service d'échanges de données. Ce développement consiste en à la création d'une interface décorée par l'attribut ExternalDataExchange et d'une implémentation de cette interface reprenant chacune des actions listées précédemment.

```
[ExternalDataExchange]
public interface IWizardService
{
    event EventHandler<WizardEventArgs> OnCancel;
    event EventHandler<WizardEventArgs> OnNext;
    event EventHandler<WizardEventArgs> OnPrevious;
    event EventHandler<WizardEventArgs> OnFinish;
}
[Serializable]
public class WizardService : IWizardService
{
    events
    public void RaiseOnNextEvent(Guid instanceld)...
    public void RaiseOnPreviousEvent(Guid instanceld)...
}
```

Si la navigation s'abonne aux événements publiés par l'interface IWizardService via l'utilisation de l'activité HandleExternalEventActivity, ces derniers seront déclenchés par la couche de présentation via l'appel des méthodes RaiseOnNextEvent ou RaiseOnPreviousEvent lors d'un clic sur le bouton approprié.

A cette étape, nous disposons maintenant d'un service en charge de l'ensemble de la navigation de notre assistant. Malheureusement, cette navigation peut être plus ou moins complexe et conditionnée par des données fournies par l'utilisateur (étape facultative). Il est donc nécessaire qu'à la fois la couche de présentation et la navigation puissent partager un même contexte de données.

Ceci est délégué à un deuxième service d'échange de données. Suite à son développement, il est envisageable de faire héberger par WF l'ensemble de la logique de l'assistant (appels de services tiers, communication avec la base de données...).

```
[ExternalDataExchange]
public interface IDataContextService
{
    DataContext Get(Guid instanceld);
}
```

L'interface IDataContextService est réduite à sa plus simple expression, une unique méthode met à disposition en retour de la saisie d'un identifiant unique, un contexte de données. La couche de présentation et la couche de navigation partageant déjà un identifiant unique créé par WF lié à l'instance de la machine à états associés à l'assistant, c'est ce dernier que nous utiliserons. Un dernier effort sera réalisé afin de faciliter la manipulation de ces différents services. Il consistera en un développement d'une classe spécifique Wizard, embarquant en son sein l'environnement d'exécution WF et en le dédiant aux machines à états à destination d'assistants.

```
public sealed class Wizard
{
    public WizardInfo WizardInfo...
    public DataContext DataContext...

    public void Start(Type wizardStateMachineWorkflowType)
    public void Next()
    public void Previous()
    public void Cancel()
}
```

Cette classe regroupe en son sein tous les mécanismes d'interaction avec la logique de navigation d'un assistant (DataContext, Start, Next, Previous, Cancel), mais également des informations quant à l'étape en cours via la propriété WizardInfo.

```
public sealed class WizardInfo
{
    public Guid StateMachineWorkflowInstanceld...
    public WizardState WizardState...
    public string CurrentStateName...
    public ReadOnlyCollection<string> PreviousStateNames...
    public bool IsNextEnabled...
    ...
}
```

Nous retrouvons au sein de cette classe la propriété StateMachineWorkflowInstanceld dont il était question lors de l'évocation d'un identifiant unique et d'un contexte de données. Les autres informations permettent de paramétrer au mieux la représentation graphique de l'étape en cours. A cette fin, la propriété CurrentStateName est publiée en vue de relier l'étape en cours à un composant graphique particulier. Les propriétés IsNextEnabled, IsCancelEnabled sont à destination de la barre de boutons liés aux actions d'un assistant.

Détails d'implémentation

Les différents services présentés précédemment ne sont pas les seuls services que la classe Wizard à en charge d'enregistrer auprès du moteur d'exécution de workflow qu'elle embarque. En effet, il est également nécessaire de modifier le comportement par défaut asynchrone de l'exécution des workflows, étant donné que la navigation d'un assistant est un processus synchrone. Cette modification se traduit par l'enregistrement auprès du moteur d'exécution du service `ManualWorkflowSchedulerService`, et par l'appel à la méthode `RunWorkflow` publiée par ce service, suite au déclenchement d'un événement listé précédemment. L'appel aura pour effet de traiter instantanément l'évènement déclenché.

Quant au service `IDataContextService`, si son enregistrement à l'instar de tout service d'échanges de données s'appuie sur l'utilisation du service dédié `ExternalDataExchangeService`, différentes solutions existent en vue de le consommer au sein d'un workflow : l'utilisation de l'activité `CallExternalMethodActiviy`, la surcharge de la méthode `void Initialize(IServiceProvider)`. La seconde méthode a l'avantage de ne pas multiplier les appels au service et la création de variables à destination de ces appels.

```
protected override void Initialize(IServiceProvider provider)
{
    IDataContextService __dataContextService = (IDataContextService)
    provider.GetService(typeof(IDataContextService));
    __dataContext = __dataContextService.Get(this.WorkflowInstanceId);
    base.Initialize(provider);
}
```

Le dernier point que nous mettrons en lumière est la sauvegarde du nom de l'état précédent à l'état courant. Nous ne pouvons conclure cet article sans relater au moins une faiblesse de la plate-forme WF, celle-ci ne propose pour les machines à états un historique d'états qu'avec l'utilisation conjointe du service `SqlTrackingService`. La classe Wizard prend donc en charge, afin de lever cette contrainte, cette historisation d'états.

Pour aller plus loin

Si toute la mécanique propre aux assistants a été mise en place, peu d'efforts ont été fait pour guider l'utilisateur lors de l'édition de la navigation. Il suffirait pourtant de développer des activités spécifiques liées aux évènements Précédent ou Suivant en spécialisant par exemple l'activité `HandleExternalEventActivity`.

Il est bon de noter également que désormais vos assistants peuvent tirer bénéfice de l'ensemble des fonctionnalités offertes par WF que cela soit les fonctions de persistance, de supervision ou de transaction.

Conclusion

Projeté dans un plan entreprise, WF permet la constitution d'une boîte à outils personnalisés, destinée au développement de workflow (activités, règles...). Il est bien évident que les assistants ne sont pas les plus représentatifs de ceux se trouvant dans le SI d'une entreprise, mais leur caractère pédagogique est indéniable.

■ Anthony Guerot et Pascal Recchia

Consultants pour le compte de la société MexEdge

Membres de l'équipe éditoriale du portail NetFxFactory.org



Daniel Cohen-Zardi, SoftFluent :

"Notre expérience du Framework 3.0 est globalement très positive"

« Nous avons collaboré à la mise en œuvre de WPF chez deux de nos clients.

Ce qui est intéressant dans WPF, au-delà de l'innovation visuelle elle-même, c'est la programmation descriptive des interfaces utilisateur. L'utilisation de XAML permet de faire évoluer l'apparence graphique de l'application sans recourir à des développeurs au sens classique. En conséquence, nous avons vu sur ces deux projets le besoin de trouver des compétences graphiques particulières pour mettre en œuvre cette nouvelle technologie. C'est d'ailleurs pour accompagner ces nouveaux graphistes que Microsoft propose la nouvelle gamme de produits Expression, mais les compétences sont encore rares sur le marché.

De plus, ayant travaillé sur des versions Bêta, nos clients ont parfois souffert de devoir passer sur différentes CTP, avec une documentation pas toujours suffisamment fournie et des bogues divers. Chez l'un d'eux, nous avons aussi rencontré une réelle difficulté liée à la gestion de la transparence avec le composant Windows Media, avec en conséquence une surconsommation de la CPU, point qui n'est pas encore résolu aujourd'hui et qui a nécessité une approche de contournement.

En ce qui concerne WCF, notre client VCS Timeless, éditeur spéciali-

sé dans la gestion des chaînes de magasin, a mis en œuvre les mécanismes dynamiques pour les appels distants entre applications. Le passage d'un protocole d'appel de procédure distante à une approche SOA par simple configuration est donc une réalité constatée sur le terrain, et porteuse de valeur également pour les clients de VCS Timeless.

Nous travaillons par ailleurs sur le Framework 3.0 en R&D, pour la prise en compte de WPF, WCF et WWF dans notre propre usine logicielle. Ainsi sur WWF, nous avons déjà rédigé il y a près d'un an dans notre magazine en ligne N° 3 un article qui détaille sa valeur ajoutée. Le moteur de Workflow est ainsi mis au cœur du système d'exploitation, ce qui permet à des outils de plus haut niveau de le piloter.

En résumé, notre expérience du Framework 3.0 est donc globalement très positive, même si, comme toute technologie nouvelle, il ne faut pas négliger l'investissement de compétences et les petits problèmes de jeunesse. Enfin, notons que, si les gens assimilent souvent l'utilisation du Framework 3.0 à un pré-requis déploiement de Windows Vista, celui-ci fonctionne également sur Windows XP SP2 et Windows Server 2003 SP1. »

■ Daniel COHEN-ZARDI Fondateur de SoftFluent

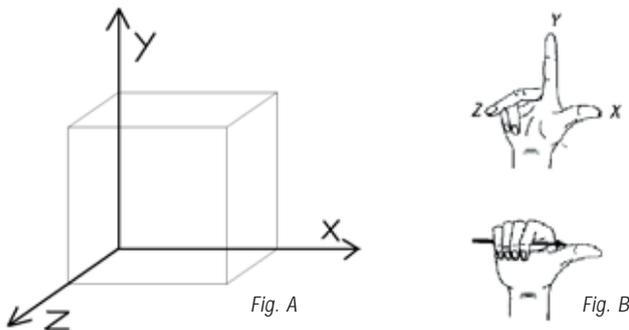
WPF et la 3D

Développer des applications utilisant de la 3D était jusqu'à présent relativement complexe. En effet, les API comme DirectX ou OpenGL demandent de bonnes connaissances en programmation et algorithmes 3D pour obtenir une simple forme 3D. Or l'utilisation de la 3D peut donner une nouvelle dimension aux applications. L'arrivée de WPF change un peu la donne.

WPF permet de faire du développement 3D de manière simple et rapide. Il s'agit d'une API de haut niveau, c'est-à-dire qu'elle se base sur d'autres, d'un niveau moindre, afin d'obtenir un niveau d'abstraction élevé et donc une simplicité de développement accrue. Cette simplicité a son revers, cette API est limitée en fonctionnalités. Pas la peine de penser faire du vertex shader ou du pixel shader avec WPF, mais ce n'est pas son but non plus. Dans cet article nous verrons comment fonctionne la 3D en WPF et comment on la met en place.

Tout est une question d'axes

Comme son nom l'indique, la 3D est composée de 3 axes, cela peut paraître anodin, mais vous vous rendrez vite compte que c'est loin d'être le cas. Voici la représentation d'un simple cube en 3D.



Ce cube est posé sur le plan formé par les axes X et Z. Les flèches indiquent l'orientation positive des déplacements (Fig.A).

Un moyen mnémotechnique pour retenir l'axe positif est d'utiliser la règle de la main droite (Fig.B).

Avec votre main droite vous formez un système de coordonnées et la direction de vos doigts vous donne la direction positive. Pour la rotation positive, vous fermez votre main droite sur l'axe et la rotation de vos doigts vous donne l'angle positif.

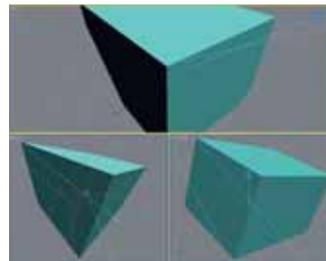
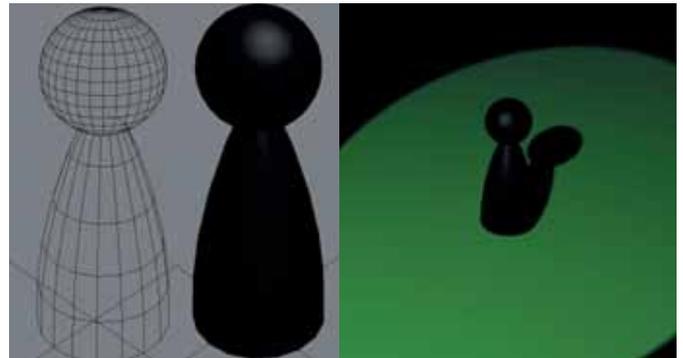
Il faut savoir aussi que les déplacements ne sont pas commutatifs en 3D, ainsi un déplacement puis une rotation ne donneront pas la même chose qu'une rotation identique puis un déplacement identique.

Je crée mon monde

Pour mettre en place un monde en 3D, on utilise le contrôle `viewport3D`. Ce contrôle sera l'endroit où la scène en 3D sera positionnée. Tout se fera dans ce monde.

Pour avoir un monde en trois dimensions il faut 3 éléments.

Premièrement l'**objet** : aussi appelé **Mesh**, cet objet est constitué de faces. Plus il y a de faces et plus l'objet est proche de la réalité. Il est aussi constitué d'une texture, ou mapping, cette texture lui donne toute la vie et toute la consistance. Mais on reviendra dessus plus tard.



Deuxièmement la **lumière** : les lumières sont de plusieurs types, on a des *lumières directionnelles* qui peuvent être associées à des lumières de soleil, tous les rayons sont droits. On a aussi les *lumières spot*, qui ont la même réaction que des projecteurs de spectacle, c'est un point qui

envoie la lumière dans une sorte de cône. Enfin, on a les *omni*, là c'est un point qui envoie de la lumière partout. Le choix de la lumière est très important car il permet de mettre des objets dans l'ombre et d'autres en avant. C'est vraiment la lumière qui donne la vie à une scène.

Enfin, la **caméra**. La caméra est l'œil du spectateur, elle est primordiale car elle détermine comment voir une scène, sous quel angle et avec quelle focale. C'est vraiment le *point de vue*.

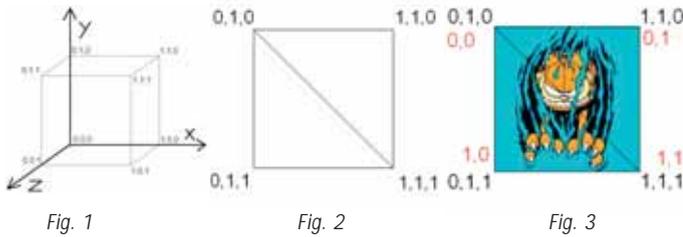
XAML 3D

Le XAML 3D permet de faire de la 3D très simplement en se basant sur un dérivé de XML. Commençons par placer notre Viewport 3D :

```
<Page xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" xmlns:sys="clr-namespace:System;assembly=microsoftcorlib" xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
  <Grid>
    <Viewport3D Name="mainViewport" ClipToBounds="True" Width="Auto" Height="Auto" MinWidth="100" MinHeight="100">
    </Viewport3D>
  </Grid>
</Page>
```

Il y a rien de spécial dans ce code, on a placé un contrôle en lui demandant de s'adapter automatiquement à la fenêtre ou à la page et que sa taille minimum soit de 100 sur 100.

On va placer dedans notre géométrie 3D. Pour cela, il faut savoir que les



faces en trois dimensions sont constituées de triangles. Chaque triangle sera déclaré dans un sens particulier. Le dessin de la figure 1 nous servira de modèle.

Rappelez vous de la règle de la main droite, si on ferme la main droite on obtient le sens positif de rotation. C'est ce sens qui est primordial ici. En fait, chaque triangle réagit comme un miroir sans tain. C'est-à-dire que d'un côté il s'affiche mais de l'autre il est invisible. On va donc créer notre cube.

```
<Page xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:sys="clr-namespace:System;assembly=mcorlib"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
<Grid>
<Viewport3D Name="mainViewport" ClipToBounds="True" Width
="Auto" Height="Auto" MinWidth="100" MinHeight="100">
<ModelVisual3D>
<ModelVisual3D.Content>
<Model3DGroup>
<GeometryModel3D x:Name="pCube1">
<GeometryModel3D.Geometry>
<MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1
1,1,1 1,1,1 1,0,1 1,0,0 1,0,0 1,1,0 1,1,1 1,1,1 0,1,1 0,0,1 0,0,1
1,0,1 1,1,1 0,1,0 1,1,0 1,0,0 1,0,0 0,0,0 0,1,0 0,0,0 1,0,0 1,0,1
1,0,1 0,0,1 0,0,0 0,0,0 0,0,1 0,1,1 0,1,1 0,1,0 0,0,0"/>
</GeometryModel3D.Geometry>
</GeometryModel3D>
</Model3DGroup>
</ModelVisual3D.Content>
</ModelVisual3D>
</Viewport3D>
</Grid>
</Page>
```

Dans ce code on a créé un model visual 3D qui peut être considéré comme une scène, un peu comme si une pièce, avec ses lampes, son éclairage et ses caméras était un model visuel, et dans ce model visuel on a ajouté un groupe de modèles 3D, qui formeraient les ensembles de la scène. Enfin on ajoute notre Mesh. Si vous mettez ce code rien n'apparaît, en effet nous avons notre objet mais il nous manque deux éléments encore.

Commençons par la lumière.

On choisira une directional light, c'est-à-dire une lumière qui envoie les rayons parallèlement, un peu comme un soleil. On a besoin d'une direction et d'une couleur, mais pas de point d'origine car avec des rayons parallèles il n'y a pas besoin :

```
<Page xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:sys="clr-namespace:System;assembly=mcorlib"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
<Grid>
<Viewport3D Name="mainViewport" ClipToBounds="True" Width
="Auto" Height="Auto" MinWidth="100" MinHeight="100">
<ModelVisual3D>
<ModelVisual3D.Content>
<Model3DGroup>
<DirectionalLight
Color="White"
Direction="-2,-3,-1" />
<GeometryModel3D x:Name="pCube1">
<GeometryModel3D.Geometry>
<MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1
1,1,1 1,1,1 1,0,1 1,0,0 1,0,0 1,1,0 1,1,1 1,1,1 0,1,1 0,0,1 0,0,1
1,0,1 1,1,1 0,1,0 1,1,0 1,0,0 1,0,0 0,0,0 0,1,0 0,0,0 1,0,0 1,0,1
1,0,1 0,0,1 0,0,0 0,0,0 0,0,1 0,1,1 0,1,1 0,1,0 0,0,0"/>
</GeometryModel3D.Geometry>
</GeometryModel3D>
</Model3DGroup>
</ModelVisual3D.Content>
</ModelVisual3D>
</Viewport3D>
</Grid>
</Page>
```

Enfin, il nous manque la caméra, après tout c'est à travers elle qu'on regarde. Elle est caractérisée par son emplacement, dans quelle direction elle regarde, avec quelle focale, et avec sa "myopie" c'est-à-dire depuis quelle distance et jusqu'à quelle distance elle peut voir. En informatique les caméras ne voient pas à l'infini et, encore plus bizarre, elles peuvent ne pas voir ce qu'il y a devant elles. C'est ce qu'on appelle le *clipping*, qui permet d'économiser les temps de calcul. Dans combien de jeux se rend-on compte qu'un arbre apparaît soudain quand on conduit sa voiture de rallye ? Il nous manque aussi la couleur de notre géométrie, car pour l'instant elle est tellement transparente qu'elle n'apparaît pas, on va donc lui ajouter une couleur en utilisant une brosse diffuse. Voici le code complet.

```
<Page xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:sys="clr-namespace:System;assembly=mcorlib"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
<Grid>
<Viewport3D Name="mainViewport" ClipToBounds="True" Width
="Auto" Height="Auto" MinWidth="100" MinHeight="100">
<Viewport3D.Camera>
<PerspectiveCamera
FarPlaneDistance="100"
LookDirection="-5,-5,-5"
UpDirection="0,1,0"
NearPlaneDistance="0"
Position="5,5,5"
FieldOfView="45" />
</Viewport3D.Camera>
```

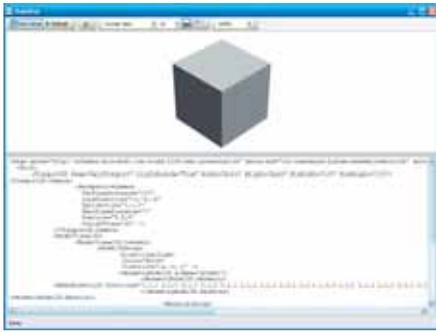


Fig.4



Fig.5



Fig.6

```
<ModelVisual3D>
  <ModelVisual3D.Content>
    <Model3DGroup>
      <DirectionalLight
        Color="White"
        Direction="-2,-3,-1" />
      <GeometryModel3D x:Name="pCube1">
        <GeometryModel3D.Geometry>
          <MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1
1,1,1 1,1,1 1,0,1 1,0,0 1,0,0 1,1,0 1,1,1 1,1,1 0,1,1 0,0,1 0,0,1
1,0,1 1,1,1 0,1,0 1,1,0 1,0,0 1,0,0 0,0,0 0,1,0 0,0,0 1,0,0 1,0,1
1,0,1 0,0,1 0,0,0 0,0,0 0,0,1 0,1,1 0,1,1 0,1,0 0,0,0"/>
        </GeometryModel3D.Geometry>
      <GeometryModel3D.Material>
        <MaterialGroup>
          <DiffuseMaterial>
            <DiffuseMaterial.Brush>
              <SolidColorBrush Color="#ffff
ffff" Opacity="1" />
            </DiffuseMaterial.Brush>
          </DiffuseMaterial>
        </MaterialGroup>
      </GeometryModel3D.Material>
    </GeometryModel3D>
  </Model3DGroup>
</ModelVisual3D.Content>
</ModelVisual3D>
</Viewport3D>
</Grid>
</Page>
```

Voici ce que donne le code dans le xaml pad (Fig.4).

Penchons nous un peu sur le mapping (ou mise en place d'un matériau sur le cube). Lorsqu'on crée un objet en 3D, on crée des faces triangulaire mises bout à bout et de différentes dimensions. Chacune de ces faces reçoit une partie du matériau qui lui aussi mis bout à bout forme le mapping total de l'objet.

Prenons la face du haut. On va limiter le code de notre objet à ceci :

```
<MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1 1,1,1"/>
```

Cette face ressemble au dessin de la Figure 2.

Dedans, je veux mettre une image en deux dimensions. Chaque point correspond a une coordonnée en 3D (coordonnées XYZ couleur noir) mais également une coordonnée de mapping (aussi appelée coordonnées U, V en couleur rouge) (Fig.3).

Pour mettre en place cette correspondance on utilise la propriété `TextureCoordinates`

```
<MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1 1,1,1"
TextureCoordinates="1,1 0,1 0,0 0,0 1,0 1,1"/>
```

On change la texture de `solidbrush` à `image brush` et voici le résultat (Fig. 5).

```
<DiffuseMaterial.Brush>
<ImageBrush ImageSource="C:\Documents and Settings\HP_Administrateur\Mes documents\Mes images\garfield.jpg" Opacity="1" />
</DiffuseMaterial.Brush>
```

Le cube au complet (Fig.6).

```
<GeometryModel3D x:Name="pCube1">
  <GeometryModel3D.Geometry>
    <MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,0 0,1,1
1,1,1 1,1,1 1,0,1 1,0,0 1,0,0 1,1,0 1,1,1 1,1,1 0,1,1 0,0,1 0,0,1
1,0,1 1,1,1 0,1,0 1,1,0 1,0,0 1,0,0 0,0,0 0,1,0 0,0,0 1,0,0 1,0,1
1,0,1 0,0,1 0,0,0 0,0,0 0,0,1 0,1,1 0,1,1 0,1,0 0,0,0"Texture
Coordinates="1,1 0,1 0,0 0,0 1,0 1,1 0,1 1,1 1,0 1,0 0,0 0,1
0,1 1,1 1,0 1,0 0,0,0,1 0,1 1,1 1,0 1,0 0,0,0,1 0,1 1,1 1,0 1,0 0,0
0,1 0,1 1,1 1,0 1,0 0,0,0,1"/>
  </GeometryModel3D.Geometry>
<GeometryModel3D.Material>
  <MaterialGroup>
    <DiffuseMaterial>
      <DiffuseMaterial.Brush>
        <ImageBrush ImageSource=
"C:\Documents and Settings\HP_Administrateur\Mes documents\Mes
images\garfield.jpg" Opacity="1" />
      </DiffuseMaterial.Brush>
    </DiffuseMaterial>
  </MaterialGroup>
</GeometryModel3D.Material>
</GeometryModel3D>
```

On peut aller encore plus loin en mettant une rotation sur le cube. Pour cela on va utiliser l'objet AxisAngleRotation3D pour mettre en place une rotation sur le cube. Juste entre </GeometryModel3D.Material> et </GeometryModel3D> on ajoute donc :

```
<GeometryModel3D.Transform>
  <Transform3DGroup>
    <RotateTransform3D>
      <RotateTransform3D.Rotation>
        <AxisAngleRotation3D x:Name="rotate" Angle="90.000
000" Axis="1.500000 1.000000 0.500000"/>
      </RotateTransform3D.Rotation>
    </RotateTransform3D>
  </Transform3DGroup>
</GeometryModel3D.Transform>
```

L'objet AxisAngleRotation3D est nommé rotate. Cela nous permettra d'y accéder à partir du storyboard. Voici le code au complet :

```
<Page xmlns="http://schemas.microsoft.com/winfx/2006/xaml/
presentation" xmlns:sys="clr-namespace:System;assembly=mcorlib"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" >
  <Grid>
<Grid.Resources>
  <ResourceDictionary>
    <Storyboard x:Key="OnLoaded">
      <DoubleAnimation Storyboard.TargetName="rotate"
BeginTime="00:00:00" Duration="00:00:15" Storyboard.Target
Property="Angle" From="0" To="360" RepeatBehavior="Forever">
    </DoubleAnimation>
  </Storyboard>
</ResourceDictionary>
</Grid.Resources>
<Grid.Triggers>
  <EventTrigger RoutedEvent="FrameworkElement.Loaded">
    <EventTrigger.Actions>
      <BeginStoryboard x:Name="OnLoaded_BeginStory
board" Storyboard="{DynamicResource OnLoaded}"/>
    </EventTrigger.Actions>
  </EventTrigger>
</Grid.Triggers>
  <Viewport3D Name="mainViewport" ClipToBounds="True" Width
="Auto" Height="Auto" MinWidth="100" MinHeight="100">
<Viewport3D.Camera>
  <PerspectiveCamera
    FarPlaneDistance="100"
    LookDirection="-5,-5,5"
    UpDirection="0,1,0"
    NearPlaneDistance="0"
    Position="5,5,5"
    FieldOfView="45" />
</Viewport3D.Camera>
<ModelVisual3D>
```

```
<ModelVisual3D.Content>
  <Model3DGroup>
    <DirectionalLight
      Color="White"
      Direction="-2,-3,-1" />
    <GeometryModel3D x:Name="pCube1">
      <GeometryModel3D.Geometry>
        <MeshGeometry3D Positions="1,1,1 1,1,0 0,1,0 0,1,1
1,1,1 1,1,1 1,0,1 1,0,0 1,0,0 1,1,0 1,1,1 1,1,1 0,1,1 0,0,1 0,0,1
1,0,1 1,1,1 0,1,0 1,1,0 1,0,0 1,0,0 0,0,0 0,1,0 0,0,0 1,0,0 1,0,1
1,0,1 0,0,1 0,0,0 0,0,0 0,0,1 0,1,1 0,1,1 0,1,0 0,0,0" Texture
Coordinates="1,1 0,1 0,0 0,0 1,0 1,1 0,1 1,1 1,0 1,0 0,0 0,1 0,1 1,1
1,0 1,0 0,0 0,1 0,1 1,1 1,0 1,0 0,0 0,1 0,1 1,1 1,0 1,0 0,0 0,1 0,1
1,1 1,0 1,0 0,0 0,1"/>
      </GeometryModel3D.Geometry>
    <GeometryModel3D.Material>
      <MaterialGroup>
        <DiffuseMaterial>
          <DiffuseMaterial.Brush>
            <ImageBrush ImageSource
="C:\Documents and Settings\HP_Administrateur\Mes documents\Mes
images\garfield.jpg" Opacity="1" />
          </DiffuseMaterial.Brush>
        </DiffuseMaterial>
      </MaterialGroup>
    </GeometryModel3D.Material>
  <GeometryModel3D.Transform>
    <Transform3DGroup>
      <RotateTransform3D>
        <RotateTransform3D.Rotation>
          <AxisAngleRotation3D x:Name
="rotate" Angle="90.000000" Axis="1.500000 1.000000 0.500000"/>
        </RotateTransform3D.Rotation>
      </RotateTransform3D>
    </Transform3DGroup>
  </GeometryModel3D.Transform>
</GeometryModel3D>
</Model3DGroup>
</ModelVisual3D.Content>
</ModelVisual3D>
</Viewport3D>
</Grid>
</Page>
```

Conclusion

La 3D en WPF est vraiment simple à mettre en œuvre, bien entendu, on est loin de ce qu'on peut faire avec DirectX 10 mais on a une bonne base pour développer des applications en 3D à partir de WPF et cela en XAML.

■ **Xavier Vanneste** - XVanneste@msn.com
<http://www.netfxfactory.org>
<http://blogs.developpeur.org/xvanneste>

Un lecteur MP3 avec WPF/E

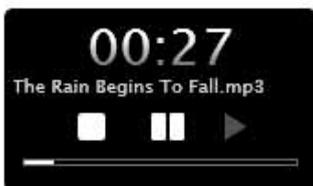
Encore au stade de CTP (Community Technology Preview) la technologie WPF/E (Windows Presentation Foundation /Everywhere) est la solution de Microsoft permettant de développer des applications riches et interactives sur le web. Cette technologie est directement mise en concurrence avec la technologie Flex d'Adobe.

Flex et WPF/E s'articulent sur les mêmes fondamentaux, un langage de description d'interface dérivé du XML et un langage de scripting. Alors que Flex se base sur le format MXML pour l'interface et Actionscript 3 pour ce qui est du langage de programmation, WPF/E s'appuie, quant à lui, sur un sous-ensemble de XAML (eXtensible Application Markup Language) en ce qui concerne la couche de présentation et JavaScript pour ce qui est du scripting.

Le plug-in WPF/E est compatible avec les plates-formes Windows et Mac. Les navigateurs supportés sont Internet Explorer, Safari, et le moteur Gecko qu'embarquent Mozilla et Firefox. WPF/E est orchestré par le langage JavaScript. C'est avec ce langage que vous pourrez animer et rendre interactives vos applications WPF/E. Des rumeurs courent sur le fait que les langages C# et VB.NET seront supportés par WPF/E, malheureusement aucune date officielle n'a encore été mentionné.

WPF/E en pratique

Au cours de cet article nous allons aborder et apprendre à mettre en place des notions telles que les Storyboards, les événements, les déclencheurs événementiels et enfin la manipulation de fichiers son ou vidéo, via un MediaElement. Pour mettre en pratique toutes ces notions nous allons développer une mini-application qui sera un lecteur MP3 en WPF/E.



Ce lecteur MP3 sera capable de lire les fichiers MP3 et WMA. Son interface sera entièrement décrite en XAML. Pour réaliser les interactions avec le XAML nous utiliserons le langage JavaScript qui est à ce jour le seul langage supporté

par WPF/E. Tous les documents et sources utilisées au cours de cet article sont disponibles à l'adresse suivante : <http://www.wpfstorm.com/medias/articles/Programmez-HSNET3.zip>

Pour débuter cet exercice nous allons mettre en place l'architecture et détailler chaque fichier du projet " Lecteur MP3 WPF/E ". Voici un schéma afin de mieux comprendre le fonctionnement.



Hoster du WPF/E avec le fichier HTML :

Le fichier HTML tel que vous le connaissez nous permet d'hoster, via des balises objet, le contrôle Activex WPF/E. Pour ce faire :

- Inclure le fichier " aghost.js " dans le HEAD de la page comme suit :

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

Ce fichier est téléchargeable via le SDK de WPF/E disponible à cette adresse : <http://msdn.microsoft.com/wpfe/> ou bien dans les exemples du site <http://www.wpfstorm.com/>. Ce fichier permet essentiellement d'insérer l'Activex WPF/E au sein de votre page HTML, mais aussi de gérer la compatibilité avec le navigateur client. C'est dans ce même fichier qu'il est possible de personnaliser le message d'erreur en cas de non installation du plug-in WPF/E sur la machine cliente.

- Inclure dans la page HTML et dans la balise BODY un élément DIV avec un ID spécifique. C'est dans ce DIV que va être inséré via JavaScript la balise OBJECT décrivant l'Activex WPF/E :

```
<div id="agControl1Host"></div>
```

- Inclure la portion de JavaScript qui décrit la balise OBJECT de l'Activex WPF/E tout de suite après le DIV renseigné ci-dessus.

```
- <script type="text/javascript">
new agHost(
  "agControl1Host", // hostElementID (The HTML element inside of which
  "agControl1", // The ID of the WPF/E ActiveX control to create.
  "400px", // The width of the control.
  "300px", // The height of the control.
  "#FFFFFF", // The background color of the control.
  null, // SourceElement (name of script tag containing XAML)
  "music.xml", // The uri of the source file that
  "false", // IsWindowless
  "30", // MaxFrameRate
  null // OnError handler.
);
var agControl = document.getElementById("agControl1");
</script>
```

La mise en place du fichier HTML et de la balise OBJECT étant terminée, nous allons pouvoir créer le fichier XAML.

La structuration d'interface avec le fichier XAML :

Le fichier XAML permet, tout comme en WPF, de décrire l'interface d'une fenêtre. Attention toutefois, la version XAML de WPF/E est un sous-ensemble de XAML pour WPF. En effet, certaines fonctionnalités

telles que la 3D, les ressources, la gestion des données etc. ont été supprimées. Nous savons d'ores et déjà que la 3D ne sera pas embarquée dans la version finale de WPF/E, mais nous pouvons imaginer que la version finale de XAML pour WPF/E sera une version plus avancée et plus aboutie que la version actuelle. Croisons les doigts !

Nous allons pouvoir à présent commencer à créer l'interface du lecteur MP3. Pour générer le XAML et donc l'interface, il est possible d'utiliser le logiciel Expression Blend (<http://www.microsoft.com/products/expression/en/Expression-Blend/default.aspx>) ou un utilitaire plus léger tel que XAMLPad, disponible avec l'installation du SDK Framework 3.0. Attention, ces éditeurs ne génèrent pas encore le XAML spécifique à WPF/E il faut donc être attentif aux balises que ses outils génèrent. Voici le fichier XAML commenté permettant de réaliser l'interface du lecteur MP3 :

```
<!-- Comme énoncé précédemment, le nœud racine d'un document WPF/E est un Canvas. Un Canvas est un conteneur qui permet de placer ses enfants en fonction de son propre coin supérieur gauche et ses axes X et Y. On pourra noter la déclaration des namespaces mais surtout l'association d'une fonction JavaScript à l'évènement de chargement du Canvas racine.-->
<Canvas
  xmlns="http://schemas.microsoft.com/client/2007"
  xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
  Loaded="javascript:onLoaded">
<!-- Création d'un MediaElement permettant de lire les fichiers Audios et Vidéos (MP3, WMA, WMV) La propriété Source permet de renseigner le fichier que l'on souhaite lire. Les chemins absolus et relatifs sont envisageables. -->
  <MediaElement x:Name="media" Source="C:\MonFichierMusical.mp3" />
<!-- Création d'un Rectangle noir arrondi en background -->
  <Canvas x:Name="background"><Rectangle Width="170" RadiusX="4.5" RadiusY="4.5" Height="100" Fill="#000000" /></Canvas>
<!-- Conteneur de type Canvas contenant la totalité des boutons du lecteur -->
  <Canvas Canvas.Top="12" Width="170" Height="20">
<!-- Création du bouton Stop avec un Rectangle et association de l'évènement MouseLeftButtonDown à la fonction JavaScript stop -->
    <Canvas x:Name="btnStop" Cursor="Hand" Opacity="1" Canvas.Top="45" Width="20" Height="20" Canvas.Left="40" MouseLeftButtonDown="javascript:stop">
      <Rectangle Width="15" Fill="White" Height="15" RadiusX="2" RadiusY="2" />
    </Canvas>
<!-- Création du bouton Pause avec deux Rectangles et association de l'évènement MouseLeftButtonDown à la fonction javascript pause-->
    <Canvas x:Name="btnPause" Cursor="Hand" Canvas.Top="45" Opacity="1" Width="20" Height="20" Canvas.Left="80" MouseLeftButtonDown="javascript:pause">
      <Rectangle Width="8" Height="15" RadiusX="1" RadiusY="1" Fill="White" />
      <Rectangle Width="8" Canvas.Left="10" Height="15" RadiusX="1" RadiusY="1" Fill="White" />
    </Canvas>
<!-- Création du bouton Lecture avec un Path représentant un triangle et association de l'évènement MouseLeftButtonDown à la fonction javascript play -->
    <Canvas x:Name="btnPlay" Canvas.Top="45" Opacity="0.3" Width="20" Height="20" Canvas.Left="120" MouseLeftButtonDown="javascript:
```

```
play">
  <Path Stretch="Fill" Fill="White" Width="12" Height="15" Data="M-34,111 L-34,169 8,140 z" />
</Canvas>
<!-- TextBlock, zone de texte contenant le temps en cours d'un fichier MP3, par défaut 00:00 -->
  <TextBlock x:Name="time" Text="00:00" Canvas.Left="45" Canvas.Top="12" FontSize="28">
<!-- Application d'un dégradé sur le texte du TextBlock -->
    <TextBlock.Foreground>
      <LinearGradientBrush EndPoint="0.4,0.4" StartPoint="0.4,1">
        <GradientStop Color="#000000" Offset="0"/>
        <GradientStop Color="#FFFFFF" Offset="1"/>
      </LinearGradientBrush>
    </TextBlock.Foreground>
  </TextBlock>
<!-- Conteneur contenant le message défilant du nom de la chanson -->
  <Canvas Canvas.Top="23" Canvas.Left="5" Height="20" Width="160">
<!-- Création d'une zone permettant de cacher tout débordement de ce conteneur via la méthode " Clip ", cela va être utile lorsque l'on va déplacer le titre de la chanson sur l'axe des X. En effet, par défaut le débordement d'un Canvas n'est pas " overflow hidden " c'est-à-dire que le débordement n'est pas caché. -->
    <Canvas.Clip><RectangleGeometry Rect="0,0,160,20" /></Canvas.Clip>
    <TextBlock x:Name="mp3play" Text="" FontSize="10" Foreground="#FFFFFF">
      <TextBlock.Triggers>
<!-- Les Triggers sont des déclencheurs -->
        <EventTrigger RoutedEvent="Canvas.Loaded">
<!-- Les EventTrigger sont des déclencheurs évènementiels, on peut ainsi associer un évènement à un déclencheur, on l'occurrence ici, le " Load " du Canvas -->
          <BeginStoryboard>
            <Storyboard x:Name="aMp3play">
<!-- Création d'une animation via un StoryBoard permettant de déplacer le titre de la chanson en train de jouer sur l'axe des X de droite à gauche de l'interface du lecteur MP3. Dans le TargetProperty on renseigne la propriété que l'on souhaite modifier et dans le TargetName le nom de l'élément. Il ne reste plus qu'à définir le From et To pour animer la propriété et le tour et joué -->
              <DoubleAnimation x:Name="daMp3playLeft" RepeatBehavior="Forever" Storyboard.TargetName="mp3play" Storyboard.TargetProperty="(Canvas.Left)" SpeedRatio="1.5" From="160" To="0" Duration="0:0:15" FillBehavior="Stop" />
              <DoubleAnimation x:Name="daMp3playOpacity" RepeatBehavior="Forever" Storyboard.TargetName="mp3play" Storyboard.TargetProperty="Opacity" SpeedRatio="1.5" From="0" To="1" Duration="0:0:15" FillBehavior="Stop" />
            </Storyboard>
          </BeginStoryboard>
        </EventTrigger>
      </TextBlock.Triggers>
    </TextBlock>
  </Canvas>
<!--Création du Rectangle de fond de la barre du temps -->
```

```

<Rectangle Canvas.Left="10" Canvas.Top="70" Fill="#000000"
Width="150" Height="4" StrokeThickness="1" Stroke="#666666" />
<!--Création du Rectangle de la barre du temps et de l'animation -->
<Rectangle x:Name="bartime" Canvas.Top="71" Canvas.Left="11"
Width="0" Height="2" Fill="#FFFFFF">
<Rectangle.Triggers>
<EventTrigger RoutedEvent="Canvas.Loaded">
<BeginStoryboard>
<Storyboard x:Name="abartime">
<DoubleAnimation x:Name="dabartime" Storyboard.Target
Name="bartime" Storyboard.TargetProperty="Width" Duration="0:0:1"
FillBehavior="Stop" />
</Storyboard>
</BeginStoryboard>
</EventTrigger>
</Rectangle.Triggers>
</Rectangle>

</Canvas>

</Canvas>

```

Nous avons à présent terminé la partie interface et la partie design du lecteur MP3, nous allons donc pouvoir passer à la partie programmation de l'application.

La couche événementielle avec le fichier JavaScript :

C'est en effet dans le fichier JavaScript que va être gérée la couche événementielle. C'est ici que vont être articulées toutes les interactions de l'utilisateur avec l'ActiveX WPF/E ou plus particulièrement le fichier XAML que nous venons de créer. Voici les fonctions JavaScript que nous avons appelées dans le XAML : Play, Stop, Pause, OnLoaded et voici la fonction JavaScript qui va nous servir à " dessiner " l'interface : Drawbar.

La fonction OnLoaded :

Cette fonction est exécutée lorsque le nœud Canvas a été chargé en intégralité.

```

onLoaded = function(sender, eventArgs)
{
    root = sender; //déclaration d'une variable global contenant le nœud
    racine de l'application.
    timerSeconds = window.setInterval("drawbar()",500); //on créé un
    timer qui va nous permettre de designer la barre de progression du
    lecteur.
    root.FindName("mp3play").Text = root.findName("media").Source;
    //on insert le text dans le TexBlock mp3play
    root.FindName("daMp3playLeft").SetValue("To", "-" + root.FindName
    ("mp3play").ActualWidth); //on renseigne la propriété "To" de la
    Double Animation "daMp3playLeft". Qui va nous permettre d'animer
    le texte du mp3 tel que pourrait le faire la balise <marquee> en
    HTML.
}

```

La fonction Play :

Cette fonction permet de lire le fichier MP3 renseigné dans le Source du MediaElement. Dans cette fonction nous nous occupons aussi d'afficher les bons curseurs et les bons boutons dans l'interface du lecteur.



Une application WPF / E sous Safari



Une application WPF / E sous FireFox (version MacOS X)

```

play = function(sender)
{
    root.FindName("media").play();
    root.FindName("btnPause").SetValue("Opacity", 1);
    root.FindName("btnPause").SetValue("Cursor", "Hand");
    root.FindName("btnPlay").SetValue("Opacity", 0.3);
    root.FindName("btnPlay").SetValue("Cursor", "Default");
    root.FindName("btnStop").SetValue("Opacity", 1);
    root.FindName("btnStop").SetValue("Cursor", "Hand");
    root.FindName("mp3play").Text = root.findName("media").Source;
    root.FindName("aMp3play").Begin();
}

```

La fonction Pause :

Cette fonction permet de mettre en pause la lecture courante du MediaElement. Dans cette fonction, nous nous occupons aussi d'afficher les bons curseurs et les bons boutons dans l'interface du lecteur.

```

pause = function(sender)
{

```

```

root.FindName("media").pause();
root.FindName("btnPause").SetValue("Opacity", 0.3);
root.FindName("btnPause").SetValue("Cursor", "Default");
root.FindName("btnPlay").SetValue("Opacity", 1);
root.FindName("btnPlay").SetValue("Cursor", "Hand");
}

```

La fonction Stop :

Cette fonction permet de stopper la lecture courante du MediaElement. Dans cette fonction nous nous occupons aussi d'afficher les bons curseurs et les bons boutons dans l'interface du lecteur.

```

stop = function(sender)
{
    root.FindName("media").stop();
    root.FindName("btnPause").SetValue("Opacity", 0.3);
    root.FindName("btnPause").SetValue("Cursor", "Default");
    root.FindName("btnPlay").SetValue("Opacity", 1);
    root.FindName("btnPlay").SetValue("Cursor", "Hand");
    root.FindName("btnStop").SetValue("Opacity", 0.3);
    root.FindName("btnStop").SetValue("Cursor", "Default");
    root.FindName("mp3play").SetValue("Text", "");
    root.FindName("mp3play").SetValue("Text", "");
    root.FindName("aMp3play").Stop();
}

```

La fonction drawbar :

Cette fonction permet de faire progresser la barre du temps en fonction du temps.

```

drawbar = function()
{
    // Récupération du Timespan
    var timespan = root.findName("media").position;
    // Modification des valeurs From et To de l'animation
    root.FindName("dabartime").SetValue("From", barcoord);
    //le naturalDuration renvoie la durée en seconde du fichier MP3 renseigné
    dans le Source
    root.FindName("dabartime").SetValue("To", timespan.seconds * 148 /
root.findName("media").naturalDuration.seconds);
    root.FindName("abartime").Begin();
    barcoord = timespan.seconds * 148 / root.findName("media").natural
Duration.seconds;
    var datetime = new Date(0, 0, 0, 0, 0, timespan.seconds)
    var hours = datetime.getHours();
    var minutes = datetime.getMinutes();
    var seconds = datetime.getSeconds();
}

```

```

if(seconds < 10){ seconds = "0" + seconds; }
if(minutes < 10){ minutes = "0" + minutes; }
var durationString;
if(hours > 0){
// Formatage du texte du TextBox 0:0:0
durationString = hours.toString() + ":" + minutes + ":" + seconds;
root.findName("time").text = durationString.toString();
} else {
durationString = minutes + ":" + seconds;
root.findName("time").text = durationString.toString();
}
}
}

```

WPF/E en conclusion

L'exercice étant terminé vous êtes à présent aptes à utiliser les fondamentaux de la technologie WPF/E. Les déclencheurs (Triggers), les déclencheurs événementiels et les animations n'ont plus aucune surprise pour vous. Comme vous l'avez sans doute remarqué cette technologie n'est pas encore aboutie tant au niveau XAML qu'au niveau plate-forme. Le support de C# et VB.NET serait une réelle valeur ajoutée quant au développement de ce type d'application.

Vous souhaitez aller plus loin avec WPF/E ?

Retrouvez l'ensemble des informations dédiées à WPF/E sur le site officiel de la MSDN : <http://msdn.microsoft.com/wpfe>

Vous pourrez retrouver aussi sur le site <http://www.wpfstorm.com/> divers cas d'utilisation typique de WPF/E avec de multiples exemples.

■ **Guillaume André** - Intégrateur.Net et Designer. **Wygwam**, Expertise technique .Net: conseil, formation, développement et solutions. <http://www.wygwam.com/> - <http://www.wpfstorm.com/>



L'avis de Thomas Lebrun

(MSP – mcad.net)

"Travailler avec le Framework .NET 3.0 est un vrai bonheur: les gains de temps, en terme de productivité, se font considérablement ressentir notamment sur les gros projets. L'apprentissage est certes un peu complexe au début, mais une fois les techniques bien maîtrisées, les différences sont là. Certaines briques du Framework .NET 3.0 sont utilisées plus souvent que les autres ce qui permet de bien se rendre compte des impacts/problèmes possibles (dans le cas de migrations par exemple) mais également de bien appréhender les différentes fonctionnalités offertes. Pour conclure, je dirais que si vous voulez faire mieux qu'avant et plus vite, alors n'hésitez pas à regarder ce que le.NET 3.0 peut vous apporter."

L'INFORMATION PERMANENTE

Programmez!

LE MAGAZINE DU DÉVELOPPEMENT

✓ **Abonnez-vous**

au format NUMERIQUE (PDF)

Tarif unique pour **LE MONDE ENTIER**

35 €

(abonnement exclusivement en ligne
www.programmez.com)

✓ **3 NUMEROS GRATUITS**

Faites des économies, abonnez-vous !

Exemple : en France, vous gagnez 20 euros, soit plus de 3 numéros !

✓ **ETUDIANTS** Tarif spécial **39€**

(France métropolitaine exclusivement)

COUPON D'ABONNEMENT page 4 de ce numéro ou www.programmez.com

ASP.NET AJAX Extensions : l'Ajax de Microsoft

Depuis le début du web, il y a eu beaucoup de changements. Les premières pages web, étaient des pages institutionnelles qui permettaient d'afficher et de donner des informations au client. C'était un nouveau moyen d'information. Une équipe rédactionnelle faisait des pages statiques qu'elle mettait en ligne au format HTML. Puis les media ont commencé à investir peu à peu les pages web, avec, au début, des images puis des vidéos.

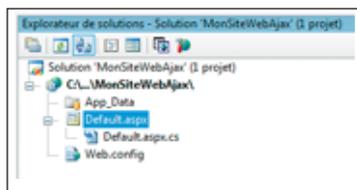
Les pages ont aussi rigoureusement changé en passant d'un stade texte simple à du texte mis en forme et en couleur. A cela est venu s'ajouter le dynamisme des pages. C'est-à-dire que le serveur a vite eu la possibilité de mettre en place dynamiquement les pages par rapport à des données stockées dans une base. D'abord en utilisant des CGI, puis des langages de script comme ASP, et enfin, en se basant sur le Framework .Net en ASP.Net. Le souci étant que la communication et les fondements même du HTTP empêchent le serveur d'envoyer directement des informations au client si ce n'est pas celui-ci qui en fait la demande. Le résultat n'est pas sans conséquences, puisque le client est obligé de demander des données à jour. De plus, jusqu'à présent, lorsqu'une partie de la page nécessitait un rafraîchissement suite à une mise à jour de données côté serveur, il fallait rafraîchir toute la page. Cela n'est plus tout à fait vrai avec AJAX. Il permet de faire des requêtes au serveur et d'obtenir les informations sans avoir à rafraîchir la page. Bien entendu, c'est toujours le client qui est l'initiateur de la communication, mais le principe permet, avec l'aide d'un timer par exemple, de rafraîchir à intervalles réguliers les informations contenues sur le serveur.

Installation d'AJAX Extensions

Pour développer avec les Ajax Extensions de Microsoft il faut, en plus d'un serveur web (IIS ou Cassini de Visual Studio 2005) et d'un IDE (Visual Studio 2005 par exemple), télécharger le runtime et le contrôle toolkit sur le site <http://ajax.asp.net> et <http://www.codeplex.com>. Sur le site <http://ajax.asp.net> vous trouverez en plus la documentation.

Lorsque vous installez le runtime, il ajoute un Template pour créer des sites qui supportent les Ajax extensions. Regardons ce que le Template comporte : Je crée un nouveau projet site web dans Visual Studio 2005. Je choisis le système de fichier pour le développement, mais j'aurais pu prendre HTTP et utiliser IIS.

Lorsque mon site est créé, il n'y a rien d'exceptionnel, à première vue.



Mais en y regardant de plus près, je me rends compte que dans ma page default.aspx j'ai un control qui était inconnu en ASP.NET 2.0 :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
```

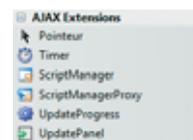
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml">  
<head runat="server">  
  <title>Untitled Page</title>  
</head>
```

```
<body>  
  <form id="form1" runat="server">  
    <asp:ScriptManager ID="ScriptManager1" runat="server" />  
    <div>  
    </div>  
  </form>  
</body>  
</html>
```

Cet ASP :ScriptManager vient du runtime Ajax. Dans le fichier Web.Config on a la configuration et la référence au runtime Ajax :

```
<pages>  
  <controls>  
    <add tagPrefix="asp" namespace="System.Web.UI" assembly="System.Web.Extensions, Version=1.0.61025.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35"/>  
  </controls>  
</pages>
```

De plus, dans la barre d'outils, on se retrouve avec des contrôles spécifiques pour Ajax (ci-contre).



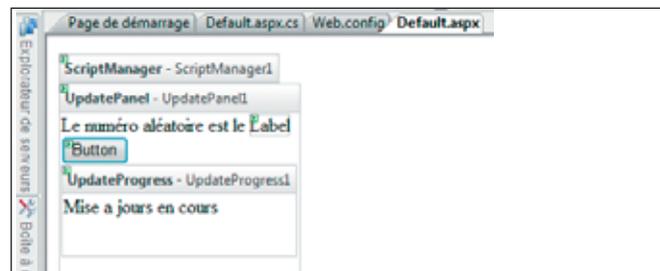
Ma première application.

Pour l'instant, nous allons nous contenter d'utiliser juste le runtime Ajax sans les contrôles toolkit. On va donc utiliser l'update panel et l'update progress.

L'update Panel

L'update panel est un panel qui permet de faire de la mise à jour dynamique suivant un événement donné. L'événement en question peut être un click sur un bouton, ou un timer. Pour notre exemple, nous allons mettre en place un Update panel avec un texte et un nombre aléatoire généré côté SERVEUR. On va aussi mettre un update progress et on verra à quel point il est simple de développer avec ce framework.

Commençons par mettre sur notre feuille un Update Panel avec l'update progress inclus dedans. Dans l'update progress, mettons en place un texte d'attente. Voilà à quoi cela ressemble :



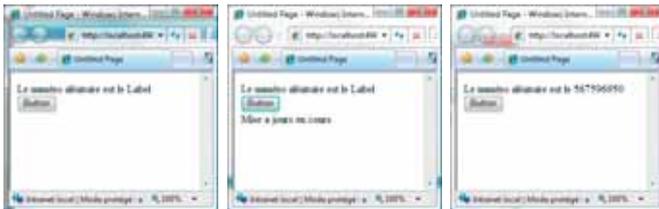
En ASP.Net pour mettre du code sur l'événement OnClick du bouton je double cliquerais dessus. En Ajax Extension c'est identique, on double clique sur le bouton et on rentre ceci dans le code c# :

```
protected void Button1_Click(object sender, EventArgs e)
{
    Label1.Text = new Random().Next().ToString();
}
```

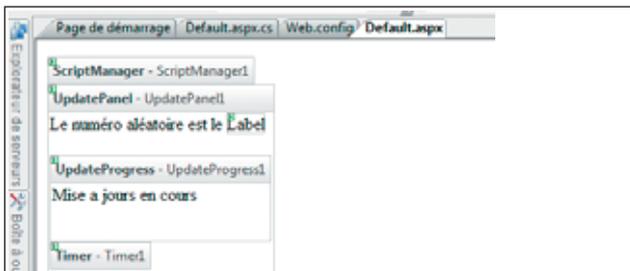
Si on exécute une première fois, on se rend compte qu'il n'y a pas de refresh de la page et que le chiffre aléatoire apparaît bien dans le label. Mais on ne voit pas l'update progress. Ajoutons alors le code suivant :

```
protected void Button1_Click(object sender, EventArgs e)
{
    System.Threading.Thread.Sleep(10000);
    Label1.Text = new Random().Next().ToString();
}
```

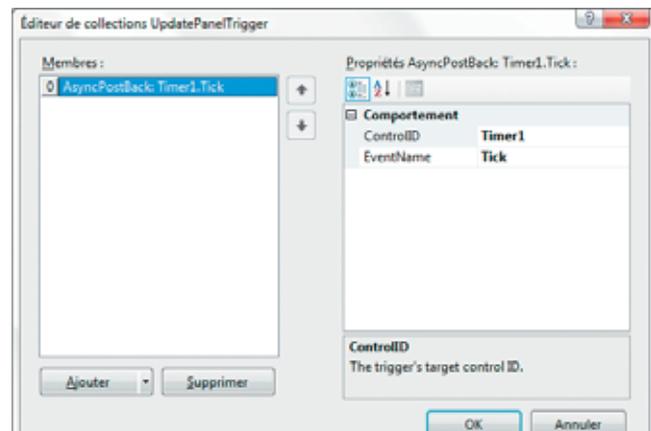
Cela ajoute une pause de 10 secondes. Maintenant qu'on exécute :



Tout est mis à jour sans rafraîchissement de la page. Continuons en ajoutant le timer Ajax avec un intervalle de 1000. Je ne veux plus mettre à jour quand je clique sur le bouton, mais quand l'événement Tick de mon timer a lieu. Je retire donc le bouton de mise à jour. Je place un timer sur ma page comme ceci :



Je vais dans les propriétés de mon updatepanel pour ajouter ceci à la collection de triggers :



J'enlève le code qui se trouvait dans le click du bouton car il n'y a plus de bouton. Je mets le code dans le form load :

```
protected void Page_Load(object sender, EventArgs e)
{
    Label1.Text = new Random().Next().ToString();
}
```

Quand je lance mon application la page est bien mise à jour (et non rafraîchie) toutes les secondes. Ceci est déjà impressionnant, imaginez développer la même chose en JavaScript en allant chercher les informations sur le net.

Quoi de plus

Evidemment, les trois contrôles fournis avec le runtime des Ajax Extensions sont bien, mais c'est tout de même un peu limité. Ajoutons donc les contrôles toolkit à notre environnement de développement préféré et voilà ce qu'on obtient dans la barre d'outils (ci-contre). 41 contrôles ont été rajoutés à ma barre d'outils. On ne pourra pas tous les passer en revue, mais quand on sait que Microsoft travaille sur d'autres contrôles, cela laisse présager qu'on n'aura bientôt plus rien à faire.

Nous allons nous intéresser à deux contrôles en particulier :

- L'AutoCompleteExtender qui permet de faire de l'auto complétion dans une textbox
- Le CascadingDropDown qui permet de faire des choix par catégorie.

Auto Complétion

L'auto complétion est une fonctionnalité qui aide beaucoup les utilisateurs et les visiteurs d'un site web. Prenons l'exemple de Google <http://www.google.fr/webhp?hl=en&complete=1> qui donne ceci lors d'une recherche :



On va mettre en place quelque chose de similaire. J'ajoute une page à mon projet. Dans cette page je mets un scriptmanager qui activera la prise en charge des AjaxExtensions sur ma page. Ensuite je mets une textbox qui sera étendue par l'AutoCompleteExtender. Ainsi j'ajoute l'AutoCompleteExtender et j'initialise la propriété TargetControlID avec le nom du contrôle TextBox que je veux étendre. Dans ma textbox, j'ai les nouvelles propriétés suivantes qui apparaissent :

Développement Web



Les deux propriétés les plus importantes sont ServiceMethod et ServicePath. En effet, cet extender se base sur un Webservice. J'ajoute donc un Webservice à mon projet. Pour mon projet, voici les informations que j'ai entrées :

Extenders	
AutoCompleteExtender1 (AutoComplete)	
BehaviorID	AutoCompleteExtender1
CompletionInterval	1000
CompletionListElementID	
CompletionSetCount	9
EnableCaching	True
MinimumPrefixLength	2
ServiceMethod	GetCompletionList
ServicePath	~/AutoCompletion.aspx

Ici, je lui ai indiqué que j'appelais la méthode du web service AutoCompletion.aspx, je lui ai spécifié que je voulais 9 enregistrements maximum et qu'il devait commencer à interroger le web service après deux caractères tapés dans la textbox. Dans le code ASP.Net on retrouve ceci :

```
<form id="form1" runat="server">
  <div>
    <asp:ScriptManager ID="ScriptManager1" runat="server">
    </asp:ScriptManager>
    <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
    <cc1:AutoCompleteExtender ID="AutoCompleteExtender1" runat="server"
      ServiceMethod="GetCompletionList" ServicePath="~/AutoCompletion.aspx"
      TargetControlID="TextBox1" CompletionSetCount="9" CompletionInterval="1000" EnableCaching="True" MinimumPrefixLength="2">
    </cc1:AutoCompleteExtender>
  </div>
</form>
```

Pour mon webservice, j'ai tout d'abord ajouté l'attribut ScriptService à ma classe.

```
[WebService(Namespace = "http://tempuri.org/")]
[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
[System.Web.Script.Services.ScriptService]
public class AutoCompletion : System.Web.Services.WebService {
```

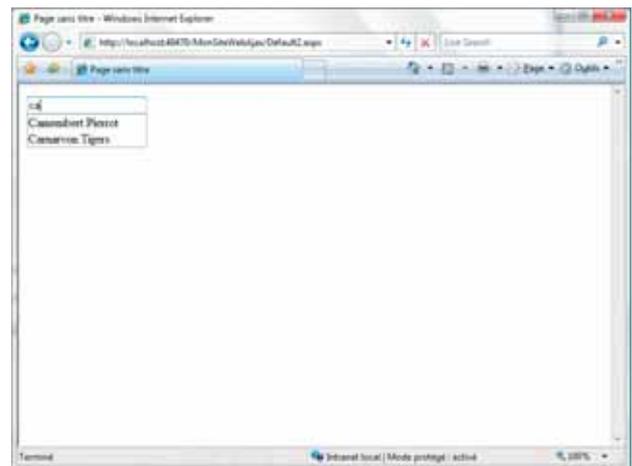
puis j'ai créé ma méthode en interrogeant la base de données Northwind pour récupérer les produits qui commencent par ce que j'ai tapé dans ma TextBox :

```
[WebMethod]
[System.Web.Script.Services.ScriptMethod]
```

```
public string[] GetCompletionList(string prefixText, int count)
{
    List<string> lstReturn=new List<string>();
    SqlConnection cnData = new SqlConnection(@"Data Source=. \
SQLEXPRESS;AttachDbFilename='|DataDirectory|NORTHWND.MDF';I
ntegrated Security=True;User Instance=True");
    cnData.Open();
    SqlCommand cmdData=new SqlCommand("select top " + count.To
String() + " productname from products where productname like '" +
prefixText + "%'",cnData);
    SqlDataReader drData=cmdData.ExecuteReader();
    while(drData.Read())
    {
        lstReturn.Add(drData[0].ToString());
    }

    return lstReturn.ToArray();
}
```

Quand j'exécute ma page, voici le résultat :



J'ai bien l'auto complétion qui a été mise en place.

Allons un peu plus loin avec les cascading drop down. Voici la table avec des données que l'on a pour le projet (Fig. A).

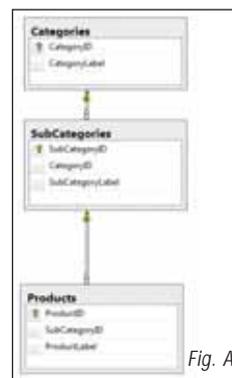


Fig. A

Je veux avoir 3 dropdown lists, une qui sélectionnera la catégorie, ce qui activera la dropdown de la sous-catégorie pré-remplie par rapport à la catégorie sélectionnée. De même pour les produits et sous-catégories. Comment mettre cela en place ?

Je crée une nouvelle page sur laquelle je mets un script manage.

Pour cette page je vais avoir 3 dropdown avec 3 cascadingDropDown. Je vais donc avoir une page de la forme suivante (Fig. B).

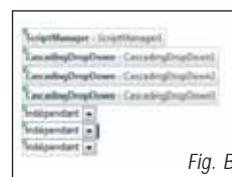


Fig. B

Pour la cascadingDropDown1 je vais initialiser la valeur de la propriété TargetControlID avec le nom de ma première dropdown. De nouvelles propriétés sont donc apparues dans ma dropdownlist. Je ne mettrais rien

dans ParentControlID car cette dropdown doit se remplir dès le départ. La propriété category est obligatoire car c'est elle qui permettra de savoir sur quelle dropdown on se trouve.

Voici la fenêtre de propriété de ma dropdownlist :



Comme l'autocomplete, elle pointe sur un webservice. La propriété Loading Text est faite pour mettre un texte d'attente lors du chargement des données. La propriété PromptText sera la première valeur de ma dropdown. Pour le web service, je pars sur le même principe que pour mon autocomplete. Je rajoute l'attribut [ScriptService] à ma classe. La signature de la méthode est la suivante :

```
public AjaxControlToolkit.CascadingDropDownNameValue[]
GetCategory(string knownCategoryValues, string category)
```

La valeur de retour est un tableau de CascadingDropDownNameValue qui permet de renvoyer la valeur et le contenu d'une des options de la dropdown. Les paramètres sont : en premier la valeur de la dropdown parent sélectionnée, pour notre première dropdown ce n'est pas important. La deuxième valeur est la category assignée à la dropdown qui vient d'être actionnée, c'est un peu comme le sender. Voilà donc la première fonction :

```
[WebMethod]
public AjaxControlToolkit.CascadingDropDownNameValue[] Get
Category(string knownCategoryValues, string category)
{
    List<AjaxControlToolkit.CascadingDropDownNameValue>
acddReturn = new List<AjaxControlToolkit.CascadingDropDownName
Value>();
    SqlConnection cnData = new SqlConnection(@"Data Source=. \
SQLEXPRESS;AttachDbFilename='|DataDirectory|ProductDB.MDF';Inte
grated Security=True;User Instance=True");
    cnData.Open();
    SqlCommand cmdData = new SqlCommand("select CategoryID,
CategoryLabel from categories", cnData);
    SqlDataReader drData = cmdData.ExecuteReader();
    while (drData.Read())
    {
        acddReturn.Add(new AjaxControlToolkit.CascadingDropDown
NameValue(drData["CategoryLabel"].ToString(), drData["CategoryID"].
ToString()));
    }

    return acddReturn.ToArray();
}
```

Pour la deuxième, il va falloir parser la valeur KnownCategoryValues car voilà ce qui s'est passé :

```
de[] GetSubCategory(string knownCategoryValues, string category)
{
    List<AjaxControlToolkit.CascadingDropDownNameValue> acdd
Return = new List<AjaxControlToolkit.CascadingDropDownNameValue>();
    SqlConnection cnData = new SqlConnection(@"Data Source=. \
SQLEXPRESS;AttachDbFilename='|DataDirectory|ProductDB.MDF';Inte
grated Security=True;User Instance=True");
    cnData.Open();
    SqlCommand cmdData = new SqlCommand("select SubCategoryID,
SubCategoryLabel from SubCategories where CategoryID=" + strDic
["Categorie"], cnData);
    SqlDataReader drData = cmdData.ExecuteReader();
    while (drData.Read())
    {
        acddReturn.Add(new AjaxControlToolkit.CascadingDropDown
NameValue(drData["SubCategoryLabel"].ToString(), drData["Sub
CategoryID"].ToString()));
    }

    return acddReturn.ToArray();
}
```

Pour cela, on utilise la fonction static ParseKnownCategoryValuesString de la classe CascadingDropDown. Cela crée un stringdictionary. Voici le code de la deuxième fonction :

```
[WebMethod]
public AjaxControlToolkit.CascadingDropDownNameValue[] GetSub
Category(string knownCategoryValues, string category)
{
    StringDictionary strDic= AjaxControlToolkit.CascadingDropDown.
ParseKnownCategoryValuesString(knownCategoryValues);
    List<AjaxControlToolkit.CascadingDropDownNameValue> acdd
Return = new List<AjaxControlToolkit.CascadingDropDownNameValue>();
    SqlConnection cnData = new SqlConnection(@"Data Source
=. \SQLEXPRESS;AttachDbFilename='|DataDirectory|ProductDB.MDF';In
tegrated Security=True;User Instance=True");
    cnData.Open();
    SqlCommand cmdData = new SqlCommand("select SubCategoryID,
SubCategoryLabel from SubCategories where CategoryID=" + strDic
["Categorie"], cnData);
    SqlDataReader drData = cmdData.ExecuteReader();
    while (drData.Read())
    {
        acddReturn.Add(new AjaxControlToolkit.CascadingDropDown
NameValue(drData["SubCategoryLabel"].ToString(), drData["Sub
CategoryID"].ToString()));
    }

    return acddReturn.ToArray();
}
```

Evidement sur la deuxième drop down, on n'oubliera pas de spécifier le parent.

Pour la troisième, c'est assez simple puisqu'elle est, à peu de chose près, identique à la deuxième. Voilà donc notre application en marche :



Cela peut être très utile si on choisit un pays, puis une région et une ville. Il peut y avoir de multiples utilisations de cette fonctionnalité.

Conclusion

Quand on voit ce que les Ajax Extensions et surtout les Ajax Control tool-kits sont capables de faire actuellement, on se dit que le monde HTML va sûrement changer dans les années à venir. Le but d'Atlas est de mettre en place un système web identique à la réaction qu'on pourrait attendre d'une application Windows. C'est-à-dire être sur une page HTML et ne pas avoir de rafraîchissement à chaque fois qu'on fait une action.

■ **Xavier Vanneste** - XVanneste@msn.com
<http://www.netfxfactory.org>
<http://blogs.developpeur.org/xvanneste>

OpenSearch : affiner les recherches

OpenSearch est une spécification créée dans le but de fournir un format de fichier commun pour les services de recherche sur internet. Cette spécification se décompose en deux sous-ensembles. Le premier est constitué d'un fichier de description du service de recherche qui permet à un client d'obtenir des informations sur le service - comme son nom, sa description - et la manière d'effectuer une recherche ; le second spécifie le format de fichier pour les réponses. Comme nous allons le voir, ces fichiers se basent sur un standard très répandu sur le net : le format XML.

Le format OpenSearch est principalement utilisé par les navigateurs récents qui permettent de lancer une recherche directement dans leur interface.

Le fichier de description

Le fichier de description permet à l'utilisateur de mieux connaître le service. Il s'agit d'un fichier XML qui commence par le nœud racine OpenSearchDescription, le namespace de la spécification est :

" <http://a9.com/-/spec/opensearch/1.1/>"

```
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
```

Plusieurs éléments obligatoires sont à renseigner, le nom court du service que l'on renseigne avec la balise ShortName puis la description via la balise Description, et enfin l'adresse de la réponse avec l'attribut template de la balise Url (son attribut type est lui aussi obligatoire, il permet d'indiquer quel format de réponse le service utilise).

Le fichier de description minimum est donc le suivant :

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription xmlns="http://a9.com/-/spec/opensearch/1.1/">
  <ShortName>Web Search</ShortName>
  <Description>Use Example.com to search the Web.</Description>
  <Url type="application/rss+xml"
    template="http://example.com/?q={searchTerms}&format=rss"/>
</OpenSearchDescription>
```

Le contenu de la balise shortName doit contenir au maximum 16 caractères, et si vous souhaitez renseigner un nom plus complet il vous faudra en plus utiliser la balise LongName. Pour décrire en détail votre service il vous faudra utiliser la balise LongDescription.

La balise Url est une des balises les plus importantes, en effet son attribut template contient l'adresse du résultat que l'on recherche. Pour renseigner les paramètres de la recherche il faut remplacer les expressions entre accolades par ce que vous recherchez. Par exemple, si l'on fait une recherche sur l'expression " voiture " avec le service décrit plus haut la réponse sera alors disponible à cette adresse <http://example.com/?q=voiture&format=rss>. Certains des paramètres peuvent être optionnels, on les reconnaît car le dernier caractère est un point d'interrogation, par exemple le paramètre startIndex est optionnel dans l'exemple ci-dessous.

```
<Url type="application/rss+xml"
  template="http://example.com/?q={searchTerms}&start={startIndex}&format=rss"/>
```

Voici les paramètres disponibles avec OpenSearch :

Nom du paramètre	Description	Restriction	Valeur par défaut
{searchTerms}	Expression à rechercher.	Doit être url-encodé	obligatoire
{count?}	Nombre de résultat par page	Entier positif	valeur de l'attribut indexOffset de la balise Url correspondante 1 si nulle
{startIndex?}	Index de la première page de recherche	Entier	valeur de l'attribut pageOffset de la balise Url correspondante 1 si nulle
{language?}	Langue des résultats	Doit être conforme * à la RFC 3066	
{inputEncoding?}	Jeu de caractère utilisé par la requête	Doit être conforme UTF-8 aux spécifications IANA Character Set Assignments	
{outputEncoding?}	Jeu de caractère utilisé pour la réponse	Doit être conforme UTF-8 aux spécifications IANA Character Set Assignments	

Vous pouvez définir d'autres paramètres grâce à l'extensibilité des namespaces XML, l'extension referrer permet par exemple d'indiquer au service de recherche quel est le logiciel client. Internet Explorer 7 et Firefox supportent cette extension.

```
<Url type="application/atom+xml"
  xmlns:referrer=http://a9.com/-/opensearch/extensions/referrer/1.0/
  template="http://example.com/{searchTerms}?src={referrer:source?}"/>
```

L'attribut type de la balise url définit le format de la réponse, par défaut OpenSearch en propose trois : RSS, Atom et XHTML. Les attributs à utiliser suivant le type de résultat que vous fournissez sont listés ci-dessous, vous pouvez, bien sûr, fournir plusieurs types de réponses dans un même document de description OpenSearch.

```
<Url type="application/atom+xml"
  template="http://example.com/?q={searchTerms}&pw={startIndex}&format=atom"/>
<Url type="application/rss+xml"
  template="http://example.com/?q={searchTerms}&pw={startIndex}&format=rss"/>
```

```
<Url type="text/html"
  template="http://example.com/?q={searchTerms}&pw={start
Page?}"/>
```

Les formats de réponses OpenSearch

Au niveau du format des réponses, OpenSearch n'a pas voulu réinventer la roue, il existe déjà un format retournant une liste d'éléments. Grâce à quelques ajouts et un namespace dédié, les formats RSS et Atom permettent de retourner des réponses pour OpenSearch. La version 1.1 de OpenSearch va plus loin en permettant de retourner du XHTML en guise de réponse. Pour RSS et Atom, il faut utiliser le namespace XML "<http://a9.com/-/spec/opensearch/1.1/>", vous pouvez alors ajouter les quatre balises OpenSearch :

- totalResults : le nombre total de résultats de la recherche ;
- startIndex : l'index du premier résultat de la réponse courante ;
- itemsPerPage : le nombre de résultats par Page ;
- query : informations sur la recherche en cours ou des recherches relatives à la recherche en cours.

```
<?xml version="1.0" encoding="UTF-8"?>
<rss version="2.0"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:atom="http://www.w3.org/2005/Atom">
  <channel>
    <title>Example.com Search: New York history</title>
    <link>http://example.com/New+York+history</link>
    <description>Search results for "New York history" at Example
.com</description>
    <opensearch:totalResults>4230000</opensearch:totalResults>
    <opensearch:startIndex>21</opensearch:startIndex>
    <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
    <atom:link rel="search" type="application/opensearchdescription
+xml"
      href="http://example.com/opensearchdescription.xml"/>
    <opensearch:Query role="request" searchTerms="New York History"
startPage="1" />
    <item>
      <title>New York History</title>
      <link>http://www.columbia.edu/cu/lweb/eguids/amerihist/nyc.
html</link>
      <description>
        ... Harlem.NYC - A virtual tour and information on
businesses ... with historic photos of Columbia's own New York
neighborhood ... Internet Resources for the City's History. ...
      </description>
    </item>
  </channel>
</rss>
```

Aucune des balises n'est obligatoire, elles sont par contre fortement conseillées. La balise Query peut être utilisée plusieurs fois, vous devez spécifier son attribut roles ainsi que searchTerms. Ces attributs définissent le rôle et les termes de la recherche relative. Il est conseillé d'inclure cette balise au moins une fois pour connaître les termes de la recherche en cours, il faut pour cela utiliser le rôle request. Voici la liste des rôles disponibles.

Attribut	Description
request	Représente la recherche qui a conduit aux résultats courants.
Example	Représente une recherche d'exemple pour illustrer le service de recherche.
related	Représente une recherche similaire, complémentaire.
Correction	Représente une recherche qui améliore la recherche active, comme par exemple une faute d'orthographe, ...
subset	Représente une recherche qui diminue le nombre de résultats.
superset	Représente une recherche qui élargit le nombre de résultats.

N'oubliez pas que vous êtes dans un document RSS/Atom et que le namespace XML par défaut n'est peut être pas celui d'OpenSearch, il vous faudra alors le spécifier pour chaque balise.

L'extension Relevance permet d'indiquer le score d'un résultat pour une recherche donnée. Cette extension est définie dans le namespace "<http://a9.com/-/opensearch/extensions/relevance/1.0/>", la balise score de cette extension doit contenir un nombre entre 0 et 1 où 1 est la plus forte note.

Voici un exemple de réponse ATOM utilisant l'extension Relevance :

```
<feed xmlns="http://www.w3.org/2005/Atom"
  xmlns:opensearch="http://a9.com/-/spec/opensearch/1.1/"
  xmlns:relevance="http://a9.com/-/opensearch/extensions/relevance
/1.0/">
  <!-- ... -->
  <opensearch:totalResults>4230000</opensearch:totalResults>
  <opensearch:startIndex>21</opensearch:startIndex>
  <opensearch:itemsPerPage>10</opensearch:itemsPerPage>
  <opensearch:Query role="request" searchTerms="New York History"
startPage="1" />
  <entry>
    <title>New York History</title>
    <link href="http://www.columbia.edu/cu/lweb/eguids/amerihist/
nyc.html"/>
    <id>urn:uuid:1225c695-cfb8-4ebb-aaaa-80da344efa6a</id>
    <updated>2003-12-13T18:30:02Z</updated>
    <content type="text">
      ... Harlem.NYC - A virtual tour and information on
businesses ... with historic photos of Columbia's own New York
neighborhood ... Internet Resources for the City's History. ...
    </content>
    <relevance:score>0.95</relevance:score>
  </entry>
  <!-- ... -->
</feed>
```

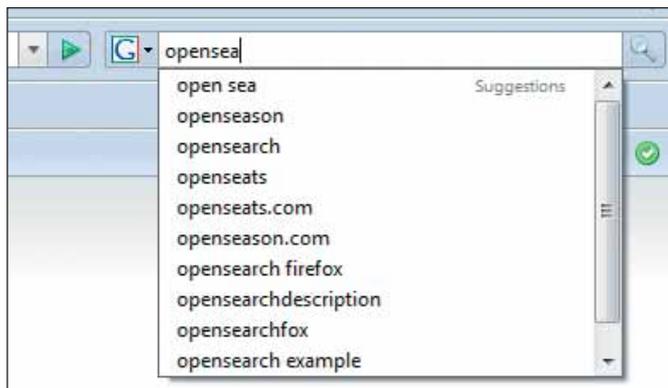
Si votre site internet possède déjà une page de recherche, il vous sera très facile d'intégrer OpenSearch, en effet en plus de RSS/Atom vous pouvez retourner une page HTML classique, dans ce cas il suffit de renseigner les attributs totalResults, startIndex, itemsPerPage via des balises meta :

```
<meta name="totalResults" content="4230000"/>
<meta name="startIndex" content="1"/>
<meta name="itemsPerPage" content="10"/>
```

On peut également définir nos propres type de retour, c'est ce que fait

Développement Web

l'extension suggestion qui retourne les résultats sous forme de JSON (JavaScript Object Notation) ce qui permet à certains clients d'avoir une textbox de suggestion, ce comportement est natif pour Firefox 2.0.



Pour arriver à un tel résultat, il faut renseigner le bon type de réponse dans le document de description du service :

```
<Url type="application/x-suggestions+json"
  template="http://example.com/suggest?q={searchTerms}"/>
```

Le format de la réponse doit être un tableau qui contient en première position le terme de la recherche, en seconde position un tableau qui contient les suggestions de recherche, et en dernière position un autre tableau qui contient le nombre de résultats pour les suggestions de recherche et enfin un dernier tableau qui contient l'url des résultats de la suggestion de recherche. Seuls les deux premiers éléments sont obligatoires. Ce qui donne :

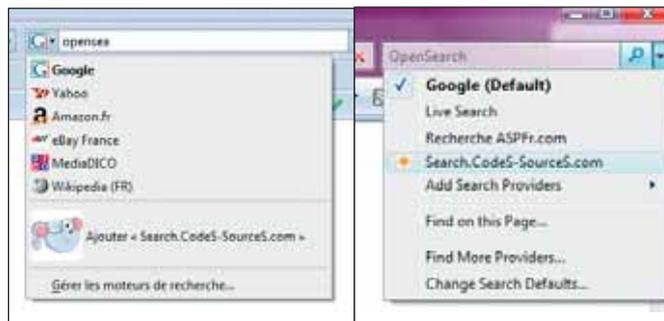
```
["sea",
  ["sears", "search engines"],
  ["7,390,000 results", "17,900,000 results"],
  ["http://example.com?q=sears", "http://example.com?q=search+engines"]
]
```

Faire connaître sa description de recherche OpenSearch

Pour faire connaître votre fichier de description de service de recherche vous pouvez mettre un lien dans vos flux RSS/Atom ou dans les pages HTML en utilisant la balise link :

```
<link rel="search"
  href="http://example.com/opensearchdescription.xml"
  type="application/opensearchdescription+xml"
  title="Content Search" />
```

La balise link est directement reconnue par les documents ATOM et HTML, pour un flux RSS il faut utiliser le namespace du format atom. Lorsque vous visitez une page HTML avec une telle balise link, on peut voir dans IE7 et Firefox que l'icône du moteur de recherche intégré change, en cliquant dessus vous pouvez directement intégrer le service de recherche dans votre navigateur préféré.



Vous pouvez aussi donner la possibilité de le rajouter via le code en utilisant la fonction JavaScript AddSearchProvider :

```
<a href="javascript:window.external.AddSearchProvider('\openSearch.xml\')" > Ajouter Search.CodeS-SourceS.com à vos moteurs de recherche</a>
```

Enfin, il est également possible d'enregistrer votre service de recherche sur le site A9.com qui permet de faire des recherches sur plusieurs sources de données via OpenSearch.

Ressources

- Spécification complète d'OpenSearch : http://www.opensearch.org/Specifications/OpenSearch/1.1/Draft_3
- L'extension Referrer : http://www.opensearch.org/Specifications/OpenSearch/Extensions/Referrer/1.0/Draft_1
- L'extension Relevance : <http://www.opensearch.org/Specifications/OpenSearch/Extensions/Relevance/1.0>
- L'extension Suggestion : <http://www.opensearch.org/Specifications/OpenSearch/Extensions/Suggestions/1.0>
- Créer un fichier de description OpenSearch très facilement : <http://www.microsoft.com/windows/ie/searchguide/en-en/default.mspx>



■ Cyril DURAND

Membre de l'équipe de développement CodeS-SourceS

<http://blogs.CodeS-SourceS.com/cyril/>

- ✓ Nouveau : rubrique humour
- ✓ L'actu quotidienne
- ✓ Le téléchargement

- ✓ Forum des développeurs
- ✓ Nouveau site d'emploi
- ✓ Recevez la Newsletter

www.programmez.com

Development - Consulting / Coaching - Solutions



.NET Experts

Ajax Vista Live.com .NET 3.0
Gadgets WCF
MOSS = WSS V3 WEB 2.0 WPF OpenXML

Des experts techniques à votre service
reconnus aux titres de

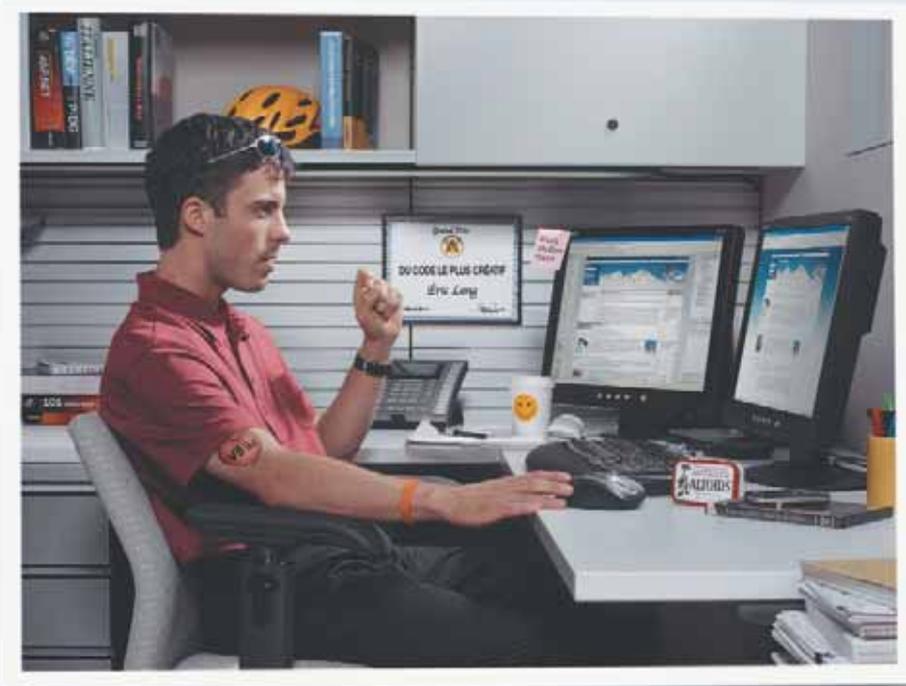
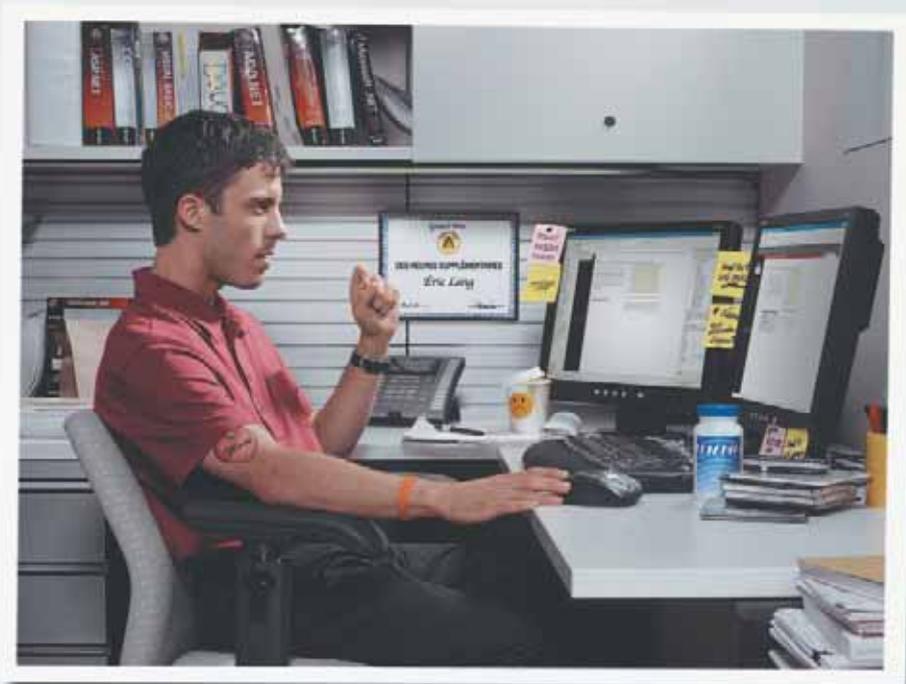


Microsoft
Regional Director
PROGRAM

www.wygwam.com

info@wygwam.com





Nouveau Visual Studio 2005. La différence saute aux yeux.

Vous voyez la différence ? Vous vous en rendez compte dès la première ligne de code. Visual Studio 2005 propose plus de 400 nouvelles fonctionnalités pour vous aider à développer vos applications. Il vous permet d'accélérer vos développements en tapant moins de code, en déboguant plus efficacement, et en profitant des outils de tests avancés. Vous pouvez ainsi vous concentrer sur l'essentiel. Pour découvrir maintenant le nouveau Visual Studio 2005, consultez : www.microsoft.com/france/vs2005

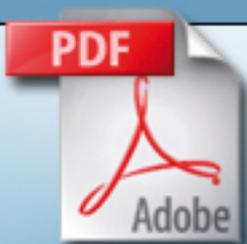
Lisez **Pro**grammez!

Le magazine du développeur

Chaque mois chez votre marchand de journaux

- les actus
- les dossiers
- le code

Achetez le numéro en vente actuellement
Trouvez le point de vente diffusant Programmez
A côté de chez vous



Achetez le numéro
en vente actuellement